Towards Lean 4 Formalization of the Free Energy Principle

2026-08-23

Contents

1 Abstract: Auditing Free Energy Principle Formalisms in Lean 4

The Free Energy Principle (FEP) connects variational inference, active inference, information geometry, Bayesian mechanics, and thermodynamics, but informal presentations often leave types, finiteness conditions, and the distance between a motivating concept and its proved surrogate implicit. We present an open, version-pinned Lean 4 package for auditing that distance. Its catalogue contains **155 stable topics** across five areas, generated from a single typed join of maintained metadata, semantic review records, and canonical Lean bodies. Every topic exposes a primary theorem, its assumptions, a non-vacuity review, an acceptance probe, and a semantic disposition. Compilation maturity and scientific adequacy are therefore reported as different axes.

The formal kernel uses Lean **`4.33.1`** on **`leanprover/lean4:v4.33.1`** with Mathlib **`v4.33.1`**. In particular, the variational-bound and KL rows use Mathlib's native `InformationTheory.klDiv` definition and chain rule, and the fixed-point row assumes a globally quantified contraction and includes a concrete halving-map witness. The current semantic audit classifies **136 of 155** rows as directly formalized at their deliberately narrowed scopes. The schema still distinguishes proxy and gap dispositions, so a later weaker row cannot be promoted by compilation alone. The generated coverage audit records theorem, definition, import, assumption, and non-vacuity coverage for every topic.

Verification evidence is fail-closed. Native compilation claims are populated only from a source-digest-bound receipt covering all 155 topics with zero warnings and zero `sorry`; the rendered native rate is **`155/155 (100.0%)`** and the evidence kind is **`native-lean`**. Hermes/OpenGauss results are a separate, optional full-run contract and enter the manuscript only when an independently validated full receipt is claim-ready (`false`). Catalogue generation by itself cannot satisfy either evidence contract. Manuscript rendering likewise validates every variable before writing to a separate build directory, so unknown placeholders and stale theorem identifiers block publication.

This work does not prove the empirical FEP, the existence of Markov blankets for generic dynamics, or the equivalence of the catalogue's many motivating formalisms. It contributes a reproducible formal-review substrate with **136** directly formalized rows and explicit conditional/structural proxies; a manifested kernel spanning Bayesian inversion, variational and active inference, finite policy trees, temporal and causal structure, native finite blankets, path and exact two-state continuous-time thermodynamics, categorical and scalar exponential-family geometry, collective inference, finite risk, and learning; **125** checked cross-topic witnesses, separated into **20** derivational relations and **105** non-implicational pairings; a warning-free acceptance target; and an explicit separation between locally satisfied capability contracts and stronger empirical or system-level claims.

Keywords: Free Energy Principle; Active Inference; Lean 4; Mathlib4; formal verification; variational inference; Bayesian mechanics; information geometry; theorem maturity; reproducible research.

2 Introduction

2.1 The Verification Gap in Mathematical Physics

The Free Energy Principle (FEP) offers a proposed unifying account of perception, action, and learning (K. Friston 2010). Strong readings associate the persistence of self-organizing systems with minimization of a variational free-energy functional; narrower readings concern explicitly specified generative models and particular dynamics. Over the past two decades, the FEP has generated a rich theoretical ecosystem spanning Active Inference (Parr, Pezzulo, and Friston 2022), Information Geometry (Amari 2016), and Bayesian Mechanics (Da Costa et al. 2024). Yet the mathematical foundations of this ecosystem—drawing simultaneously on measure theory, stochastic differential equations, differential geometry, and category theory—remain difficult to parse, verify, and extend. A working researcher reconstructing a derivation from a flagship paper must typically cross-reference textbooks in four distinct subfields, reconcile inconsistent notation, and silently patch over assumptions the author judged too obvious to state.

This difficulty is not merely pedagogical. Recent critiques (Biehl, Pollock, and Kanai 2021; Aguilera et al. 2022; Andrews 2021) raise substantive concerns about the mathematical status of key FEP claims: the uniqueness of Markov blanket decompositions, the conditions under which steady-state densities exist, and the extent to which variational bounds apply beyond specific model classes. Such debates expose a **verification gap** in mathematical physics: informal review does not provide a mechanically replayable check of every inference. In a literature where results are repeatedly reused, an unstated hypothesis can propagate into later derivations. Formal verification provides a precise checksum: an accepted theorem is derivable from the stated definitions, hypotheses, imported lemmas, and Lean's trusted kernel. That certificate does not establish that the definitions model the world correctly, that the hypotheses hold empirically, or that a narrowed theorem captures the motivating scientific claim. A catalogue of theorem statements, semantic audits, and reproducible compiler evidence lets later work inspect those three questions separately. When the free energy F is claimed to upper-bound surprise, the argument hinges on a chain of measure-theoretic manipulations:

$$F[q, p] = \mathrm{KL}[q(\psi \mid m) \,\|\, p(\psi \mid s, m)] - \log p(s \mid m) \geq -\log p(s \mid m), \tag{1}$$

(Equation 1.) The inequality holds because Kullback–Leibler divergence is non-negative by Gibbs' inequality. Each of those symbols—q, p, KL, $\log p(s \mid m)$—carries type-theoretic weight: q is a probability measure absolutely continuous with respect to p, KL is the Radon–Nikodym-derivative integral $\int \log \frac{dq}{dp}\, dq$, and $\log p(s \mid m)$ is a real-valued random variable. In a journal proof these conditions are implicit; in a theorem prover they must be declared, and the compiler rejects the proof if they are not.

2.2 Why Formal Verification of FEP Matters for Cognitive Science

The stakes are concrete. Active Inference is now used to model cortical processing, motor control, psychiatric conditions, and the behavior of multi-agent biological systems. When a clinical claim rests on the "free energy minimization" rationale, the underlying inequality should be correct by construction rather than by editorial consensus. Three specific benefits follow from machine-checked FEP mathematics:

1. **Unambiguous statements.** Each theorem forces explicit declaration of the measurable space, the dominating measure, and the policy type, eliminating the category errors that (Andrews 2021) identify as pervasive in the literature.
2. **Compositional reasoning at scale.** Once a lemma (for example, KL non-negativity) is kernel-checked, downstream proofs may reuse its statement without replaying its implementation, while still inheriting its exact hypotheses. A community library of FEP theorems would give each new manuscript a springboard rather than a restart.
3. **Automated differentiation of hype from theorem.** When Lean refuses to close a goal, the author sees precisely which hypothesis is missing. This provides a principled interface between informal intuition and formally defensible claim.

2.3 Interactive Theorem Provers as Resolution Mechanism

Interactive Theorem Provers (ITPs) such as Lean 4 (Moura and Ullrich 2021) address this challenge directly. Lean 4's dependent type system implements the Calculus of Inductive Constructions, so accepted theorems are checked against the kernel. A theorem proven in Lean produces a *proof object*—a certificate that an independent verifier can re-check. Its community library, Mathlib4 (The mathlib Community 2020), covers topology, measure theory, algebra, geometry, and many other areas. Notable successes include the Lean 4 formalization of the polynomial Freiman–Ruzsa proof ("Polynomial Freiman–Ruzsa in Lean 4" 2023) and the Liquid Tensor Experiment (Scholze and Commelin 2022). Stochastic process foundations—critical for FEP path integrals—remain uneven in Mathlib4 at large; the shipped catalogue rows are nonetheless `sorry`-free under this project's maturity policy, while broader SDE and continuous-time stochastic infrastructure remains aspirational where the pinned revision does not supply a reviewed interface (see §4.17.7).

Lean 4 was selected for project-specific, reproducible reasons rather than a mutable cross-prover ranking:

Requirement	Evidence at the pinned Lean/Mathlib revision	Project consequence
Measure-valued probability	`Measure`, Radon–Nikodym derivatives, finite measures, and Markov kernels	State probability claims at their native measure-theoretic level

Requirement	Evidence at the pinned Lean/Mathlib revision	Project consequence
Information theory	`InformationTheory.klDiv` plus self, zero-characterization, Gibbs, and kernel chain-rule lemmas	Reuse native KL results in fep-002 and fep-014
Executable proof tooling	Lean elaborator, kernel, and tactics available through the pinned Lake workspace	Produce replayable per-topic and aggregate compiler evidence
Research-tool integration	Lean source is text, compiler results are structured, and local tooling can isolate invocations	Keep optional LLM commentary outside the acceptance boundary

This is a fitness claim for the present catalogue, not a claim that Lean dominates Coq, Isabelle/HOL, Agda, or another prover for every physical theory.

2.4 Origins and Context: FEP and Lean 4 / Mathlib4 Maturity

The FEP was introduced by Karl Friston in a sequence of papers between 2005 and 2010 (K. J. Friston 2005; K. Friston, Kilner, and Harrison 2006; K. Friston 2010), extending Helmholtz-machine and predictive-coding ideas. Active-Inference accounts subsequently developed perception–action formulations (K. Friston et al. 2017), while recent Bayesian-mechanics work developed path-integral and non-equilibrium-statistical-mechanics connections under explicit modeling assumptions (Da Costa et al. 2024; K. Friston et al. 2023). The core variational objective is often written as

$$F = \mathbb{E}_q[\log q(s) - \log p(o, s)] \tag{2}$$

(Equation 2.) It is the negative evidence lower bound under the usual variational-Bayes construction. Connections to Helmholtz free energy require a specified energy map, normalization, and units; §5.4.2 states one such bridge and separates it from the current Lean result.

Lean 4 reached a comparable inflection point in parallel. Lean 4.0.0 was released in 2023, Mathlib4 completed its migration from Lean 3 in the same year, and the pinned toolchain used in this work (**`leanprover/lean4:v4.33.1`**, Mathlib4 **`v4.33.1`**) includes mature measure theory and a native Kullback–Leibler divergence API. The catalogue imports `Mathlib.InformationTheory.KullbackLeibler.Basic` and `.ChainRule` directly. This matters methodologically: library availability is established from the pinned source and compiled declarations, not inferred from secondary roadmap commentary.

2.5 LLM-ITP Integration: Beyond Problem Solving

The integration of Large Language Models with ITPs has produced strong results in parallel. Systems such as LeanDojo and ReProver (Yang et al. 2024), LEGO-Prover (Xin et al. 2024b), DeepSeek-Prover (Xin et al. 2024a), and the more recent DeepSeek-Prover-V2 (Xin et al. 2025) demonstrate that LLMs can effectively navigate proof search spaces, while AlphaProof (AlphaProof team, Google DeepMind 2024) solved International Mathematical Olympiad problems at a silver-medal level. Lean Copilot (Song, Yang, and Anandkumar 2025) brings LLM-assisted tactic suggestion directly into the developer workflow. The pinned Lean 4 **`4.33.1`** toolchain (`lean/lean-toolchain`) and Mathlib4 **`v4.33.1`** supply automation including `grind` (SMT-style), `positivity`, and related tactics, expanding the proof capabilities available to our pipeline.

Our work targets the **axiomatization of a physical theory** in a proof assistant: turning informal FEP statements into well-typed Lean specifications. Related formalization efforts exist nearby (e.g., categorical ontology definitions or classical simulation boundaries; see (Namjoshi 2026)); this catalogue is a systematic, template-integrated slice focused on FEP-facing rows. The task demands domain knowledge spanning neuroscience, statistical mechanics, and measure theory—material that must be *translated* from informal mathematical physics into formal specifications, not merely retrieved from Mathlib4.

2.6 The FEP Lean Pipeline

The package has three intentionally separate evidence modes. Catalogue mode produces deterministic offline projections—including the coverage report and formalism atlas—and carries no verification claim. Native mode compiles selected canonical topic bodies with the pinned Lean/Mathlib workspace, can write a digest-bound topic receipt, and has a separate declaration/axiom audit for reviewed formal witnesses. Full mode additionally requires Hermes and OpenGauss capabilities and produces a report bundle whose hashes, source digests, topic roster, and verification provenance are independently checked. A success bit from one mode is never reused as evidence for another.

This separation is central to the research design. The Lean kernel, not an LLM, decides whether a topic body type-checks. The semantic audit then asks a different question: whether the proved statement reaches the topic-facing scientific claim. Finally, receipt validation asks whether a reported run corresponds to the current source. The manuscript renderer consumes only those typed projections and refuses unresolved variables. The current rendered native compilation rate is **`155/155 (100.0%)`**; the evidence kind is **`native-lean`**, and full-run claim readiness is **`false`**.

2.7 Research Contributions

This manuscript makes five principal contributions:

- **(C1) A distributable formalism catalogue.** The `fep_lean` package exposes 155 stable topics through one import namespace. Static metadata, semantic review, Lean bodies, YAML, and the appendix follow an explicit generation graph rather than competing as sources of truth.
- **(C2) A semantic maturity audit.** Each topic records a primary theorem, assumption review, non-vacuity argument, and acceptance probe. The audit distinguishes direct formalization from conditional, structural, and scope-limited proxies; `mathlib_status: real` means a body is intended to compile, not that the motivating scientific claim is complete.
- **(C3) Deeper and compositional formal statements.** The variational and KL rows use Mathlib's native KL definition, identity law, zero characterization, and Markov-kernel chain rule. The manifested foundations now span measure and finite Bayesian inversion, variational duality, controlled and temporal inference, causal interventions, generalized predictive coding, finite path thermodynamics, categorical and optimization geometry, collective inference, concentration, and model evidence in addition to the original finite active-inference kernel. Manifested composition leaves prove **125** cross-topic witnesses rather than inferring them from shared imports; the graph distinguishes **20** derivations or identifications from **105** checked pairings, and `composed.lean` is only their import aggregate.
- **(C4) Evidence-separated verification.** Native Lean and full Hermes/OpenGauss evidence have different receipt schemas and claim predicates. Both bind evidence to the current source, while catalogue-only output is ineligible by construction.
- **(C5) Publication contract audits and visualization.** Generated breadth/depth coverage, an offline accessible formalism atlas, declaration/axiom checking, theorem-reference checking, placeholder validation, and source-to-build rendering make drift visible before publication.

The machine-facing evidence for C1–C3 is the generated catalogue and appendix (§5; Appendix 13). The methodological evidence for C4–C5 is described in §4.20 and §4.21. At this rendered snapshot, native evidence is **`native-lean`** with rate **`155/155 (100.0%)`**; full-run claim readiness is **`false`**.

What this contribution is, and is not. This is not a proof that the FEP is empirically true or that its many formulations are mutually equivalent. Lean establishes exactly the propositions stated under exactly their hypotheses. The high direct-formalization count reflects disciplined narrowing to exact local contracts, not an end-to-end theorem of self-organization. The contribution is therefore a formal review instrument upstream of empirical and dynamical validation, not a substitute for either.

2.8 Paper Organization

The remainder of this paper is organized as follows.

§3 reviews FEP and Active Inference background, the relevant Lean 4 / Mathlib4 capabilities, and adjacent ITP efforts.

§4 details the Lean 4 / Mathlib4 methodology and pipeline architecture. Six deep-dive subsections (§4.16–§4.21) cover Lean 4 fundamentals, Mathlib4 coverage, the `sorry` maturity taxonomy, the Hermes AI agent, native `lake env lean` compilation, and the orchestration DAG.

§5 presents results for the 155-topic catalogue across the five theoretical areas—FEP foundations, Active Inference, Information Geometry, Bayesian Mechanics, and Thermodynamics—and then isolates the reusable finite kernel in §5.10. The first ten seven-topic expansion families are synthesized in §6; §7 adds finite-sample risk, finite policy trees, native blanket transfer, exponential-family dual geometry, and exact two-state continuous time. The injected `compile_rate` metrics (from `manuscript_vars.yaml`) are reported alongside Hermes and native verification statistics in §5.5.

§8 examines Mathlib4 coverage gaps, the execution-integrity standard, and implications for the FEP debate. **§10** concludes with an engineering-outcomes analysis and future directions.

Appendix 12 orients readers to the catalogue, anchors, and injection path. **Appendix 13** is the unified per-topic catalogue: each stable topic juxtaposes the full Lean body with typeset statement signatures and deterministic section/equation anchors for cross-references.

2.9 Notation

The following notation is used throughout this paper:

Symbol	Definition	First use
$\mathrm{F}[q, p]$	Variational free energy functional	§3.1.2, Eq. 4
$\mathrm{G}(\pi)$	Expected free energy under policy π	§3.1.6, Eq. 14
$\mathrm{KL}[q\|p]$	Kullback-Leibler divergence from q to p	§3.1.2, Eq. 4
$\mathrm{H}[q]$	Shannon entropy of distribution q	§3.1.6
$\mathbb{E}_q[\cdot]$	Expectation under distribution q	§3.1.2, Eq. 4
$\Omega, \mathcal{F}, P$	Sample space, sigma-algebra, and probability measure	§4.16

Symbol	Definition	First use
$q \ll p$	Absolute continuity (q is abs. continuous w.r.t. p)	§4.16
$\frac{dq}{dp}$	Radon-Nikodym derivative	§4.17
`sorry`	Lean 4 tactic admitting a goal without proof	§4.18
∇	Gradient operator (on statistical manifold or $\mathbb{R}^n$)	§3.1.6
Γ	Solenoidal flow operator	§3.1.6
$Q = -Q^\top$	Skew-symmetric (solenoidal) matrix	§3.1.6, Eq. 70
F, U, T, S	Helmholtz free energy, internal energy, temperature, entropy	§5.4

3 Background and Related Work

3.1 The Free Energy Principle and Its Mathematical Structure

A strong reading of the Free Energy Principle (FEP) holds that a bounded dynamical system that persists over time can be interpreted, under suitable assumptions, as performing approximate Bayesian inference (K. Friston, Kilner, and Harrison 2006). This proposal, developed by Karl Friston (K. Friston 2010) and extended into a physics-of-self-organization program (K. Friston 2019), has been applied from single-cell homeostasis to human cognition through Active Inference (K. Friston et al. 2017; Parr, Pezzulo, and Friston 2022). Behind the strongest claim lie at least two mathematical obligations: a specified variational free-energy functional and a theorem connecting the selected system dynamics to a gradient flow or related descent on that functional. Neither persistence nor the name "free energy" supplies that connection automatically.

3.1.1 Variational Free Energy as an Evidence Lower Bound (ELBO)

The machine learning literature (Blei, Kucukelbir, and McAuliffe 2017) introduces the same quantity under the name *evidence lower bound* (ELBO). For a latent variable z, observation x, generative model $p(x, z)$, and variational posterior $q(z)$,

$$\mathrm{ELBO}(q) = \mathbb{E}_{q(z)}[\log p(x, z) - \log q(z)] = \log p(x) - \mathrm{KL}[q(z) \,\|\, p(z \mid x)]. \tag{3}$$

(Equation 3.) After identifying $z \leftrightarrow \psi$, $x \leftrightarrow s$, and matching the joint, posterior, support, and integrability assumptions, the associated variational free energy is the negative ELBO. This correspondence is central, but it is not automatic: Mathlib4 results about KL divergence and expectations become reusable only after their typed measures, codomains, and finiteness hypotheses have been connected to the chosen FEP model.

3.1.2 Formal Definition: Variational Free Energy

The variational free energy F for an agent with recognition density $q(\psi \mid m)$, generative model $p(s, \psi \mid m)$, and sensory observations s is defined as:

$$\mathrm{F}[q, p] \;=\; \underbrace{\mathrm{KL}[q(\psi \mid m) \,\|\, p(\psi \mid s, m)]}_{\geq 0} \;-\; \underbrace{\log p(s \mid m)}_{\text{log-evidence}} \tag{4}$$

Because KL divergence is non-negative by Gibbs' inequality, this immediately yields the **variational bound**:

$$\mathrm{F}[q, p] \;\geq\; -\log p(s \mid m) \;=\; \text{surprise} \tag{5}$$

Equivalently, the free energy admits an **energy-entropy decomposition**:

$$\mathrm{F}[q, p] \;=\; \underbrace{\mathbb{E}_q[-\log p(s, \psi \mid m)]}_{\text{energy}} \;-\; \underbrace{\mathrm{H}[q(\psi \mid m)]}_{\text{entropy}} \tag{6}$$

These dual decompositions—(1) as KL plus log-evidence and (3) as energy minus entropy—are the starting point for all subsequent formalisms in this paper. In Mathlib4 parlance, Eq.~6 is a statement about `∫ (fun ψ => -Real.log (p (s,ψ))) ∂(q)` together with `MeasureTheory.entropy q`; the mere act of writing this expression forces declaration of a measurable space `α` and an integrability hypothesis, neither of which typically appears in journal papers.

3.1.2.1 Three Equivalent Decompositions of Variational Free Energy

Before proceeding to Active Inference, it is worth exhibiting three forms of F that are algebraically equivalent when the displayed densities, posterior, support, and integrability conditions exist, since each form motivates a different slice of the Lean 4 catalogue. Let o denote observations (written s elsewhere), s the hidden (latent) states (written ψ elsewhere), $p(o, s)$ the generative model, and $q(s)$ the recognition density.

(1) Surprise bound (posterior-tracking form). Starting from Bayes' rule $p(s \mid o) = p(o, s)/p(o)$ and adding and subtracting $\log q(s)$ inside an expectation under q,

$$F[q] \;=\; -\log p(o) \;+\; \mathrm{KL}\big[q(s) \,\|\, p(s \mid o)\big] \;\geq\; -\log p(o). \tag{7}$$

This is the "free energy bounds surprise" identity: the first term is the (negative log) model evidence—the Shannon surprise $-\log p(o)$ that the agent cannot change by rearranging beliefs—and the second is a non-negative KL gap that vanishes iff $q = p(\cdot \mid o)$.

(2) Energy–entropy form. Multiplying out the logarithm gives

$$F[q] \;=\; \mathbb{E}_{q(s)}\big[-\log p(o, s)\big] \;+\; \mathbb{E}_{q(s)}\big[\log q(s)\big] \;=\; U_q \;-\; H[q], \tag{8}$$

where $U_q := \mathbb{E}_q[-\log p(o, s)]$ is the (cross-)energy under the joint generative model and $H[q] := -\mathbb{E}_q[\log q(s)]$ is the Shannon entropy of the recognition density. This is the form most directly connected to statistical-mechanical free energy (Helmholtz $F = U - TS$, with $T = 1$ in natural units).

(3) ELBO form. Because $F[q] = \mathbb{E}_q[\log q(s) - \log p(o, s)]$, one has

$$F[q] \;=\; -\mathrm{ELBO}(q), \qquad \mathrm{ELBO}(q) \;=\; \mathbb{E}_{q(s)}\big[\log p(o, s) - \log q(s)\big]. \tag{9}$$

This identity is what licenses the direct reuse of the machine-learning ELBO apparatus: variational Bayes, amortized inference, and the reparameterization trick all minimize $-$ELBO, which is exactly F.

Why F is useful to minimize. Eq.~7 exhibits F as surprise plus a nonnegative KL gap. With the generative model fixed and the recognition family rich enough to contain the posterior, minimizing over q attains the exact posterior. Model-parameter learning is more delicate because the posterior and KL term generally change with those parameters; evidence maximization follows directly only after optimizing the variational family exactly, or under a separately justified coordinate-ascent argument. A single gradient step therefore cannot be described unconditionally as both exact perception and evidence accumulation. The maintained finite kernel proves the posterior-attainment statement and its support-qualified uniqueness, while parameter-learning and marginal-likelihood optimization remain distinct obligations.

3.1.3 Predictive Coding as Precision-Weighted Prediction Error

The FEP additionally predicts a specific microscopic form for belief updates. Assuming a generative model whose likelihood is a nonlinear Gaussian $p(x \mid s) = \mathcal{N}(x;\, g(s),\, \Sigma_\varepsilon)$ with precision $\Pi_\varepsilon = \Sigma_\varepsilon^{-1}$, a point-mass or Laplace recognition density concentrated at μ, and differentiable g, the gradient of F with respect to the mean takes the canonical precision-weighted prediction-error form

$$\dot{\mu} \;=\; -\frac{\partial F}{\partial \mu} \;=\; \big(\partial_\mu g(\mu)\big)^\top \Pi_\varepsilon\, \varepsilon \;-\; \Pi_s\,(\mu - \mu_{\text{prior}}), \qquad \varepsilon := x - g(\mu), \tag{10}$$

where ε is the sensory prediction error, Π_s is the prior precision on μ, and μ_{prior} is the prior expectation. Here $\Pi_\varepsilon \cdot \varepsilon$ is the precision-weighted prediction error: each component of the error is rescaled by how confident the model is about that channel, so that more reliable sensory dimensions drive belief updates more aggressively. The first term on the right drives μ to reduce sensory prediction error; the second term anchors μ to its prior.

This equation ties three separate strands of the catalogue together. (i) It is a **gradient flow** on F and therefore shares the contraction structure formalized for quadratic descents in **fep-032** (`descent_contracts`, `grad_sq_nonneg`, `fixed_point`). (ii) Under the Laplace approximation introduced next, the energy U_q reduces to a sum of quadratics in ε weighted by the precisions Π; this is exactly the quadratic-minimum structure formalized in **fep-016** (`sq_nonneg`, `minimum at mode`, `precision-weighted quadratic`). (iii) Message passing across hierarchical layers of a predictive-coding network (K. Friston, Parr, and Vries 2018) propagates the same form recursively, which is the structural content of **fep-045** (`ConjugateFamily`, `fold`, `single_update`) and the monotone-composition lemmas in **fep-048**. A reader who wants to know where in the Lean 4 catalogue the "prediction error" half of the FEP lives should therefore look at the intersection of these four rows.

3.1.4 The Laplace Approximation and the Quadratic Form of F

The FEP in its most widely used form does not carry a fully nonparametric q; it typically employs the **Laplace assumption**—that q is Gaussian, fully parameterized by its mean μ and covariance Σ:

$$q(s) \;=\; \mathcal{N}(s;\, \mu,\, \Sigma). \tag{11}$$

When F is the negative log posterior and $H_F(\mu) := \partial^2 F / \partial\mu\partial\mu^\top$ is positive definite at the mode, the local Laplace covariance is

$$\Sigma^* \;=\; H_F(\mu^*)^{-1} \;=\; \left[-\nabla^2 \log p(\mu, o)\big|_{\mu^*}\right]^{-1}. \tag{12}$$

Thus Σ^* is the inverse observed information at the MAP estimate. Substituting this local Gaussian approximation back into Eq.~8 gives an entropy log-determinant and a second-order energy expansion. In common Gaussian observation models this yields a **precision-weighted quadratic** in the prediction errors,

$$F_{\text{Laplace}}(\mu) \;\approx\; \tfrac{1}{2}\,\varepsilon^\top \Pi_\varepsilon\,\varepsilon \;+\; \tfrac{1}{2}\,(\mu - \mu_{\text{prior}})^\top \Pi_s\,(\mu - \mu_{\text{prior}}) \;-\; \tfrac{1}{2}\log\det(\Pi_\varepsilon\,\Pi_s) \;+\; \text{const}, \tag{13}$$

plus terms that vanish at μ^*. Precision-weighted quadratic objectives are common in Laplace and predictive-coding implementations, but **fep-016** formalizes only their scalar algebraic substrate: square nonnegativity, the value zero at $x = \mu$, nonnegativity after multiplication by a nonnegative precision, symmetry, and expansion of $(x-\mu)^2$. It does not prove uniqueness when the precision may be zero, derive the quadratic from a likelihood or Hessian, or assemble the matrix expression in Equation~13. The catalogue keeps this scaffolding separate from claims about the full F functional.

3.1.5 The Active Inference Perception–Action Loop

Active Inference extends the FEP by positing that agents minimize not only present free energy but *expected* free energy under each available policy π:

$$\mathrm{G}(\pi) = \underbrace{\mathbb{E}_{q(o|\pi)}[-\log p_C(o)]}_{\text{pragmatic cost}} - \underbrace{\mathbb{E}_{q(o,s|\pi)}[\log q(s \mid o, \pi) - \log q(s \mid \pi)]}_{\text{epistemic value}} \tag{14}$$

where o are predicted observations, s are hidden states, and p_C is a normalized preferred-outcome law. The second term is the mutual information $I_q(s; o \mid \pi)$ and therefore rewards information gain by lowering G; the first is preference cross-entropy and rewards policies that predict preferred outcomes. This sign convention is one among several used in the literature (Millidge, Tschantz, and Buckley 2021). The maintained finite kernel fixes it explicitly and derives the equivalent risk–ambiguity form under full-support hypotheses. Policies are then selected by a softmax:

$$P(\pi) \propto \exp(-\gamma \cdot \mathrm{G}(\pi)), \tag{15}$$

(Equation 15.) with precision parameter $\gamma > 0$. Formalizing this loop in Lean 4 requires a `Fin n → Action` policy type, a measurable space of observations, and a summation over the (finite) policy set—all of which are available in `Algebra.BigOperators.Group.Finset` and `Data.Fin`.

3.1.6 The Theoretical Landscape

The FEP ecosystem is heavily stratified mathematically, progressing from foundational variational calculus to cutting-edge stochastic physics. Each stratum is anchored by a distinguished mathematical object, and each object demands a different slice of Mathlib4:

1. **Foundational Variational Bounds** (Eqs.~4–6): Built upon Kullback–Leibler (KL) divergence and the Evidence Lower Bound (ELBO) originating in machine learning. Variational free energy F serves as a tractable upper bound on surprise under the associated variational construction (K. Friston et al. 2007; K. Friston, Trujillo-Barreto, and Daunizeau 2008), and generalized free energy extends the objective to accommodate model uncertainty (Parr and Friston 2019). A Boltzmann–Gibbs density $p^*(\Gamma) \propto \exp[-F(\Gamma)]$ becomes a bridge to statistical mechanics only after an energy map, units, normalization, and temperature convention are fixed; an equilibrium law and a Bayesian posterior are not identical merely because both can be written in exponential form.

2. **Active Inference** (EFE and policy objectives; Eq.~14): Introduces temporal policies in which organisms take physical action to minimize Expected Free Energy (K. Friston et al. 2015), decomposed into epistemic and pragmatic terms (Sajid et al. 2021). The relationship between VFE and EFE, and the hypotheses needed to move among published EFE decompositions, are nontrivial (Millidge, Tschantz, and Buckley 2021). Champion et al. (Champion et al. 2026) compare four formulations and give conditions for a unifying likelihood mapping. Topic fep-021 fixes an extended-nonnegative-real convention, $G = C \dot{-} \mathrm{IG}$, while the maintained finite carrier derives real-valued pragmatic-minus-epistemic and risk-plus-ambiguity identities and stage-updated open-loop EFE. The controlled expansion adds finite soft-control recursions and one exact two-stage observation-dependent feedback witness; the later policy-tree family defines observation-contingent trees at arbitrary finite depth, finite Bellman optima, open-loop embedding and dominance, and a treewise EFE decomposition (K. Friston et al. 2020). The collective expansion adds explicitly independent product-agent and fixed-consensus laws. None establishes equivalence to every published EFE formulation, infinite-horizon or continuous-belief planning, or emergent collective agency. Central informal object: a policy selector $\pi^* = \arg\min_\pi \mathbb{E}[G(\pi)]$.

3. **Information Geometry** (§5.3.5, §4.17): Models belief updates as traversal of statistical manifolds governed by the Fisher Information Metric and natural gradients (Amari 1983, 2016). The space of probability distributions becomes a Riemannian manifold whose geometry encodes local statistical distinguishability. The Fisher Information Metric is defined as

$$g_{ij}(\theta) = \mathbb{E}_{p(x|\theta)}\left[\frac{\partial \log p(x \mid \theta)}{\partial \theta^i} \cdot \frac{\partial \log p(x \mid \theta)}{\partial \theta^j}\right], \tag{16}$$

(Equation 16.) Natural gradient descent replaces ∇F by $g^{-1}\nabla F$. Under the regularity, nondegeneracy, and transformation assumptions that make the statistical manifold and inverse metric well-defined, this vector field is coordinate-covariant and represents metric steepest descent. The catalogue constructs the complete one-parameter Bernoulli instance and a finite categorical score carrier with explicit full-rank and null directions. It proves simplex-tangent Fisher positivity, pullback, scalar Cramér–Rao, invertible-chart natural-gradient equivariance, mirror and affine Bregman identities, and replicator equivalence. The later full-support scalar exponential family adds log-partition gradient/Hessian, centered score, Fisher–variance equality, KL–Bregman duality, and interval-local mean-coordinate injection. It does not yet define arbitrary smooth atlases, affine connections, curvature, or general geodesic existence.

4. **Bayesian Mechanics** (§5.3.6): Develops inference-related dynamics using Fokker–Planck equations, non-equilibrium steady states (NESS), and decompositions with skew components (Parr, Da Costa, and Friston 2018; K. Friston et al. 2021). Sakthivadivel (Sakthivadivel 2023) frames this as "a physics of and by beliefs," while Friston et al. (K. Friston et al. 2023) develop a path-integral treatment of particular kinds. The catalogue formalizes measure and finite Bayesian inversion, filtering and smoothing, hierarchy and model averaging, finite blanket and intervention laws, and finite stationary currents. It does not derive the general stochastic-analytic Fokker–Planck construction or causal identification from observational data.

5. **Thermodynamic Foundations**: Relates FEP language to thermodynamic potentials, finite path-law fluctuation and Jarzynski identities (Jarzynski 1997), Landauer's principle (Landauer 1961), and nonequilibrium thermodynamics (Prigogine and Nicolis 1977; Pavliotis 2014). The standard Helmholtz relation $\mathcal{F} = -k_{\mathrm{B}}T \log Z$ and the variational identity $F_{\text{var}} = \mathrm{KL}(q\|p(\cdot \mid o)) - \log p(o)$ live in different modeling layers. Identifying them requires an explicit Boltzmann generative model, units, normalization, and temperature scaling; the present catalogue proves scoped finite identities but does not prove that physical identification.

3.1.7 The Formalization Gap

This project addresses a narrower, directly inspectable **verification gap**: familiar FEP labels often sit several definitions and hypotheses away from the Lean propositions that can currently be stated against the pinned library. It does not claim priority over every earlier formal-methods treatment. Table 1 reports the present catalogue's scope, rather than attempting an unbounded census of prior work or numerical software.

Topic-facing concept	Current formal object	Semantic disposition
Variational free-energy bound (fep-002)	Surprisal plus native $\mathbb{R}\geq 0\infty$ KL; bound and posterior self-case	`formalized` at this narrowed abstraction
KL laws (fep-014)	Native KL self/zero laws and composition-product chain rule	`formalized`
EFE convention (fep-021)	Explicit `ENNReal` pragmatic-cost-minus-epistemic-value convention with monotonicity and balance	`formalized` at that convention
Four-block partition (fep-005)	Pairwise-disjoint finite fibers with unique state membership; no blanket independence claim	`formalized` at partition scope
Nonequilibrium current (fep-025)	Transition-induced antisymmetric current, stationarity conservation, and a nonzero three-cycle witness	`formalized` in finite state space
Fisher metric (fep-004/fep-038)	Positive-definite finite weighted metric and an exact Bernoulli statistical-family specialization	`formalized` at finite/Bernoulli scope
Finite softmax (fep-028)	Support-aware full finite probability vector with exact zero off-support and global normalization	`formalized`
Conjugate update (fep-045)	Exact normalized Bernoulli posterior closure and parameter update under positive binary evidence	`formalized` at binary scope

Current semantic status of representative catalogue rows. The generated coverage report is authoritative for all 155 rows.

3.2 Interactive Theorem Proving: Lean 4 and Mathlib

Lean 4 is a functional programming language and interactive theorem prover (ITP) based on dependent type theory (Moura and Ullrich 2021). Machine-checked projects such as CompCert (Leroy 2009), the perfectoid-spaces development (Buzzard, Commelin, and Massot 2020), the Liquid Tensor Experiment (Scholze and Commelin 2022), and the polynomial Freiman–Ruzsa proof ("Polynomial Freiman–Ruzsa in Lean

4" 2023) illustrate different scales and styles of formal verification. This work targets **Mathlib4** (The mathlib Community 2020) through the pinned **`leanprover/lean4:v4.33.1`** toolchain and Mathlib **`v4.33.1`** revision recorded in `lean/lean-toolchain` and `lean/lakefile.lean`; it does not rely on mutable upstream declaration counts.

3.2.1 Type Theory Foundations and Tactic Mode

Lean 4 rests on the Calculus of Inductive Constructions (CIC), a dependent type theory in which propositions and types are the same kind of object. A proof of proposition `P` is a term of type `P`; consequently, the kernel that type-checks terms is the same kernel that verifies proofs. Writing a proof in Lean is therefore identical to writing a program whose type is the statement of the theorem. Users rarely write proof terms directly; instead, they invoke *tactics*—metaprograms that incrementally build the term. A proof in tactic mode takes the following shape:

```
theorem kl_nonneg {α : Type*} [MeasurableSpace α] (μ ν : Measure α)
    : 0 ≤ InformationTheory.klDiv μ ν := by
  exact zero_le _
```

Here `by` enters tactic mode and `zero_le` discharges non-negativity because Mathlib's native divergence takes values in the nonnegative extended reals. The type, rather than an informal convention, rules out a negative result. This style is introduced in depth in §4.16.

3.2.2 Why Lean 4 for Physical Theories?

Lean 4 offers four properties critical for formalizing the FEP:

1. **Dependent types with universe polymorphism**: Types can depend on values, enabling precise encoding of parameterized families of probability measures, finite policy spaces, and transition kernels across the universes used by Mathlib.
2. **Mathlib4's measure-theoretic and information-theoretic stack**: The pinned library provides Bochner integration, Radon–Nikodym derivatives (`MeasureTheory.Measure.rnDeriv`), sigma-algebra constructions, finite and probability measures, Markov kernels, and `InformationTheory.klDiv : Measure α → Measure α → ℝ≥0∞`. It also supplies self-divergence, a zero characterization for finite measures, Gibbs' inequality, and a composition-product chain rule. The extended codomain preserves infinite divergence; any real-valued FEP identity still needs explicit finiteness before applying `.toReal`.
3. **Advanced proof automation**: The pinned Lean 4 `leanprover/lean4:v4.33.1` toolchain and Mathlib4 `v4.33.1` supply automation including the `grind` tactic (SMT-style), `positivity`, and related solvers. These capabilities expand the range of FEP theorems that can be verified without manual proof construction; newer releases add further improvements.
4. **Available LLM–ITP tooling**: LeanDojo (Yang et al. 2024), Lean Copilot (Song, Yang, and Anandkumar 2025), and LEGO-Prover (Xin et al. 2024b) provide relevant examples of Lean-facing proof-search or assistance. Their existence motivates the optional review stage; it supplies no correctness evidence for this catalogue.

Modern physical theories, however, often outpace formalized repositories. The catalogue and maintained foundations now cover finite Markov dynamics, a one-parameter smooth statistical instance, arbitrary finite-dimensional score geometry, native posterior kernels, generic conditional-independence laws, and a concrete finite blanket factorization with typed dynamics. Full continuous Bayesian mechanics still requires stochastic processes, Fokker–Planck evolution, smooth manifold structure, and system-specific blanket-existence arguments. Rows are therefore narrowed to the strongest exact theorem supported by their objects instead of adding local axioms solely to obtain a green compiler result.

3.2.3 Lean 4 Release Cadence and Tactic Evolution

Lean 4 has maintained a rapid release cadence. The repository pins **`leanprover/lean4:v4.33.1`** in `lean/lean-toolchain`; later releases add further automation and ergonomics beyond what this paper's lemmas rely on.

This evolution shapes maintenance, but tactic availability is not the semantic maturity criterion. A row can be warning-free and `sorry`-free while proving only a weak proxy. Upgrades therefore require both a native compile and a renewed semantic review of the theorem's relation to its advertised FEP claim.

3.2.4 Prior Formalization Work in Adjacent Domains

A range of prior efforts have formalized parts of cognitive science, statistical mechanics, or information theory in interactive theorem provers. None targets the FEP directly, but each offers methodological lessons:

Project	System	Domain	Scope	Relevance to FEP
Hammers for Helmholtz (Avigad, Hölzl, and Serafin 2017)	Lean 3	Real analysis / measure theory	Foundational	Provides the measure-theoretic backbone reused in Mathlib4

Project	System	Domain	Scope	Relevance to FEP
Information-theoretic inequalities (Mehta et al. 2021)	Isabelle/HOL	Classical information theory	Shannon entropy, mutual information	Complementary to KL-based FEP work
Formalized statistical mechanics (Paulson 2022)	Isabelle/HOL	Classical thermodynamics	Partition functions, entropy	Thermodynamic catalogue rows build on analogous ideas
LeanDojo (Yang et al. 2024)	Lean 4	Proof search benchmarks	Retrieval-augmented LLM	Demonstrates tractability of LLM ↔ Lean interfaces
Mathlib information theory (The mathlib Community 2020)	Lean 4	Measure-theoretic information theory	Native KL divergence and chain rules	Direct library substrate for fep-002 and fep-014
Categorical ontology / classical simulation (Namjoshi 2026)	Lean 4	Foundations	Definitions of classical / quantum systems	Adjacent formalization of physical theories

The landscape shows that adjacent domains have proven tractable. We have not conducted the systematic, date-bounded search needed to establish whether this is the first catalogue-scale FEP formalization, so we make no priority claim. The inspectable contribution is the particular versioned catalogue, semantic audit, and receipt boundary reported here.

3.3 The LLM–ITP Bridge

Recent systems illustrate several ways to connect language models with interactive theorem provers:

System	Year	Approach	Benchmark	Key innovation
LeanDojo (Yang et al. 2024)	2024	Retrieval-augmented	LeanDojo Benchmark	Grounded tactic suggestion via premise retrieval
Baldur (First et al. 2023)	2023	Whole-proof generation	miniF2F	End-to-end proof generation + repair
LEGO-Prover (Xin et al. 2024b)	2024	Modular growing libraries	miniF2F	Reusable lemma construction
DeepSeek-Prover (Xin et al. 2024a)	2024	RL from proof feedback	miniF2F	RLPAF — reinforcement learning from proof assistant feedback
AlphaGeometry (Trinh et al. 2024)	2024	Neuro-symbolic	IMO Geometry	Synthetic data + symbolic deduction
AlphaProof (AlphaProof team, Google DeepMind 2024)	2024	Gemini + AlphaZero	IMO 2024	Silver-medal level problem solving
Lean Copilot (Song, Yang, and Anandkumar 2025)	2025	Editor integration	N/A	Real-time tactic suggestion in VSCode
DeepSeek-Prover-V2 (Xin et al. 2025)	2025	RL + subgoal decomposition	miniF2F, ProofNet	Reinforcement learning for structured proof planning

These systems largely begin with already formalized targets and concentrate on proof search. Translating a physical theory adds an earlier modeling burden: choosing definitions, types, scope, and assumptions for concepts such as Markov blankets, solenoidal flows, and Expected Free Energy decompositions. The maintained kernel in this work emphasizes that translation and review problem. The optional Hermes path can explain or refine candidate Lean, but it supplies no correctness or authorship claim without subsequent compilation and a claim-ready full receipt.

3.3.1 The Axiomatization vs. Problem-Solving Distinction

The distinction between proof search and theory axiomatization is worth making precise:

- **Proof search** (LeanDojo, DeepSeek-Prover): given a well-typed theorem statement, find a proof term. The statement already exists in a formal language; the challenge is navigating the proof space.

- **Theory axiomatization** (this work): given an informal mathematical theory (published in journals, written in natural language with embedded LaTeX), produce well-typed theorem statements, definitions, and structural lemmas. The challenge is *translation from informal to formal*, not proof search.

This distinction explains why the pipeline reports statement design, proof completion, and semantic adequacy separately. Every shipped body is `sorry`-free at its reviewed scope, but proving a narrowed proposition does not validate the modeling choice that produced it. The contribution is therefore both executable mathematics and an auditable translation process, rather than a benchmark whose only outcome is proof-completion rate.

3.4 The FEP Debate and the Case for Formalization

The mathematical status of the FEP has been actively debated in the literature, along three principal lines of critique:

1. **Blanket conditions.** The partition of states into internal, external, sensory, and active components is not always well-defined for arbitrary dynamical systems ((Biehl, Pollock, and Kanai 2021)). Specifically, Biehl et al. demonstrate that *"various definitions of the 'Markov blanket' proposed in different works are not equivalent"* and that crucial vector-field rewritings are not generally correct absent previously unstated assumptions. The canonical Markov-blanket claim is that the state space admits a factorization $X = \Psi \times B \times H$ (external, blanket, internal) such that the blanket $b = (s, a)$ renders external and internal states conditionally independent,

$$p(\psi, \eta \mid b) \;=\; p(\psi \mid b)\, p(\eta \mid b) \qquad \text{(conditional independence given the blanket).} \tag{17}$$

   Eq.~17 is a *statistical* condition about a specific family of joint densities, and whether a given dynamical system satisfies it depends on the system's drift, diffusion, and steady-state structure—properties that are not determined by the choice of state-space partition alone. The blanket partition sits at the intersection of two very different objects: an algebraic/set-theoretic decomposition of the state space, and a probabilistic conditional-independence statement about its dynamics. Conflating the two is precisely the locus of Biehl et al.'s critique. Accordingly, **fep-005** formalizes only the algebraic side: a four-part disjoint cover with unique membership. The maintained `FEP.MarkovBlanket` foundation adds a separate normalized finite joint of the form $P(b)P(i \mid b)P(e \mid b)$, proves positive-mass conditional factorization and zero internal–external mutual information, and constructs a nontrivial transition whose allowed dependencies are encoded in its types. These results settle a concrete finite instance; they neither derive the blanket from fep-005's arbitrary labels nor prove that generic stochastic dynamics admit or preserve such a structure.

2. **Particular partitions.** The broader applicability of the "particular physics" framework has been questioned, with arguments that certain assumptions about steady-state densities are unduly restrictive ((Aguilera et al. 2022)). Concretely, the "particular-physics" construction assumes that a stochastic system with dynamics $\mathrm{d}x = f(x)\mathrm{d}t + \sigma\,\mathrm{d}W_t$ admits a non-equilibrium steady-state (NESS) density $p^*(x)$ such that the probability current decomposes as

$$J(x) \;=\; -\big(D + Q(x)\big)\,\nabla F(x), \qquad F(x) := -\log p^*(x), \tag{18}$$

   with D symmetric positive-semidefinite (the dissipative component), $Q(x)$ **antisymmetric**, i.e. $Q(x)^\top = -Q(x)$ (the solenoidal component), and the steady-state divergence condition $\nabla \cdot J(x) = 0$. Establishing these conditions simultaneously requires both algebraic hypotheses and analytic/stochastic results about the density and current. **fep-025** now proves a finite continuity-equation instance instead: forward-minus-reverse edge current is antisymmetric and globally conserved, a normalized transition with a stationary mass vector has zero node divergence, and a directed three-cycle has nonzero current despite stationarity. This separates finite stationarity from detailed balance, but it does not construct $Q(x)$, a stationary density on a continuous space, or the SDE/PDE decomposition above.

3. **Math and territorialism** (a deliberate nod to the classic "map and territory" distinction and to the "territorialism" of notational regionalisms across fields). It has been argued that FEP derivations sometimes conflate distinct mathematical objects—using the same notation for different quantities in different contexts ((Andrews 2021)). Lean's type system makes many such conflations visible: for example, a `Measure ℝ` cannot be passed where an $\mathbb{R}$-valued function is required without an explicit bridge. It cannot prevent a modeler from choosing an inadequate definition or assigning the same underlying type to semantically different quantities.

These debates motivate the present work: formal verification in Lean 4 does not adjudicate the underlying semantic disagreements. It makes statement-level hypotheses and proof dependencies inspectable, while the separate semantic audit records whether those formal objects reach the motivating claim.

3.5 Recent Developments (2024–2026)

Recent FEP work includes path-integral and geometric treatments of Bayesian mechanics (Sakthivadivel 2023; K. Friston et al. 2023) and a comparison and unification of Expected Free Energy formulations (Champion et al. 2026). Translating such claims into Lean requires more than proof search: the modeler must choose probability spaces, codomains, finiteness conditions, stochastic-process interfaces, and a semantic acceptance target. This catalogue makes those choices reviewable one row at a time. Compiler acceptance answers derivability of the stated proposition; the disposition and assumption review answer the separate question of how far that proposition reaches toward its topic label.

4 Methodology and System Architecture

The project is organized around one methodological rule: a claim is no stronger than the evidence object that supports it. Catalogue generation, native Lean compilation, and a live Hermes/OpenGauss run are therefore separate modes with separate acceptance predicates. A deterministic catalogue build can establish source consistency; a native receipt can establish warning-free, `sorry`-free typechecking against a pinned toolchain; only a validated full-run receipt can support claims about model output, provider latency, or persisted sessions.

Readers new to Lean should begin with §4.16. The detailed components in §4.17–§4.21 cover the library substrate, semantic maturity, optional LLM stage, native compiler bridge, and execution pipeline.

4.1 System Architecture Overview

The architecture has three source layers and three evidence layers.

Layer	Canonical input	Derived output	What it can establish
Authorship	roster/family metadata, maturity and novelty review, authored relations, family-owned topic bodies, and manifested formal resources	checkout/wheel `topics.yaml`; topic aggregate; foundation, composition-leaf, and import-aggregate Lean; coverage, atlas, and numerical dashboard	roster, source parity, declared semantic scope, maintained dependency structure, and witnessed composition
Native verification	exact topic roster plus generated catalogue/formal modules and pinned Lake workspace	native topic receipt plus formalism declaration/axiom receipt	compiler exit status, warnings, `sorry`/`sorryAx`, declaration resolution, toolchain and source digests
Full pipeline	live credentials, Hermes result, OpenGauss session state, native verification	hashed report bundle and independently validated full receipt	only the observed LLM/session/run claims

`FEPPipeline` itself has four stages: Load Catalogue, Environment Validation, Gauss Sessions, and Manuscript Artifacts. In `catalogue` mode the Gauss stage is explicitly `not_run`. In `full` mode all selected topics must produce successful Hermes-backed results and clean Lean verification before the run is complete. `Reporter` executes only after a complete pipeline result; incomplete runs do not receive a successful report directory.

4.2 The Command-Line Toolchain

The installed `fep-lean` command is the public interface:

```
uv run fep-lean catalogue
uv run fep-lean atlas --check
uv run fep-lean verify --fail-on-warnings --receipt output/native-verification.json
uv run python scripts/audit_formalisms.py --receipt output/formalism-audit.json
uv run fep-lean preflight
uv run fep-lean run
```

The first two commands are deterministic and offline: catalogue mode builds publication inputs, while the atlas command checks the coverage visualization. The native compiler command writes an atomic, source-bound topic receipt that records the actual compiler output and resolved Mathlib revision; the formalism audit separately resolves primary/evidence declarations, applies a versioned trusted-axiom policy, and inspects evidence axioms. `preflight` is a read-only capability check. `run` is the credentialed Hermes/OpenGauss path and fails closed when required capabilities are absent. Topic and area filters are useful during development, but a filtered native receipt is never publication-claim-ready for the full catalogue.

Maintenance scripts expose non-mutating drift checks:

```
uv run python scripts/_maint_build_topics_catalogue.py --check
uv run python scripts/_maint_build_fep_all_lean.py --check
uv run python scripts/_maint_build_formal_modules.py --check
uv run python scripts/build_formalism_coverage.py --check
uv run fep-lean atlas --check
uv run python scripts/build_formal_kernel_dashboard.py --check
uv run python scripts/theorem_maturity_audit.py
uv run python docs/theorem_ref_audit.py
```

4.3 The Hermes Agent and LLM Integration

`fep_lean.llm.hermes.HermesExplainer` is an optional reviewer of a curated theorem body, not the owner of the formal kernel. It sends a system/user prompt pair to an OpenAI-compatible endpoint, parses the explanation and refined Lean block, and records retries and model-chain advances. The prompt requires preservation of imports, namespaces, tactic hints, and the absence of `sorry` in an already complete sketch.

Provider/model rosters are runtime configuration rather than scientific results. They are authoritative only in a validated report bundle. With no OpenRouter or Anthropic credential, full-mode preflight fails; an offline fixture result may exercise code paths in tests, but it is ineligible for publication claims.

See §4.19 for the session protocol and the three distinct fallback mechanisms.

4.4 The Native Lean Compilation Engine

`fep_lean.verification.lean_verifier.LeanVerifier` writes each selected body to a temporary file in the pinned Lake workspace and invokes `lake env lean`. Leading imports are preserved; a shared preamble is used only for a sketch that has no explicit imports. Results retain compiler output, separated warnings and errors, `sorry` detection, duration, toolchain version, and an advisory failure class.

Native claim readiness is stricter than an exit code. The receipt must cover the exact ordered roster of 155 topics, every row must compile, warning and `sorry` counts must both be zero, and the catalogue, canonical family-body sources, Lean toolchain, and Mathlib tag must match the live tree. Revalidation independently recomputes those conditions rather than trusting a stored `complete` flag.

The complementary formalism audit imports `FepSketches.composed`, checks every reviewed primary plus semantic/formal witness, and runs `#print axioms` over the evidence roster. Projection drift, unresolved names, warnings, timeouts, compiler failures, or `sorryAx` make that receipt incomplete. Standard logical axioms reported by Mathlib are retained rather than hidden.

4.5 OpenGauss Workflow and State Integration

OpenGauss here refers to the Math, Inc. Lean-workflow tool, not the similarly named database product. `GaussRunner` persists project sessions in SQLite, records the system/user/assistant turns, stores the refined sketch and structured compiler artifact, and closes each session with explicit Hermes and Lean outcomes. Raw compiler diagnostics remain in the artifact rather than being presented as model dialogue.

The state store improves auditability but does not confer correctness. Full-report validation checks the selected topic roster, verification source, summary/manifest agreement, current source and configuration digests, and hashes of required artifacts. A stale, partial, catalogue-only, or fixture-backed report cannot supply manuscript metrics.

4.6 The Unified Execution Pipeline

The three supported execution contracts are intentionally non-interchangeable:

1. `catalogue`: validates sources and writes deterministic figures/manuscript data; it performs no verification.
2. `verify`: runs the native compiler without an LLM or session store and may emit a native receipt.
3. `run`: requires live credentials and state/toolchain capabilities, runs Hermes and native verification per topic, and emits a full report only on strict completion.

This separation removes the former ambiguity in which a catalogue-only `complete` flag could be read as empirical pipeline evidence. The manuscript renderer selects a current, independently validated native receipt for compile claims and a claim-ready full report for Hermes/OpenGauss claims; otherwise it emits explicit unavailable/false values.

4.7 Standard Reproducibility Workflow

For native formal results:

```
uv sync --locked --extra dev
uv run python scripts/_maint_build_topics_catalogue.py --check
uv run python scripts/_maint_build_fep_all_lean.py --check
uv run python scripts/_maint_build_formal_modules.py --check
uv run python scripts/build_formalism_coverage.py --check
uv run fep-lean atlas --check
uv run python scripts/build_formal_kernel_dashboard.py --check
uv run python scripts/audit_formalisms.py --receipt output/formalism-audit.json
uv run fep-lean verify --fail-on-warnings --receipt output/native-verification.json
uv run fep-lean catalogue
uv run python scripts/render_manuscript.py --check
```

For a full live run, configure an accepted credential, run `uv run fep-lean preflight`, then `uv run fep-lean run`. The resulting report must pass the receipt validator before its Hermes or timing fields can enter manuscript variables. Neither path mutates authored manuscript Markdown: rendering writes to `output/manuscript/` only after validating every placeholder across the full source set.

4.8 Area-Specific Methodological Constraints

The five areas are organizational lenses, not five independent foundations. Each row declares its actual imports, primary theorem, assumptions, non-vacuity argument, acceptance probe, and semantic disposition in generated coverage. The following summaries state the present boundary rather than an aspirational one.

4.8.1 FEP Methodology (41 topics)

The FEP area combines measurable variational integrands, native extended-real KL divergence, exact scalar Laplace kernels, finite quadratic optimization dynamics, variational duality, generalized predictive-coding corrections, and finite learning/model-evidence bounds. fep-001 and fep-002 prove KL-remainder variational bounds; fep-015 supplies the measurable integrand contract; fep-016 exposes the exact Gaussian normalizer; and fep-032 compiles the closed-form gradient-descent iterate and its stable-step convergence. Later families add Gibbs duality, mean-field and importance-weighted bounds, precision-weighted prediction-error dynamics, PAC-Bayes, posterior-concentration, mixture-regret, Bayes-factor laws, and finite-law Laplace squared/Brier-risk transfer on explicit carriers. The reusable finite active-inference model additionally derives posterior-form VFE, its evidence lower bound, exact-posterior attainment, and uniqueness even for posteriors with zero-mass states. No theorem identifies all of these finite and measure-valued conventions without the bridges stated in their composition leaves.

4.8.2 Active Inference Methodology (31 topics)

Active-Inference rows combine discounted pragmatic cost, an explicit `ENNReal` EFE sign convention, policy-indexed reachable laws, finite-horizon Bellman and desirability recursions, native measure-KL information gain, normalized sum-product messages, support-aware softmax policies, and collective product-agent and consensus laws. The maintained kernel proves pragmatic-minus-epistemic and risk-plus-ambiguity decompositions under explicit support, normalized prior-weighted policy selection, action pushforward, chronological state rollout, recursively accumulated stage-dependent EFE, and a concrete two-stage Boolean witness in which feedback strictly improves on open-loop control. The later policy-tree foundation proves arbitrary finite-depth observation-contingent recursion on finite carriers, Bellman optimizer existence, open-loop embedding and dominance, treewise EFE decomposition, and the same strict Boolean witness. No result claims infinite-horizon or continuous-belief optimality, equivalence with every EFE formulation, or emergent agency for a collective.

4.8.3 Information Geometry Methodology (21 topics)

This area uses native `InformationTheory.klDiv`, including self-zero, finite-measure separation, and the composition-product chain rule. It contains a finite positive-definite weighted Fisher metric, the complete one-dimensional Bernoulli family, and a categorical score model with explicit full-rank and null-direction witnesses. The maintained geometry modules prove Fisher Gram PSD and conditional PD on simplex tangents, pullback laws, scalar Cramér–Rao under unbiasedness and score regularity, natural-gradient equivariance under an invertible chart, a mirror-descent three-point identity, an affine-projection Bregman Pythagorean law, and replicator–natural-gradient equivalence. The scalar exponential-family foundation adds normalization/full support, log-partition gradient and Hessian, centered score, Fisher–variance equality, KL–Bregman duality, and interval-local mean-coordinate injection. These sources do not define arbitrary smooth manifolds, affine connections, curvature, or general geodesic existence.

4.8.4 Bayesian Mechanics Methodology (41 topics)

These rows include generic conditional-independence laws, reversible-kernel invariance, measure-theoretic and finite posterior reconstruction, transition–observation filtering and smoothing, hierarchical and posterior-predictive kernel composition, Bernoulli sufficient statistics and conjugate closure, finite causal interventions, and a nonzero stationary probability current. The maintained blanket and causal modules connect explicit finite carriers to exact conditional factorization, zero conditional mutual information, typed blanket-respecting dynamics, intervention normalization, descendant change, and named non-descendant invariance witnesses. The finite-to-native family further proves weighted-Dirac law/kernel embeddings, expectation and prediction transfer, native `CondIndepFun`, measurable endpoint coarsening, and rowwise factorized-transition closure. These sources do not prove blanket existence, mixture-level preservation, or causal identification for arbitrary systems, nor derive continuous Fokker–Planck dynamics.

4.8.5 Thermodynamics Methodology (21 topics)

Thermodynamic rows cover the equilibrium Helmholtz derivative identity, finite conserved currents, binary entropy optimality, normalized Gibbs weights, Gaussian thermal entropy response and heat capacity, diagonal and edgewise entropy production, and a logical-erasure derivation of Landauer heat/work bounds from explicit second-law premises. The path-space family additionally normalizes forward and reverse finite path laws, proves reversal and fluctuation identities, states a finite Jarzynski equality with explicit work, inverse temperature, and normalization, and links reversible one-step dynamics to KL dissipation. The exact Boolean continuous-time family adds stochastic kernels, the Chapman–Kolmogorov semigroup, both master equations, stationarity, detailed balance, exponential relaxation, and quadratic Lyapunov decay for positive

two-state rates. These finite laws do not constitute general constrained equilibrium statistical mechanics, SDE/PDE theory, or a microscopic thermodynamic derivation of the FEP.

4.9 Catalogue Authorship Pipeline

The source graph is explicit:

```
config/catalogue_metadata.yaml             stable titles/areas/library hints
config/theorem_maturity.yaml               primary theorem + semantic review
config/formalism_novelty.yaml              expansion deltas + required bridges
config/formalism_relations.yaml            authored relations + capability history
src/fep_lean/catalogue/bodies/*.py          family-owned canonical Lean bodies
src/fep_lean/catalogue/registry.py          validated body manifest and merger
src/fep_lean/catalogue/latex.py             theorem-signature projection
src/fep_lean/formal/**/*.lean              manifested foundations + composition leaves
                     \ | /
                      v
src/fep_lean/catalogue/generation.py
        |                          |
        v                          v
config/topics.yaml       src/fep_lean/data/topics.yaml
        |
        +--> lean/FepSketches/fep_all.lean
        +--> lean/FepSketches/{foundations,compositions,aggregate}.lean
        +--> docs/formalism-coverage.{md,json}
        +--> docs/formalism-atlas.{svg,html}
        +--> docs/formal-kernel-dashboard.{svg,html}
```

The braces in this schematic mean the explicitly manifested formal-module set, not a literal filename or an inferred directory glob. `scripts/_maint_build_formal_modules.py` owns the exact source-to-workspace projection. The atlas displays authored scientific relations and module-import dependencies as separate edge classes; the dashboard evaluates deterministic numerical witnesses but supplies no proof evidence.

Both YAML copies are byte-identical generated projections: one supports checkout workflows and one ships in the wheel. Strict loaders reject missing/extra rows, order drift, unknown semantic values, theorem/signature count mismatches, and a primary theorem absent from the canonical body. The generator restores its own provenance header, so formatting churn cannot silently create a second authoring source.

4.10 Verification Workflow and Cache Strategy

The project pins `leanprover/lean4:v4.33.1` and Mathlib `v4.33.1`. `lake exe cache get` may populate dependency `.olean` files; no validation command implicitly downloads or rebuilds them. Verification checks the existing workspace and returns a structured failure when the toolchain or cache is absent.

`verify_batch` is deliberately serialized (`max_workers=1`) to avoid races over temporary files and build artifacts. Warm-cache duration is environmental and is therefore reported only from a receipt, never as a fixed value in prose. The aggregate/composed targets provide complementary whole-library checks that catch namespace collisions, broken seams, and warning-producing declarations. The declaration/axiom probe then confirms that names cited as evidence resolve through those exact modules.

4.11 execution-integrity Testing Policy

Tests pin observable contracts rather than publication constants. Catalogue tests exercise strict schema and source parity; native tests invoke the real Lean subprocess where marked; SQLite tests use isolated real databases; HTTP behavior is tested through controlled local servers or is guarded by explicit live credentials. Distribution tests build a wheel, install it into an isolated environment, import `fep_lean`, execute CLI help, load packaged data, and confirm that obsolete top-level names such as `catalogue` are not exported.

External services remain external evidence. Offline fixture responses are useful for parser and failure-path tests, but cannot make a full report claim-ready.

4.12 Namespace Convention and Topic Isolation

Python code lives under one `fep_lean` namespace. Lean declarations use an inner stable topic namespace (`FEP002`, `FEP014`, and so on); the generated aggregate adds an outer `fep_fepNNN` namespace so helper declarations from different self-contained sketches cannot collide. The coverage audit counts declarations and imports from canonical bodies, while theorem-reference auditing ensures that manuscript identifiers resolve to actual declarations.

4.13 PYTHONPATH Isolation

The installable package no longer relies on path precedence among generic top-level names such as `catalogue`, `pipeline`, `llm`, or `output`. Checkout commands run through `uv`, and installed commands import `fep_lean.*`. The wheel includes its generated catalogue as package data, so loading the default catalogue does not depend on locating the repository root.

4.14 Parallelism Model

Formal compilation is serialized. In full mode, optional single-worker prefetch may overlap the next Hermes request with current Lean verification, but it does not reorder persisted topic results or permit multiple concurrent compiler processes. Receipt topic order remains the requested catalogue order, which makes completeness and digest validation deterministic.

4.15 Detailed Methodology Sub-Sections

The following sub-sections give the necessary depth without changing these contracts: the Lean primer (§4.16), pinned Mathlib surface (§4.17), proof/semantic maturity (§4.18), optional Hermes stage (§4.19), native receipt (§4.20), and full pipeline (§4.21).

4.16 Lean 4: A Primer for Active Inference Researchers

4.16.1 Why Formal Verification Matters for the FEP

The Free Energy Principle (K. Friston 2010; Parr, Pezzulo, and Friston 2022) rests on deep intersections of measure theory, stochastic calculus, differential geometry, and information theory. Informal mathematical proofs in this space — written in natural language with $\forall$ and $\exists$ symbols — are powerful but can harbor subtle errors: interchanging limits and expectations without justification, conflating almost-sure and sure convergence, or silently assuming absolute continuity of measures. When the FEP community claims that "variational free energy upper-bounds surprise," every step of the derivation must be airtight — yet in practice verification depends on peer review by a small number of domain experts.

Lean 4 is an interactive theorem prover (ITP) that makes this bottleneck explicit. Every elaborated inference step is checked by Lean's kernel against declared axioms and definitions. Acceptance therefore certifies the exact formal statement under the recorded toolchain and trust base; it does not by itself certify that the statement faithfully represents the intended scientific claim. That second obligation is why this project records a separate semantic disposition and assumption review.

For Active Inference, this yields:

- **Verifiable variational bounds.** The claim $F \geq -\log p(s|m)$ is checked all the way down to the axiom level.
- **Dimension safety.** Type-checking prevents integrating over the wrong measure space or mixing distributions over incompatible state spaces.
- **Compositional scaling.** Proven lemmas compose into larger theorems without re-verification.
- **Reproducibility.** A Lean proof file is itself a reproducible artifact, unlike a journal PDF.

Throughout this primer — and the full 155-topic catalogue — the pinned toolchain is **Lean 4 `leanprover/lean4:v4.33.1`** with **Mathlib4 `v4.33.1`**. Every lemma name, module path, and tactic behavior cited resolves against that exact pin; version drift is prevented by the `lean/lean-toolchain` and `lean/lakefile.lean` files in the repository.

4.16.2 Propositions as Types: The Curry-Howard Correspondence

Lean 4 is built on a deep correspondence between logic and computation called the **Curry-Howard isomorphism**. The slogan — **propositions as types, proofs as programs** — captures the whole design: every mathematical proposition is reified as a type, every proof of that proposition is reified as a term (a program) inhabiting that type, and the compiler's type-checker *is* the proof-checker. Verifying "theorem T is correct" reduces mechanically to verifying "the program `t` has type `T`".

Logic	Type Theory (Lean 4)
Proposition P	Type `P`
Proof of P	Term `p : P` (a program of type `P`)
$P \implies Q$	Function type `P → Q` (a program that transforms proofs)
$\forall x, P(x)$	Dependent function `(x : α) → P x`
$P \wedge Q$	Product type `P × Q`
$P \vee Q$	Disjunction `Or P Q` (proof-irrelevant; `P ⊕ Q` is the data-level analogue)
Modus ponens	Function application `f a : Q` given `f : P → Q`, `a : P`
$\bot$ (false)	Empty type `False`

In Lean 4, proving a theorem is equivalent to constructing a program whose type is the proposition being proved. If the program type-checks, the theorem is proven. This is fundamentally different from computer algebra systems (Mathematica, SymPy) which *compute* with symbols but cannot *prove* that a result holds for all inputs universally. The Curry-Howard lens is also what justifies the FEP verifier treating `lake env lean` exit code 0 as *proof*: successful type-checking of the proof term is, by construction, a checked proof of the stated theorem.

4.16.3 Universe Polymorphism and Dependent Types

Standard type systems distinguish `Int`, `Float`, `String`. Lean 4 supports **dependent types** where types can depend on *values*:

```
-- A vector of exactly n elements
def Vector (α : Type) (n : Nat) : Type := ...

-- A probability measure on a measurable space
structure ProbMeasure (α : Type) [MeasurableSpace α] where
  μ : Measure α
  total : μ Set.univ = 1
```

For Active Inference, dependent types naturally encode:

- **Parameterized distributions.** `Distrib (S ω)` — distributions indexed by world-states.
- **Transition kernels.** `(s : State) → Measure (Action s)` — action distributions whose type depends on the current state.
- **Finite policy spaces.** `Fin n → Action` — a policy over exactly n time steps.

Lean also leans heavily on **universe polymorphism** (`Type u`, `Type v`). In measure theory this prevents Russell's paradox by ensuring that the collection of all measurable spaces lives strictly above any individual space. FEP researchers encounter this most often as `{α : Type*}` in theorem signatures; the `*` simply means the type can live in any universe.

4.16.4 Tactics: How Proofs Are Constructed

Lean 4 proofs are written using **tactics**—commands that transform proof goals step by step. Key tactics used in FEP formalizations:

Tactic	What it does	FEP usage
`exact h`	Close goal exactly with term `h`	Apply `measure_union_le`, `measure_mono` directly
`rw [h]`	Rewrite goal using equation `h`	Substitute `IsProbabilityMeasure.measure_univ`
`apply f`	Apply function/lemma `f`, leaving subgoals	Apply `Real.exp_le_exp.mpr` for Gibbs monotonicity
`simp [h]`	Simplify using `h` and simp lemmas	Reduce `Finset.sum_div`, `Finset.card_range`
`intro x`	Introduce ∀/implication hypothesis into context	Start proofs of `∀ x, P x` or `P → Q` goals
`constructor`	Split a goal into its structural pieces	Build `And`/`Iff`/structure goals (e.g., softmax normalization ∧ non-negativity)
`linarith`	Linear arithmetic over ordered fields	Derive bounds from KL ≥ 0 by rearranging
`nlinarith [h₁, h₂]`	Nonlinear arithmetic with hints	Prove quadratic contraction in gradient flow
`positivity`	Prove `0 ≤ e` or `0 < e` automatically	Non-negativity of sum of squares, `Real.sqrt`
`ring`	Prove equalities in commutative rings	Algebraic identities in free energy decompositions
`norm_num`	Evaluate numeric expressions	Verify `(2 : ℝ) > 0` or specific constants
`have h : P := ...`	Introduce intermediate lemma `h : P`	Build step-by-step proofs for complex bounds
`calc`	Chain transitivity steps	$a \leq b \leq c$ derivations in energy bounds
`sorry`	Admit goal without proof	Mark aspirational proof steps (compile flag)

The catalogue's 155 Lean bodies collectively exercise the major tactic families enumerated above. `exact`, `simp`, `rw`, `linarith`, `positivity`, and `have` dominate by frequency, while `nlinarith`, `ring`, `norm_num`, `intro`, `constructor`, and `calc` appear in specific topic families — for example, `calc` anchors fep-021's energy-bound derivation, and `constructor` structures fep-028's softmax lemma. The exhaustiveness claim is intentionally conservative: a small handful of niche tactics (e.g. `omega` and `decide`) are used opportunistically rather than uniformly.

How `lake env lean` verification works in the pipeline.

`LeanVerifier` writes each sketch to a temporary file under `lean/FepSketches/`. Before the subprocess call, `_wrap_lean_code` prepends `import Mathlib`, adds the standard `open MeasureTheory` line plus any area-specific opens, and wraps the body in a `namespace FEP<NNN> ... end FEP<NNN>` block, where `<NNN>` is the topic's three-digit identifier. It then invokes:

```
lake env lean lean/FepSketches/FepCheck_fep001.lean
```

This executes inside the Lake build environment rooted at `lean/`, which provides access to the pre-compiled Mathlib4 `v4.33.1` `.olean` files. Exit code 0 signals compilation success. The verifier captures `stdout` and `stderr`, sets `compiles` to `True` or `False`, detects a `sorry` tactic in the source text, and records the resulting `VerifyResult` in SQLite. With a warm Mathlib4 cache, each verification takes about 1–2 seconds.

> **Namespace isolation.** The `namespace FEP<NNN> ... end FEP<NNN>` wrapper prevents theorem-name collisions during the 155-topic aggregate compilation: when sketches are concatenated into `fep_all.lean` for the batch build, identically named helpers (for example two topics both declaring `aux_lemma`) live in disjoint namespaces and never clash. Every row in `config/topics.yaml` follows this pattern.

Mathlib4 module map for FEP topics:

Topic Area	Key Mathlib4 Modules
FEP / measure theory	`MeasureTheory.Measure.MeasureSpace`, `MeasureTheory.Measure.NullMeasurable`
Probability	`Probability.Notation`, `MeasureTheory.Measure.ProbabilityMeasure`
Special functions	`Analysis.SpecialFunctions.Log.Basic`, `Analysis.SpecialFunctions.Exp`
Linear algebra	`LinearAlgebra.Matrix.Transpose`, `Analysis.InnerProductSpace.Basic`
Finite sums	`Algebra.BigOperators.Group.Finset`, `Data.Finset.Basic`
Metric spaces	`Topology.MetricSpace.Basic`, `Topology.MetricSpace.PseudoMetric`
Real arithmetic	`Analysis.SpecialFunctions.Pow.Real`, `Mathlib.Tactic`

4.16.5 From Informal Bound to Lean Statement: A Minimal Walk-Through

Before the full ELBO, consider the simplest FEP-flavoured informal claim and its literal translation into Lean.

Informal (textbook):

> *Claim.* For any two events A, B in the state space of an agent, the probability that the world is in $A \cup B$ is at most the sum of the individual probabilities: $\mu(A \cup B) \leq \mu(A) + \mu(B)$. *(This is the countable-subadditivity bound that underwrites the union step in most FEP surprise-minimization arguments.)*

Formalization, step by step:

1. *Name the space.* Informal "state space" becomes `{α : Type*} [MeasurableSpace α]` — a type equipped with a σ-algebra.
2. *Name the measure.* Informal "probability" becomes `(μ : Measure α)` — a Mathlib4 `Measure` over that space.
3. *Name the events.* Informal "events A, B" become `(s t : Set α)`.
4. *State the inequality.* `μ (s ∪ t) ≤ μ s + μ t`.
5. *Discharge the proof.* Invoke the Mathlib4 lemma `measure_union_le` via the `exact` tactic.

Formal (Lean 4, Mathlib4 `v4.33.1`, catalogue row fep-001):

```
import Mathlib

open MeasureTheory

namespace FEP001

theorem fep001_union_bound {α : Type*} [MeasurableSpace α]
    (μ : Measure α) (s t : Set α) :
    μ (s ∪ t) ≤ μ s + μ t := by
  exact measure_union_le s t

end FEP001
```

This three-line proof is a real (sorry-free) catalogue row: every implicit assumption of the informal claim has been made explicit (the type, its σ-algebra, and the two sets), and the inequality is discharged by a single pre-verified Mathlib4 lemma. The `FEP001` namespace prevents name collisions in the 155-topic aggregate build, and `open MeasureTheory` brings `Measure` and `measure_union_le` into scope without fully-qualified names. This is the template every catalogue body follows.

4.16.6 Concrete Example: Informal vs Formal ELBO

Informal (journal paper):

> *Theorem*. The Evidence Lower Bound maximizes model evidence: $\log p(s) \geq -F[q, p]$. *Proof*. By definition, $F = \mathrm{KL}[q \| p(\psi|s)] - \log p(s)$. Since KL divergence is non-negative, the result follows immediately by rearranging terms. ∎

Formal kernel used by fep-002 (Lean 4 with Mathlib4 `v4.33.1`):

```
import Mathlib.InformationTheory.KullbackLeibler.Basic

namespace FEP002
variable {α : Type*} [MeasurableSpace α]
open MeasureTheory
open scoped ENNReal

noncomputable def fep002_variationalFreeEnergy
    (q posterior : Measure α) (surprisal : ENNReal) : ENNReal :=
  surprisal + InformationTheory.klDiv q posterior

theorem fep002_vfe_ge_surprisal
    (q posterior : Measure α) (surprisal : ENNReal) :
    surprisal ≤ fep002_variationalFreeEnergy q posterior surprisal := by
  simp only [fep002_variationalFreeEnergy]
  exact le_add_right (le_refl surprisal)
end FEP002
```

The topic-row version fixes the measurable space, both measures, the order of KL arguments, and the nonnegative extended-real codomain. Its inequality is deliberately narrower than the informal theorem: it treats surprisal and posterior as inputs rather than constructing them from a joint model. The separate finite `GenerativeModel` foundation closes that bridge for one normalized finite carrier by constructing evidence and posterior, proving Bayes reconstruction, and deriving posterior-form VFE and its evidence bound. Keeping the two declarations separate makes the difference between a native measure-level remainder theorem and a finite model theorem visible in their types.

Catalogue note. The 155 committed bodies in the family modules under `src/fep_lean/catalogue/bodies/` carry targeted `import Mathlib.…` lines. The validated registry fixes their order and source identity. `LeanVerifier._wrap_lean_code` preserves a body with leading imports and supplies the shared preamble only when imports are absent (§4.20; Appendix B).

4.16.7 Reading Type Error Messages

When translating FEP physics into Lean, compilation errors typically fall into three categories:

1. **Type mismatch.** `application type mismatch: expected 'Measure R', got 'R'` — raised when passing a scalar prediction error into a function expecting a full belief distribution.
2. **Missing instance.** `failed to synthesize instance 'MeasurableSpace α'` — Lean refuses to integrate over a space that has not been declared measurable.
3. **Unsolved goal.** `unsolved goals: ⊢ q ≪ p` — the theorem requires absolute continuity, but no proof or hypothesis supplies it.

`LeanVerifier.classify_failure_kind` maps these patterns (plus `missing_import`, `renamed_identifier`, and `timeout`) onto the `FailureKind` enum carried inside every `VerifyResult`. These errors are not bugs in the pipeline; they are the compiler enforcing mathematical rigor.

4.17 Mathlib4 and Measure-Theoretic Probability

4.17.1 What Is Mathlib4? A Short Orientation

Mathlib4 (The mathlib Community 2020) is Lean 4's community mathematical library. This project treats the pinned source as the authority on available definitions and theorem signatures. Lean `leanprover/lean4:v4.33.1` and Mathlib `v4.33.1` are fixed by the Lake workspace so a later upstream addition cannot be cited as though it existed at the publication pin.

4.17.2 Core Measure Theory and Stochastic Foundations

The relevant substrate includes measurable spaces, extended-nonnegative-real-valued measures, probability-measure type classes, Radon–Nikodym derivatives, kernels, integrals, and information-theoretic KL divergence. Preconditions are part of theorem types: finiteness, sigma-finiteness, measurability, and Markov-kernel instances cannot be elided by prose.

At the pin, KL divergence is the native `InformationTheory.klDiv : Measure α → Measure α → ℝ≥0∞`. The extended codomain preserves infinite divergence. A conversion through `.toReal` would map infinity to zero unless finiteness is separately established, so the strengthened catalogue statements stay in `ℝ≥0∞`.

4.17.3 Key Mathlib4 Lemmas Referenced by the Catalogue

The generated coverage report derives the exact import surface. Representative direct dependencies include measure monotonicity and subadditivity, probability mass of the universe, finite-sum order laws, real log/exp facts, metric inequalities, matrix transpose laws, and the following KL declarations:

- `InformationTheory.klDiv_self`;
- `InformationTheory.klDiv_eq_zero_iff` for finite measures;
- `InformationTheory.klDiv_compProd_eq_add` for finite measures and Markov kernels.

The catalogue wraps these declarations in topic-local theorems rather than copying their proofs.

The project-local finite information layer goes further at its explicitly finite, full-support scope. It proves conditional entropy and conditional KL definitions, a joint/channel KL chain rule, the resulting prior-marginal KL bound, independent-product additivity, and the entropy identity and cardinality bound for mutual information. These are finite-sum theorems over the shared generative carrier; they are not presented as generic measure-theoretic data-processing or disintegration results.

4.17.4 Coverage Map and Dependency Graph

`docs/formalism-coverage.md` reports the area/disposition matrix, per-topic theorem and definition counts, exact imports, and open semantic obligations. Shared imports are library-incidence edges, not proof dependencies among the independently namespaced topics.

4.17.4.1 Coverage by FEP Area FEP and Bayesian rows use the measure/probability stack; Active Inference relies heavily on finite sums and order; Information Geometry combines native KL with elementary inner-product and metric facts; Thermodynamics uses real log/exp and ordered algebra. The semantic audit shows where those library facts remain only substrates for the advertised domain concept.

4.17.4.2 FEP and Active Inference (41 + 31 topics across areas) Finite-policy existence, nonnegative aggregation, and basic probability normalization are well supported. The maintained finite carrier now supplies a common probabilistic model, mutual-information epistemic value, both central expected-free-energy decompositions, and support-explicit nonnegativity. The remaining scope boundary is equivalence to alternative measure-valued or generalized-coordinate EFE formulations, not absence of a shared finite model.

4.17.4.3 Geometry, Mechanics, and Thermodynamics (21 + 41 + 21 topics) Native KL and posterior laws are direct, and the Bernoulli family has an exact Fisher metric, natural gradient, Fisher–Rao distance, and Hellinger divergence. Finite categorical score geometry supplies tangent positivity, full-rank and null witnesses, pullback, scalar Cramér–Rao, chart-equivariant natural gradients, mirror descent, Bregman projection, and replicator equivalence. The later scalar exponential family adds full-support normalization, log-partition derivatives, Fisher–variance equality, KL–Bregman duality, and interval-local mean-coordinate injection. Finite NESS current, path-law fluctuation identities, Jarzynski normalization, reversible KL dissipation, and an exact two-state continuous-time semigroup/master equation are direct at their stated finite scopes. The remaining frontier is general smooth-manifold structure, generic CTMC and continuous stochastic path measures, and PDE-level stationarity.

4.17.5 The Import Pattern Strategy

Each canonical topic body declares the narrow Mathlib modules it uses. Narrow imports make missing APIs and accidental dependencies visible and keep generated coverage meaningful. `LeanVerifier` passes a body with leading imports through as an independent source unit.

4.17.6 A Worked Example: KL Divergence and the ELBO

For a generative model and variational posterior, the familiar ELBO identity is:

$$\mathrm{ELBO}(q) = \log p(o) - \mathrm{KL}(q \,\|\, p(\cdot \mid o)) \,. \tag{19}$$

Equivalently,

$$F(q) = -\mathrm{ELBO}(q) = -\log p(o) + \mathrm{KL}(q \,\|\, p(\cdot \mid o))\,. \tag{20}$$

fep-002 formalizes the algebraic KL remainder in the native extended codomain: surprisal plus `klDiv` is at least surprisal, and equality holds at the posterior under `SigmaFinite`. That row does not derive `p(o)` or the posterior from a joint measure. fep-014 separately proves KL nonnegativity, self-zero, zero characterization, and the composition-product chain rule. The maintained finite model closes the corresponding discrete bridge by constructing predicted evidence and a posterior from normalized kernels, then proving VFE attainment and a support-qualified equality characterization. This layered design avoids pretending that a library theorem alone supplies the generative-model interpretation or that the finite real-valued result automatically lifts to arbitrary measures.

4.17.7 Gap Analysis: What Mathlib4 Does Not Yet Provide

At this pin, the project does not claim a generic measure-KL data-processing theorem. It does provide native measure-KL information gain; finite entropy, KL, and mutual information; both EFE decompositions on a shared policy-conditioned carrier; a complete Bernoulli Fisher–Rao instance; arbitrary finite-dimensional Fisher score geometry; a scalar exponential family; and a normalized finite blanket factorization whose weighted-Dirac embedding satisfies native `CondIndepFun`. The remaining gaps are a general finite-to-measure conditional-information equivalence, smooth manifold and connection theory, generic blanket existence or invariance, and Itô/Langevin and Fokker–Planck developments beyond the exact Boolean continuous-time chain. The coverage audit distinguishes these project obligations from confirmed upstream-library limitations.

4.18 The `sorry` Mechanism and Formalization Maturity

4.18.1 What `sorry` Does

Lean's `sorry` closes an arbitrary proof goal by introducing an admitted placeholder. The file may still compile, but Lean emits a warning and the declaration depends on a generated axiom. Compilation with `sorry` is therefore not proof completion.

```
theorem expected_free_energy_decomposition (π : Policy) :
    EFE π = risk π + ambiguity π := by
  sorry
```

The project scans source text and compiler diagnostics because an exit code alone cannot distinguish a complete declaration from one accepted with `sorry`.

4.18.2 Three Maturity Levels

`mathlib_status` records syntactic/proof completion of a row, with values `real`, `partial`, or `aspirational`. All 155 shipped rows are tagged `real`; no shipped canonical body contains `sorry`. This field is intentionally **not** the semantic maturity verdict.

A second axis, `semantic_disposition`, records whether the primary theorem directly captures its narrowed claim or is a proxy/gap. The current generated counts include 136 `formalized`, 13 `conditional_proxy`, and 6 `structural_proxy` rows. A row can therefore be `mathlib_status: real` without being semantically direct; the schema and firewall preserve that distinction rather than promoting it from compilation.

4.18.2.1 Level 1 — `real` (Kernel-Complete at the Stated Scope) A `real` body has no admitted proof and is accepted by the pinned Lean/Mathlib environment. For example:

```
import Mathlib.MeasureTheory.Measure.MeasureSpace

namespace FEP001
open MeasureTheory

theorem fep001_measure_union_le {α : Type*} [MeasurableSpace α]
    (μ : Measure α) (s t : Set α) :
    μ (s ∪ t) ≤ μ s + μ t :=
  measure_union_le s t
end FEP001
```

This proves exactly the measure union bound. The topic title may motivate an FEP interpretation, but the kernel does not transfer that interpretation into the theorem type. Semantic review must do that separately.

4.18.2.2 Level 2 — `partial` (Typed Statement With Admitted Subgoals)

A `partial` row would contain one or more localized `sorry` terms. Such a row can be useful on a research branch because Lean checks the surrounding types, but it is excluded from the shipped catalogue and from claim-ready evidence. The preferred publication behavior is either to prove the missing lemma or to narrow the statement to an honest complete theorem and record the remaining gap in the semantic audit.

4.18.2.3 Level 3 — `aspirational` (Target Signature Only)

An `aspirational` row documents a desired signature whose proof is not present. The project keeps such targets in prose or coverage-roadmap fields rather than in the verified catalogue. This prevents a compilable file containing an admitted axiom from resembling a completed result.

4.18.3 The Zero-`sorry` Policy

The canonical catalogue, generated aggregate, native receipt, and publication variables all enforce zero `sorry`. A native receipt is claim-ready only when it covers the exact ordered roster of 155 topics and independently reports zero warnings and zero admitted bodies. A filtered or stale receipt cannot satisfy the full-catalogue predicate.

Zero `sorry` is necessary but not sufficient. The semantic audit additionally checks the exact primary theorem, assumption quality, non-vacuity rationale, and acceptance probe. This blocks the common failure mode of replacing a difficult domain theorem with an easy tautology while retaining the stronger title.

4.18.4 The Compilation Gate: How Zero-Sorry Is Enforced

The enforcement layers are complementary:

1. `FEPTopicCatalogue` rejects malformed or incomplete generated rows and theorem/signature count drift.
2. `LeanVerifier` detects non-comment `sorry`, invokes the real pinned compiler, and retains warnings/errors in a structured result.
3. `uv run fep-lean verify --fail-on-warnings --receipt output/native-verification.json` requires clean per-topic results and atomically writes a source-digest-bound receipt.
4. `lake build FepSketches` checks the topic aggregate and manifested maintained formal modules together, exposing namespace collisions, broken dependencies, and declaration warnings.
5. `scripts/_maint_build_fep_all_lean.py --check` and `scripts/_maint_build_formal_modules.py --check` reject projection drift without rewriting tracked files.
6. The receipt validator recomputes roster, counts, digests, toolchain, and claim readiness instead of trusting stored booleans.
7. The formalism declaration/axiom audit, semantic audit, and theorem-reference audit reject unresolved evidence names, `sorryAx`, missing primary declarations, and stale manuscript theorem names.

These checks distinguish proof completeness, compiler cleanliness, source freshness, and semantic fidelity rather than collapsing them into one "green" status.

4.18.4.1 ✓ `real` (What It Does and Does Not Mean)

For a canonical row, `real` means:

- the statement and proof term are accepted at the pinned toolchain;
- the body contains no `sorry`;
- imported declarations resolve; and
- the generated source/signature projections agree.

It does **not** mean that the topic title is proved at its broadest reading, that assumptions hold in a biological system, or that the theorem is empirically adequate. For example, fep-028 directly proves finite softmax normalization, while fep-005 proves only properties of a supplied finite label assignment. Both are kernel-complete; their semantic reach differs.

4.18.5 Migration From `partial` to `real`: A Worked Example

The important migration is not merely "make the compiler green." fep-014 previously used generic measure-set inequalities as a proxy for KL behavior. Inspection of the pinned Mathlib source showed that native `InformationTheory.klDiv`, self-zero, finite-measure zero characterization, and the composition-product chain rule were already available. The row was replaced with direct wrappers of those declarations and its semantic disposition was upgraded only after the theorem type, assumptions, and non-vacuity were reviewed.

The workflow for such a migration is:

```
uv run python scripts/_maint_build_topics_catalogue.py
uv run python scripts/_maint_build_fep_all_lean.py
uv run fep-lean verify --topic fep-014 --fail-on-warnings
uv run python scripts/theorem_maturity_audit.py
uv run python scripts/build_formalism_coverage.py
```

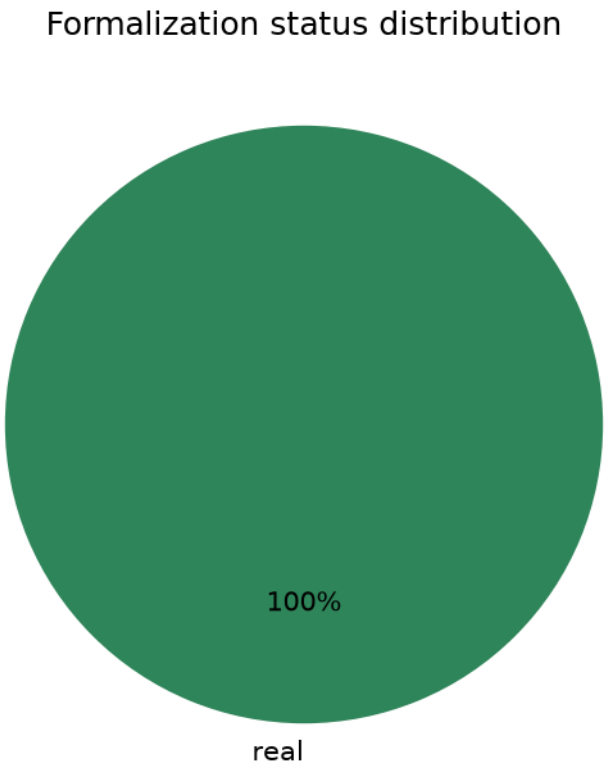


Figure 1: Syntactic proof-status distribution across the 155 catalogue rows. This figure reports `mathlib_status`, not semantic disposition; the separate coverage report supplies the semantic matrix.

This sequence updates the authoring graph, proves the focused row, and refreshes the semantic projection. A full receipt is generated only after the entire catalogue passes.

4.18.6 Maturity by FEP Area

Per-area `mathlib_status` is uniform in this release and therefore not very informative. The generated area-by-`semantic_disposition` matrix in `docs/formalism-coverage.md` is the meaningful comparison: it distinguishes 136 direct rows from 13 conditional and 6 structural proxies. The fep-036 topic adds a finite binomial PMF, an outcome-indexed Laplace estimator, exact shrinkage, and deterministic consistency transfer. The statistical-convergence module discharges its asymptotic premise, the learning family separately proves finite concentration and model-evidence laws, and `fep-121-fep-127` add finite-law Laplace squared/Brier-risk and bad-event transfer. These results still do not supply posterior contraction, minimax or empirical calibration, or a marginal-likelihood optimum for the estimator.

4.18.7 Why Aspirational Proofs Are Rejected

Shipping admitted targets would blur three distinct activities: specifying a desirable theorem, proving it, and showing it represents the scientific claim. The project instead keeps the executable corpus axiom-free with respect to `sorry`, records stronger targets as explicit gaps, and permits narrow complete proxies only when their reduced scope is visible in the catalogue and manuscript. This produces fewer headline theorems, but a much clearer evidentiary boundary.

4.19 The Hermes AI Agent and LLM-Assisted Formalization

4.19.1 Architecture Overview

`fep_lean.llm.hermes.HermesExplainer` reviews a curated `TopicEntry`. It does not own the catalogue, overwrite a canonical topic body, or establish a theorem merely by returning text. Family modules under `src/fep_lean/catalogue/bodies/` own those bodies and `registry.py` validates their ordered union; `config/topics.yaml` is the generated projection. A live model response becomes relevant only after its extracted Lean block is compiled and its complete run bundle passes independent receipt validation.

The client uses standard-library HTTP against an OpenAI-compatible endpoint. Configuration is resolved from explicit environment values, project settings, and built-in defaults. The exact configured model and fallback roster are operational facts in `config/settings.yaml`

and the validated run manifest; copying them into the paper would create a second, rapidly stale owner.

4.19.2 Gauss Session Protocol

For each selected topic, `GaussRunner` performs the following state transition:

1. create an SQLite session containing the topic id, area, workflow, and canonical body;
2. retrieve an unexpired, toolchain-salted Hermes cache entry or make a live request;
3. persist the system, user, assistant, and optional reasoning turns;
4. reject an unsuccessful response or one without an extracted Lean block;
5. compile the refined block with `LeanVerifier`;
6. persist a structured artifact containing Hermes and compiler fields; and
7. close the session with an explicit success or failure status.

An exception cannot silently leave a successful session: the runner closes an active session with the exception information. Cache writes require both model success and a nonempty extracted sketch, preventing a truncated response from poisoning later runs. Cache keys include the topic body, workflow stage, model, exact rendered-message digest, and Lean/Mathlib salt, so a prompt change cannot inherit an older response accidentally. Persisted request/response turns use contiguous indices and retain the exact prompt that generated the cached or live result.

```
flowchart LR
    source[canonical TopicEntry] --> session[SQLite session]
    session --> hermes[Hermes request or valid cache]
    hermes --> parsed{nonempty Lean block?}
    parsed -- no --> failed[closed failed session]
    parsed -- yes --> lean[lake env lean]
    lean --> artifact[structured artifact]
    artifact --> receipt{full receipt valid?}
    receipt -- no --> ineligible[not manuscript evidence]
    receipt -- yes --> eligible[full-run claims eligible]
```

4.19.3 FEP-Domain System Prompt

The system prompt asks for a short mathematical explanation and one refined Lean block. Its load-bearing constraints are concrete: copy the original imports, preserve the `FEPNNN` namespace, retain explicit tactic hint lists, and never introduce `sorry` into a source that was already `sorry`-free. Workflow preambles may request draft, prove, review, or verify behavior, but they do not weaken the compiler or receipt gates.

Prompt constraints reduce predictable damage; they do not certify semantic fidelity. A model may preserve a compiling theorem while changing its meaning, or provide persuasive commentary for a weak proxy. In the two-pass `review` workflow, the initial candidate must compile cleanly with no warnings or `sorry` before the prose-only review is requested; a review that returns a replacement Lean block is rejected rather than silently changing the compiled candidate. The canonical theorem remains researcher-maintained, and the semantic audit remains independent of both model confidence and compiler success.

4.19.4 Hermes vs Native Lean: Compilation Diagnostics

Hermes success and Lean success are orthogonal. A well-formed explanation can accompany a non-compiling block; a compiling block can express a weak or irrelevant theorem. Accordingly, the report schema records the model response, direct refined-sketch outcome, final compiler fields, and semantic disposition separately. No aggregate rate is inferred from a catalogue-only run.

4.19.5 Compiler Output and the `VerifyResult` Dataclass

`LeanVerifier.verify_sketch` returns a structured `VerifyResult`: `compiles`, `has_sorry`, warnings, errors, stdout/stderr, duration, Lean version, skip reason, and an advisory failure class. Its status projection distinguishes `compiles_clean`, `compiles_with_sorry`, `compile_error`, and `skipped (...)`. The failure classifier helps triage missing imports, renamed identifiers, tactic failures, arity/type mismatches, and timeouts, but it is not proof evidence itself.

The full workflow does not treat the canonical body as a hidden success fallback after Hermes failure. If Hermes is disabled, lacks credentials, returns no usable candidate, or exhausts its model attempts, the topic fails full mode. Native verification of the canonical catalogue is available as a separate `fep-lean verify` contract.

4.19.6 Token Usage and Cost Profile

Token, model, cache, retry, and latency fields are populated only from an independently validated full report. At render time the current full-claim predicate is `false`. If it is false, model totals and timings are unavailable rather than borrowed from an old or catalogue-only directory. This policy is important because provider latency, context accounting, and model availability are mutable external conditions.

For a claim-ready run, `summary.json` remains the primary quantitative record: it contains per-topic model identity, token count, duration, cache status, same-model retries, and chain-advance reason. The manuscript may summarize those values but never substitutes configuration defaults for observations.

4.19.7 Model Fallback Chain and Degradation

`HermesExplainer` builds an ordered chain from the configured primary and deduplicated configured/built-in fallbacks. A configurable cap can bound attempted models. A successful response records the model actually used; failure records the last error rather than inventing content.

The client validates obvious key/endpoint mismatches and uses a bounded preflight probe. A fatal authentication/client error may disable Hermes for the remainder of a run. Transport failures and rate limits are bounded rather than retried indefinitely. These are availability mechanisms, not mechanisms for choosing the mathematically best answer.

4.19.8 Three Classes of Fallback

The implementation distinguishes mechanisms that are often conflated in LLM reports:

Mechanism	Trigger	What changes	Recorded field	Evidentiary interpretation
Same-model retry	HTTP `429` or a transient transport failure	network attempt only	`network_retries`, aggregated as `network_retry_count`	bounded availability recovery on the same model
Cross-model advance	empty content, wall-clock timeout, non-retriable HTTP failure, transport failure, or parse failure	provider model	`chain_advance_reason`, `model_used`, aggregated as `model_fallback_count` and `chain_advance_reasons`	the configured primary did not supply the accepted response
Hermes-refined Lean outcome	the returned refinement is compiled with `lake env lean`	no provider retry; this occurs after the LLM stage	`hermes_lean_compiled`	a failed refinement remains a topic failure; no baseline body is substituted
Native canonical verification	the curated catalogue is compiled independently of Hermes	execution mode	source-bound native receipt	separate non-LLM evidence contract

`HermesResult` carries the per-call retry count and labeled chain-advance cause, and `TopicRunResult.as_dict()` propagates them into `summary.json`. The manuscript renderer aggregates only a claim-ready full report. Native verification is deliberately not counted as Hermes success; keeping the two planes separate prevents a curated canonical body from masking a failed live model experiment.

4.20 Native Lean 4 Compilation and Execution-Integrity Verification

Native verification is a compiler operation, not a catalogue property. The rendered evidence kind is `native-lean`; claim readiness is `true`; the full-catalogue rate is `155/155 (100.0%)`, with 0 warnings and 0 admitted proofs in the selected receipt. If no matching receipt exists, these fields render as unavailable rather than borrowing a `complete` flag from catalogue mode.

4.20.1 Why Simulated Compilation Fails

A generated success value cannot establish that imports resolve, types unify, tactics close their goals, or the pinned Mathlib API contains a cited declaration. The package therefore sends the exact canonical topic body to the pinned Lean binary and records compiler output. Tests may isolate subprocess behavior, but publication evidence must come from an actual native receipt.

4.20.2 The `lean_verifier.py` Architecture

The compiler bridge lives in `fep_lean.verification.lean_verifier`. Its public contract is intentionally small: isolate a sketch, invoke Lean through Lake, and return a structured `VerifyResult`. The source remains authoritative for implementation details.

4.20.2.1 Stage 1: Sketch Isolation and Temporary File Construction Each topic is written to its own temporary file inside the Lake workspace. Isolation prevents declarations from one topic satisfying another topic accidentally and permits the temporary source to be removed after the result is captured.

4.20.2.2 Stage 2: Native Shell Invocation

The verifier invokes the pinned workspace through:

`lake env lean FepSketches/<temporary-file>.lean`

Executable discovery respects explicit overrides and the toolchain named by `lean/lean-toolchain`; it does not silently download dependencies. Acquisition belongs to the explicit setup command.

4.20.2.3 Stage 3: Output Parsing and `VerifyResult`

`VerifyResult` separates process success, `sorry` detection, compiler warnings, compiler errors, elapsed time, and an advisory failure category. Native claim readiness is stricter than process exit zero: every requested row must compile, no source may contain `sorry`, and warnings must be absent.

Result dimension	Why it remains separate
`compiles`	Lean accepted elaboration and checking
`has_sorry`	The source admitted an unproved goal
`warnings`	Lean accepted the file but reported a quality or migration issue
semantic disposition	Human-reviewed relationship between theorem and topic claim

4.20.3 Aggressive Mathlib4 Caching

The local `.lake` tree is an untracked build cache. Verification refuses a visibly incomplete Mathlib workspace and instructs the operator to run the explicit setup path. The cache changes latency, not theorem meaning, and it is never publication evidence.

4.20.4 Measured Compilation Headline

The only headline used here is receipt-derived: **`155/155 (100.0%)`** for receipt `native-566a22fce86c`. Its 155 results contain 155 compiler successes, 0 failures, 0 warnings, and 0 uses of `sorry`. `mathlib_status: real` is not substituted for any of these numbers.

4.20.5 Preflight: `LeanVerifier.check_mathlib_built()`

Preflight checks that the pinned Lake workspace and required Mathlib object files exist before a batch begins. It is intentionally read-only. A failed preflight stops verification rather than producing one misleading import failure per topic.

4.20.6 Verbose Mode: `FEP_LEAN_VERIFY_VERBOSE=1`

Verbose mode increases diagnostic logging for local reproduction. The stable automation surface is the CLI JSON and optional native receipt; prose should not depend on incidental log formatting.

4.20.7 Sequential Batching: `verify_batch(max_workers=1)`

The shared Lake workspace is verified sequentially. This favors reproducibility over throughput because concurrent compiler processes can contend over the same build tree. Parallelism would require isolated workspaces and a separate evidence review.

4.20.8 Cache Timing: Cold vs Warm vs Cached

Compiler timing depends on hardware and cache state. This manuscript reports only the elapsed values present in the selected receipt: 556.106 seconds total and 3.588 seconds per result. It does not promote local cold/warm observations into general performance claims.

4.20.9 The Execution-Integrity Standard Applied

Execution integrity means that evidence is collected at the boundary where the claim is decided:

- Lean claims come from the Lean process and source scan;
- source identity comes from content digests;
- full-run claims come from reconciled and hashed report artifacts;
- semantic adequacy comes from maintained review records;
- manuscript values come from validated projections.

No one boundary can certify another.

4.20.10 Methodological Assumptions and Limits

Lean checks derivability in the imported formal system. It does not choose the uniquely correct formalization of an informal FEP statement, validate an empirical model, establish physical realizability, or infer composition merely because separately namespaced bodies share imports. Composition is established only where the maintained foundations or named leaf-composition witnesses state it explicitly; every other bridge remains a semantic or scientific obligation.

4.20.10.1 Formal Definition of Execution Integrity A native claim is accepted exactly when a schema-valid receipt identifies the canonical ordered roster, reconciles every result row, reports complete warning-free and `sorry`-free compilation, and matches the current catalogue, family-body source manifest, Lean toolchain, and Mathlib pin. A full claim additionally requires a valid full-mode report with artifact hashes and current source/configuration digests.

4.20.10.2 Per-Stage Success Criteria

Boundary	Acceptance criterion	Failure meaning
Catalogue generation	Typed sources join and generated projections are current	Authoring or drift error
Native Lean	Selected compiler results satisfy the native receipt	Formal compilation evidence unavailable
Full pipeline	Required capabilities and every selected full-mode row succeed	No Hermes/OpenGauss publication claim
Manuscript rendering	Every placeholder and theorem identifier resolves	Publication build blocked

4.20.10.3 Three Levels of Truth The project keeps three claims ordered but non-interchangeable:

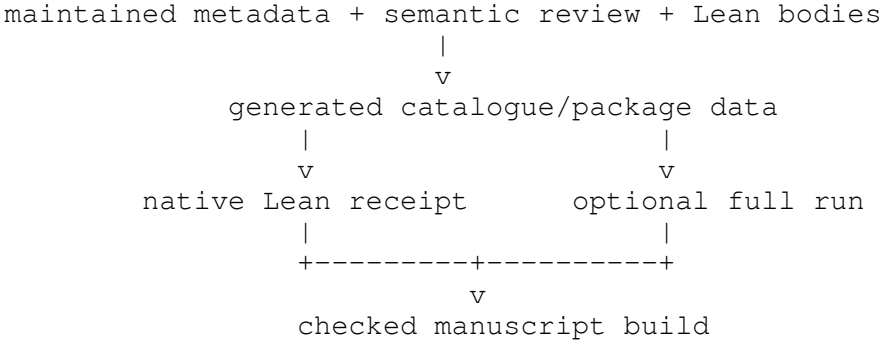
1. **Syntactic and kernel evidence:** the stated Lean proposition compiles without warnings or admissions.
2. **Semantic adequacy:** the proposition matches the reviewed topic claim under explicit assumptions and has a non-vacuity argument where needed.
3. **Scientific adequacy:** the formal object captures the intended empirical or physical system.

The current work directly automates the first, maintains an explicit audit of the second, and does not claim the third.

4.21 Pipeline Architecture and Execution Profile

4.21.1 The Central Execution and Orchestration DAG

The architecture is organized around evidence boundaries rather than one undifferentiated success flag:



Catalogue generation is offline. Native verification requires the pinned Lean workspace. Full mode additionally requires the declared Hermes and OpenGauss capabilities. Manuscript rendering consumes validated evidence but cannot create it.

4.21.2 Expression Lifecycle: YAML → Manuscript → Lake

The direction of ownership is fixed:

1. `config/catalogue_metadata.yaml`, `config/theorem_maturity.yaml`, `config/formalism_novelty.yaml`, and the family-owned `fep_lean.catalogue.bodies` modules are maintained inputs.
2. The catalogue generator joins metadata, semantic review, and bodies strictly by stable topic ID, validates every required novelty bridge against the maintained composition leaves, and writes checkout and wheel data from identical bytes.
3. Native verification reads the generated catalogue, invokes Lean, and may write a digest-bound receipt.

4. Manuscript variables are projected from the catalogue plus validated receipts.
5. The renderer validates all placeholders before writing numbered chapters to a separate build directory.

Generated YAML, aggregate Lean, coverage reports, manuscript variables, and appendices are projections; hand edits to them are not a valid repair.

4.21.3 Sequence Diagram: Single Topic Execution

For native verification, one selected topic follows catalogue load → isolated temporary source → `lake env lean` → structured result. In full mode, Hermes may propose or explain a sketch before native checking, but the kernel outcome remains decisive. A failed or unavailable provider cannot be converted into a native or full success through baseline substitution.

4.21.4 Persistent State: Dual-Mode Storage

Offline catalogue mode needs no external state. Native receipts are standalone JSON artifacts. Full mode may persist session and model interaction state through OpenGauss-compatible SQLite storage, while its publication contract is the independently validated report directory. The distinction lets readers verify compiler evidence without trusting a database or provider.

4.21.4.1 Per-topic audit trail Canonical topic content lives in the generated package catalogue; semantic review lives in the maintained maturity source. Native receipt rows add compiler status, warnings, admissions, elapsed time, and toolchain identity. Full report rows add provider and orchestration provenance. The layers are linked by topic ID and digests, not by copied prose.

4.21.4.2 Run-level artifacts The report validator, rather than this manuscript, owns the exact artifact roster. It rejects missing required files, unsafe paths, hash mismatches, disagreement among summary/run/verification manifests, incomplete topic rosters, and drift from the live source tree.

4.21.5 SQLite Storage Boundaries

The optional persistence layer separates sessions, dialogue turns, artifacts, logs, and cached provider responses. This separation exists so a compiler result is not reconstructed from an LLM transcript and a cached provider response is not mistaken for a fresh run. The schema in `fep_lean.gauss.client` is canonical; duplicating its columns here would invite drift.

4.21.6 Environment Variable Reference

Environment variables configure credentials, provider budgets, timeouts, and storage locations. The supported names and defaults belong to the configuration and CLI documentation. Two principles matter for reproducibility:

- credentials enable full mode but do not alter native theorem truth;
- changing a timeout or model invalidates any claim that a previous full receipt describes the new configuration.

The package now uses the unambiguous `fep_lean` namespace, so correctness no longer depends on placing collision-prone top-level modules such as `catalogue`, `pipeline`, or `output` first on `PYTHONPATH`.

4.21.7 Representative Run Statistics

No unvalidated run is called representative. The renderer reports native evidence `native-lean` with `155/155` compiler successes, 0 warnings, and 0 admissions. It reports full readiness as `false`; Hermes fields remain unavailable unless that predicate is true.

4.21.8 Execution Metrics: Representative Run

For receipt `native-566a22fce86c`, recorded native compiler time is 556.106 seconds in total and 3.588 seconds per result. Provider latency, if present in a claim-ready full receipt, is reported independently. These observations are machine-specific and are not extrapolated into general performance bounds.

4.21.9 Reproducibility Checklist

From the project root:

1. synchronize the locked Python environment with `uv sync --locked`;
2. explicitly prepare the pinned Lean workspace with `uv run fep-lean setup` when its cache is absent;
3. regenerate and drift-check catalogue, aggregate Lean, semantic audit, and coverage projections;
4. run `uv run fep-lean verify --fail-on-warnings --receipt output/native-verification.json` for native evidence;
5. generate manuscript variables, run the placeholder and theorem-reference audits, and render to `output/manuscript/`;

6. run full mode only when its external capabilities and credentials are intentionally supplied, then validate its report before citing it.

Hermes wording may vary. A fixed Lean source under the pinned toolchain is reproducible; a native receipt additionally binds that result to the current catalogue bytes.

5 Formalisms, Model Specifications, and Results

The core artifact is a generated catalogue of 155 topic-scoped Lean bodies plus semantic review metadata, accompanied by reusable maintained foundations. Canonical topic bodies live in the family modules under `fep_lean.catalogue.bodies`; generated YAML and the unified appendix are projections. Native evidence is `native-lean` with rate `155/155 (100.0%)`. This section discusses mathematical scope; it does not infer full-pipeline evidence from catalogue generation.

5.1 Foundational Dynamics: Free Energy Principle (41 topics)

A common variational identity writes free energy as surprisal plus a KL remainder:

$$F[q] = -\log p(o) + \mathrm{KL}[q \,\|\, p(\cdot \mid o)]\,. \tag{21}$$

The catalogue now represents the nonnegative-extended-real core of this identity directly. `fep002_variationalFreeEnergy` defines surprisal plus `InformationTheory.klDiv`; `fep002_vfe_ge_surprisal` proves the upper bound; and `fep002_vfe_exact_at_posterior` uses Mathlib's self-divergence theorem under `SigmaFinite`. This is a meaningful strengthening over a theorem that merely assumed an unnamed nonnegative scalar remainder.

The topic result is deliberately narrower than the displayed FEP equation: it does not construct the observation marginal, conditional posterior, or real logarithmic evidence term from a joint generative model. The maintained finite kernel supplies exactly that discrete construction on normalized finite laws, proves Bayes reconstruction, and shows that posterior-form VFE bounds outcome surprisal, with exact-posterior attainment and support-qualified uniqueness. A bridge from this finite real carrier to the topic row's general measure-valued `ENNReal` statement remains intentionally explicit rather than being inferred from matching notation.

5.1.1 Core Mathematical Formalisms and Theoretical Definitions

The FEP-area rows provide heterogeneous substrates: measure operations and measurability, strict concavity of the logarithm, finite sums, quadratic bounds, minima, conjugate updates, contraction fixed-point theory, and finite-law risk transfer for the outcome-indexed Laplace estimator. They should not be read as one already-composed proof. The generated coverage report gives the exact primary theorem, disposition, assumptions, and imports for each row; the appendix gives the statements and proofs.

The familiar Jensen route to an ELBO remains useful context:

$$\log p(o) = \log \mathbb{E}_{q(s)}\left[\frac{p(o,s)}{q(s)}\right] \geq \mathbb{E}_{q(s)}\left[\log \frac{p(o,s)}{q(s)}\right] = -F[q]. \tag{22}$$

No catalogue theorem currently proves this whole chain with its integrability and almost-everywhere conditions. fep-035 now imports Mathlib's strict-concavity result for `Real.log` and proves the exact two-point strict Jensen inequality for distinct positive inputs and positive weights summing to one. That theorem is a genuine finite convexity fact, but it is not the expectation-level Jensen step above. fep-002 supplies a native KL remainder identity at a narrower abstraction boundary.

Likewise, the elementary quadratic update used as a local descent model is:

$$x_1^2 = (1-\eta)^2 x_0^2 \leq x_0^2 \qquad \text{when } |1-\eta| \leq 1. \tag{23}$$

fep-032 now proves more than the one-step inequality: for the centered scalar quadratic it derives the exact n-step displacement $(1-\eta)^n(x_0 - x^\star)$, proves energy descent for $0 \leq \eta \leq 2$, and proves convergence to the center for $0 < \eta < 2$. `FEPComposed.fep032_update_is_fep043_gradientStep` identifies that update with a gradient step on fep-043's positive-curvature quadratic, whose unique minimizer, critical-point equivalence, and Hessian are independently checked. fep-048 supplies the complementary general Banach-style contract and a concrete halving witness. None of these results establishes convergence for an arbitrary variational objective.

Cross-topic composition is now an explicit proof surface. `FEPComposed.fep002_vfe_compProd_chain_rule` expands fep-002 variational free energy through fep-014's native KL chain rule; `FEPComposed.fep024_regularizer_is_fep014_kl` pins the regularizer convention to the same divergence; and the quadratic theorem above connects landscape, gradient, iteration, and convergence without copied definitions. These are among **20** declaration-backed seams. The finite foundation additionally supplies a generative-model-to-evidence theorem on one discrete real-valued carrier; the stronger open target is a measure-valued bridge for the full displayed FEP equation, not the absence of any constructed model.

The `fep-121-fep-127` family closes a different finite boundary. On the same add-one estimator introduced by fep-036, it proves the exact error and bias decompositions, absolute- and squared-error transfer, a normalized finite-law risk bound, the Bernoulli Brier excess-risk identity, a Laplace Brier-risk bound, and bad-event containment. A nonzero-bias boundary and a nondegenerate sampling law keep those implications from being vacuous. These are finite sampling-risk theorems, not posterior contraction, minimax optimality, empirical calibration, or marginal-likelihood optimization; the exact declarations and hypotheses are tabulated in §7.2.

The FEP rows therefore establish exact local facts, not the Free Energy Principle as a theorem of self-organization. Their scientific value lies in exposing which bridges—generative model to posterior, posterior to KL objective, objective to dynamics, dynamics to steady state, and finite risk to asymptotic or empirical guarantees—still require formal statements.

5.2 Intermediate Dynamics: Active Inference (31 topics)

Transitioning into temporally extended behavior, Active Inference (K. Friston et al. 2017, 2016) introduces action policies and predicted consequences, including motor-control accounts in which prediction errors drive action (Adams, Shipp, and Friston 2013). A common discrete formulation has an inner inference problem for variational free energy and an outer policy problem for expected free energy (Da Costa et al. 2023). The 31 catalogue rows map finite-sum, update, policy-selection, controlled-kernel, finite-belief, policy-tree, collective, and consensus substrates for that formulation. The maintained kernel composes the central objects in one `GenerativeModel`: policy-conditioned prediction, evidence and posterior, posterior-form VFE, one-stage and cumulative EFE, prior-weighted policy selection, action pushforward, and open-loop rollout. Separate foundations add normalized controlled kernels, positive-evidence belief updates, reachable finite-belief reduction, soft Bellman and desirability recursions, a prior-weighted control posterior, arbitrary finite-depth observation-contingent policy trees with optimality and open-loop dominance, a strict feedback witness, and product-agent and consensus laws. These finite constructions are not a refinement proof for an executable agent, an infinite-horizon or continuous-belief theorem, a posterior over policy trees, or a theorem for arbitrary multiagent networks.

5.2.1 Generative Model, Variational Inference, and Policy Evaluation

Active Inference begins with the same generative model that underwrites the static FEP, but partitioned into observation and latent components with an explicit decomposition into likelihood and prior:

$$p(o, s) \;=\; p(o \mid s)\, p(s), \tag{24}$$

and extended to sequences $(o_{1:T}, s_{1:T})$ under a (possibly time-varying) transition kernel $p(s_{t+1} \mid s_t, a_t)$ with action a_t. Writing out the full sequential generative model,

$$p(o_{1:T}, s_{1:T} \mid a_{0:T-1}) \;=\; p(s_1) \prod_{t=1}^{T} p(o_t \mid s_t) \prod_{t=1}^{T-1} p(s_{t+1} \mid s_t, a_t), \tag{25}$$

makes the *factor graph* structure of the problem explicit. Inference is often performed by local message passing. fep-007 proves positivity and exact normalization of one finite local sum-product message; fep-047 packages a full finite forward pass as `Matrix.mulVec` and proves exact two-pass composition; fep-017 supplies Mathlib's posterior kernel and joint-reconstruction law; and fep-034 composes transition and observation kernels into a normalized filter. These results still do not define graph topology or a loopy update schedule, so tree exactness and iterative convergence are not inferred.

The agent's variational posterior $q(s)$ approximates the true posterior $p(s \mid o)$ by minimizing the standard variational free energy (Eq. 21); under mean-field or structured factorizations this motivates message-passing updates. The catalogue distinguishes three exact layers: finite normalized local messages (fep-007), algebraically compositional finite propagation (fep-047), and measure-valued Bayesian posterior/filter kernels (fep-017/fep-034). `FEPComposed.fep034_filter_is_fep017_posterior` proves that the filter is precisely the native posterior at the transition-predicted prior. The finite kernel adds a policy-conditioned normalized state–outcome joint and exact posterior reconstruction. What remains absent is a typed factor-graph topology with loopy-message schedules and a theorem equating those schedules to the kernel posterior, not a shared finite generative model itself.

5.2.2 Expected Free Energy and Policy Selection

Many discrete Active-Inference formulations evaluate candidate policies with an **expected free energy** $G(\pi)$ over predicted states and outcomes. Because several inequivalent conventions appear in the literature (Millidge, Tschantz, and Buckley 2021; Champion et al. 2026), we fix the convention used by the maintained finite kernel. Let $q_\pi(s)$ be the predicted state law, $\ell(o \mid s)$ a normalized likelihood, $q_\pi(s, o) = q_\pi(s)\ell(o \mid s)$ the predictive joint, $q_\pi(s \mid o)$ its positive-evidence posterior, and $p_C(o)$ a positive preferred-outcome law. Then

$$G(\pi) \;=\; \mathbb{E}_{q_\pi(s,o)}[\log q_\pi(s) - \log q_\pi(s \mid o) - \log p_C(o)]\,. \tag{26}$$

The first two logarithms have expectation $-I_{q_\pi}(S;O)$, while the last contributes preference cross-entropy. Thus Equation 26 is exactly the pragmatic-minus-epistemic convention formalized by the finite carrier. For a horizon, a common stage-additive extension is

$$G_{1:T}(\pi) \;=\; \sum_{\tau=1}^{T} G_\tau(\pi), \tag{27}$$

where each G_τ is evaluated after replacing the model's initial law by the state prediction from the preceding stage. `cumulativeExpectedFreeEnergyFrom` implements this recursion. Lean proves exact decomposition at every prefix–suffix boundary and nonnegativity under a recursive stagewise full-support contract. That carrier remains open loop. fep-071 uses a separate finite belief/action/observation carrier to reoptimize after each observation-dependent update and proves a two-stage Boolean case where feedback strictly beats every fixed open-loop second action. The `fep-128-fep-134` family then defines observation-indexed policy trees at every finite depth, proves recursive Bellman minimization and optimal-tree existence, embeds open-loop plans, proves weak closed-loop dominance, transports the risk–ambiguity EFE identity under full support, and supplies a strict Boolean feedback witness (K. Friston et al. 2020).

5.2.2.1 Risk–Ambiguity Decomposition Using $q_\pi(s,o) = q_\pi(s)\ell(o \mid s)$, the entropy identity $I(S;O) = H(O) - H(O \mid S)$, and the cross-entropy identity $\mathrm{CE}(q_\pi(o), p_C) = H(q_\pi(o)) + D_{\mathrm{KL}}(q_\pi(o)\|p_C)$ yields the **risk–ambiguity** decomposition:

$$G(\pi) \;=\; \underbrace{D_{\mathrm{KL}}(q_\pi(o)\,\|\,p_C(o))}_{\text{preference risk}} \;+\; \underbrace{\mathbb{E}_{q_\pi(s)}[H(\ell(\cdot \mid s))]}_{\text{ambiguity}}, \tag{28}$$

where $H[\ell(\cdot \mid s)] = -\sum_o \ell(o \mid s)\log \ell(o \mid s)$. Risk is a divergence between predicted and preferred outcomes, not merely expected preference surprise; the latter is the pragmatic cross-entropy and differs from risk by the predicted-outcome entropy. Ambiguity is the expected entropy of the observation channel: a high-ambiguity policy favors states whose likelihood is dispersed. In the finite kernel, both terms are nonnegative, so this decomposition also proves $G(\pi) \geq 0$ under the stated full-support contract.

5.2.2.2 Epistemic–Pragmatic Decomposition The defining rearrangement is the **epistemic–pragmatic** form:

$$G(\pi) \;=\; \underbrace{\mathrm{CE}(q_\pi(o), p_C(o))}_{\text{pragmatic cost}} \;-\; \underbrace{I_{q_\pi}(S;O)}_{\text{epistemic value}}. \tag{29}$$

Here mutual information measures anticipated information gain about the hidden state, and cross-entropy measures expected negative log preference. Expanding the two finite sums gives

$$G(\pi) = \sum_o q_\pi(o)[-\log p_C(o)] - \sum_{s,o} q_\pi(s,o)\log \frac{q_\pi(s,o)}{q_\pi(s)q_\pi(o)}. \tag{30}$$

The maintained finite information layer defines the second sum as KL divergence from the joint to the product of its marginals. Under full support, Lean proves that Equations 28 and 29 agree exactly. Topic fep-021 remains a distinct `ENNReal` convention: pragmatic cost truncated-subtract epistemic value. It proves $G + \mathrm{IG} = C$ under the visible premise $\mathrm{IG} \leq C$, monotonicity in cost, antitonicity in information value, and the zero boundary; `FEPComposed.fep021_informationGain_balance` instantiates its information input with fep-041's native measure KL. The finite real-to-`ENNReal` bridge uses `ENNReal.ofReal`, so negative real values would be truncated rather than silently identified.

5.2.2.3 Softmax Policy Selection with Precision Policies are then selected via a **Boltzmann/softmax posterior** over G:

$$q(\pi) \;=\; \frac{p_0(\pi)\exp(-\gamma\, G(\pi))}{\sum_{\pi'} p_0(\pi')\exp(-\gamma\, G(\pi'))}, \qquad \gamma \geq 0, \tag{31}$$

with prior policy law p_0 and precision γ controlling the exploration–exploitation balance. At $\gamma = 0$, the posterior equals p_0 rather than being uniformly distributed unless the prior is uniform. In a finite policy space, the large-precision limit concentrates on prior-supported EFE minimizers, with relative mass among tied minimizers inherited from p_0. The maintained kernel proves finite normalization for the one-stage EFE posterior, existence of MAP and EFE-minimizing policies, antitone ordering by EFE for equal priors and nonnegative precision, and exact pushforward of policy mass to an action law. fep-070 separately normalizes a prior-weighted exponential action score for finite control. The later policy-tree family proves deterministic finite-tree optimality but does not define a probability posterior over trees or infer precision.

Some Active-Inference models infer precision γ rather than treating it as fixed. That hierarchical precision update is not present in fep-028. The row defines a support-aware finite softmax for any real γ, proves nonnegativity and a unit upper bound, proves exact zero outside the selected support, and proves that the complete finite policy vector sums to one. `FEPComposed.fep012_softmax_entropyRegularizedCost_le` then feeds this law into fep-012's Shannon-entropy-regularized objective. Neither theorem proves precision learning or the asymptotic temperature limits stated informally above.

5.2.2.4 Exploration Bonus and Information Gain

Equation 32 motivates **fep-041**, which now defines information gain directly as Mathlib's measure-valued `InformationTheory.klDiv`:

$$\mathrm{IG}(\pi) \;=\; \mathbb{E}_{q(o|\pi)}\big[\mathrm{KL}\big[q(s \mid o,\pi)\,\big\|\,q(s \mid \pi)\big]\big] \;=\; I(s;o \mid \pi) \;\geq\; 0. \tag{32}$$

fep-041 proves nonnegativity, finite-measure separation, and vanishing expected information gain when a posterior kernel equals the prior almost everywhere under the predictive observation measure. The maintained finite carrier now builds the policy-conditioned predictive joint and defines epistemic value as its mutual information. The remaining bridge is a theorem identifying that finite joint-KL definition with fep-041's measure-valued posterior-averaged KL under matched embeddings and support assumptions. Claims about epistemic foraging also require a behavioral or dynamical model beyond either information identity.

5.2.2.5 Markov Decision Processes as a Special Case

A common comparison recovers reward-like policy objectives from log prior preferences. If the prior over preferred outcomes is written as

$$p(o \mid C) \;=\; \begin{cases} \exp(r(o))/Z & o \in \mathcal{G}, \\ 0 & o \notin \mathcal{G}, \end{cases} \tag{33}$$

or more generally $\log p(o \mid C) = r(o) - \log Z$, the pragmatic term resembles negative expected reward up to a constant. fep-033 now defines a deterministic transition-aware finite-horizon value in `ENNReal` and proves its exact Bellman recursion, monotonicity in stage/terminal costs, and zero-cost and zero-discount boundaries. This is a genuine dynamic-programming contract, but without a shared reward/preference transformation and stochastic policy model it still does **not** prove that Active Inference contains finite-horizon MDPs.

5.2.3 Perception vs Action in the Catalogue

The rows used in this section can be grouped heuristically along a perception/action interface; fep-017 and fep-034 are Bayesian-Mechanics rows included here as cross-area dependencies:

- **Perception-side topics**: fep-007 and fep-047 provide normalized local messages and compositional finite propagation; fep-017 and fep-034 provide native posterior and transition–observation filter kernels.
- **Action-side topics**: fep-003 (discounted pragmatic cost), fep-008 (finite minimizer), fep-012/fep-028 (entropy-regularized and softmax policies), fep-021 (explicit EFE convention), fep-023 (policy-indexed reachable laws), fep-033 (Bellman recursion), fep-041 (native KL information gain), fep-065–071 (controlled kernels, belief updates, soft control, and sophisticated EFE), and fep-128–134 (finite policy-tree recursion, optimality, open-loop embedding, treewise EFE, and strict feedback).
- **Collective topics**: fep-107–113 cover independent product-agent laws, additive VFE/EFE, unit-weight product-of-experts pooling, consensus mass conservation and contraction, and coupled-potential descent under their explicit independence, overlap, and fixed-mixing premises.
- **Optimization bridge**: fep-032 and fep-043 prove exact quadratic gradient dynamics and criticality; fep-048 supplies a general contraction contract and convergence witness.

5.2.3.1 Belief Propagation on Factor Graphs (fep-007)

The sequential generative model of Section 5.2.1 admits a canonical factor-graph representation: variable nodes for each s_t and o_t, factor nodes for the prior $p(s_1)$, each transition $p(s_{t+1} \mid s_t, a_t)$, and each emission $p(o_t \mid s_t)$. The *sum–product* (belief propagation) algorithm computes the marginal $q(s_t) \approx p(s_t \mid o_{1:T})$ by local message passing on this graph: forward messages $\alpha_t(s_t) = \sum_{s_{t-1}} p(s_t \mid s_{t-1}, a_{t-1})\,\alpha_{t-1}(s_{t-1})\,p(o_{t-1} \mid s_{t-1})$ and backward messages $\beta_t(s_t)$ combine into the posterior $q(s_t) \propto \alpha_t(s_t)\,\beta_t(s_t)$. In the linear-Gaussian case this reduces exactly to the Kalman filter/smoother.

fep-007 retains the local product/order lemmas and adds a support-aware normalized message. Strictly positive factors and incoming values over a nonempty support give a positive normalizer, pointwise nonnegative output, and support mass exactly one. Exactness on a tree and convergence on a loopy graph still require a formal graph and update schedule rather than follow from local normalization.

5.2.3.2 Categorical Belief Updates (fep-034)

For discrete state and observation spaces, the single-step Bayesian update has the canonical form

$$q(s \mid o) \;\propto\; p(o \mid s)\,q(s), \qquad q(s \mid o) \;=\; \frac{p(o \mid s)\,q(s)}{\sum_{s'} p(o \mid s')\,q(s')}, \tag{34}$$

where the denominator is the evidence. **fep-034** now defines the transition-predicted prior as kernel–measure composition and applies Mathlib's posterior construction to the observation kernel. It proves predictive and posterior mass one, reconstruction of the transition-prediction/observation joint law, and recovery of the predicted prior. The composition theorem identifies it definitionally with fep-017's posterior at that predicted prior.

The normalization assumptions are carried by Mathlib's finite/Markov-kernel typeclasses rather than by a manually divided finite vector. This supplies an exact one-step Bayesian filter, not a numerical filtering algorithm or approximation theorem. The maintained finite kernel separately composes policy transitions into an open-loop rollout, accumulates EFE against each successive predicted state law, and reconstructs a terminal posterior at positive evidence. The later `PolicyTree` carrier conditions its deterministic continuation choice on each finite observation branch, but it remains a separate finite model and does not identify every intermediate tree belief with this native posterior kernel.

5.2.4 Policies, Optimality, and Affordances

5.2.4.1 Affordances and Reachability (fep-023) The term *affordance*, borrowed from ecological psychology (Gibson 1979), has a precise formal meaning within Active Inference: the set of outcomes that an agent can render accessible through some choice of policy. Given a family of admissible policies Π and a predictive distribution $q(\cdot \mid \pi)$ over outcomes for each $\pi \in \Pi$, the *affordance set* is

$$\mathcal{A}(\Pi, q) \;=\; \{\, o : \exists \pi \in \Pi,\; q(o \mid \pi) > 0 \,\} \;=\; \bigcup_{\pi \in \Pi} \operatorname{supp}(q(\cdot \mid \pi)). \tag{35}$$

fep-023 encodes reachability as the set of probability laws generated by an allowed finite policy set. It proves membership for an allowed policy, monotonicity under policy-set inclusion, and transfer of probability normalization to every reachable law. It does not define outcome support, optimize over the reachable-law set, or connect that set to fep-041's policy-conditioned information gain, so curiosity and niche-construction interpretations remain outside the row.

5.2.4.2 Optimal Policy Existence (fep-008) Policy selection reduces to minimization of G over the finite policy set. **fep-008** certifies that such a minimizer exists and that all minimizers share a common EFE value. The proof invokes `Finset.exists_min_image` to obtain an explicit minimizer $\pi^\star$ with $G(\pi^\star) \le G(\pi)$ for every $\pi \in \Pi$, and then `min_agrees_on_value` plus `le_antisymm` to show that any two minimizers $\pi_1^\star, \pi_2^\star$ satisfy $G(\pi_1^\star) = G(\pi_2^\star)$. The discrete, finite setting matches real Active-Inference implementations that enumerate a finite policy horizon; fep-008 is thus the small but essential existence theorem that downstream results (commitment to a specific action, deterministic policy extraction, value iteration on the EFE lattice) rely upon.

5.2.4.3 Mathlib Footprint and Verification Status

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-003	Discounted `ENNReal` pragmatic cost with exact horizon increment	`formalized`	`Data.ENNReal.Inv`	0
fep-007	Positive, support-normalized finite sum-product message	`formalized`	`Algebra.BigOperators`	0
fep-008	Finite nonempty-set minimizer existence and value agreement	`formalized`	`Data.Finset`	0
fep-020	Normalized two-state transition, stationarity, exact iterates, convergence	`formalized`	`Analysis.SpecificLimits.Normed`	0
fep-021	Explicit `ENNReal` EFE convention with balance and order laws	`formalized`	`Data.ENNReal.Inv`	0
fep-023	Policy-indexed reachable probability laws and normalization transfer	`formalized`	`MeasureTheory.Measure.Typeclasses.Probability`	0

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-028	Support-aware full finite softmax probability law	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-033	Deterministic transition-aware Bellman recursion	`formalized`	`Data.ENNReal.Inv`	0
fep-034	Native normalized transition–observation posterior filter	`formalized`	`Probability.Kernel.Posterior`	0
fep-041	Native measure-KL information gain and zero-expectation law	`formalized`	`InformationTheory.KullbackLeibler.Basic`	0
fep-047	`Matrix.mulVec` sum-product propagation with exact composition	`formalized`	`Data.Matrix.Mul`	0

Representative formalization — *Expected Free Energy (fep-003, Eq. 14)*: On the maintained finite policy-conditioned carrier, EFE decomposes into pragmatic cost minus epistemic value:

$$\mathrm{G}(\pi) = \underbrace{\mathrm{CE}(q_\pi(o), p_C(o))}_{\text{pragmatic cost}} - \underbrace{I_{q_\pi}(S;O)}_{\text{epistemic value}}.$$

The production fep-003 sketch defines finite-horizon discounted pragmatic cost directly in `ENNReal`, proves its exact successor-horizon increment, and establishes monotonicity in both stage costs and horizon. fep-021 then combines that cost with an epistemic value under the package's explicit sign convention, and the composed theorem checks the balance. Those topic rows do not derive the probabilistic expression above; the maintained finite carrier does so separately for a real-valued one-step policy-conditioned joint under full support. See §5.10.3, §13.3 in Appendix B, and §14.3 in Appendix~14.

Representative formalization — *Optimal Policy Existence (fep-008)*: On a nonempty finite policy set, EFE achieves its minimum. The sketch invokes `Finset.exists_min_image` (producing a certified minimizer) and then proves that any two minimizers agree on G via `le_antisymm`. This turns the *soft* statement "some policy is best under G" into a compiler-verifiable existence theorem — a small but important piece of machinery for downstream results where the agent commits to a specific action. Note the discrete setting matches real active-inference implementations that enumerate a finite policy horizon. Typeset statements: §14.8 in Appendix~14; Lean: §13.8 in Appendix B.

Representative formalization — *Affordances as Reachable Laws (fep-023)*: The sketch maps policies to probability measures and defines the reachable set as the image of an allowed finite policy set. It proves set monotonicity and that every reachable law inherits probability normalization. Inclusion is weak: added policies may induce an already-reachable law, so strict growth is not claimed. See §14.23 in Appendix~14 and §13.23 in Appendix B.

Depth landmarks. fep-028 is a complete finite softmax law; fep-008 is a finite-set minimizer theorem; fep-021 fixes an explicit `ENNReal` EFE convention; fep-034 is a normalized posterior filter; and fep-041 uses native measure KL. The finite kernel supplies their missing shared model at a distinct real-valued finite layer and adds stage-updated cumulative EFE. fep-071 adds exact observation-dependent reoptimization; fep-128–134 add arbitrary finite-depth policy trees, optimality, open-loop dominance, treewise EFE, and a strict feedback witness; and fep-107–113 add independent-product and fixed two-agent collective laws. Remaining limitations include the explicit real/`ENNReal` bridge, measure-level equivalence, probabilistic selection or learning over policy trees, infinite or continuous carriers, arbitrary network topologies, and refinement to executable agents. Syntactic `mathlib_status`, semantic disposition, and current native evidence are reported separately in generated coverage and receipts.

Mathlib4 module footprint (Active Inference). Finite aggregation routes through `Algebra.BigOperators.Group.Finset`, while finite policy indices and images use `Data.Fin` and `Data.Finset`. The generated coverage report owns the exact topic-to-import incidence table. Conceptually, these modules supply the finite sums, support sets, minimizer existence, and image constructions on which the discrete policy layer depends; they do not by themselves establish a probabilistic EFE interpretation.

5.2.5 What the Active Inference Theorems Collectively Establish

The 31 rows can be grouped by intended role. At topic level they remain local facts; the maintained finite kernel composes their central probabilistic counterparts into one bounded agent model:

1. **EFE inputs and convention** — fep-003 provides discounted pragmatic cost, fep-041 provides native KL information gain, and fep-021 composes them under one explicit `ENNReal` sign convention.
2. **Inference operators** — fep-007 provides normalized local messages, fep-047 exact finite operator composition, and fep-017/fep-034 native posterior/filter kernels.
3. **Policy selection** — fep-008 and fep-028 cover deterministic finite argmin existence and stochastic softmax normalization; fep-012 adds the finite Shannon entropy regularizer.
4. **Multi-step planning** — fep-033 proves an exact deterministic Bellman recursion and boundary laws in `ENNReal`; the finite kernel composes policy-conditioned transitions, updates the predicted state law stage by stage, accumulates real-valued EFE, and proves prefix–suffix decomposition and support-qualified nonnegativity.
5. **Information value** — fep-041 proves KL nonnegativity, finite-measure separation, and a zero expected-information boundary; the finite kernel defines policy-conditioned mutual information, while equivalence to the native posterior-averaged KL remains bridge work.
6. **Optimization dynamics** — fep-032/fep-043 provide exact stable quadratic gradient dynamics, while fep-048 packages a global contraction interface.
7. **Stochastic dynamics** — fep-020 gives a normalized two-state Markov evolution with stationarity, exact iterates, and convergence.
8. **Distributional reachability** — fep-023 makes policies reach probability laws and transfers normalization, without yet supplying embodiment or an optimization theorem over that set.
9. **Controlled and sophisticated planning** — fep-065–071 add controlled-kernel normalization, positive-evidence belief updates, reachable finite-belief value preservation, soft Bellman/desirability recursion, a control posterior, and a strict two-stage feedback advantage; fep-128–134 extend this to arbitrary finite-depth observation-contingent trees, prove optimal-tree existence and open-loop dominance, and preserve an explicit strict-feedback boundary.
10. **Collective inference** — fep-107–113 prove independent-agent VFE/EFE additivity, positive-normalizer unit-weight product-of-experts pooling, mass conservation, fixed-matrix consensus contraction, and coupled-potential descent without inferring emergent agency.

No single theorem in this list proves Active Inference as a universal theory, and the package is not a certificate for a particular discrete agent or MDP implementation. It provides normalized updates, native posterior kernels, a shared finite policy-conditioned generative model, posterior VFE, both central EFE decompositions, prior-weighted policy/action selection, state-updating cumulative open-loop EFE, native information gain, controlled finite-belief recursion, arbitrary finite-depth observation-contingent policy trees with deterministic optimality, a strict feedback witness, and scoped collective laws. The remaining high-level obligations are distributions and learning over policy trees, infinite-horizon or continuous-carrier planning, comparison across alternative EFE conventions, a finite-to-measure conditional-information bridge, arbitrary multiagent interaction structures, and a refinement relation to executable code.

All 31 rows carry **`mathlib_status: real`**, a source classification that is distinct from semantic maturity. The current native area rate, when backed by a claim-ready native receipt, is **`31/31 (100.0%)`**. Reproduce it with `uv run fep-lean verify --fail-on-warnings --receipt output/native-verification.json`; full Hermes/OpenGauss evidence remains a separate contract.

5.3 Sophisticated Dynamics: Information Geometry and Bayesian Mechanics

Two catalogue areas target especially demanding mathematics: **Information Geometry (21 topics)** and **Bayesian Mechanics (41 topics)**—together 62 of the 155 bodies. Their motivating theories involve statistical manifolds (Amari 1983, 2016), stochastic dynamics, and nonequilibrium steady states (K. Friston 2019). The catalogue constructs Bernoulli and categorical Fisher geometry, Cramér–Rao and geometric-optimization laws, a finite scalar exponential family with log-partition and KL–Bregman identities, native measure and finite Bayesian inversion, temporal smoothing, causal interventions, finite blankets with a native `CondIndepFun` transfer, hierarchical kernels, and a divergence-free nonequilibrium current. The package still stops short of arbitrary smooth connections and curvature, generic blanket existence, general causal identification, and continuous-state SDE/PDE steady-state theory. The semantic audit records that boundary theorem by theorem. See §4.17 and the generated formalism coverage map.

5.3.1 Langevin Dynamics and the Fokker–Planck Equation

Beyond the static-bound formulation of the FEP, adaptive systems are naturally described by *stochastic* dynamics on the variational parameters. The canonical continuous-time object is the overdamped **Langevin stochastic differential equation**:

$$\dot{x}(t) \;=\; -\nabla F(x(t)) \;+\; \sqrt{2D}\,\xi(t), \qquad \langle \xi(t) \rangle = 0, \;\; \langle \xi(t)\,\xi(t')^{\top} \rangle = \mathbb{I}\,\delta(t-t'), \tag{36}$$

where $F : \mathbb{R}^n \to \mathbb{R}$ is the free-energy functional, $D \geq 0$ is the diffusion coefficient, and ξ is standard Gaussian white noise. Equation 36 expresses the principle that adaptive parameters *drift* along the free-energy gradient while continuously exploring a neighborhood of the current state—reconciling deterministic gradient flow (§5.3.2) with Bayesian posterior sampling.

The probability density $\rho(x,t)$ of trajectories solving Equation 36 evolves according to the **Fokker–Planck equation**:

$$\partial_t \rho(x,t) \;=\; \nabla \cdot (\rho\, \nabla F) \;+\; D\, \nabla^2 \rho \;=\; -\nabla \cdot J(x,t), \tag{37}$$

with probability current $J = -\rho\, \nabla F - D\, \nabla \rho$. Under normalizability, boundary, regularity, and constant positive-diffusion assumptions, a zero-current stationary solution has Gibbs form $\rho^\star(x) \propto \exp(-F(x)/D)$. This links the chosen potential to a Boltzmann weight; it does not by itself identify that potential with variational free energy (§5.4).

5.3.2 Gradient Flow in Measure Space

A Wasserstein-gradient-flow formulation reinterprets Equation 37 as descent of a free-energy functional on the space of probability measures:

$$\partial_t \rho \;=\; \nabla \cdot (\rho\, \nabla \tfrac{\delta F}{\delta \rho}), \qquad F[\rho] \;=\; \mathbb{E}_\rho[U(x)] \;+\; D\, \mathbb{E}_\rho[\log \rho(x)], \tag{38}$$

where $\delta F / \delta \rho$ is the L^2 functional derivative. Equation 38 elevates the FEP from a bound on scalars to a *geometric flow on a manifold of distributions*, with the Wasserstein-2 metric playing the role of Riemannian structure. Topic fep-038 constructs the one-parameter Bernoulli Fisher metric and its inverse-metric natural gradient, while topic fep-018 constructs the corresponding variance-stabilizing coordinate distance. Those finite one-dimensional results are exact; they are not a Wasserstein gradient-flow theorem.

5.3.3 Ergodicity, Invariant Measures, and Mathlib4

Under an appropriate combination of existence, uniqueness, recurrence, invariant-measure, and integrability hypotheses, a Langevin process may be **ergodic**: time averages then converge to ensemble averages under a stationary measure $\rho^\star$:

$$\lim_{T\to\infty} \frac{1}{T} \int_0^T \phi(x(t))\, dt \;=\; \int \phi(x)\, \rho^\star(x)\, dx \quad \text{a.s.}, \tag{39}$$

for observables satisfying the theorem's integrability conditions. Equation 39 motivates a bridge between single trajectories and ensemble statistics, but no catalogue row defines the continuous-state stochastic process or almost-sure time-average limit needed to prove it. The current Lean sources instead formalize discrete-time measurable semigroups (fep-006), invariant Markov kernels (fep-010), exact discrete two-state relaxation and autocorrelation decay (fep-020 and fep-037), convergent quadratic descent (fep-032), and an exact continuous-time two-state Markov semigroup with master equation and exponential relaxation (fep-149–155). These finite-state results do not imply Langevin ergodicity or a pathwise ergodic theorem.

5.3.4 Finite Markov Relaxation, Not Langevin (fep-020)

Topic fep-020 defines the normalized symmetric Boolean transition that flips state with probability α and stays put with probability $1-\alpha$. For $0 \le \alpha \le 1$, every transition mass is nonnegative and each row sums to one. If p is the current mass on `true`, one step is the affine map

$$T_\alpha(p) = \alpha + (1-2\alpha)p, \qquad T_\alpha^{;n}(p) - \tfrac{1}{2} = (1-2\alpha)^n (p - \tfrac{1}{2}). \tag{40}$$

Lean proves that $1/2$ is stationary and that $T_\alpha^{;n}(p) \to 1/2$ whenever $0 < \alpha < 1$. The strict interior excludes the alternating endpoint $\alpha = 1$ and the frozen endpoint $\alpha = 0$. This is an exact finite-state distributional convergence theorem, but not a Langevin discretization. The later fep-149–155 family adds a different exact two-state carrier in continuous time, including a semigroup, master equation, detailed balance, relaxation, and strict Lyapunov witness. Neither carrier defines Gaussian noise, a diffusion limit, a continuous-state SDE, or a Fokker–Planck PDE.

5.3.5 Information Geometry Results (21 topics)

Information geometry treats parametric families of probability distributions $\mathcal{M} = \{p_\theta\}_{\theta \in \Theta}$ as Riemannian manifolds, with the Fisher information tensor as the canonical metric. The 21 catalogue rows build an exact Bernoulli family deeply, extend it with a categorical score carrier, and add a full-support finite scalar exponential family. The categorical expansion proves Fisher positivity on simplex tangents with explicit full-rank and null directions, pullback, an unbiased scalar Cramér–Rao bound under score regularity, natural-gradient equivariance under an invertible full-rank chart, a mirror-descent three-point identity, an affine-projection Bregman Pythagorean law, and replicator–natural-gradient equivalence. The scalar exponential-family expansion proves normalization, affine log-density ratios, first and second log-partition derivatives, centered scores, Fisher–variance equality, KL–Bregman duality, and interval-local mean-coordinate injection. These finite laws complement native KL, Hellinger, and Bernoulli Fisher–Rao results (Amari 1983). Arbitrary smooth coordinate atlases, multidimensional dual connections, curvature, and general geodesics remain outside scope.

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-004	Finite diagonal Fisher metric: symmetry, positive semidefiniteness, and positive definiteness	`formalized`	`Algebra.Order.BigOperators.Group.Finset`	0
fep-014	Native KL nonnegativity, self-zero, zero characterization, and composition-product chain rule	`formalized`	`InformationTheory.KullbackLeibler.ChainRule`	0
fep-017	Native posterior kernel: normalized fibers, joint reconstruction, prior recovery, and Bayes density	`formalized`	`Probability.Kernel.Posterior`	0
fep-018	Bernoulli Fisher–Rao coordinate distance, symmetry, triangle inequality, and separation	`formalized`	`Analysis.SpecialFunctions.Trigonometric.Inverse`	0
fep-024	Native KL-regularized objective and its exact zero-weight and self-prior laws	`formalized`	`InformationTheory.KullbackLeibler.Basic`	0
fep-029	Quadratic Bregman definition, squared-distance identity, nonnegativity, and point separation	`formalized`	`Analysis.Convex.Basic`	0
fep-038	Bernoulli score, Fisher information and metric, natural gradient, and coordinate pullback	`formalized`	`Analysis.Calculus.Deriv.Basic`	0
fep-044	Bernoulli squared Hellinger divergence: nonnegativity, symmetry, separation, and relabeling invariance	`formalized`	`Data.Real.Sqrt`	0
fep-100	Categorical Fisher positivity on simplex tangents with rank/null witnesses	`conditional_proxy`	`LinearAlgebra.Matrix.Notation`	0
fep-101	Fisher pullback under a finite reparameterization	`formalized`	`Data.Matrix.Mul`	0

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-102	Unbiased scalar Cramér–Rao under score regularity and positive Fisher information	`formalized`	`Analysis.InnerProductSpace.Basic`	0
fep-103	Natural-gradient equivariance under an invertible full-rank chart	`formalized`	`LinearAlgebra.Matrix.NonsingularInverse`	0
fep-104	Mirror-descent three-point identity	`formalized`	`Analysis.Convex.Basic`	0
fep-105	Bregman Pythagorean law for an affine information projection	`conditional_proxy`	`LinearAlgebra.AffineSpace.AffineSubspace.Basic`	0
fep-106	Replicator–natural-gradient equivalence on the finite simplex	`conditional_proxy`	`LinearAlgebra.Matrix.Notation`	0
fep-142	Full-support finite scalar exponential-family normalization	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-143	Affine log-density ratio within one supported exponential family	`formalized`	`Analysis.SpecialFunctions.Log.Basic`	0
fep-144	Log-partition derivative equals the sufficient-statistic mean	`formalized`	`Analysis.SpecialFunctions.ExpDeriv`	0
fep-145	Centered scalar score with zero model expectation	`formalized`	`Algebra.Order.BigOperators.Group.Finset`	0
fep-146	Log-partition Hessian and Fisher information equal variance, with rank and zero boundaries	`formalized`	`Analysis.Calculus.Deriv.Basic`	0
fep-147	Exponential-family KL equals the log-partition Bregman divergence	`formalized`	`InformationTheory.KullbackLeibler.Basic`	0
fep-148	Positive-variance mean coordinate is strictly monotone and injective on an interval	`formalized`	`Analysis.Calculus.Deriv.Monotone`	0

5.3.5.1 Fisher Information Metric (fep-004) On a statistical manifold $\mathcal{M} = \{p_\theta\}_{\theta \in \Theta}$ with $\Theta \subset \mathbb{R}^n$ open, the **Fisher information metric** is the Riemannian metric whose components in the coordinate basis are given by the expected outer product of the score function $s_i(x;\theta) := \partial_{\theta_i} \log p_\theta(x)$:

$$g_{ij}(\theta) \;=\; \mathbb{E}_{p_\theta}\!\left[\frac{\partial \log p_\theta}{\partial \theta_i}\cdot\frac{\partial \log p_\theta}{\partial \theta_j}\right] \;=\; -\,\mathbb{E}_{p_\theta}\!\left[\frac{\partial^2 \log p_\theta}{\partial \theta_i \partial \theta_j}\right], \tag{41}$$

where the second equality uses the identity $\mathbb{E}_{p_\theta}[\partial_i s_j] + \mathbb{E}_{p_\theta}[s_i s_j] = 0$ that follows from differentiating the normalization $\int p_\theta \, d\mu = 1$ twice under the integral. The matrix $I(\theta) = [g_{ij}(\theta)]$ is the **Fisher information matrix** and, under regularity conditions that exchange differentiation and integration, is symmetric and positive semidefinite.

The Fisher metric is distinguished among Riemannian metrics on $\mathcal{M}$ by Chentsov's uniqueness theorem: up to scalar, it is the only metric invariant under sufficient statistics (Markov embeddings). Geometrically, it captures how distinguishable two nearby distributions p_θ and $p_{\theta+d\theta}$ are: the squared infinitesimal KL divergence is

$$2\, D_{\mathrm{KL}}(p_\theta \,\|\, p_{\theta+d\theta}) \;=\; g_{ij}(\theta)\, d\theta^i\, d\theta^j + O(\|d\theta\|^3), \tag{42}$$

so the Fisher metric *is* the Hessian of the KL divergence at coincident parameters — connecting fep-004 directly to fep-014 and fep-024. The **Cramér–Rao bound** states that for any unbiased estimator T of θ, $\mathrm{Var}_{p_\theta}[T] \succeq I(\theta)^{-1}$ in the positive-semidefinite order; no unbiased estimator can resolve parameters more finely than the Fisher geometry permits. In the FEP context, $I(\theta)$ plays the role of a **precision matrix** — the natural metric for measuring how "far" two nearby beliefs are from each other, and the object weighted by attention / synaptic-gain parameters in predictive coding.

The fep-004 row defines a finite diagonal Fisher metric with explicitly supplied information weights and proves symmetry, positive semidefiniteness for nonnegative weights, and positive definiteness for strictly positive weights. The authored composition theorem `fep004_bernoulliMetric_specialization` then shows that its one-coordinate specialization is definitionally the Fisher metric computed in fep-038. fep-100 derives the metric from categorical scores and exposes both full-rank and null directions. fep-102 adds the scalar Cramér–Rao inequality under explicit unbiasedness, score regularity, finite sums, and positive Fisher information; it is not the unrestricted matrix bound quoted above. None of these rows constructs a general smooth manifold or justifies differentiation under an arbitrary integral. See §13.4 in Appendix B; typeset signatures in §14.4 of Appendix~14.

5.3.5.2 Natural Gradient (fep-038) The ordinary Euclidean gradient $\nabla_\theta F$ of a loss $F : \Theta \to \mathbb{R}$ is *not* coordinate-invariant on a statistical manifold: reparameterizing $\theta \mapsto \phi(\theta)$ produces an update that depends on the Jacobian of ϕ, so two modelers using different parameterizations of the same family will take genuinely different steps. Amari (Amari 1998) resolved this by defining the **natural gradient** as the steepest-descent direction with respect to the Fisher-induced Riemannian metric:

$$\tilde{\nabla}_\theta F(\theta) \;=\; I(\theta)^{-1}\, \nabla_\theta F(\theta). \tag{43}$$

Geometrically, $\tilde{\nabla}_\theta F$ is the unique vector such that $\langle \tilde{\nabla}_\theta F, v\rangle_{I(\theta)} = dF(\theta)[v]$ for all tangent v, where $\langle u, v\rangle_I := u^\top I v$ is the Fisher inner product. The natural gradient update

$$\theta_{t+1} \;=\; \theta_t \;-\; \eta\, I(\theta_t)^{-1}\, \nabla_\theta F(\theta_t) \tag{44}$$

is coordinate-covariant under the regularity and invertibility assumptions that make the Fisher metric and transformation law well-defined. Its conditioning and statistical-efficiency properties are model- and algorithm-dependent rather than unconditional convergence guarantees. In Active-Inference accounts, $I(\theta_t)^{-1}$ is interpreted as a precision-weighted gain; fep-038 formalizes the inverse-metric update for the Bernoulli family but not that neurobiological interpretation.

The fep-038 row defines the normalized Bernoulli mass, its pointwise parameter derivative, the score, Fisher information as expected squared score, the scalar Fisher metric, and the inverse-metric natural gradient. Lean proves expected-score zero, the closed form $I(p) = 1/[p(1-p)]$ on the interior, positive definiteness, metric–natural-gradient duality, and the pullback law through the Fisher–Rao coordinate Jacobian. The maintained foundations generalize the score field to arbitrary finite dimension, prove matrix symmetry, Gram PSD, conditional PD, functorial and positive pullbacks, equivalence of expected-score and matrix-lowered forms, and bilinearity. Under invertibility they construct the unique inverse-Fisher natural gradient and prove metric duality and energy. fep-103 then proves equivariance through an explicitly invertible full-rank chart, and fep-106 identifies the corresponding categorical natural gradient with replicator dynamics on the simplex. Arbitrary smooth atlases, connections, and curvature remain successor work.

5.3.5.3 KL Divergence (fep-014) and the I- / M-Projection Asymmetry The **Kullback–Leibler divergence** between probability measures $q \ll p$ on a measurable space $(X, \mathcal{F})$ is

$$D_{\mathrm{KL}}(q \,\|\, p) \;=\; \int_X q \,\log\frac{q}{p}\, d\mu \;=\; \mathbb{E}_q\!\left[\log\frac{dq}{dp}\right]. \tag{45}$$

Three structural properties lift it from a mere functional to the canonical statistical discrepancy:

1. **Nonnegativity** (Gibbs' inequality): $D_{\mathrm{KL}}(q \,\|\, p) \geq 0$ with equality iff $q = p$ a.e. This follows from Jensen's inequality applied to the concave function log: $\mathbb{E}_q[\log(p/q)] \leq \log \mathbb{E}_q[p/q] = 0$.
2. **Chain rule**: a joint or kernel-composed divergence decomposes into a marginal term and a conditional term under appropriate finiteness and kernel assumptions.
3. **Data-processing inequality (DPI)**: for a measurable transformation or channel, postprocessing cannot increase KL. This is distinct from the chain rule and is not formalized by the pinned fep-014 row.
4. **Asymmetry**: in general $D_{\mathrm{KL}}(q \,\|\, p) \neq D_{\mathrm{KL}}(p \,\|\, q)$. This asymmetry is not a defect but carries the **mode-covering / mode-seeking** distinction that governs variational inference:
   - Minimizing the "reverse" direction $D_{\mathrm{KL}}(q\|p)$ over a restricted approximation family is often described as mode-seeking, but the actual optimizer depends on the family and support constraints.
   - Minimizing the "forward" direction $D_{\mathrm{KL}}(p\|q)$ is often mass-covering or moment-matching in common exponential-family settings; this too is not a distribution-free theorem.

The common variational form uses $D_{\mathrm{KL}}(q(s) \,\|\, p(s \mid o))$, but mode-seeking behavior depends on the approximation family and optimization setting and is not itself a theorem in this catalogue. fep-014 directly uses Mathlib's measure-level `InformationTheory.klDiv`. It proves codomain nonnegativity, self-zero under `SigmaFinite`, equality characterization for finite measures, and `klDiv_compProd_eq_add` for finite measures and Markov kernels. The maintained finite information layer separately proves a joint/channel chain rule, independent-product additivity, and the induced prior-marginal bound. Neither layer proves a generic DPI or the variational projection behavior described above.

5.3.5.4 Rényi / Tsallis α-Divergences (fep-044) The one-parameter **Chernoff α-divergence family** interpolates between forward and reverse KL and recovers Hellinger distance at its midpoint:

$$D_\alpha(p \,\|\, q) \;=\; \frac{1}{\alpha(1-\alpha)} \left[1 - \int p^\alpha \, q^{1-\alpha} \, d\mu\right], \qquad \alpha \in (0,1). \tag{46}$$

Endpoint limits and distinguished values:

α	Limit	Behavior
$\alpha \to 1$	$D_{\mathrm{KL}}(p \,\|\, q)$	M-projection, mass-covering
$\alpha \to 0$	$D_{\mathrm{KL}}(q \,\|\, p)$	I-projection, mode-seeking
$\alpha = 1/2$	$4\,H^2(p,q)$	Symmetric Hellinger distance squared

The family sits within the broader class of Csiszár f-divergences. The fep-044 row selects the distinguished Bernoulli Hellinger instance rather than pretending to cover the entire α family. It defines squared Hellinger divergence with the conventional factor $1/2$ and proves nonnegativity, symmetry, zero iff equal parameters on $[0,1]$, and invariance under exchanging success with failure. Measure-level α-divergence, endpoint limits, and general density integration remain separate extensions.

5.3.5.5 Bregman Divergences and Mirror Descent (fep-029) For a strictly convex, differentiable **potential** $\phi : \mathcal{C} \to \mathbb{R}$ on a convex domain $\mathcal{C} \subseteq \mathbb{R}^n$, the **Bregman divergence** is the gap between ϕ and its linear approximation at q:

$$B_\phi(p,q) \;=\; \phi(p) \;-\; \phi(q) \;-\; \langle \nabla\phi(q),\, p - q\rangle. \tag{47}$$

Key properties: $B_\phi(p,q) \geq 0$ with equality iff $p = q$; B_ϕ is convex in its first argument; and it obeys the **generalized Pythagorean theorem** $B_\phi(p,r) = B_\phi(p,q) + B_\phi(q,r) + \langle \nabla\phi(r) - \nabla\phi(q),\, q - p\rangle$, which reduces to the familiar identity when q is the Bregman projection of p onto a convex set. Distinguished instances:

- $\phi(p) = \frac{1}{2}\|p\|^2$ recovers **squared Euclidean distance** $B_\phi(p,q) = \frac{1}{2}\|p - q\|^2$.
- $\phi(p) = \sum_i p_i \log p_i$ (negative Shannon entropy) on the probability simplex recovers the **KL divergence** $B_\phi(p,q) = D_{\mathrm{KL}}(p \,\|\, q)$ — positioning KL as one instance of a general convex-analytic family.
- $\phi(p) = -\log\det(P)$ on positive-definite matrices recovers the **LogDet / Burg divergence** used in covariance estimation.

Mirror descent is gradient descent in the dual geometry induced by ϕ: $\nabla\phi(\theta_{t+1}) = \nabla\phi(\theta_t) - \eta\nabla F(\theta_t)$, with an equivalent proximal form under the usual convexity and regularity conditions. fep-029 defines the one-dimensional quadratic Bregman instance and proves nonnegativity and separation. fep-104 adds the exact mirror-descent three-point identity on its maintained finite inner-product carrier. fep-105 adds an affine-projection Bregman Pythagorean equality under explicit membership, orthogonality, and minimizing premises. These are algebraic and affine finite laws, not a general convergence theorem for mirror descent or existence theorem for projections onto arbitrary convex sets.

5.3.5.6 KL Regularization (fep-024) and Bayesian Update Geometry (fep-017)

Topic fep-024 isolates the elementary log-identities that license variational bounds: the log-ratio decomposition $\log(p/q) = \log p - \log q$, monotonicity of log on $(0, \infty)$, and the anchor $D_{\mathrm{KL}}(p \,\|\, p) = 0$ via $\log 1 = 0$. These three facts are the Lean-level primitives behind the ELBO decomposition $\log p(o) = \mathbb{E}_q[\log p(o, s)] - \mathbb{E}_q[\log q(s)] + D_{\mathrm{KL}}(q \,\|\, p(\cdot \mid o))$ and thus behind every variational FEP bound in the catalogue.

Topic fep-017 now wraps Mathlib's native posterior kernel. Lean proves every posterior fiber has mass one, reconstructs the swapped likelihood–prior joint from predictive mass followed by the posterior, recovers the prior after a Markov likelihood and its posterior, and states the countable-space Radon–Nikodym Bayes-density formula almost everywhere under the predictive law. The row therefore formalizes normalized Bayesian inversion at kernel level. Its remaining geometric extension is to prove a projection characterization for a selected divergence and approximation family.

5.3.5.7 Statistical Manifold Geodesics and Dual Connections (fep-018)

Amari's information geometry equips $\mathcal{M}$ not with a single Levi–Civita connection but with a one-parameter **α-connection family** $\nabla^{(\alpha)}$, with two distinguished members forming a **dually flat pair** $(\nabla^{(+1)}, \nabla^{(-1)})$:

- The **exponential (e-) connection** $\nabla^{(+1)}$: its geodesics are log-linear interpolations $\log p_t = (1-t)\log p_0 + t \log p_1 + \text{const}$. In exponential families these become straight lines in natural parameters.
- The **mixture (m-) connection** $\nabla^{(-1)}$: its geodesics are convex mixtures $p_t = (1-t)p_0 + tp_1$, i.e. straight lines in the probability simplex.

Dual connections yield Pythagorean identities under the relevant flatness and projection hypotheses and help explain convergence proofs for particular information-geometric algorithms. The fep-018 row defines the Bernoulli variance-stabilizing coordinate $2 \arcsin \sqrt{p}$ and the absolute coordinate difference. It proves nonnegativity, symmetry, the triangle inequality, and separation on the unit interval, giving the exact Fisher–Rao distance for this one-dimensional family. fep-105 supplies a finite affine Bregman Pythagorean law, but neither row defines α-connections or geodesic equations on a general manifold, nor identifies its projection with unrestricted measure-valued KL geometry.

Representative formalization — *Bernoulli Fisher Geometry (fep-038 with fep-004 and fep-018)*: The family mass differentiates exactly, the expected score vanishes, its Fisher information is $1/[p(1-p)]$, the metric is positive on nonzero tangents, the natural gradient is inverse-metric dual, and the Fisher–Rao coordinate pulls the metric back to Euclidean form. Two authored composition theorems connect the generic finite metric and the distance-separation law to this same family. See §13.38 in Appendix B and §14.38 in Appendix~14.

5.3.6 Bayesian Mechanics Results (41 topics)

Bayesian mechanics rests on a particular partition of system states. A **Markov blanket partition** of a finite state space $\mathcal{S}$ is a decomposition into four mutually disjoint blocks $\mathcal{S} = \mathcal{I} \sqcup \mathcal{B}_s \sqcup \mathcal{B}_a \sqcup \mathcal{E}$ — **internal** ($\mathcal{I}$), **sensory blanket** ($\mathcal{B}_s$), **active blanket** ($\mathcal{B}_a$), and **external** ($\mathcal{E}$) — such that internal and external states are conditionally independent given the blanket $\mathcal{B} = \mathcal{B}_s \cup \mathcal{B}_a$:

$$p(\mu, \eta \mid b) \;=\; p(\mu \mid b)\, p(\eta \mid b), \qquad \mu \in \mathcal{I},\ \eta \in \mathcal{E},\ b \in \mathcal{B}. \tag{48}$$

Equation 48 states statistical conditional independence for a specified law. It does not by itself entail causal arrows, sensory/active directionality, or an inference interpretation. Active-Inference graphical and dynamical models add such directional assumptions through their transition structure; NESS treatments add further drift, diffusion, regularity, and stationarity hypotheses. Keeping those additions separate is essential to the blanket critique.

Topic **fep-005** defines four `Finset.filter` blocks from an arbitrary assignment `Fin 20 → Fin 4`, proves disjointness for unequal labels, characterizes membership, and proves that every state belongs to exactly one block. Topic **fep-009** uses Mathlib's conditional-expectation-based `CondIndep` predicate: it proves symmetry and supplies the trivial-σ-algebra case as a concrete inhabitant, while retaining basic measure-mass laws. The topic declarations do not select blanket σ-algebras from fep-005's labels or construct a joint law for a particular system. The maintained `FEP.MarkovBlanket` foundation supplies a separate normalized finite joint, proves Equation 48 at every positive-mass blanket state, proves zero conditional mutual information, and constructs typed nontrivial dynamics. Its factorization theorem is the named formal witness for the reviewed fep-005→fep-009 edge. The `FEP.NativeBlanket` family then embeds normalized finite laws as weighted Dirac measures, proves singleton and expectation reflection, aligns finite prediction with native measure–kernel composition, and proves Mathlib's native `CondIndepFun` for the factorized static blanket, including measurable endpoint coarsening and rowwise transition preservation. This closes one finite native instance without turning an arbitrary partition into a generic blanket-existence, invariance, or biological-boundary theorem.

In a NESS decomposition, skew-symmetry of a current supplies cancellations, but antisymmetry alone does not imply a continuous-space stationarity equation for an arbitrary field and density. Topic fep-025 now defines finite probability current as forward flow minus reverse flow, defines its divergence, proves antisymmetry and global conservation, and shows that a normalized transition matrix with a stationary distribution has zero divergence at every state. A directed three-state cycle witnesses the genuinely non-equilibrium case: its current is divergence-free yet nonzero, so stationarity does not force detailed balance. The row does not claim a diffusion, density PDE, or continuous Helmholtz decomposition; see §5.4.

Mathlib4 module footprint (Bayesian Mechanics): The area now uses several concrete backbones. Finite partitions and probability currents use `Finset` and `Matrix`; blanket independence reaches `Probability.Independence.Conditional`; reversibility reaches `Probability.Kernel.Invariance`; posterior, predictive, and hierarchical laws use native kernel composition and composition products; empirical-prior estimation uses `Probability.ProbabilityMassFunction.Binomial` plus exact sequence limits. Gaussian laws and derivatives use `Probability.Distributions.Gaussian.Real` and `Analysis.SpecialFunctions.Log.Deriv`. These imports expose real structures, while the authored cross-topic theorems certify which structures actually compose.

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-005	Four-label finite partition with disjointness and unique total membership	`formalized`	`Data.Finset.Basic`	0
fep-009	Conditional-independence symmetry and a trivial-σ-algebra witness, plus basic measure laws	`formalized`	`Probability.Independence.Conditional`	0
fep-010	Reversible Markov kernel implies invariant measure; identity-kernel witness and invariant composition	`formalized`	`Probability.Kernel.Invariance`	0
fep-019	Native prior-predictive measure, mass preservation, normalization, and sequential association	`formalized`	`Probability.Kernel.Composition.MeasureComp`	0
fep-027	Normalized hierarchical joint, exact marginals, and three-level associativity	`formalized`	`Probability.Kernel.Composition.MeasureComp`	0
fep-022	Posterior-predictive kernel plus exact proper Bernoulli Brier-score decomposition	`formalized`	`Probability.Kernel.Composition.MeasureComp`	0
fep-036	Finite binomial sampling, outcome-indexed Laplace prior, shrinkage identity, and consistency transfer	`formalized`	`Probability.ProbabilityMassFunction.Binomial`	0
fep-040	Native Gaussian law and moments; entropy monotonicity, derivatives, and heat capacity	`formalized`	`Probability.Distributions.Gaussian.Real`	0

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-042	Bernoulli sufficient statistic and exact likelihood factorization	`formalized`	`Data.List.Count`	0
fep-046	Recursive finite stick weights, exact mass conservation, and residual bounds	`formalized`	`Algebra.BigOperators.Ring.Lift`	0
fep-135	Weighted-Dirac embedding preserves singleton masses and normalization	`formalized`	`MeasureTheory.Measure.WithDensityFinite`	0
fep-136	Embedded finite laws are injective and preserve finite expectations	`formalized`	`MeasureTheory.Integral.Bochner.Basic`	0
fep-137	Embedded finite prediction agrees with native measure–kernel composition	`formalized`	`Probability.Kernel.Composition.MeasureComp`	0
fep-138	Embedded static blanket law has exact rectangle factorization	`formalized`	`MeasureTheory.Constructions.Pi`	0
fep-139	Factorized static blanket satisfies native `CondIndepFun`, with a correlated nontrivial witness	`formalized`	`Probability.Independence.Conditional`	0
fep-140	Measurable endpoint maps preserve native blanket conditional independence	`formalized`	`Probability.Independence.Conditional`	0
fep-141	A factorized finite transition row preserves the native blanket property	`formalized`	`Probability.Kernel.Composition.MeasureComp`	0

5.3.6.1 Hierarchical Generative Models and Predictive Coding (fep-027) A **hierarchical generative model** of depth L over observations o and latent state stacks $s^{(1)}, \dots, s^{(L)}$ factors the joint as a Markov chain on levels:

$$p\big(o, s^{(1)}, \dots, s^{(L)}\big) \;=\; p(o \mid s^{(1)}) \prod_{l=1}^{L-1} p(s^{(l)} \mid s^{(l+1)}) \;\cdot\; p(s^{(L)}). \tag{49}$$

In Friston's predictive coding realization, each conditional $p(s^{(l)} \mid s^{(l+1)})$ is Gaussian, $s^{(l)} = g^{(l)}(s^{(l+1)}) + \omega^{(l)}$ with $\omega^{(l)} \sim \mathcal{N}(0, \Pi_l^{-1})$, so that inference reduces to gradient descent on precision-weighted squared prediction errors $\varepsilon^{(l)} = s^{(l)} - g^{(l)}(\mu^{(l+1)})$. This gives the canonical **top-down predictions / bottom-up prediction errors** dynamic:

$$\dot{\mu}^{(l)} \;=\; -\,\Pi_l\, \varepsilon^{(l)} \;+\; \partial_{\mu^{(l)}} g^{(l-1)}\, \Pi_{l-1}\, \varepsilon^{(l-1)}, \tag{50}$$

a neurobiologically suggestive message-passing algorithm in which precisions Π_l gate the influence of each level. Marginalization preserves probability mass only after the joint law, measurability, and normalization hypotheses have been supplied.

The fep-027 row defines a hierarchical joint as Mathlib's native measure–kernel composition product. Probability parents and Markov child kernels give mass one; the first marginal recovers the parent; the second marginal is exactly the prior-predictive law; and a three-level hierarchy is associative up to the canonical measurable product equivalence. The authored `fep027_priorPredictive_is_fep019` theorem connects its child marginal to fep-019 without copying either definition. The row does not yet represent arbitrary-depth dependent products or the Gaussian conditional kernels in Equation 49.

5.3.6.2 Gaussian Entropy, Variance as Temperature, and Heat Capacity (fep-040) For a univariate Gaussian $\mathcal{N}(\mu, \sigma^2)$ the differential entropy admits the closed form

$$H(\mathcal{N}(\mu, \sigma^2)) \;=\; \tfrac{1}{2}\log(2\pi e\,\sigma^2) \;=\; \tfrac{1}{2}\log(2\pi e) + \tfrac{1}{2}\log\sigma^2, \tag{51}$$

with multivariate generalization $H(\mathcal{N}(\mu, \Sigma)) = \frac{1}{2}\log\det(2\pi e\,\Sigma)$. Entropy is monotone in the variance — higher σ^2 encodes higher uncertainty. The **equipartition / heat-capacity** analogy is the bridge to statistical mechanics: identify σ^2 with thermodynamic temperature T (each quadratic degree of freedom carries $\frac{1}{2}k_BT$ of energy at equilibrium), so that

$$U \;=\; \langle F\rangle \;=\; \tfrac{1}{2}k_BT, \qquad C_V \;=\; \frac{\partial U}{\partial T} \;=\; \tfrac{1}{2}k_B, \qquad S \;=\; \int \frac{C_V}{T}\,dT \;=\; \tfrac{1}{2}k_B\log T + \text{const.} \tag{52}$$

The shared logarithmic form motivates an **analogy** between variance and temperature after choosing units and a concrete probabilistic/thermodynamic model. It is not an identity of physical quantities: belief variance need not have temperature units, and precision need not be a thermodynamic inverse temperature. The catalogue does not formalize that model bridge.

The fep-040 row wraps Mathlib's normalized real Gaussian law and proves its total mass, mean, and variance from native theorems. It defines the one-dimensional positive-variance entropy formula, proves strict monotonicity and derivative $1/(2v)$, then composes variance $v = \kappa T$ to obtain entropy derivative $1/(2T)$ and dimensionless heat capacity $1/2$. The authored `fep013_gaussianHelmholtz_derivative` theorem feeds this entropy into the Helmholtz first-law identity. The remaining boundary is multivariate covariance/determinant entropy and a physical units model, not the absence of a Gaussian object.

5.3.6.3 Stick-Breaking Priors and Dirichlet Processes (fep-046) Sethuraman's **stick-breaking construction** gives an explicit, almost-surely-valid sample from a **Dirichlet process** $\mathrm{DP}(\alpha, G_0)$:

$$V_k \overset{\text{iid}}{\sim} \mathrm{Beta}(1,\alpha), \qquad \pi_k \;=\; V_k\prod_{j<k}(1-V_j), \qquad \theta_k \overset{\text{iid}}{\sim} G_0, \qquad G \;=\; \sum_{k=1}^{\infty}\pi_k\,\delta_{\theta_k}. \tag{53}$$

The construction proceeds by iteratively "breaking" a unit stick: at step k, fraction V_k of the remaining stick $\prod_{j<k}(1-V_j)$ is assigned to component k. Two algebraic invariants make this well-posed: the retained mass $v(1-v) \in [0,1]$ is nonnegative (each break gives a valid proportion), and the remaining stick $\prod_{j\le k}(1-V_j)$ decreases monotonically in k (monotone convergence to zero a.s. when $\alpha < \infty$), so $\sum_k \pi_k = 1$ almost surely. The resulting random measure G is a draw from $\mathrm{DP}(\alpha, G_0)$; the concentration parameter α controls the rate at which new atoms appear (small α = few large atoms, large α = many small atoms following G_0).

Dirichlet processes are the foundational prior of **Bayesian nonparametrics**: they place a prior over discrete probability measures whose support size is itself random and grows with data. In Active Inference, DP priors enable **infinite-dimensional generative models** that can add latent causes as observations demand — a formalization of conceptual novelty and structure learning. The Chinese-restaurant-process representation of the same prior gives a coherent sampling / inference scheme (Neal's Algorithm 8 for DP mixtures). Hierarchical Dirichlet processes (HDPs) extend this to shared-atom structure across groups (topic models, multi-task learning), and Pitman–Yor processes generalize to power-law atom-size distributions relevant to linguistic data.

The fep-046 row recursively defines every allocated weight and the residual mass for an arbitrary finite list of breaks. Lean proves the exact conservation identity "allocated mass plus remainder equals one" without inequality assumptions, then proves the remainder lies in $[0,1]$ when every break fraction lies in $[0,1]$. This completes the finite stick-breaking algebra. It does not certify an infinite random sequence, almost-sure vanishing of the remainder, or a Dirichlet-process law; those require a distribution on breaks and countable-limit theory.

Representative formalization — *Finite Label Partition (fep-005)*: Given an assignment from twenty states to four labels, the row defines each block with `Finset.filter`, proves unequal labels give disjoint blocks, characterizes membership, and proves existence and uniqueness of the block containing every state. It does not prove conditional independence or that any dynamical system admits a Markov blanket. The distinction is central to the Biehl et al. discussion (§9.5.1). See §13.5 in Appendix B and §14.5 in Appendix~14.

Representative formalization — *Hierarchical Generative Models (fep-027)*: Hierarchical models sit at the junction of Bayesian mechanics and measure-theoretic probability. The row uses Mathlib's composition product $\mu \otimes_m \kappa$ rather than an independent product measure: it proves normalization under probability/Markov hypotheses, exact parent and predictive marginals, and associativity of a third conditional level. This is a kernel-valued hierarchy with real conditional dependence, while a particular predictive-coding factorization and blanket structure remain model choices.

5.3.7 Synthesis: What the 62 Theorems Establish

Taken together, the 21 Information Geometry rows and 41 Bayesian Mechanics rows map a broad set of probabilistic and geometric interfaces. They do not yet span the full infrastructure required for a complete FEP formalization.

The 21 Information Geometry theorems establish the **differential-geometric substrate**:

- **Fisher geometry** — fep-038 derives the Bernoulli score, Fisher information and natural gradient; fep-004 abstracts its finite weighted metric; fep-018 gives the corresponding coordinate distance and separation law.
- **Categorical and optimization geometry** — fep-100–106 establish simplex-tangent Fisher positivity, pullback, scalar Cramér–Rao, invertible-chart natural-gradient equivariance, mirror-descent and affine Bregman identities, and replicator equivalence with full-rank and null boundaries.
- **Scalar exponential-family geometry** — fep-142–148 establish full-support normalization, affine log-density ratios, log-partition differentiation, centered scores, Fisher–variance equality, KL–Bregman duality, and interval-local mean-coordinate injection, with nonconstant positive-rank and constant-statistic zero boundaries.
- **Core information divergence** — fep-014 proves native KL laws and a composition-product chain rule; fep-024 proves scalar log identities rather than another KL definition.
- **Divergence layers** — fep-029 treats the exact scalar quadratic Bregman instance; fep-044 constructs Bernoulli squared Hellinger divergence; fep-104/fep-105 add finite mirror and affine-projection identities; fep-014 provides native measure KL.
- **Metric-space realization** — fep-018 proves Bernoulli Fisher–Rao nonnegativity, symmetry, triangle inequality, and separation, without claiming general α-geodesics.
- **Normalized Bayesian inversion** — fep-017 uses Mathlib's posterior kernel and proves normalization, reconstruction, recovery, and a countable Bayes-density law.

The 41 Bayesian Mechanics theorems establish the **probabilistic substrate**:

- **Finite partitions** — fep-005 proves a four-label disjoint cover with unique membership but not the conditional-independence structure $p(\mu, \eta \mid b) = p(\mu \mid b)p(\eta \mid b)$.
- **Conditional independence and measure substrates** — fep-009 proves generic `CondIndep` symmetry and a trivial-σ-algebra witness. The maintained blanket foundation connects the fep-005-style four-block structure to a normalized finite law, positive-mass conditional factorization, zero conditional mutual information, and typed nontrivial dynamics. fep-135–141 embed this carrier into native measures and prove one concrete sigma-algebra-valued `CondIndepFun` statement, measurable endpoint coarsening, and rowwise transition preservation. Generic blanket existence, invariance under arbitrary mixtures, and biological interpretation remain absent.
- **Predictive and hierarchical kernels** — fep-019 and fep-027 prove native normalized prediction, exact marginals, and associativity; fep-022 adds posterior prediction and a proper Brier-score decomposition.
- **Measure and finite Bayesian inversion** — fep-051–057 cover Radon–Nikodym reconstruction, posterior-kernel swapping and disintegration at their stated measure-theoretic scopes, plus finite normalization, Bayes reconstruction, conjugacy, and zero-evidence boundaries.
- **Temporal inference** — fep-072–078 prove finite forward filtering, backward information, smoothing, normalized variational updates, hierarchical prediction, and model averaging.
- **Causal blankets and interventions** — fep-079–085 connect finite blanket factorization, conditional mutual information, ordered parent factorization, intervention kernels, non-descendant invariance, and local Markov laws while retaining explicit zero-evidence and identification limits.
- **Finite stick arithmetic** — fep-046 proves exact allocated-plus-residual mass conservation and unit-interval residual bounds, while leaving the infinite random process open.
- **Sufficient statistics** — fep-042 proves Bernoulli Fisher–Neyman factorization through success and failure counts.
- **Empirical-prior model** — fep-036 defines a finite binomial sampling PMF and its outcome-indexed Laplace prior, proves interiority, monotonicity, exact shrinkage, and deterministic consistency transfer, and composes the estimate with a normalized Bernoulli posterior. The statistical-convergence foundation supplies almost-sure Boolean and finite-atom strong laws, a whole-law L^1 limit, and convergence of every finite observable. The learning family supplies concentration and model-evidence laws, while fep-121–127 add finite-law squared/Brier-risk and bad-event transfer for the same Laplace estimator. Posterior contraction, minimax or empirical calibration, and a marginal-likelihood optimum remain open.
- **Gaussian thermodynamics** — fep-040 constructs the native Gaussian law and moments and proves entropy/temperature derivatives and heat capacity in one dimension.
- **Finite-state stationarity** — fep-010 proves that measure–kernel detailed balance implies invariance, witnesses reversibility with the identity Markov kernel, and proves invariant-kernel composition. The finite path family adds fluctuation and dissipation laws; fep-149–155 add an exact two-state continuous-time semigroup, master equation, detailed balance, relaxation, and Lyapunov decay. These do not establish irreducibility in general, continuous-state stochastic dynamics, or pathwise SDE theory.

The intended theory interlocks information geometry with probabilistic generative, temporal, causal, and conditional-independence structures. The topic rows and maintained foundations close several interfaces explicitly: score expectations produce categorical Fisher geometry, pullbacks, scalar Cramér–Rao, chart-qualified natural gradients, and finite scalar exponential-family dual identities; posterior kernels compose with prediction, hierarchy, smoothing, and model averaging; finite blanket laws connect to interventions, local Markov structure, and one native `CondIndepFun` transfer; and empirical-law limits coexist with finite concentration, risk-transfer, and evidence bounds. The remaining frontiers are multidimensional smooth manifolds and dual connections, generic blanket existence and measure-level causal identification, continuous-state SDE/PDE semantics, infinite random measures, and posterior contraction or empirical calibration on a shared carrier. 62 catalogue rows therefore provide both breadth and deep executable instances without presenting finite closure as a universal FEP theorem.

5.4 Non-equilibrium Thermodynamics (21 topics)

The thermodynamic interpretation of the FEP draws analogies to statistical mechanics. The 21 catalogue rows formalize an exact Helmholtz differential identity, finite non-detailed-balance stationary currents, binary maximum entropy and normalized Gibbs weights, discrete fluctuation–response decay, finite entropy-production laws, a conditional Landauer derivation, normalized forward/reverse path laws, detailed and integral fluctuation identities, a finite Jarzynski equality, local-current cancellation, reversible one-step KL dissipation, and an exact two-state continuous-time Markov semigroup with master equation and Lyapunov decay. The motivating continuous-state and physical theories remain broader than these deliberately scoped models.

Thermodynamics area (current pin). Every row carries `mathlib_status: real`; semantic dispositions include direct and deliberately proxied scopes and are reported by the generated maturity projection. The receipt-backed native area rate is **`21/21 (100.0%)`** when `true` is true. Timing is read from that receipt rather than fixed in prose. The area combines calculus, finite sums and matrices, geometric limits, logarithms, exponential weights, native entropy results, finite path-space probability, and one explicit finite-state continuous-time kernel. Its finite and scalar models do not imply continuous-state trajectory theory or continuum statistical mechanics.

5.4.1 Thermodynamic Free Energy and Partition Structure

For a canonical equilibrium ensemble with the usual definitions of mean energy, entropy, and partition function, the thermodynamic **Helmholtz free energy** takes the equivalent forms:

$$\mathcal{F} \;=\; U \;-\; T\,S \;=\; -\,k_B\,T\,\log Z, \qquad Z \;=\; \sum_i \exp\!\big(-\beta\,E_i\big),\; \beta = 1/(k_B T). \tag{54}$$

Equation 54 motivates a formal analogy with variational free energy, but thermodynamic Helmholtz energy does not generically "upper-bound surprise." The connection requires an explicit choice of energy $E = -k_B T \log p$ and compatible normalization. fep-031 proves positivity and energy monotonicity of exponential weights, positivity of a nonempty finite partition sum, nonnegativity of the normalized weights, and that they sum to one on the selected support. It does not derive the Gibbs form from maximum entropy or construct a thermodynamic ensemble beyond this finite mass function.

5.4.2 Helmholtz Free Energy Bridge (fep-013): Full Derivation

The following calculation states one precise bridge under a fixed observation, common base measure, integrability, and positive temperature. Let $p(s,o)$ be a generative joint density and $q(s)$ a normalized approximate posterior. Define

$$U_q \;:=\; \mathbb{E}_q\big[-\log p(s,o)\big], \qquad H[q] \;:=\; -\,\mathbb{E}_q\big[\log q(s)\big], \qquad T \;=\; \frac{1}{k_B\,\beta}. \tag{55}$$

Here U_q is the *internal energy* interpretation of the negative log-joint (each configuration (s,o) is assigned an energy $E(s,o) = -\log p(s,o)$ in natural units), and $H[q]$ is the *Boltzmann entropy* of the posterior q. Substituting these definitions into the variational free energy $F_{\text{var}}[q] = \mathbb{E}_q[-\log p(s,o)] - H[q]$ (in nats) yields

$$F_{\text{var}}[q] \;=\; U_q \;-\; H[q] \;=\; \frac{1}{k_B T}\Big(\widetilde{U}_q \;-\; T\,\widetilde{S}_q\Big), \tag{56}$$

where $\widetilde{U}_q = k_B T\,U_q$ and $\widetilde{S}_q = k_B\,H[q]$. Calling the bracketed quantity $\mathcal{F}[q]$ gives the bridge and its Bayesian decomposition:

$$\boxed{F_{\text{var}}[q] \;=\; \frac{\mathcal{F}[q]}{k_B T} \;=\; D_{\text{KL}}(q(s)\,\|\,p(s\mid o)) - \log p(o)} \tag{57}$$

Here the equilibrium free energy is $\mathcal{F}_{\text{eq}} = -k_B T \log p(o)$ and the excess $\mathcal{F}[q] - \mathcal{F}_{\text{eq}}$ is $k_B T$ times the KL divergence. Positive rescaling preserves the argmin:

$$\operatorname*{argmin}_q F_{\text{var}}[q] \;=\; \operatorname*{argmin}_q \mathcal{F}[q]. \tag{58}$$

Under the stipulated construction and natural scaling, the minimizer is the posterior because the KL term vanishes there. This is a mathematical analogy obtained by defining the energy landscape from the generative density; it is not by itself a physical claim that biological inference is thermodynamic equilibration.

What fep-013 formalizes. It defines $F(T) = U(T) - TS(T)$, proves the exact derivative $F' = U' - S - TS'$, and reduces it to $F' = -S$ under the explicit equilibrium first-law identity $U' = TS'$. Boundary, order, and finite-difference laws are retained. The authored `fep013_gaussianHelmholtz_derivative` theorem instantiates this calculus with fep-040's Gaussian thermal entropy and $U(T) = T/2$. The row still does not identify this Helmholtz object with fep-002's variational free energy or prove the density-level bridge in Equation 57.

5.4.3 Jarzynski Equality and Fluctuation Theorems

For a system driven by an external protocol that transitions the Hamiltonian from H_0 to H_1 in finite time, the non-equilibrium work W is a random variable with distribution $P(W)$. The **Jarzynski equality** (Jarzynski 1997) provides an *identity* — not merely an inequality — linking the exponential average of W to the *equilibrium* free energy difference:

$$\langle e^{-\beta W} \rangle \;=\; \int P(W)\, e^{-\beta W}\, dW \;=\; e^{-\beta\, \Delta\mathcal{F}}, \qquad \Delta\mathcal{F} = \mathcal{F}_1 - \mathcal{F}_0. \tag{59}$$

Equation 59 can hold for protocols driven arbitrarily far from equilibrium, but not without a model: standard derivations assume an initial canonical ensemble and dynamics with the required microscopic reversibility or local-detailed-balance structure, along with a well-defined work functional. Applying Jensen's inequality to Equation 59 recovers the mean-work bound

$$\langle W \rangle \;\geq\; \Delta\mathcal{F}, \tag{60}$$

i.e., the mean work performed is at least the equilibrium free energy difference. Equality requires zero dissipated work almost surely; quasistatic reversible driving is the standard limiting realization. Equation 59 is stronger than Equation 60: it fixes an exponential moment of W, not merely its mean.

A companion result, the **Crooks fluctuation theorem** (Crooks 1999), relates forward and time-reversed work distributions at the level of *individual trajectories*:

$$\frac{P_F(W)}{P_R(-W)} \;=\; \exp\!\big(\beta\,(W - \Delta\mathcal{F})\big), \tag{61}$$

where $P_F(W)$ is the probability density of performing work W under the forward protocol (Hamiltonian swept from H_0 to H_1) and $P_R(-W)$ is the probability density of performing work $-W$ under the time-reversed protocol (swept from H_1 to H_0). Equation 61 quantifies the *exponential asymmetry* between a dissipative trajectory and its time-reverse: work excursions above $\Delta\mathcal{F}$ are exponentially more likely in the forward direction, while excursions below $\Delta\mathcal{F}$ are exponentially more likely in reverse. The Jarzynski equality is a direct corollary of Crooks: rearranging Equation 61 to $P_R(-W) = P_F(W)\, e^{-\beta(W-\Delta\mathcal{F})}$ and integrating $\int P_R(-W)\, dW = 1$ yields Equation 59 immediately.

Both fluctuation identities require trajectory measures and microscopic-reversibility assumptions. fep-010 states detailed balance at Mathlib's measure–Markov-kernel level and proves that reversibility implies invariance; the identity kernel supplies a non-vacuous witness. fep-037 treats a symmetric two-state relaxation and response model. The path-space expansion then constructs normalized finite forward and reversed path laws, proves involutive reversal and full-support ratios, defines entropy production as finite KL, and derives detailed and integral fluctuation identities. fep-097 proves a finite Jarzynski equality from an explicit inverse temperature, work functional, free-energy difference, and pointwise exponential normalization. These are exact finite path-law identities. They do not establish the continuous work-density Crooks theorem in Equation 61 without the corresponding pushforward and protocol structure.

5.4.4 NESS Solenoidal Flow (fep-025): Target Fokker–Planck Structure

The Fokker–Planck equation describes the time evolution of the probability density $p(x,t)$ of a stochastic process $\dot{x} = f(x) + \sqrt{2D}\,\xi(t)$ with drift f and diffusion D:

$$\frac{\partial p(x,t)}{\partial t} \;=\; -\nabla \cdot J(x,t), \qquad J(x,t) \;=\; f(x)\, p(x,t) \;-\; D\,\nabla p(x,t). \tag{62}$$

The vector $J(x,t)$ is the **probability current**: the net flux of probability mass through a point. At a stationary distribution $p^*(x)$ with $\partial_t p^* = 0$, Equation 62 reduces to the continuity constraint

$$\nabla \cdot J^*(x) \;=\; 0 \qquad \text{(stationarity)}. \tag{63}$$

Under the regular constant-diffusion model written above, two important stationary cases are:

1. **Zero-current equilibrium:** $J^*(x) \equiv 0$. With the additional reversibility assumptions connecting the diffusion to its stationary law, this is the detailed-balance case.
2. **Nonequilibrium steady state (NESS):** $J^*(x) \not\equiv 0$ but $\nabla \cdot J^*(x) = 0$. A nonzero stationary current breaks zero-current detailed balance; positive thermodynamic entropy production requires further constitutive and time-reversal structure.

NESS models are often used when describing driven, dissipative biological systems, but that application is a modeling choice rather than a consequence of stationarity alone. In one simplified constant-diffusion ansatz, a drift is written using a **Helmholtz–Ao-type decomposition** (Ao 2004):

$$f(x) \;=\; -\,(D \,+\, Q(x))\,\nabla F(x), \qquad F(x) \;=\; -\log p^*(x), \qquad Q(x)^\top \;=\; -Q(x), \tag{64}$$

where $F(x)$ is the negative log-stationary density up to normalization, D is symmetric positive-semidefinite, and $Q(x)$ is antisymmetric. Within this ansatz, substituting $p^* \propto e^{-F}$ into the current cancels the two D terms and leaves $J^* = -Q\,\nabla F\,p^*$. This algebra does not show that an arbitrary drift admits the ansatz, that the density exists, or that the remaining current is divergence-free; state-dependent diffusion conventions can also introduce additional terms.

What antisymmetry cancels. Expanding the candidate current's divergence shows which terms vanish algebraically and which require an additional hypothesis:

$$\frac{1}{p^*}\,\nabla \cdot (Q\,\nabla F\,p^*) \;=\; \mathrm{tr}\big(Q \cdot \nabla^2 F\big) \;-\; (\nabla F)^\top\, Q\,(\nabla F) \;+\; (\nabla \cdot Q)^\top \nabla F. \tag{65}$$

The trace term vanishes when F is sufficiently smooth because Q is antisymmetric and the Hessian is symmetric. The quadratic term also vanishes because an antisymmetric bilinear form is zero on a repeated vector. The remaining $(\nabla \cdot Q)^\top \nabla F$ term does not vanish from antisymmetry alone; it needs, for example, spatially constant Q or a separate orthogonality condition. Thus even this restricted ansatz requires an explicit divergence check. It does not imply that removing Q forces every stochastic system to detailed-balance equilibrium.

What fep-025 formalizes. It defines a finite edge current $J_{ij} = \pi_i P_{ij} - \pi_j P_{ji}$ and its node divergence. Lean proves antisymmetry, zero self-current, global conservation, and the pointwise zero-divergence consequence of row normalization and stationarity. A directed three-state cycle then proves existence of a nonzero divergence-free stationary current, formally separating stationarity from detailed balance. This is a finite NESS witness. It does not formalize the trace cancellation, a Hessian, a state-dependent Q, or any PDE/SDE in Equation 65.

5.4.5 Maximum Entropy (fep-030): Jaynes' Derivation

Jaynes' **maximum-entropy principle** (Jaynes 1957) provides a constructive answer to the question "*given that I know only the expectation values $\langle f_i \rangle = c_i$ of certain observables, which probability distribution should I assign?*" The principle says: assign the distribution p^* that maximizes the Shannon/Boltzmann entropy $H[p] = -\mathbb{E}_p[\log p]$ subject to the constraints, and no others. This is the least-biased inference consistent with the data: any other distribution would implicitly assume information the data did not provide.

The constrained-optimization problem is

$$p^* \;=\; \operatorname*{argmax}_p \; H[p] \quad \text{subject to} \quad \sum_x p(x) = 1, \;\; \sum_x p(x)\, f_i(x) = c_i \;\; (i = 1, \ldots, k). \tag{66}$$

For a finite feasible problem whose optimum lies in the positive interior and satisfies the relevant constraint qualification, introducing Lagrange multipliers λ_0 for normalization and λ_i for each constraint yields the stationarity condition $-\log p^*(x) - 1 - \lambda_0 - \sum_i \lambda_i f_i(x) = 0$. The resulting candidate has **Boltzmann–Gibbs form**:

$$p^*(x) \;=\; \frac{1}{Z(\lambda)}\, \exp\Big(-\sum_{i=1}^{k} \lambda_i\, f_i(x)\Big), \qquad Z(\lambda) \;=\; \sum_x \exp\Big(-\sum_{i=1}^{k} \lambda_i\, f_i(x)\Big). \tag{67}$$

When a matching multiplier exists, enforcing the constraints relates derivatives of the log partition function to the prescribed moments. This is the familiar maximum-entropy route to exponential families, subject to feasibility, support, boundary, and dual-attainment qualifications. fep-030 and fep-031 do not prove that general optimization-to-Gibbs bridge. They compose at one exact boundary: for a two-state Gibbs law at zero

inverse temperature, hence the infinite-temperature limit, `FEPComposed.fep031_zeroBeta_binary_maxEntropy` rewrites the selected probability to $1/2$ and invokes fep-030's binary-entropy equality characterization. The separate fep-142–148 family constructs a full-support finite scalar exponential family and proves log-partition, Fisher–variance, and KL–Bregman identities, but it assumes the family rather than deriving it as the solution of Equation 66. Here $\beta = 0$ must not be described as zero physical temperature.

Two canonical special cases. (i) *Uniform distribution:* with no constraints beyond normalization, the maximizer is $p^*(x) = 1/n$ on a finite set of size n, achieving $H[p^*] = \log n$. This is the classical "principle of insufficient reason". (ii) *Canonical ensemble:* with the single constraint $\langle E \rangle = \bar{E}$, the maximizer is $p^*(x) \propto \exp(-\beta E(x))$ with $\beta = \lambda_1$ identified as inverse temperature.

What fep-030 formalizes. It imports Mathlib's binary entropy, proves the global bound $\text{binEntropy}(p) \leq \log 2$, and proves equality exactly at $p = 1/2$. It also retains the explicit finite-uniform normalization and entropy-equals-$\log n$ calculation. The binary theorem is a genuine maximum-entropy result; the arbitrary finite-simplex and constrained Jaynes problems remain unproved. The composed zero-inverse-temperature theorem connects the binary Gibbs special case to this maximum, but it does not derive general Gibbs weights from constrained optimization.

5.4.6 Entropy Production and the Variational–Thermodynamic Bridge

In a one-flux/one-force schematic, a local **entropy-production contribution** is often written as the pairing of a thermodynamic flux J with a conjugate force F; sign, mobility, integration, and time-reversal conventions are model-dependent:

$$\sigma \;=\; J \cdot F \;=\; \mathbf{J} \cdot \nabla \log p \;\geq\; 0, \tag{68}$$

The conditions for nonnegativity and equality depend on the constitutive relation and inner product; nonzero flux alone does not imply a positive scalar product with force. Equation 68 is motivating thermodynamics, not a consequence of scalar sign assumptions alone.

Topic **fep-049** formalizes two finite constitutive models. For diagonal linear response $J_i = L_i X_i$, entropy production is the quadratic form $\sum_i L_i X_i^2$; Lean proves nonnegativity for $L_i \geq 0$ and zero iff all forces vanish for $L_i > 0$. Independently, strictly positive forward/reverse edge fluxes have production $(f - r)\log(f/r) \geq 0$, with equality exactly at detailed balance $f = r$. These are genuine finite second-law instances under explicit constitutive assumptions, not a microscopic derivation or a theorem for arbitrary cross-coupled Onsager matrices.

The bridge calculation above becomes exact only after a generative density is used to define an energy landscape and units are fixed. The present Lean rows do connect selected thermodynamic interfaces: fep-025 currents feed fep-049's nonnegative diagonal dissipation, and fep-040 thermal entropy feeds fep-013's Helmholtz derivative. They still do not identify biological self-organization with physical relaxation, because no theorem maps a concrete generative model, stationary law, thermodynamic force, and variational objective into one shared dynamics.

5.4.7 Landauer Bound (fep-050): Information–Thermodynamic Interpretation

Landauer's principle (Landauer 1961) asserts that erasing one bit of information from a physical memory has an irreducible thermodynamic cost of at least

$$W_{\text{erase}} \;\geq\; k_B T \,\log 2 \;\approx\; 2.9 \times 10^{-21}\,\text{J at } T = 300\,\text{K}. \tag{69}$$

In the standard idealized symmetric-memory model, an initially equiprobable bit has Shannon entropy $\log 2$ nats and a reset state has zero logical entropy. A cyclic isothermal implementation must compensate that entropy decrease in the environment; in the reversible limit this gives the $k_B T \log 2$ minimum. More general Landauer results are formulated through nonequilibrium free-energy changes and depend on the physical memory, control protocol, reservoir, and error tolerance. The bound is therefore not obtained from information entropy plus the first law without those modeling assumptions.

Landauer's bound applies to logically irreversible erasure under physical assumptions; it is not a cost per Bayesian update, per bit observed, or per nat of KL reduction. Deriving a neural-metabolic rate limit would require an explicit physical implementation, a count of irreversible erasures, temperature/control assumptions, and a link from algorithmic state changes to thermodynamic operations. Neither the general principle nor fep-050 supplies that bridge.

What fep-050 formalizes. It defines a noninjective Boolean reset, computes the entropy lost by erasing an unbiased bit as $k_B \log 2$, and models environmental entropy change as Q/T. From positive temperature and the explicit second-law hypothesis $\Delta S_{\text{total}} \geq 0$, Lean derives $Q \geq k_B T \log 2$; a separate work-at-least-heat hypothesis yields the work bound. Thus the thermodynamic inference is explicit and checkable, while the second law and implementation energetics remain assumptions rather than a microscopic Hamiltonian derivation.

5.4.8 Lean 4 Formalization: Entropy, Boltzmann, and Landauer

The thermodynamic topics encode six distinct, composable layers of the theory:

- **Binary maximum entropy plus uniform calculation (fep-030)** — native `Real.binEntropy` is bounded by $\log 2$ with equality exactly at $1/2$; uniform finite weights are separately normalized and their entropy expression equals $\log n$. General finite-simplex maximality is not proved.
- **Finite Boltzmann–Gibbs weights (fep-031)** — Exponential-weight positivity, energy-ordered monotonicity at positive inverse temperature, partition-sum positivity, pointwise nonnegativity after normalization, and exact mass one on a nonempty selected support.
- **Differential Helmholtz law (fep-013)** — $F = U - TS$ is differentiated exactly, and the equilibrium first-law premise reduces F' to $-S$; fep-040 supplies a concrete Gaussian thermal instance.
- **Stationarity, response, and production (fep-025, fep-037, fep-049)** — finite nonzero stationary currents, exact two-state response decay, quadratic entropy production, and edge detailed-balance separation are theorem-level objects.
- **Conditional Landauer derivation (fep-050)** — erasure entropy loss plus an explicit second-law premise derives the heat bound, and work-at-least-heat derives the work bound.
- **Finite path thermodynamics (fep-093–099)** — normalized forward/reverse path laws, involutive reversal, path-ratio and fluctuation identities, finite Jarzynski normalization, local current cancellation, and reversible KL dissipation with explicit irreversible and zero-rate boundaries.
- **Two-state continuous time (fep-149–155)** — a positive-rate Boolean Markov kernel, identity and Chapman–Kolmogorov laws, left and right master equations, a nonuniform detailed-balance stationary law, exact exponential relaxation, and quadratic Lyapunov decay with a strict nonstationary benchmark.

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-013	Helmholtz definition, exact derivative, and equilibrium reduction to $-S$	`formalized`	`Analysis.Calculus.Deriv.Basic`	0
fep-025	Finite probability current, conservation, stationarity, and nonzero three-cycle witness	`formalized`	`LinearAlgebra.Matrix.Notation`	0
fep-030	Native binary-entropy maximum/equality characterization and exact uniform finite entropy	`formalized`	`Analysis.SpecialFunctions.BinaryEntropy`	0
fep-031	Finite Gibbs-weight positivity, energy monotonicity, normalization, and mass one	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-037	Two-state autocorrelation recurrence/decay and exact fluctuation–response kernel	`formalized`	`Analysis.SpecificLimits.Normed`	0
fep-049	Finite quadratic and edge-flux entropy production with equilibrium separation	`formalized`	`Analysis.SpecialFunctions.Log.Basic`	0
fep-050	Boolean erasure entropy loss and conditional Landauer heat/work derivation	`formalized`	`Analysis.SpecialFunctions.BinaryEntropy`	0

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-093	Normalized finite forward/reverse path laws, reversal, and full-support ratio	`formalized`	`Probability.Kernel.Invariance`	0
fep-094	Finite path entropy production identified with finite KL	`structural_proxy`	`InformationTheory.KullbackLeibler.Basic`	0
fep-095	Detailed finite fluctuation symmetry from the path-law ratio	`conditional_proxy`	`Analysis.SpecialFunctions.Exp`	0
fep-096	Integral fluctuation identity under normalized full-support path laws	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-097	Finite Jarzynski identity with explicit work, inverse temperature, and normalization	`conditional_proxy`	`Analysis.SpecialFunctions.Exp`	0
fep-098	Local detailed balance and edge-current cancellation	`formalized`	`Probability.Kernel.Invariance`	0
fep-099	Reversible one-step KL dissipation and irreversible positive-production witness	`formalized`	`InformationTheory.KullbackLeibler.ChainRule`	0
fep-149	Positive-rate two-state continuous-time Markov kernel and nonuniform benchmark	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-150	Continuous-time transition is the identity at time zero	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-151	Exact Chapman–Kolmogorov semigroup law	`formalized`	`LinearAlgebra.Matrix.Multiplication`	0
fep-152	Entrywise left and right continuous-time master equations	`formalized`	`Analysis.SpecialFunctions.ExpDeriv`	0
fep-153	Stationarity and detailed balance for the exact semigroup	`formalized`	`Probability.Kernel.Invariance`	0

Topic	Actual Lean content	Semantic disposition	Mathlib navigation hint	`sorry` count
fep-154	Exact exponential relaxation from an arbitrary normalized Boolean law	`formalized`	`Analysis.SpecialFunctions.Exp`	0
fep-155	Exact quadratic Lyapunov law, derivative, and strict benchmark decay	`formalized`	`Analysis.SpecialFunctions.Exp` `Deriv`	0

The strongest direct facts form a chain rather than an isolated roster: fep-030 and fep-031 meet at the binary infinite-temperature maximum; fep-025's finite currents feed fep-049's dissipation law; fep-040's Gaussian thermal entropy feeds fep-013's Helmholtz derivative; and fep-050 derives Landauer heat and work bounds from named physical premises. Each chain is narrower than the full physical theory, but each seam is checked in Lean.

Key formalization — Helmholtz free energy $F = U - TS$ (fep-013): The row represents temperature-dependent internal energy and entropy as differentiable real functions, proves the product-rule derivative $U' - S - TS'$, and derives $F' = -S$ from $U' = TS'$. Boundary and monotonicity laws remain available. The separate fep-040 composition supplies a Gaussian thermal witness; no theorem identifies thermodynamic and variational free energy.

Mathlib4 module footprint (Thermodynamics): The area routes through `Analysis.SpecialFunctions.BinaryEntropy`, real differentiation, geometric-limit results, logarithms and exponentials, finite big operators, and matrix notation. The generated coverage report, rather than this illustrative roster, is the exact import authority.

Thermodynamic vs variational free energy — formally distinct Lean objects: The thermodynamic object fep-013 consumes $U(T)$, T, and $S(T)$, whereas fep-002 defines variational free energy as surprisal plus native measure KL. Their types and assumptions are deliberately distinct. The Gaussian composition closes one thermodynamic instance, and the fep-002/fep-014 composition closes the KL chain rule, but no theorem identifies these two free energies. Such a theorem would need a concrete density, energy map, units, normalization, and integrability contract.

Representative formalization — *Helmholtz differential identity (fep-013)*: The row proves the exact derivative law and its equilibrium reduction under explicit differentiability and first-law assumptions. The composition with fep-040 supplies a nontrivial entropy function rather than an arbitrary scalar witness. It does not derive the first law or the physical meaning of temperature. See §13.13 in Appendix B and §14.13 in Appendix~14.

Representative formalization — *Finite stationary probability current (fep-025, Eq. 70)*: Some NESS constructions posit a Helmholtz/Ao decomposition (Ao 2004) with a skew component:

$$\dot{\rho} = -\nabla \cdot (\rho \nabla \mathrm{F} + Q\rho) = 0, \qquad Q = -Q^{\top} \tag{70}$$

The row proves the finite continuity analogue directly: a stationary normalized transition has zero current divergence, and a directed three-cycle has both zero divergence and nonzero current. Antisymmetry alone still does not discharge the state-dependent continuum term, and the row has no Hessian or PDE. See §13.25 in Appendix B and §14.25 in Appendix~14.

Representative formalization — *Conditional Landauer derivation (fep-050)*: The row proves the reset is noninjective, computes unbiased-bit entropy loss, derives the heat bound from total-entropy nonnegativity, and derives the work bound from work-at-least-heat. The physical premises are visible theorem hypotheses rather than hidden in prose.

5.4.9 Closing Synthesis: What the Thermodynamics Theorems Establish

The 21 rows form a finite, executable thermodynamic layer:

- **Free energy and finite ensembles.** fep-013 proves Helmholtz calculus; fep-030 proves the binary entropy maximum; fep-031 normalizes finite Gibbs weights; composed theorems connect the Gaussian and binary special cases.
- **Stationarity and response.** fep-010 proves reversible-kernel invariance; fep-025 constructs a nonzero divergence-free finite current; fep-037 proves exact discrete two-state correlation and response decay; and fep-149–155 prove an exact two-state continuous-time semigroup, master equation, detailed balance, relaxation, and Lyapunov decay.
- **Production and erasure.** fep-049 proves finite quadratic and edge-flux production laws with equality characterizations; fep-050 derives Landauer heat and work bounds from explicit second-law and work-transfer premises.
- **Path thermodynamics.** fep-093–099 construct finite path laws, entropy production, detailed/integral fluctuation and Jarzynski identities, local detailed-balance currents, and reversible KL dissipation on the shared finite Markov carrier.

The remaining theorems are substantive: the measure-theoretic Helmholtz/variational bridge, a general constrained maximum-entropy derivation of Gibbs laws, continuous or singular path measures, the full protocol-level Crooks work-distribution theorem, generic CTMC construction, continuous-state NESS existence, cross-coupled constitutive response, and a microscopic finite-time erasure model. A fresh warning-free exact-roster receipt would establish acceptance of the present finite mathematics at the pinned toolchain; even then, it would remain a scoped thermodynamic formalization layer rather than a universal physical derivation of the FEP.

5.5 Quantitative Execution Metrics

This section reports three surfaces separately: catalogue coverage, native Lean evidence, and optional full-pipeline evidence. A count from one surface is never substituted for another. The generated formalism coverage audit is authoritative for source structure; native and full receipts are authoritative for execution claims.

The area-distribution figure visualizes catalogue breadth only. It does not measure scientific maturity or compilation success.

5.5.1 Aggregate Catalogue Metrics

The catalogue contains 155 topics across 5 areas. `mathlib_status` records whether a row is intended to use available pinned-library infrastructure; semantic disposition records whether the primary theorem reaches the topic-facing claim. The two fields answer different questions. Maintained foundation/composition declarations are counted separately from topic declarations before the package total is formed:

Source layer	Modules	Theorem declarations
Topic catalogue	155 topic modules	487
Maintained foundations	22	475
All maintained formal resources, including composition	34	602
Package total, without double-counting resources	—	1089

Area	Topics	Native receipt rate
FEP	41	`41/41 (100.0%)`
Active Inference	31	`31/31 (100.0%)`
Bayesian Mechanics	41	`41/41 (100.0%)`
Information Geometry	21	`21/21 (100.0%)`
Thermodynamics	21	`21/21 (100.0%)`
Total	**155**	**`155/155 (100.0%)`**

The runtime projection reports the following receipt-derived values:

Field	Value
Evidence kind	`native-lean`
Claim-ready native/full Lean evidence	`true`
Receipt identifier	`native-566a22fce86c`
Topics with compiler results	155
Compiled	155
Failed compilation	0
Warnings	0
`sorry`	0
Failed topic IDs	none
Measured compiler time	556.106 s
Mean measured time per result	3.588 s
Lean / Mathlib pin	`leanprover/lean4:v4.33.1/v4.33.1`

A native receipt becomes claim-ready only for the exact ordered roster of 155 topics with every result compiling, zero errors, zero warnings, zero `sorry`, actual Lean output matching the configured pin, the resolved Mathlib commit, finite timing evidence, and source/toolchain digests matching an explicitly supplied live tree. A valid subset or structurally validated unbound receipt remains useful diagnostic evidence but cannot populate the full-catalogue headline.

5.5.2 Maturity Distribution by Area

All rows currently carry `mathlib_status: real`, but that is a compile-intent classification, not a semantic result. The separate semantic audit reports 136 directly formalized, 13 conditional, and 6 structural rows at their stated scopes. The generated area-by-disposition matrix and per-topic assumptions live in `docs/formalism-coverage.md`; reproducing that matrix here would create a second owner.

5.5.3 Formalism Composition and Capability Ledger

The authored semantic graph contains **133** reviewed edges. Of these, **20** are derivational formal edges and **105** are checked pairings whose theorem certifies both endpoint laws without claiming that one follows from the other. Together they form **125** qualified Lean declaration witnesses. The remaining edges are explicitly conceptual or blockers and therefore carry no proof claim. The retained capability ledger contains 48 nodes: 0 open, 0 partial, and 48 satisfied. "Unresolved" includes both open and partial nodes, so its generated count is 0.

`docs/formalism-atlas.svg` and `docs/formalism-atlas.html` visualize this exact join. The atlas renders formal-module import dependencies separately from authored scientific relations, so code reuse cannot masquerade as a theorem witness. `docs/formal-kernel-dashboard.svg` and its interactive HTML companion evaluate one deterministic numerical witness for each of the fifteen expansion families, including support, normalization, contraction, rank, risk-transfer, feedback, conditional-independence, duality, master-equation, and tail-bound boundaries. Every witness exposes typed checks with individual tolerances rather than one undifferentiated residual. Those values are validation diagnostics, not proof evidence. The declaration/axiom audit must still resolve formal witnesses, while the native receipt covers the exact 155 topic bodies and the maintained modules compile through their manifested projection.

5.5.4 Hermes LLM Performance

Hermes commentary is optional and has no bearing on native Lean acceptance. Manuscript Hermes metrics are exposed only from an independently validated, claim-ready full report. At render time, full-run claim readiness is `false`.

Full-run field	Value
Report run ID	`| | Processed topics | 0 | | Successful Hermes calls | 0 | | Primary model |`
Models used	
Model-chain advances	0
Same-model network retries	0
Hermes-refined sketches compiling	0
Mean measured topic time	0.0 s
Token total	0

When `full.claim_ready` is false, zero or empty cells above mean *unavailable evidence*, not an observed zero-event experiment. Catalogue mode cannot populate this table.

5.5.5 Lean 4 Verification Timing

Only receipt-recorded elapsed time is reported: 556.106 seconds in total and 3.588 seconds per result for `native-566a22fce86c`. We do not generalize these machine- and cache-dependent observations into universal cold/warm timing ranges.

5.5.6 Error Category Distribution

`LeanVerifier` retains compiler errors, warnings, and an advisory failure category for each topic. The current receipt summary reports 0 compile failures, 0 warnings, and 0 admitted proofs. Per-topic rows in the receipt, rather than a manually maintained chart, are authoritative.

5.5.7 Live Verification Error Taxonomy: Hermes-Assisted Run

No Hermes-assisted result is described as live unless `full.claim_ready` is true. Full receipts must bind every selected row to a successful provider session and model, direct `hermes_refined` compilation, byte-identical refined/final Lean and its digest, actual compiler output, the resolved Mathlib commit, complete per-topic artifacts, redundant manifest parity, and current source/configuration digests. The renderer currently reports full claim readiness as `false`.

Historic LLM failures are useful engineering anecdotes but are not current mathematical evidence. The canonical failure rows remain in validated report bundles rather than being copied into a static taxonomy.

5.5.8 Baseline Comparison: Hermes-Assisted vs Manual Drafting

This project has not run a controlled human-versus-Hermes authoring study. It therefore makes no comparative claim about proof length, first-pass success, or authoring speed. A defensible comparison would require preregistered tasks, identical library context, blinded adjudication of semantic adequacy, and separate compiler and wall-clock outcomes.

5.6 Maturity Migration Pathways

Rows migrate only through evidence, not elapsed calendar time. A proxy becomes `formalized` when its primary theorem states the reviewed invariant at the advertised scope, its assumptions are explicit, a non-vacuous witness exists where appropriate, and native acceptance remains clean. Mathlib pin changes, new definitions, or more permissive prose do not automatically promote a row.

5.7 Error Taxonomy: LLM Failure Modes

The useful distinction is contractual:

- transport and provider failures prevent full-mode completion;
- malformed or invented Lean fails compiler verification;
- `sorry` fails native claim readiness even if elaboration succeeds;
- a compiling theorem with the wrong scope is a semantic-audit failure;
- a stale or tampered artifact is a receipt-validation failure.

These failure classes are independently observable and should not be collapsed into one success rate.

5.8 Cross-Area Mathlib Dependency Analysis

The generated coverage audit derives the exact Mathlib module-to-topic incidence relation from canonical imports. Shared imports indicate library reuse, not logical dependence between catalogue topics. Conceptual relationships must be reviewed separately, and both derivational relations and formal pairings must name checked declarations that consume both endpoint namespaces; the project does not infer any relation from common tokens such as `State`, `Policy`, or `Measure`.

5.9 Semantic Closure, Composition, and Visual Validation

The catalogue is evaluated as a typed scientific graph rather than a pile of compiling files. A topic node is *directly formalized* only when its primary theorem states the reviewed, deliberately bounded invariant, its assumptions are explicit, and its non-vacuity or concrete-model obligation has been discharged. A relation is *formal* only when a qualified declaration in a manifested composition leaf derives or identifies a shared result across both endpoints. A *formal pairing* is weaker: its checked theorem exposes both endpoint laws in one context but deliberately asserts no implication. A capability is *satisfied* only when its declared topic and relation requirements are closed. These conditions prevent theorem count, import count, and compilation success from substituting for semantic depth.

For the current canonical sources, 136 of 155 topics are directly formalized; the remainder retain explicit conditional or structural proxy classifications. The semantic graph contains 133 reviewed relations: 20 derivational formal seams, 105 checked non-implicational pairings, and the remaining conceptual or blocker edges. Its 48 capability nodes resolve to 48 satisfied, 0 partial, and 0 open. A separate reusable kernel contains 34 maintained formal modules and 602 declarations. It combines almost-sure empirical-law convergence with separately scoped finite concentration, finite Laplace/Brier-risk transfer, and model-evidence laws; it still makes no posterior-contraction, minimax, empirical-calibration, or generic marginal-likelihood optimization claim.

5.9.1 The Formalism Atlas

The interactive offline atlas presents the same canonical join with searchable node detail. Neither view infers scientific relations from shared imports or vocabulary. Topic metadata comes from the generated catalogue, relation kinds and witness names come from `config/formalism_relations.yaml`, and capability state is recomputed from exact declarations. Authored cross-topic witnesses live in manifested composition leaves and resolve through `FepSketches.composed`; capability evidence may additionally resolve in any manifested foundation. Purple module-import arrows occupy a separate edge class and table from teal scientific theorem edges. The SVG and HTML are projections, so their `--check` gate detects drift instead of accepting manual edits.

5.9.2 Established Core Theorem Strands

The original core is organized into the following ten mutually connected strands. The first ten expansion families and their distinct carriers are treated in §6; the five families added for topics 121–155 are treated in §7. Neither expansion is compressed into this historical core table:

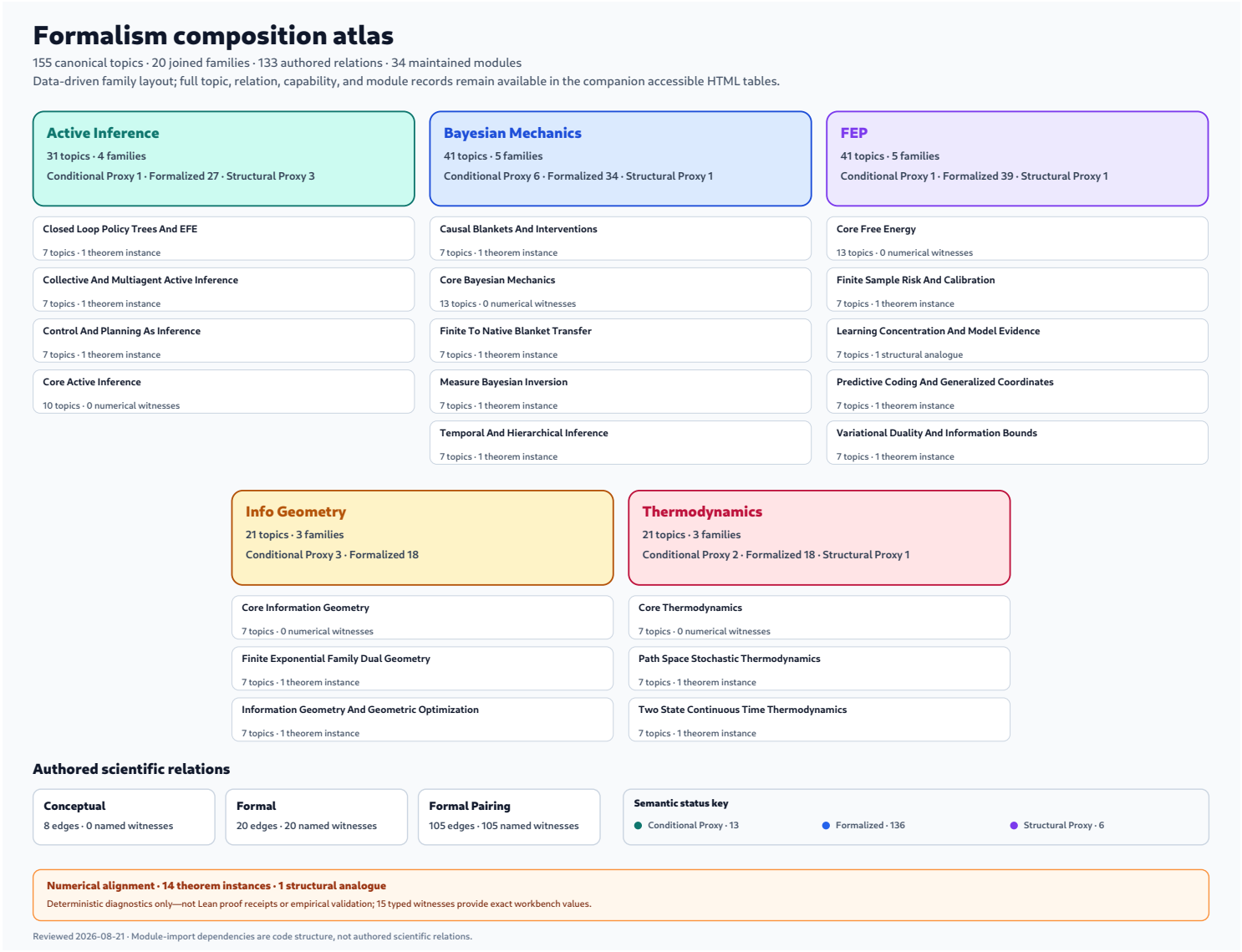


Figure 2: Formalism atlas generated from the topic, relation, and capability registries. Derivational relations, checked formal pairings, and conceptual relations are separately labeled, while the schema retains an explicit blocker notation for future gaps.

Strand	Direct objects and laws	Representative witnessed seams
Finite probability	Normalized laws and kernels, products, marginals, Bayes reconstruction, identity and associative kernel composition, predictive functor laws	`FEP.FiniteKernel.comp_assoc`; `FEP.FiniteKernel.predictive_comp` `FEP.ActiveInference.posteriorSta`
Finite information	Zero-safe entropy; support-free separation for totalized normalized finite KL and mutual information; prior-kernel and independent-product KL chain laws under their logarithmic support assumptions; explicit disjoint-support boundary values	`FEP.FiniteInformation.finiteKL_e` `FEP.FiniteInformation.finiteKL_d` `FEP.FiniteInformation.mutualInfo`
Variational inference	Outcome surprisal, posterior-form VFE, evidence lower bound, exact-posterior attainment, and uniqueness even with zero-mass posterior states	`FEP.ActiveInference.negative_var` `FEP.ActiveInference.variationalF`
Decision, policy, and planning	Both EFE decompositions, support-gated prior-weighted policy posterior, MAP/minimizer existence, a transition-consistent infer–select–act joint, prior-sensitive Boolean witness, and state-updating cumulative finite-horizon EFE	`FEP.ActiveInference.inferSelectA` `FEP.ActiveInference.symmetricBoo` `FEP.ActiveInference.cumulativeEx`
Blanket structure	Positive-mass conditional factorization, zero blanket-conditioned internal–external mutual information, typed four-component dynamics, and row-wise transfer from dynamics to the static factorization	`FEP.MarkovBlanket.transition_row` `FEP.MarkovBlanket.transition_row`
Information geometry	Multidimensional centered scores, Fisher PSD/PD conditions, compositional pullbacks, exact matrix lowering, inverse-Fisher metric duality and energy, an exact Bernoulli carrier, and an explicit rank-deficient score family	`FEP.InformationGeometry.bernoull` `FEP.InformationGeometry.duplicat` `FEP.InformationGeometry.pullback`
Statistical convergence	Topic-independent Boolean and finite-atom SLLNs, simultaneous atom convergence, whole-law L^1 and every finite-observable expectation convergence, plus a composed Laplace-smoothing bridge	`FEP.StatisticalConvergence.empir` `FEP.StatisticalConvergence.empir` `FEPCom-` `posed.fep036_smoothedRate_strong`
Discrete dynamics and optimization	Measurable semigroups, invariant kernels, two-state relaxation, exact quadratic descent, contraction convergence	`fep037_autocorrelation_tracks_fe` `fep032_update_is_fep043_gradient`
Thermodynamics	Helmholtz derivative, Gaussian thermal entropy, Gibbs weights, binary maximum entropy, finite NESS current, entropy production, Landauer derivation	`fep013_gaussianHelmholtz_derivat` `fep031_zeroBeta_binary_maxEntrop` `fep025_current_dissipation_nonne`
Structural composition	Unique finite partitions, additive local/global energy, matrix–vector message propagation, finite stick-mass conservation	`fep039_partitionEnergy_conservat` plus the exact local definitions and laws of fep-047 and fep-046

This organization shows both closure and non-closure. The posterior/filter/hierarchy strand shares native kernel objects, so its composition is definitional or theorem-level. The finite-current/entropy-production strand shares a current function explicitly. The finite blanket kernel gives a theorem-witnessed bridge from a static partition to conditional factorization and zero conditional mutual information; fep-135–141 further embed that carrier and prove one native `CondIndepFun` instance. The generic measure-theoretic topic remains broader: the package does not claim that every sigma-algebra-valued conditional-independence formulation arises from its finite factorization, nor that every dynamics admits or preserves a blanket.

5.9.3 Validation as a Layered Scientific Contract

Validation proceeds through independent failure surfaces:

1. **Canonical generation:** topic YAML, package data, the topic aggregate, every manifested formal resource, and the import aggregate must equal their generated projections.

2. **Semantic firewall:** every topic retains an assumption review, non-vacuity note, and acceptance probe. A future `scope_gap` or `assumption_gap` row must additionally point through a `blocked_by` edge to a concrete unresolved capability; satisfied capabilities cannot be used as blockers. Conditional and structural proxies preserve their narrower scope without falsely reopening a capability whose stated requirements are already witnessed.
3. **Native elaboration:** all topic bodies and every manifested formal resource compile at the pinned Lean/Mathlib revision with zero warnings and no admitted proofs.
4. **Declaration and axiom audit:** every primary theorem, derivational relation witness, and formal-pairing witness resolves by qualified name; its declaration kind and axiom footprint are recorded in a source-bound receipt.
5. **Projection integrity:** coverage JSON/Markdown and atlas SVG/HTML regenerate byte-for-byte from the same source graph.
6. **Publication integrity:** manuscript variables resolve fail closed, theorem references resolve against the canonical declaration inventory, citations and links pass their audits, and the rendered manuscript receives local copies of its generated visual assets.
7. **Distribution integrity:** an isolated wheel install must expose the `fep_lean` namespace, package data, and CLI without relying on the repository checkout or collision-prone top-level modules.

No layer subsumes another. Lean can reject an invalid proof but cannot decide whether a theorem title overclaims its type. The semantic firewall can expose a missing scientific capability but cannot certify elaboration. The atlas can make topology inspectable but cannot create a theorem edge. A native receipt can prove compilation for exact bytes but cannot authenticate an optional provider run. The package therefore publishes these receipts and projections side by side rather than collapsing them into one success percentage.

5.9.4 Residual Boundary and Extension Rule

The empirical-Bayes capability is closed at a deliberately finite scope: a Mathlib binomial sampling law, count-derived Laplace prior, exact shrinkage identity, almost-sure Boolean and finite-atom strong laws under explicit premises, L^1 empirical-law convergence, convergence of every finite-state observable, smoothing preservation, normalized posterior update, and finite-law squared/Brier-risk plus bad-event transfer. The learning family separately proves scoped concentration, PAC-Bayes, posterior-gap, mixture-regret, and Bayes-factor results. No theorem yet turns these carriers into posterior contraction, minimax or empirical-calibration guarantees, or a generic marginal-likelihood optimum. Likewise, continuous-state active inference, generic blanket existence, multidimensional dual-affine geometry, and SDE/PDE dynamics must enter through new typed objects and witnessed seams. This rule turns "deepen the formalism" into a falsifiable acceptance contract rather than an invitation to accumulate unrelated lemmas.

5.10 A Checked Finite FEP and Active-Inference Kernel

The catalogue's 155 topic bodies are deliberately heterogeneous. To test whether their central ideas can inhabit coherent formal substrates, the package maintains an explicit resource manifest containing reusable foundations, leaf cross-topic compositions, and an import-only aggregate. The foundations contribute 475 checked declarations, while all maintained formal resources contribute 602 declarations. These declarations are counted separately from the 487 topic theorems, so the reported total of 1089 cannot hide duplication between generated examples and the reusable kernel.

The kernel follows the finite-state presentation common in discrete active inference (Da Costa et al. 2023), but it does not treat finiteness as evidence for an unrestricted FEP. Every probability carrier records nonnegativity and normalization; every logarithmic identity states its support hypotheses; every posterior states positive evidence; every asymptotic result states its integrability and independence assumptions. This is also a response to critiques that blanket, stationarity, and inference claims can become stronger than their premises (Biehl, Pollock, and Kanai 2021).

Open the offline interactive validation dashboard. Its filters expose all fifteen family diagnostics, while accessible tables retain the exact parameters, theorem mirrors, typed per-check relations and tolerances, and boundary observations without requiring a network connection.

5.10.1 Probability and information carriers

`FEP.FiniteLaw α` stores a real mass function with proofs of

$$p(x) \geq 0, \qquad \sum_{x \in \alpha} p(x) = 1.$$

Point masses, uniform laws, independent products, marginals, and deterministic pushforwards preserve this carrier. `FEP.FiniteKernel α β` similarly stores normalized rows. Its identity kernel and sequential composition satisfy left identity, right identity, and associativity, while predictive propagation satisfies the matching identity and composition laws. Consequently, multi-step prediction is not a fresh normalization argument at each horizon; it is inherited from a checked kernel algebra.

The information layer uses Mathlib's zero-safe `Real.negMulLog` and `InformationTheory.klFun`. It proves entropy nonnegativity, KL nonnegativity, support-free separation for normalized finite laws, conditional KL separation under positive input-law support but without reference-row support, cross-entropy identities, exact prior–kernel KL chain rules, marginal KL domination, independent-product KL additivity, entropy chain and product laws, and support-free mutual-information separation. The logarithmic identities and chain laws retain

Figure 3: One deterministic numerical witness for each expanded formal family. The panels exercise reconstruction, normalization, update, rank, contraction, and concentration boundaries, but they are explanatory projections rather than proof receipts.

the positive-reference assumptions used by their derivations. This matters because the real-valued definition is totalized: for disjoint Boolean point masses its cross-entropy is zero and its KL value is one, not positive infinity. In particular, for finite laws p and q,

$$I(p \otimes q) = 0$$

holds without adding artificial positivity assumptions at zero-mass atoms.

5.10.2 Variational inference and the evidence bound

A single `GenerativeModel Policy State Outcome` owns the initial state law, policy-conditioned transition, likelihood, preferred-outcome law, and policy prior. From that carrier the library derives predicted states, predicted state–outcome joints, predicted outcomes, and exact posterior states. Bayes reconstruction is an equation, not a prose contract:

$$Q(s \mid o, \pi)P(o \mid \pi) = Q(s \mid \pi)P(o \mid s).$$

For any recognition law R and positive-evidence outcome, posterior-form variational free energy is

$$F[R, o, \pi] = D_{\mathrm{KL}}(R \,\|\, P(s \mid o, \pi)) - \log P(o \mid \pi).$$

The checked theorems prove $-\log P(o \mid \pi) \le F[R, o, \pi]$, equivalently $-F[R, o, \pi] \le \log P(o \mid \pi)$. The exact posterior attains the bound, and equality is equivalent to $R = P(s \mid o, \pi)$ even when the posterior has zero-mass states. Thus the package contains a genuine variational minimization result rather than only a definition named "free energy."

5.10.3 Expected free energy, policy selection, and planning

The same model defines preference risk, likelihood ambiguity, epistemic value as mutual information, pragmatic cost, and real-valued expected free energy. Under the explicit `FullSupport` contract it proves both central decompositions

$$G(\pi) = \text{pragmaticCost}(\pi) - \text{epistemicValue}(\pi) = \text{risk}(\pi) + \text{ambiguity}(\pi),$$

and hence nonnegativity in this finite convention. The literature contains several EFE formulations whose equivalence depends on modeling assumptions (Millidge, Tschantz, and Buckley 2021; Champion et al. 2026); the theorem therefore fixes its sign and support conventions in the type-checked statement rather than relying on terminology.

A prior-weighted Boltzmann law

$$Q(\pi) \propto P(\pi) \exp[-\gamma G(\pi)]$$

has a strictly positive partition and a normalized posterior under full model support; a zero-mass preferred outcome is explicitly rejected before this surface is available. Equal-prior policies are ordered antitonically by EFE for nonnegative precision, a finite MAP policy exists, and an EFE minimizer exists. An action interface requires each emitted action to recover its policy's transition. The resulting infer–select–act joint is normalized, factors into posterior policy mass and action-state kernel mass, and has exactly the advertised policy and action marginals. A symmetric two-policy, two-state, two-observation model has uniform predictions, zero preference risk, and EFE equal to uniform Boolean entropy. Because both policies have equal EFE, changing the prior changes the posterior mass of `true` from $3/4$ to $1/4$ at every precision, pinning the prior term numerically.

For temporal depth, a list of policies composes the transition kernels chronologically. Empty plans preserve the initial law; concatenated plans obey prefix–suffix prediction; and a positive-mass terminal observation yields an exact planned posterior reconstruction. A second recursion updates the predicted state law after every stage and accumulates stage-dependent EFE. Its value decomposes exactly at every plan concatenation and is nonnegative under a recursive stagewise support contract. This carrier is finite-horizon and open-loop. The controlled-Markov foundation first adds an exact two-stage observation-dependent feedback witness. The later `FEP.PolicyTrees` foundation defines arbitrary finite-depth observation-contingent trees on finite carriers, proves recursive Bellman optimality and optimizer existence, embeds open-loop plans, proves closed-loop weak dominance, and lifts the EFE decomposition treewise. It does not supply an infinite-horizon partially observed stochastic game or learn a posterior over trees.

5.10.4 Blanket factorization and dynamics

The static blanket carrier constructs

$$P(b, i, e) = P(b)P(i \mid b)P(e \mid b).$$

At positive blanket mass, division by $P(b)$ yields the exact conditional product. Independently, the blanket-indexed conditional law has zero internal–external mutual information. This gives a concrete finite conditional-independence witness and strengthens the catalogue-level bridge between blanket partitions and conditional independence. The later `FEP.NativeBlanket` foundation embeds this carrier into native measures and kernels, transfers singleton masses, expectations, and prediction, and proves Mathlib's `CondIndepFun` for the embedded static joint plus measurable endpoint coarsening and rowwise factorized transitions. It still does not prove that arbitrary dynamics admit or preserve a blanket under arbitrary mixtures.

The dynamic carrier makes the permitted dependency graph part of the transition type: internal and active updates see the current internal–sensory pair, while sensory and external updates see the current external–active pair. Their product is a normalized four-component transition with an exact factorization theorem. Every transition row is equal to a static joint derived from the current state; under positive next-blanket mass it therefore inherits conditional factorization and zero conditional mutual information. A Boolean witness changes state with probability one, ruling out a vacuous identity-only realization. These are row-wise results and do not assert stationarity or preservation after an arbitrary prior mixture.

5.10.5 Geometry and asymptotics

A d-dimensional centered score field induces a symmetric Fisher matrix and a Gram metric. Positive semidefiniteness is unconditional; positive definiteness requires full support and score identifiability; pullback positivity additionally requires an injective Jacobian; and composite Jacobians obey a functorial pullback law. Matrix lowering is proved equal to the original expected-score metric. When the Fisher matrix is invertible, the library constructs the inverse-Fisher natural gradient and proves its matrix duality, metric duality, Fisher-energy identity, and uniqueness. An interior Bernoulli model realizes the one-dimensional case with Fisher entry $1/[p(1-p)]$ and natural-gradient scaling $p(1-p)$; at $p = 1/2$ its scores are ± 2 and its Fisher entry is four. A duplicated-score two-parameter model supplies the opposite witness: an all-four Fisher matrix, a nonzero null tangent, zero Fisher norm, and failed identifiability. The nondegeneracy premises are therefore mathematically active.

For data, the topic-independent foundation instantiates Mathlib's almost-sure strong law for Boolean indicators and finite-atom indicators. Integrability, pairwise independence, and identical distribution remain theorem premises. It proves raw empirical-rate convergence, simultaneous convergence of every atom outside one null set, convergence of the whole finite empirical law in L^1, and convergence of empirical expectations and their absolute errors for every real observable on the finite carrier. The Laplace-smoothing transfer is separately proved in a composition leaf because it consumes fep-036. These are asymptotic consistency results. The learning family adds scoped finite-sample concentration and regret statements; the later empirical-risk family connects the exact Laplace estimator to finite-law squared/Brier-risk transfer and concentration-event containment. Neither surface proves posterior contraction, minimax optimality, empirical calibration, or empirical adequacy for an observed data set.

5.10.6 What the kernel proves—and what it does not

The kernel makes several high-value seams executable in one carrier: Bayes reconstruction, the variational evidence bound, both EFE decompositions, policy normalization and action pushforward, horizon composition, blanket conditional information, Fisher raising/lowering, and strong-law consistency. The numerical dashboard above complements this deductive surface with one diagnostic for each later expansion family; it is not a sample-based proof of every seam in this chapter. The formalism atlas shows the generated module-dependency graph separately from authored scientific relations: purple import arrows describe code reuse, while teal edges require named theorem witnesses.

The remaining boundary is deliberate. State spaces and policy-tree carriers are finite; logarithmic EFE identities and treewise decomposition require stated full support even though finite-KL and VFE separation do not; the real-to-`ENNReal` bridge truncates negative values through `ENNReal.ofReal`; native blanket transfer is rowwise rather than a generic existence theorem; Fisher inversion and mean-coordinate injection retain nondegeneracy premises; finite Laplace risk is not posterior contraction; and the exact continuous-time family is Boolean rather than an SDE/PDE model. These constraints are acceptance conditions for future extensions, not footnotes that disappear from the formal claims.

6 Expanded Formalism Program

The first expansion from 50 to 120 topics is organized around ten theorem families rather than a flat list of adjacent vocabulary. Each family has one reusable formal owner, seven topic-scoped projections, a nontrivial finite or measure-theoretic witness, and at least one bridge to an earlier declaration. This organization matters mathematically: a normalization lemma, a variational equality, a contraction theorem, and a path-space identity answer different questions even when all four are described informally as "free-energy minimization."

6.1 Bayesian inversion at two proof scales

The measure-theoretic layer starts from absolute continuity. For measures $\mu \ll \nu$, the Radon–Nikodym derivative reconstructs the dominated measure,

$$\nu\left[\frac{\mathrm{d}\mu}{\mathrm{d}\nu}\right] = \mu.$$

Posterior kernels then reconstruct the swapped prior–likelihood joint and, over standard Borel spaces, a conditional kernel disintegrates its joint measure (Rokhlin 1949). These are almost-everywhere statements with explicit finiteness and measurability premises. The finite layer has a different support boundary: at an evidence atom y with positive predictive mass,

$$q(x \mid y)\, q(y) = p(x)\, k(y \mid x).$$

The development keeps those two carriers connected without pretending that a finite pointwise quotient proves unrestricted regular conditional probability.

6.2 Variational duality and information bounds

For a finite reference law p with full support and a bounded potential f, the Gibbs variational identity has an explicit optimizer $q_f(x) \propto p(x)e^{f(x)}$:

$$\log \sum_x p(x)e^{f(x)} = \sum_x q_f(x)f(x) - D_{\mathrm{KL}}(q_f \| p).$$

This finite equality is the proof-relevant specialization of the variational formula associated with Donsker and Varadhan (Donsker and Varadhan 1975). It is not an asymptotic large-deviation theorem. The same family treats coordinate ELBO decomposition, mean-field coordinate minimization, channel data processing, and rate–distortion weak duality. Its importance-weighted row fixes a positive sample count and proves the Jensen lower bound for the product proposal law; it does not inherit neural-network or estimator-quality claims from the IWAE application (Burda, Grosse, and Salakhutdinov 2016).

6.3 Control, planning, and temporal inference

The controlled-kernel family distinguishes a normalized action-conditioned transition from a policy theorem. A finite belief index interprets each reachable node as a normalized state law, so finite-horizon belief-state reduction never asserts that the entire real probability simplex is finite. The soft Bellman and desirability recursions expose the passive dynamics, temperature, terminal value, and KL control cost used by linearly solvable control (Todorov 2006) and path-integral control (Kappen 2005). A separate sophisticated expected-free-energy recursion permits future observations to change a later action; an open-loop rollout is not accepted as a substitute.

The temporal family proves forward filtering, backward information messages, forward–backward smoothing, and smoothing normalization for a finite hidden Markov model. These equations share the sum–product structure underlying the classical finite-chain inference literature (Baum et al. 1970), but the formal statements concern exact finite recursions rather than parameter learning or asymptotic identification. A one-step normalized variational update, a two-level predictive factorization, and model-averaged prediction complete the family.

6.4 Causal blankets and predictive coding

Causal statements use an ordered finite factorization with named parent sets and intervention kernels. The local Markov theorem is therefore proved for that carrier only. An intervention witness changes a descendant while leaving a named non-descendant invariant. It does not claim general graphical d-separation or observational identifiability. This scoped construction is kept distinct from causal blankets defined through dynamical sufficient statistics (Rosas et al. 2020) and from the conditional-independence blanket carrier used elsewhere in the catalogue. This distinction also prevents the finite intervention carrier from inheriting the stationary-diffusion claims of Bayesian mechanics (Da Costa et al. 2021).

Predictive coding is represented by an explicit precision-weighted quadratic energy, its prediction-error gradient, a finite-jet shift, and a truncated generalized-filtering correction equation. Under the maintained quadratic curvature and step-size hypotheses, the correction decreases the energy. The formal target is thus a named finite update, not the claim that every neural implementation performs the hierarchical inference proposed in predictive coding accounts (K. Friston and Kiebel 2009).

6.5 Path thermodynamics and geometric optimization

The path-space family constructs normalized forward and reversed finite path laws before defining entropy production as their KL divergence. Detailed and integral fluctuation identities are then consequences of an explicit path-law ratio, while the Jarzynski (Jarzynski 1997) and Crooks (Crooks 1999) rows state the work, inverse-temperature, and normalization assumptions needed for their finite forms. A reversible-chain one-step KL dissipation theorem is deliberately narrower than a generic housekeeping– excess decomposition for time-dependent nonequilibrium processes.

The geometric family moves from one-dimensional Bernoulli witnesses to a finite categorical score model. Fisher positivity is restricted to simplex tangents; Cramér–Rao states unbiasedness and score regularity; natural-gradient equivariance requires an invertible full-rank chart; and the Bregman Pythagorean law names its affine projection and minimizer. These hypotheses are the formal content that a generic appeal to information geometry (Amari 1998, 2016) would otherwise hide.

6.6 Collective inference, dynamics, and learning

Collective inference begins with product generative laws and proves exactly when VFE and EFE add. The unit-weight product-of-experts pool has its own positive normalizer; it is not mislabeled as the half-exponent construction of an equal-weight logarithmic opinion pool. Consensus uses a stochastic averaging kernel with an explicit strict contraction coefficient, so convergence follows from a geometric bound rather than from the word "consensus." A shared finite-dynamics foundation proves Chapman–Kolmogorov composition, invariance and reversibility under powers, Dobrushin-bound multiplication, total-variation contraction, and master-equation mass conservation for the same normalized carrier. The collective interpretation is motivated by explicitly modeled multiagent systems (Heins, Klein, et al. 2022; K. J. Friston et al. 2024), while the contraction theorem uses the narrower stochastic-consensus mathematics reviewed by (Olfati-Saber, Fax, and Murray 2007). The formal result therefore establishes neither an emergent group agent nor a shared generative model unless those structures are supplied as hypotheses.

The learning family separates concentration from Bayesian algebra. It proves a sub-Gaussian empirical-mean tail bound, a simultaneous finite-alphabet frequency bound, and a finite-hypothesis PAC-Bayes loss-gap bound in the lineage of McAllester's PAC-Bayesian guarantees (McAllester 1999). The last result is deterministic conditional on a Gibbs loss-gap certificate and a log-MGF confidence budget; it does not itself derive that budget with high probability over sampled data. Its bounded-variable specialization is scoped by the independence and range hypotheses of Hoeffding's inequality (Hoeffding 1963). Posterior-odds recursion, likelihood-gap concentration, mixture log-loss regret, and Bayes-factor multiplication then expose their prior-support and likelihood premises. None of these results is empirical evidence that an FEP model fits a biological system.

6.7 Evidence boundary

For the preceding 120-topic snapshot, each family was assessed through four independent checks: native Lean compilation with no `sorry`, a trusted-axiom/declaration audit, a deterministic numerical witness, and semantic review against the cited formulation. The current dashboard schema expresses numerical acceptance through typed checks with per-check tolerances. Those retained receipts remain historical for their exact source; this chapter does not promote them to evidence for the 155-topic roster. A numerical witness can reveal a sign error, normalization failure, rank boundary, or unstable parameter region. It cannot upgrade a scoped finite theorem into a physical law, and it is never counted as proof evidence.

The five families that extend this program from 120 to 155 topics are treated in §7; their local evidence does not retroactively change the receipts for this first expansion.

7 From 120 to 155: Risk, Feedback, Native Blankets, Dual Geometry, and Continuous Time

The second catalogue expansion adds thirty-five stable rows, `fep-121` through `fep-155`, in five seven-topic families. Its purpose is not numerical growth. Each family closes a boundary that was explicit in the preceding 120-topic snapshot: finite-sample control for the Laplace estimator, genuinely observation-contingent policy trees, a bridge from finite blanket factors to Mathlib's native conditional-independence predicate, a differentiable scalar exponential family with dual identities, and an exact continuous-time Markov semigroup. The maintained novelty ledger requires every new row to consume at least one earlier topic through a named `FEPComposed` theorem. The result is a directed extension of the existing kernel rather than a second list of similarly named lemmas.

The five carriers remain deliberately narrow. Sampling laws, action spaces, blanket coordinates, and exponential-family outcomes are finite. The continuous-time family is an exact Boolean chain rather than a generic continuous-state diffusion. These restrictions are theorem hypotheses and scope boundaries, not implementation accidents.

7.1 Toolchain and evidence identity

The authoring workspace pins **`leanprover/lean4:v4.33.1`** and Mathlib **`v4.33.1`**, with the exact dependency revision recorded in `lean/lake-manifest.json`. Lean 4.33.0 introduced the relevant release line (Lean FRO 2026a), while the package uses the subsequent stable Lean patch (Lean FRO 2026b) and matching Mathlib release (The mathlib Community 2026). A release tag, source file, or

successful one-file probe is not a catalogue receipt. A native claim for this expansion requires a fresh exact-roster receipt whose ordered IDs, source digests, actual compiler version, resolved Mathlib revision, warnings, and `sorry` count all reconcile against the live 155-topic source.

Four evidence planes must therefore be read separately:

Plane	What it can establish	What it cannot establish
Semantic review	The maintained invariant, assumptions, non-vacuity argument, and disposition for each topic	Kernel acceptance or empirical adequacy
Native Lean and declaration receipts	Acceptance of the exact propositions at the pinned compiler; declaration resolution and trusted-axiom closure when the corresponding receipt validates	That hypotheses hold in a biological system, that a numerical plot is representative, or that a provider ran
Deterministic numerical witnesses	Exact finite evaluations of named theorem instances and boundary checks	Deductive proof, stochastic calibration on data, or extension beyond the sampled carrier
Provider-backed full execution	Hermes/OpenGauss execution for the exact source only when a complete report independently validates	A stronger theorem, physical truth of the FEP, or evidence for any later source snapshot

The current schema-4 declaration/axiom receipt resolves all 823 required formal-resource declarations, including 699 evidence declarations, under Lean 4.33.1 and the locked Mathlib revision with zero warnings, no `sorryAx`, and no untrusted axiom. The current exact-roster native receipt validates all 155 topics under the same toolchain with zero failures, warnings, or `sorry`. The schema-3 Python receipt binds the complete canonical collected-node roster, zero failures or errors, and line coverage at or above the maintained 89% floor; the schema-4 Chrome replay binds six accepted screenshots to all 20 families, 155 topics, and 15 typed witnesses. The retained provider report binds an earlier 50-topic snapshot and is not silently promoted to current 155-topic provider evidence. At render time, `true` and `false` report the independently validated native and full evidence states; unavailable evidence remains unavailable rather than being filled from an older run.

7.2 Finite-sample risk and calibration (`fep-121`–`fep-127`)

Let K be the number of successes in a positive sample of size n, let $\hat{p} = K/n$, and let the add-one estimate be $\tilde{p} = (K+1)/(n+2)$. The `FEP.EmpiricalRisk` foundation exposes the exact error decomposition

$$\tilde{p} - p = \frac{n}{n+2}(\hat{p} - p) + \frac{1-2p}{n+2}. \tag{71}$$

For $p \in [0,1]$, the offset has absolute value at most $1/(n+2)$. Squaring with the explicit factor-two inequality and integrating under any normalized finite law gives

$$\mathbb{E}[(\tilde{p} - p)^2] \le 2\left(\frac{n}{n+2}\right)^2 \mathbb{E}[(\hat{p} - p)^2] + \frac{2}{(n+2)^2}. \tag{72}$$

For a Bernoulli target, the Brier score has the exact excess-risk identity

$$\mathrm{BS}(p,q) - \mathrm{BS}(p,p) = (q-p)^2, \tag{73}$$

which specializes Equation 72 to finite-law Brier excess risk. This formal target follows the proper probabilistic scoring rule introduced by Brier (Brier 1950), while making no empirical forecast-calibration claim.

Topic	Primary and boundary declarations	Explicit assumptions and non-vacuity
`fep-121`	`fep121_laplaceError_identity`; `fep121_shrinkage_mem_unitInterval`	$n > 0$, $K \le n$, and $p \in [0,1]$. The shrinkage coefficient is in the unit interval; the identity is algebraic, not a consistency theorem.
`fep-122`	`fep122_laplaceBias_abs_le`; `fep122_nonzero_boundary_witness`	$p \in [0,1]$. At $n = 2, p = 0$, the offset is exactly $1/4 \neq 0$, so smoothing is not mislabeled as unbiased.

Topic	Primary and boundary declarations	Explicit assumptions and non-vacuity
fep-123	fep123_laplaceAbsoluteError_le; fep123_error_nonnegative	A raw absolute-error certificate is supplied in addition to the positive-sample and unit-interval hypotheses. The theorem transfers that certificate; it does not derive a tail rate.
fep-124	fep124_laplaceSquaredError_le; fep124_laplaceSquaredRisk_le	A normalized finite sampling law and bounded success counts. Both raw error and the pseudo-count offset can contribute positively to the bound.
fep-125	fep125_brierExcess_eq_sqError; fep125_brierExcess_self	The polynomial identity is real-valued; its probability interpretation additionally uses $p, q \in [0, 1]$. Unequal forecasts have positive squared excess, while $q = p$ is the zero boundary.
fep-126	fep126_laplaceBrierRisk_le; fep126_sampling_mass_one	The same finite-law, count, positivity, and target hypotheses as the squared-risk theorem. A nondegenerate law yields a genuine weighted risk rather than a single substituted value.
fep-127	fep127_laplaceBadEvent_subset; fep127_laplaceBadEvent_probability_le	A raw event-probability bound is supplied. Event containment transfers the bound monotonically; it does not manufacture a concentration inequality or posterior-contraction rate.

The family closes a finite risk-transfer gap for the exact Laplace estimator. It does **not** prove minimax optimality, frequentist calibration from observed data, posterior contraction, or a marginal-likelihood optimum. Those would require a sampling model and theorem target beyond the finite-law transfer proved here.

7.3 Closed-loop policy trees and expected free energy (fep-128–fep-134)

A depth-indexed `PolicyTree Action Observation d` chooses an action at each node and an observation-indexed continuation tree. Given a finite `PolicyTreeModel`, its recursive value is

$$V_{d+1}(b; (a, \tau)) = c_d(b, a) + \sum_o P(o \mid b, a) V_d(u(b, a, o); \tau(o)) \,. \tag{74}$$

Finite nonempty actions make the Bellman minimizer concrete at every node. Open-loop plans embed by using the same continuation after every observation, so the optimal policy-tree value is no greater than the value of any embedded open-loop plan. The treewise EFE row lifts the package's existing one-step risk-plus-ambiguity identity under its full-support contract. This is the finite recursive structure motivated by sophisticated active inference (K. Friston et al. 2020), not a claim about every planning algorithm or EFE convention.

Topic	Primary and supporting declarations	Explicit assumptions and non-vacuity
fep-128	fep128_policyTreeValue_node; fep128_policyTreeValue_leaf	Finite belief, action, and observation carriers and a finite natural-number depth. Observation laws and deterministic updates are supplied. Distinct observation branches may select distinct continuations.
fep-129	fep129_optimalTreeValue_eq_min; fep129_optimalTreeAction_le	The finite action carrier is nonempty. The selected action is proved no worse than every alternative, not merely named as an optimizer.
fep-130	fep130_exists_optimalPolicyTree; fep130_policyTree_carrier_finite	All carriers and the horizon are finite. The witness is a concrete depth-indexed tree attaining the recursive optimum.

Topic	Primary and supporting declarations	Explicit assumptions and non-vacuity
fep-131	fep131_openLoopEmbedding_value; fep131_openLoopEmbedding_leaf	Both sides share one model, belief, and depth. Positive-depth embedding repeats one continuation across all observation branches; the zero-depth carriers agree at the leaf.
fep-132	fep132_optimalTree_le_openLoop; fep132_finite_horizon_depth	Finite nonempty actions and a common finite cost model. The theorem proves weak dominance for every embedded plan; strictness is supplied separately by fep-134.
fep-133	fep133_policyTree_efe_eq_risk_ambiguity; fep133_optimalValues_agree	Each belief-indexed generative model satisfies the maintained FullSupport contract. The result transports an existing identity; it does not validate the EFE model empirically.
fep-134	fep134_boolFeedback_strictlyBetterThanOpenLoop; fep134_feedback_continuation_gap_eq	A fair Boolean observation and mismatch cost. Feedback has value 0, each fixed second action has value $1/2$, and the continuation action differs across the two observations.

The Boolean witness rules out a vocabulary-only notion of feedback: its two continuations are propositionally unequal and its value gap is exactly one half. The remaining planning frontier is not finite policy-tree existence. It is inference or learning over policy-tree distributions, infinite or continuous belief spaces, uncertain model parameters, alternative EFE equivalence conditions, and refinement to an executable online agent.

7.4 Finite-to-native blanket transfer (fep-135–fep-141)

For a normalized finite law p, FEP.NativeBlanket.embeddedLaw constructs the native measure

$$\mu_p = \sum_x \text{ofReal}(p(x))\, \delta_x. \tag{75}$$

Singleton evaluation reflects the original mass, equality of embedded measures is injective back to finite laws, native integration agrees with the finite weighted sum, and finite prediction agrees with Mathlib measure–kernel composition. For a static blanket model, the embedded joint retains the rectangle factorization

$$\mu(b, i, e) = p_B(b)\, p_I(i \mid b)\, p_E(e \mid b). \tag{76}$$

The key step is not a zero-mutual-information proxy. The theorem fep139_staticJoint_condIndepFun states Mathlib's native CondIndepFun for the internal and external coordinate maps conditioned on the blanket coordinate.

Topic	Primary and supporting declarations	Explicit assumptions and non-vacuity
fep-135	fep135_embeddedLaw_apply_singleton; fep135_embeddedLaw_normalized	A finite discrete carrier and a normalized nonnegative finite law. A nonuniform law preserves distinct native singleton masses while total mass remains one.
fep-136	fep136_embeddedLaw_injective; fep136_embeddedLaw_expectation	A finite discrete carrier and real observable. Distinct point masses remain distinguishable, and a nonconstant observable yields a nontrivial transferred expectation.
fep-137	fep137_embeddedPredictive_eq_comp	Finite discrete input/output carriers and normalized kernel rows. A nondegenerate prior with state-dependent rows yields a genuine mixture on both carriers.
fep-138	fep138_staticJoint_rectangle_factorization	Finite blanket, internal, and external coordinates with normalized conditional rows. The correlated Boolean model has two distinct positive blanket/joint atoms.

Topic	Primary and supporting declarations	Explicit assumptions and non-vacuity
`fep-139`	`fep139_staticJoint_condIndepFun`, `fep139_correlatedBlanket_nonvacuous`	Finite discrete standard-Borel carriers, nonempty internal/external types, and exact factorized rows. The Boolean witness has two positive correlated blanket regimes, so independence is not obtained by collapsing the entire law to one point.
`fep-140`	`fep140_condIndepFun_measurable`	Measurable maps of internal and external coordinates. Identity maps recover `fep-139`; noninjective coarsenings can merge endpoint states without changing the conditional-independence conclusion.
`fep-141`	`fep141_prediction_preserves_natural_blanket`	A finite factorized dynamics row and one supplied current state. Multiple positive next-blanket regimes and nonconstant endpoint laws are permitted. Arbitrary mixtures over uncertain current states are not covered.

This family closes one concrete finite-to-native conditional-independence seam. It does not prove that every system admits a blanket, that arbitrary mixtures preserve rowwise factorization, that a blanket is stationary or attracting, or that the authored finite coordinates identify a biological boundary.

7.5 Finite exponential-family dual geometry (`fep-142`–`fep-148`)

For a finite nonempty outcome carrier, positive base weights $h(x) > 0$, and a real sufficient statistic $T(x)$, the scalar family is

$$p_\theta(x) = \frac{h(x)e^{\theta T(x)}}{Z(\theta)}, \qquad A(\theta) = \log Z(\theta). \tag{77}$$

The maintained source proves normalization and full support, the affine log-density ratio, $A'(\theta) = \mathbb{E}_\theta[T]$, the centered-score identity, and

$$A''(\theta) = \mathrm{Var}_\theta[T] = I(\theta), \qquad D_{\mathrm{KL}}(p_\theta \| p_\eta) = A(\eta) - A(\theta) - A'(\theta)(\eta - \theta). \tag{78}$$

These are the finite scalar counterparts of the metric and affine structures that motivate information geometry (Amari 1983). They do not by themselves construct the full manifold, dual affine connections, or curvature described in the general theory.

Topic	Primary and supporting/boundary declarations	Explicit assumptions and non-vacuity
`fep-142`	`fep142_exponentialFamily_sum_ofReal_pos`, `fep142_exponentialFamily_pointwise_pos`	Finite nonempty outcomes and strictly positive base weights. A maintained nonconstant three-state family has unequal positive masses away from zero.
`fep-143`	`fep143_logDensityRatio_eq`	Full support makes every logarithm finite. Distinct parameters with a nonconstant statistic produce outcome-dependent ratios.
`fep-144`	`fep144_logPartition_hasDerivAt`	A finite sum of positive exponential weights. No infinite-sum interchange or arbitrary-manifold differentiability is inferred.
`fep-145`	`fep145_score_eq_statistic_sub_mean`, `fep145_score_mean_zero`	The scalar natural-parameter score is defined on the full-support family. A nonconstant statistic yields positive and negative scores whose weighted mean cancels.

Topic	Primary and supporting/boundary declarations	Explicit assumptions and non-vacuity
`fep-146`	`fep146_logPartition_secondDeriv_eq_variance`; `fep146_fisher_eq_variance`; `fep146_threeState_variance_pos`; `fep146_constantStatistic_zero_variance`	The finite scalar family and the proved first derivative. For statistic $(0, 1, 2)$ with unit basis weight, variance at zero is $2/3 > 0$; a constant statistic has exactly zero variance and Fisher information.
`fep-147`	`fep147_exponentialFamily_KL_eq_zero_iff`; `fep147_exponentialFamily_fullSupport`	Two members of the same supported scalar family. Equal parameters give the zero-KL boundary through the shared finite-KL separation theorem. The companion three-state family has variance $2/3$, while the constant-statistic family has zero variance; this row does not add a separate strict-KL witness for unequal parameters.
`fep-148`	`fep148_meanParameter_strictMono`; `fep148_meanParameter_injective`	Variance is explicitly positive throughout a stated closed interval. The theorem is interval-local and does not infer positivity for degenerate statistics.

The three-state and constant-statistic witnesses make the rank premise visible: strict mean-coordinate monotonicity is supported by positive variance, while a constant statistic supplies an exact zero-information boundary.

7.6 Two-state continuous-time thermodynamics (`fep-149`–`fep-155`)

Let $a > 0$ be the false-to-true rate and $b > 0$ the true-to-false rate. The generator and stationary law are

$$Q = \begin{pmatrix} -a & a \\ b & -b \end{pmatrix}, \qquad \pi = \left(\frac{b}{a+b}, \frac{a}{a+b} \right). \tag{79}$$

Using $\rho(t) = e^{-(a+b)t}$, the source defines the four entries of an exact transition matrix P_t. For nonnegative time the rows are nonnegative and normalized; for all real s, t, the closed form satisfies

$$P_{s+t} = P_s P_t, \qquad \frac{\mathrm{d}}{\mathrm{d}t} P_t = Q P_t = P_t Q. \tag{80}$$

The stationary law is invariant and obeys detailed balance. If m_t is the true-state mass from an arbitrary initial finite law, then

$$m_t - \pi_1 = e^{-(a+b)t}(m_0 - \pi_1), \qquad L(t) = (m_t - \pi_1)^2 = e^{-2(a+b)t} L(0), \qquad L'(t) = -2(a+b)L(t). \tag{81}$$

Topic	Primary and supporting declarations	Explicit assumptions and non-vacuity
`fep-149`	`fep149_twoStateSemigroup_rowSums`; `fep149_twoStateSemigroup_nonnegativity`; `fep149_benchmarkRates_exact`	Positive rates and nonnegative time for stochasticity. Rates 0.7 and 0.3 give decay rate one and a nonuniform stationary law.
`fep-150`	`fep150_twoStateSemigroup_zero`	Positive rates. All four Boolean entries distinguish the identity boundary: diagonal one, off-diagonal zero.
`fep-151`	`fep151_twoStateSemigroup_add`	Positive rates; the stochastic interpretation uses nonnegative times. Unequal rates and positive times give nonidentity kernels whose product is still exact.
`fep-152`	`fep152_twoStateSemigroup_hasDerivAt`	The maintained explicit Boolean generator and transition formulas. Both left and right generator products are proved entrywise; no general CTMC existence theorem is inferred.

Topic	Primary and supporting declarations	Explicit assumptions and non-vacuity
fep-153	fep153_twoStateSemigroup_stationary; fep153_twoStateSemigroup_detailedBalance	Positive rates and nonnegative time for the kernel. Unequal rates give a nonuniform stationary law while opposing fluxes balance exactly.
fep-154	fep154_twoStateRelaxation_exact; fep154_benchmarkInitial_nonstationary	An arbitrary normalized Boolean initial law. The false point mass differs from the benchmark stationary law, so the relaxed coordinate starts nonzero.
fep-155	fep155_twoStateLyapunov_exact; fep155_twoStateLyapunov_hasDerivative; fep155_benchmarkLyapunov_strictlyDecreasing	The scalar squared stationary deviation for the exact two-state chain. At the nonstationary benchmark, the derivative at time zero is strictly negative.

This family adds genuine continuous time, a semigroup, and a master equation at one exact finite-state scope. It is not a Langevin SDE, a Fokker–Planck PDE, a generic finite-state CTMC construction, a nonequilibrium driven steady state, or an identification of the quadratic Lyapunov function with thermodynamic or variational free energy.

7.7 Composition ledger for the thirty-five rows

Each new row has a manifested leaf theorem that consumes the new topic and at least one earlier endpoint. The bridge names below are part of the novelty contract; their existence prevents an import graph from being reported as a scientific derivation.

New topics	Manifested composition declarations
fep-121-fep-127	fep121_laplaceError_extends_fep036; fep122_laplaceBias_extends_fep036; fep123_laplaceAbsoluteError_extends_fep036; fep124_laplaceSquaredRisk_combines_fep036_fep11 fep125_brierExcess_refines_fep022; fep126_laplaceBrierRisk_combines_fep022_fep036; fep127_laplaceConcentration_combines_fep036_fep
fep-128-fep-134	fep128_policyTreeRecursion_extends_fep071; fep129_policyTreeBellman_extends_fep033; fep130_optimalPolicyTree_extends_fep008; fep131_openLoopEmbedding_extends_fep033; fep132_closedLoopDominance_extends_fep071; fep133_treewiseEFE_extends_fep021; fep134_feedbackWitness_extends_fep071
fep-135-fep-141	fep135_embeddedLaw_extends_fep017; fep136_embeddedExpectation_extends_fep015; fep137_embeddedPredictive_extends_fep019; fep138_rectangleFactorization_extends_fep079; fep139_nativeCondIndep_connects_fep009_fep079; fep140_measurableCoarsening_extends_fep009; fep141_blanketTransition_extends_fep080
fep-142-fep-148	fep142_exponentialNormalization_extends_fep031; fep143_logDensityRatio_extends_fep026; fep144_logPartitionGradient_extends_fep040; fep145_centeredScore_extends_fep038; fep146_fisherVariance_extends_fep100; fep147_KLBregman_connects_fep014_fep104; fep148_meanCoordinate_extends_fep103

New topics	Manifested composition declarations
fep-149-fep-155	fep149_continuousKernel_extends_fep020; fep150_semigroupZero_extends_fep006; fep151_semigroupAdd_extends_fep006; fep152_masterEquation_extends_fep020; fep153_continuousDetailedBalance_extends_fep010 fep154_continuousRelaxation_extends_fep020; fep155_lyapunovDecay_extends_fep032

These bridges are primarily formal_pairing witnesses: they preserve both endpoint laws in one checked conclusion without claiming that one topic follows from the other. A derivational formal edge is reserved for a theorem that actually derives or identifies its target.

7.8 Validation and visualization contract

The dashboard now supplies one typed deterministic witness for each of the fifteen expansion families. The five new witnesses are not generic plots:

Witness	Exact checks and boundary
laplace-brier-risk	finite sampling mass equals one; Brier risk is below the transferred bound; the $n = 2, p = 0$ bias equals $1/4 \neq 0$
policy-tree-feedback	feedback value equals zero; fixed plans equal one half; feedback is no worse; the two continuation actions differ
native-blanket-transfer	the two positive regimes sum to one; rectangle factorization is exact; the off-regime has zero mass; every conditional product row is present
exponential-family-duality	both laws normalize; the score is centered; finite KL equals log-partition Bregman divergence; three-state variance is $2/3$; the constant-statistic boundary is zero
two-state-master-equation	row normalization, semigroup addition, master equation, detailed balance, and relaxation all close numerically; the benchmark Lyapunov derivative is negative

Each check has a typed relation (eq, le, ge, or predicate), explicit operands, and its own tolerance. Acceptance is the conjunction of all checks and the named boundary observation. This representation avoids hiding a failed inequality behind an unrelated small residual. The static SVG and interactive HTML are projections of the same immutable witness model; the HTML retains the exact check table for screen-reader and nonvisual inspection.

The formalism atlas answers a different question. It separates authored scientific relations from module-import dependencies, preserves capability status, and names every formal witness. Agreement between the atlas and the dashboard is useful diagnostic evidence, but neither replaces an exact-roster native receipt or declaration/axiom audit.

7.9 Residual frontier after 155 topics

The expansion changes the location of the open boundary. It is no longer accurate to say that the package has no finite-sample Laplace risk theorem, no finite policy-tree carrier, no native blanket conditional-independence bridge, no differentiable exponential family, or no continuous-time Markov example. The remaining claims are broader and materially harder:

1. posterior contraction, minimax or calibration guarantees, and empirical marginal-likelihood objectives on a shared sampling model;
2. policy-tree distributions, parameter learning, alternative EFE equivalence, continuous beliefs, infinite horizon, and executable-agent refinement;
3. blanket existence, stationarity, and preservation under arbitrary prior mixtures or general stochastic dynamics;
4. multidimensional exponential families, dual affine connections, curvature, geodesics, and general constrained maximum entropy; and
5. generic CTMC construction, continuous-state SDE/PDE existence, driven nonequilibrium steady states, and an explicit physical free-energy model.

Progress on these targets requires new carriers and witnessed seams, not a stronger gloss on the five finite families above.

8 Discussion: Ecosystem Maturity and Formalization Impacts

The central finding is not a binary judgment that Mathlib either does or does not support the FEP. Support is layered. At the pinned revision, the library supports direct native implementations of posterior kernels, KL divergence, Gaussian laws, conditional independence, Markov-kernel invariance, differentiation, finite information geometry, and convergence arguments. Project-authored definitions and composition theorems turn those primitives into 136 directly formalized topic scopes. The package now includes one exact finite-state continuous-time chain; general continuous-state stochastic dynamics and system-level FEP claims still require new domain interfaces.

8.1 Maturity Assessment of the Mathlib Ecosystem

Two maturity axes must remain separate. `mathlib_status` records whether a catalogue body targets available pinned infrastructure. Semantic disposition records whether its primary theorem matches the reviewed topic claim. All 155 rows currently have `mathlib_status: real`, while 136 are classified as directly formalized and the remainder retain explicit conditional or structural scopes. The fep-036 topic uses Mathlib's finite binomial PMF and sequence-limit APIs to prove an outcome-indexed Laplace prior, exact shrinkage, consistency transfer, and normalized posterior closure. The convergence module generates the empirical-frequency premise and whole-law limits; the learning module separately proves sub-Gaussian and finite-alphabet concentration, PAC-Bayes, posterior-gap, regret, and Bayes-factor laws; and fep-121–127 prove finite-law squared/Brier-risk and bad-event transfer for the same estimator. Posterior contraction, minimax or empirical-calibration guarantees, and a marginal-likelihood optimum remain absent.

8.1.1 Coverage by Area

The generated coverage report computes area counts, semantic disposition totals, declarations, imports, capability states, and authored relation witnesses from canonical sources. It should be consulted instead of a hand-maintained roster. Shared library coverage is strongest for finite sums, real analysis, measures, probability kernels, logarithms, exponentials, differentiation, matrices, and strong laws. The project now exercises all of these surfaces directly. Coverage remains narrower for smooth statistical manifolds, continuous stochastic processes, and generic blanket-existence claims.

8.1.2 Module-Level Maturity and Compilation Outcomes

The module-to-topic incidence table in `docs/formalism-coverage.md` is derived from actual `import` commands. An import is evidence that a topic uses a library surface; it is not evidence that two topics depend logically on each other, nor that a named scientific construct has been formalized. Compilation outcomes come only from the selected receipt (`native-lean`, rate `155/155 (100.0%)`).

8.1.3 The Mathlib Frontier for Deeper Formalization

The pinned Mathlib revision already provides `InformationTheory.klDiv` in $\mathbb{R}\geq 0\infty$, self-divergence, zero characterization for finite measures, Gibbs' inequality, composition-product chain rules, native posterior kernels, and a real strong law. Rows fep-002, fep-014, fep-017, fep-034, and fep-041 use those measure APIs directly, while the maintained finite kernel supplies a real KL carrier, posterior VFE, mutual-information EFE decomposition, and exact policy model. The frontier is therefore no longer "define KL, Bayes, or EFE somehow"; it is comparison across alternative EFE conventions, data processing at the pin, and measure-valued identities under explicit finiteness and absolute-continuity conditions.

8.1.4 Identified Mathlib Gaps

The remaining high-leverage obligations are:

1. **Stronger empirical-Bayes statistical guarantees.** The finite strong-law layer supplies almost-sure atomwise, whole-law, and finite-observable expectation consistency; the learning layer supplies scoped concentration and model-evidence bounds; and the risk family supplies finite-law Laplace squared/Brier-risk transfer. Posterior contraction, minimax or empirical-calibration guarantees, and a named empirical marginal-likelihood optimum remain distinct extensions.
2. **Measure-valued conditional information and EFE comparison.** The finite carrier proves mutual-information and risk–ambiguity decompositions. Native random-variable or kernel statements are still needed to compare broader EFE formulations and continuous models.
3. **Continuous-state stochastic dynamics.** Langevin, Itô, Fokker–Planck, invariant-measure, and continuous path-space results require substantially more structure than the exact discrete semigroups, finite path laws, two-state continuous-time chain, and descent systems now present.
4. **Smooth statistical-manifold geometry.** The finite score carrier has matrix-valued metrics, categorical tangent positivity, pullbacks, scalar Cramér–Rao, invertible-chart natural gradients, mirror/Bregman identities, and replicator equivalence; the finite scalar exponential family adds log-partition differentiation, Fisher–variance, KL–Bregman, and local mean-coordinate duality. Multidimensional smooth atlases, dual connections, curvature, geodesics, and general optimizer convergence remain to be developed.

5. **Generic dynamical Markov blankets.** A concrete finite joint and typed transition now exist, and the weighted-Dirac embedding proves one native measure-theoretic `CondIndepFun` bridge plus rowwise closure. An invariant blanket predicate, closure under arbitrary current-state mixtures, and a blanket-existence theorem remain open.

Most are project-level model and theorem choices rather than confirmed Mathlib absences. The capability graph distinguishes a blocked project obligation from an upstream-library gap so responsibility is not displaced automatically onto Mathlib.

8.1.5 A 6–12 Month Maturity Roadmap

The heading is retained for continuity, but the roadmap is dependency-ordered rather than calendar-promised:

1. connect the finite Laplace risk-transfer and learning-family concentration results to posterior contraction, minimax or empirical calibration, or a named empirical marginal-likelihood theorem on one sampling model;
2. add probability laws or learning over the finite policy-tree carrier, compare alternative EFE formulations there, and then lift successful equivalences to native measure-valued conditional information;
3. extend the concrete native blanket transfer to an invariant predicate, arbitrary-mixture closure, and one blanket-existence theorem for a specified dynamics;
4. generalize the exact two-state continuous-time chain to a finite CTMC construction, then choose one continuous-state diffusion, state existence and regularity assumptions, and prove its invariant law before attempting continuous path measures or Fokker–Planck equations;
5. lift the finite scalar exponential family to a multidimensional smooth family with dual connections, curvature, and coordinate-change theorems;
6. lift finite stick conservation and finite NESS currents to countable/probabilistic limit theorems only after their convergence contracts are explicit.

A row changes semantic disposition only after exact theorem review, native acceptance, assumption analysis, and a non-vacuity check. A new Mathlib release alone is not a promotion event.

8.1.6 Comparison to Other Mathlib4 Formalization Projects

Large Lean projects demonstrate that difficult mathematics can be organized into reviewable, kernel-checked libraries ("Polynomial Freiman–Ruzsa in Lean 4" 2023; Scholze and Commelin 2022). The present catalogue is smaller and has a different purpose: it maps a heterogeneous scientific theory into explicit proof obligations and records semantic distance. It should not be compared by theorem count or compilation rate alone; the relevant measure is whether each mechanized statement captures the intended scientific claim under defensible assumptions.

9 Execution integrity

Every success claim in this project is tied to a concrete artifact or an observable computation. Catalogue mode validates and renders the 155 source entries, but it makes no Lean or Hermes claim. Full mode first validates every required capability and then records the Hermes response, refined Lean source, compiler result, SQLite session, and report manifest for each selected topic.

9.1 Evidence boundaries

The evidence boundary is deliberately explicit:

1. YAML is loaded by the strict catalogue schema and checked against the sealed family-body registry and generated aggregate.
2. Lean evidence comes only from `lake env lean` using the pinned workspace.
3. Hermes evidence comes from an HTTP response accepted by the configured response parser; a missing credential or invalid response is an error.
4. SQLite evidence is written transactionally and exported files are checked against their recorded hashes.
5. Report manifests bind the run to source and configuration digests, the Lean/Mathlib pins, capabilities, topic counts, and artifact hashes.

An unavailable capability therefore ends a full run before a success report is created. The result is diagnostic JSON with `complete: false` and a structured `failure_reason`. Offline work is available only through the explicitly named `catalogue` mode, whose result always reports zero verified topics.

9.2 Lean verification

`LeanVerifier` writes a temporary source file, invokes the real Lake command, captures diagnostics, removes the temporary file, and records the result. In full mode, a topic succeeds only when the Hermes-refined source compiles without warnings or a proof placeholder; in native mode, the same compiler boundary checks the canonical source directly. A prose-only review cannot replace the candidate that actually compiled. The original catalogue body and any refined source are retained as separate fields; the original is never presented as the refined result.

The committed `lean/FepSketches/fep_all.lean` is generated from the canonical catalogue source, while manifested foundations, leaf compositions, and their import-only aggregate are projected from `src/fep_lean/formal/`. Regeneration checks fail when either

projection differs, and the Lean build compiles the library against the exact pinned toolchain. The formalism audit then resolves reviewed declarations and checks their reported axioms; aggregate success alone does not supply that semantic evidence roster.

9.3 HTTP and persistence

Hermes tests use a real loopback HTTP server and exercise response parsing, timeouts, retries, model-chain advancement, and review calls over sockets. SQLite tests use temporary on-disk databases, concurrent writers, atomic file publication, restart recovery, and SHA-256 consistency checks. These boundaries make provider, compiler, and filesystem drift visible in test results.

9.4 Reproducibility

Use `uv.lock`, `fep-lean setup`, and the documented `fep-lean preflight` command to reproduce the pinned environment. A complete run produces `run_manifest.json`, `verification_manifest.json`, `summary.json`, Markdown reports, deterministic figures, and manuscript projections. These artifacts become evidence only when the independent receipt validator reconciles their rosters, digests, and hashes against the live source; no stored `complete` flag is authoritative by itself.

9.5 Implications for the FEP Debate

Formalization can locate a disagreement more sharply, but it cannot decide which formal object best represents a biological system. The catalogue therefore treats the critical literature as a source of proof obligations, not as a set of objections already answered by compilation. Its semantic audit asks, for every row, what is actually invariant, which assumptions do the work, whether the statement has a non-vacuous witness, and what stronger theorem would have to compile before the advertised interpretation was justified.

9.5.1 Blanket Conditions (Biehl et al.) → fep-005 Response

Biehl, Pollock, and Kanai (Biehl, Pollock, and Kanai 2021) challenge the generic use of Markov blankets in stochastic dynamics. At least three claims must be distinguished:

1. a finite set can be assigned to disjoint named blocks;
2. the random variables associated with those blocks satisfy a conditional-independence relation; and
3. a target dynamical system admits and preserves the relevant partition.

fep-005 proves the first claim exactly. Given `assign : Fin 20 → Fin 4`, it defines each block by filtering the finite universe, proves blocks with unequal labels are disjoint, characterizes membership, and proves that every state belongs to one and only one block. The assignment remains an input. No probability measure, conditional distribution, transition kernel, or dynamical invariance appears in the theorem type.

The row is consequently a complete finite-partition theorem rather than a rebuttal to Biehl et al. The maintained `FEP.MarkovBlanket` foundation addresses the next bounded layer with new objects: it constructs a normalized law $P(b)P(i \mid b)P(e \mid b)$, proves positive-mass internal–external factorization and zero conditional mutual information, and defines a nontrivial one-step transition whose allowed dependencies are enforced by kernel source types. This evidence satisfies the deliberately finite blanket capability, while the fep-005-to-fep-009 edge remains conceptual because no theorem identifies that finite factorization with the generic measure-theoretic conditional-independence predicate. This is a concrete finite model, not a derivation from every fep-005 assignment or an existence/invariance theorem for arbitrary stochastic dynamics.

9.5.2 Particular Partitions (Aguilera et al.) → fep-025 Response

Aguilera et al. (Aguilera et al. 2022) question whether the "particular partition" and NESS constructions used in Bayesian mechanics apply generically. A typical informal decomposition invokes a stationary density, a drift decomposition, positive diffusion, an antisymmetric solenoidal operator, and regularity/divergence conditions. Proving an elementary fact about one of those ingredients does not prove the conjunction.

fep-025 now formalizes a finite continuity-equation instance. It defines forward-minus-reverse probability current and node divergence, proves antisymmetry and global conservation, and derives zero divergence from normalized transition rows and a stationary mass vector. A directed three-state cycle has a uniform stationary law, zero divergence, and nonzero current, proving that finite stationarity need not imply detailed balance. It does not:

- construct a stationary density;
- prove existence or uniqueness of a NESS;
- express a state-dependent diffusion or solenoidal field;
- derive a Fokker–Planck or continuity equation; or
- establish that a physical flow admits the asserted decomposition.

This is a deliberate boundary. The row is a genuine finite NESS witness under its stated transition-matrix model, but it is not the continuous Helmholtz/Ao decomposition challenged in the literature. That stronger target must quantify the continuous state space, generator or SDE, stationary measure, boundary and regularity assumptions, and every term in the decomposition in one shared model.

9.5.3 Math and Territorialism (Andrews) → Type System Response

Andrews (Andrews 2021) criticizes mathematical notation that shifts meaning or hides assumptions. Lean addresses part of this problem mechanically: a `Measure α`, a density, an `ℝ`-valued objective, and an `ℝ≥0∞` divergence cannot be silently identified. Hypotheses are explicit inputs, and theorem names resolve to concrete declarations at a pinned library revision.

Two examples show both the benefit and the limit.

First, fep-014 uses `InformationTheory.klDiv μ ν` with the order of its measure arguments fixed by syntax. It proves non-negativity in the native extended codomain, self-divergence under `SigmaFinite`, the zero/equality characterization under finite-measure instances, and Mathlib's composition-product chain rule for finite measures and Markov kernels. This is substantially stronger than a theorem about unrelated measure monotonicity. It still does not say that a particular pair of FEP distributions satisfies the intended generative-model interpretation.

Second, fep-031 proves `0 < exp (-βE)` without assumptions because the exponential is always positive, while its monotonicity theorem requires the explicit hypothesis `0 < β`. On any explicitly nonempty finite support, it also proves the partition sum positive and the normalized weights sum to one. These signatures expose both the support and positive-inverse-temperature conditions that prose often leaves implicit; they do not derive the Gibbs law from a maximum-entropy principle.

Type discipline therefore blocks some category errors and makes assumptions inspectable. It does not prove that the chosen types capture the target science, and it does not rescue a theorem whose conclusion merely repeats a hypothesis.

9.5.4 Colombo & Seriès and the Empirical-Adequacy Critique

Empirical-adequacy critiques ask whether an FEP model rules out observations or predicts specific neural/behavioral phenomena. Lean has no access to those observations. It can help by fixing the mathematical content of a model, exposing auxiliary assumptions, and making alternative formal readings comparable. The empirical step still requires an observation model, data provenance, parameter-identification procedure, and a falsifiable statistical test outside the proof kernel.

9.5.5 Falsifiability and Precision

Formalization improves three forms of auditability:

1. **Syntactic precision:** every symbol has a type and every cited theorem has a resolvable identifier.
2. **Deductive auditability:** the kernel checks that the conclusion follows from the stated hypotheses.
3. **Assumption localization:** competing readings can be compared by diffing their types and hypotheses.

It does not automatically improve empirical falsifiability. A theorem can be perfectly proved yet vacuous, conditional on the desired conclusion, or disconnected from measurable data. That is why the semantic disposition and non-vacuity fields are co-equal with the native compile result.

9.5.6 Theorems That Address Contested Claims

The most informative current rows occupy different maturity levels:

- **fep-002** directly formalizes a narrow KL-remainder variational bound in `ℝ≥0∞`, including exactness when the approximation equals the posterior. The finite active-inference foundation separately constructs evidence and posterior from one normalized joint and proves posterior VFE attainment; a finite-to-measure bridge remains open.
- **fep-009** directly wraps generic conditional-independence symmetry and supplies conditional independence from the trivial σ-algebra as a non-vacuity witness. The finite blanket foundation supplies a nontrivial factorized law and zero conditional mutual information; fep-135–141 add a weighted-Dirac embedding and one native `CondIndepFun` transfer. Neither result turns every fep-005 assignment or arbitrary dynamics into a blanket.
- **fep-010** directly proves that a reversible Markov kernel preserves its reference measure and witnesses the contract with the identity kernel. Exact two-state relaxation, finite path-law fluctuation and reversible-dissipation identities, and empirical-law strong laws exist elsewhere, but generic ergodic convergence for fep-010's measure-kernel carrier and continuous path-space theory remain absent.
- **fep-014** directly wraps native KL non-negativity, self-zero, zero characterization, and the composition-product chain rule.
- **fep-030/fep-031** prove a binary entropy maximum and normalized finite Gibbs weights, with a composed theorem showing that the two-state zero-inverse-temperature Gibbs law attains the binary maximum.
- **fep-035** directly proves strict two-point Jensen for the logarithm under positive, nondegenerate weights; it does not discharge expectation-level integrability obligations.
- **fep-048** wraps native contraction uniqueness, supplies a concrete halving-map contraction witness, identifies its unique fixed point, and proves convergence of all iterates to zero.
- **fep-005** supplies a finite partition substrate but not probabilistic blanket independence.
- **fep-017/fep-034** use Mathlib's posterior kernel to prove normalized Bayesian inversion and a one-step transition–observation filter with joint reconstruction and prior recovery.
- **fep-018/fep-038/fep-044** provide a complete Bernoulli geometry instance: Fisher information, natural gradient, Fisher–Rao distance, and Hellinger separation. fep-100–106 extend the finite score carrier with categorical tangent positivity, rank/null witnesses, pullback,

scalar Cramér–Rao, invertible-chart equivariance, mirror and affine Bregman identities, and replicator equivalence. fep-142–148 add a finite scalar exponential family with log-partition derivatives, Fisher–variance and KL–Bregman identities, and local mean-coordinate injection.

- **fep-025/fep-049** provide a finite non-detailed-balance stationary current and nonnegative current dissipation under an explicit diagonal resistance law.
- **fep-021** defines expected free energy as pragmatic cost minus epistemic information in `ENNReal`, proves exact reconstruction under the visible value-at-most-cost premise, and composes with native KL information gain. The maintained real-valued finite carrier separately derives both pragmatic-minus-epistemic and risk-plus-ambiguity forms under full support; equivalence to every literature convention and the truncating real-to-`ENNReal` bridge remain explicit limitations.
- **fep-028/fep-012** prove a full support-aware stochastic policy law and connect it to an entropy-regularized objective; optimality and sampling dynamics remain separate claims.
- **fep-121–127** prove exact Laplace error and bias decompositions, finite-law squared/Brier-risk transfer, and bad-event containment, with a nonzero-bias boundary; they do not prove posterior contraction or empirical calibration.
- **fep-128–134** prove arbitrary finite-depth policy-tree recursion, Bellman minimization, optimal-tree existence, open-loop embedding and dominance, treewise EFE, and a strict Boolean feedback witness; they do not define distributions or learning over trees.
- **fep-149–155** construct an exact positive-rate two-state continuous-time semigroup, master equation, detailed-balance stationary law, exponential relaxation, and strict Lyapunov witness; they do not construct a continuous-state SDE or Fokker–Planck PDE.

The generated `docs/formalism-coverage.md` is the canonical row-by-row account; prose examples are illustrative rather than an alternate maturity registry.

9.5.7 Synthesis: What Formalization Reveals

The critical lines converge on a dependency problem. The maintained finite carrier discharges the first part of that order: a common probabilistic model produces posterior/evidence identities and both EFE decompositions; the policy-tree carrier supplies finite observation-contingent control; and the blanket carrier exposes an exact factorized joint, typed one-step dynamics, and one native conditional-independence transfer. The next layers still require probability laws or learning over trees, EFE comparison on a shared carrier, blanket existence or invariance under broader dynamics and mixtures, and observation/refinement bridges before any of this becomes empirically discriminating. Proving downstream algebra without those upstream objects would still produce valid lemmas rather than the advertised system-level theory.

This dependency order is a constructive result of formalization. It turns "more rigor is needed" into specific interfaces and theorem obligations, and it prevents compiler success at a shallow layer from being mistaken for closure at a deeper one.

9.5.8 Concrete Formalization Vignettes

Native KL (fep-014). The central zero characterization is:

```
theorem fep014_kl_eq_zero_iff (μ ν : Measure α)
    [IsFiniteMeasure μ] [IsFiniteMeasure ν] :
    InformationTheory.klDiv μ ν = 0 ↔ μ = ν :=
  InformationTheory.klDiv_eq_zero_iff
```

The theorem is strong and non-vacuous at its stated measure-theoretic scope. Its instances are not decoration: removing finite-measure assumptions changes the available result.

Variational remainder (fep-002). The project defines variational free energy as surprisal plus native KL and proves that it is at least surprisal. Exactness follows when both measure arguments are the same posterior under `SigmaFinite`. The proof does not smuggle a real-valued KL through `.toReal`, so infinite divergence remains visible.

Global contraction (fep-048). Mathlib's `ContractingWith c f` quantifies over every pair of real inputs and packages the strict coefficient bound. The concrete map `x ↦ x/2` witnesses the contract, has zero as its unique fixed point, and its iterates tend to zero from every real start. This repairs the common vacuity pattern in which a "contraction" premise applied only to two already fixed points or was never shown realizable, while keeping the claim narrower than general variational-dynamics convergence.

Witnessed composition. Both derivational graph edges and formal pairings require a qualified Lean declaration, not just prose. The 125 current witnesses span posterior filtering and Bayesian inversion, variational duality, controlled and temporal inference, causal blankets and interventions, predictive coding, finite path thermodynamics, Fisher and geometric optimization, collective inference, learning/model evidence, and the original entropy, policy, Gaussian, convergence, and partition-energy seams. Only 20 assert a derivation or identification; the other 105 deliberately place separately proved endpoint laws together without implication. The maintained foundations add deeper within-carrier chains; these are counted separately from authored topic-to-topic edges. Conceptual edges remain visibly non-proof evidence, and the graph schema retains an explicit blocker kind for any future gap.

These vignettes also explain why **136** catalogue rows currently carry the `formalized` disposition: directness, assumption quality, and a non-vacuity witness are required in addition to compilation. For fep-036, posterior closure was insufficient until an explicit finite binomial law,

outcome-indexed estimator, shrinkage identity, and consistency-transfer theorem were added. The generated coverage projection is authoritative if that count changes.

9.5.9 Limitations: What Formalization Does Not Do

The current contribution does not:

- validate the FEP empirically;
- establish a general Markov-blanket theorem for stochastic dynamics;
- construct a continuous-state NESS or derive Fokker–Planck evolution;
- lift the finite scalar exponential-family and categorical Fisher carriers to multidimensional smooth statistical manifolds with dual connections and curvature;
- establish equivalence across broader EFE conventions and probability laws or learning over the finite policy-tree carrier;
- make an LLM response trustworthy without independent compilation and review; or
- make every warning-free theorem semantically representative of its topic title.

The catalogue's breadth is therefore a map of formalization surfaces, not 155 end-to-end domain theorems. Its value lies partly in refusing to erase that distinction.

9.5.10 Future: Machine-Verifiable Proofs in Journals

A publication-ready formal theory should grow in dependency order:

1. connect the finite Laplace risk-transfer and learning-family concentration laws to posterior contraction, minimax or empirical calibration, or an empirical marginal-likelihood theorem on one shared sampling model;
2. add probability laws or learning over the finite policy-tree carrier and compare alternative EFE formulations under explicit equivalence hypotheses;
3. extend the native finite blanket result to blanket existence, invariance, and arbitrary-mixture closure for a concrete stochastic dynamics;
4. lift the finite scalar exponential-family and categorical Fisher carriers to a multidimensional smooth family with dual connections and curvature;
5. generalize the exact two-state continuous-time chain and add one continuous-state stochastic/PDE model with existence, invariant-law, and NESS obligations; and
6. bind formal parameters to an auditable empirical model and falsification protocol.

Each step should land with a narrowed theorem, a non-vacuity witness or concrete model, a semantic review, native evidence, and manuscript references that resolve to the exact declaration. That workflow would let proponents and critics contribute competing formal statements without confusing "the kernel accepts this implication" with "nature satisfies these hypotheses."

9.6 Comparative Analysis with Existing LLM-ITP Systems

FEP Lean is best understood as a domain catalogue and evidence pipeline, not as a new automated prover. Its canonical Lean bodies are researcher-curated; the optional model reviews those bodies; the kernel checks the resulting term; and a semantic audit separately asks whether the compiled statement captures the advertised concept. Comparing only "proof success" would therefore misstate the system's task.

System family	Primary task	Starting point	Typical success evidence	Relation to FEP Lean
LeanDojo / LEGO-Prover (Yang et al. 2024; Xin et al. 2024b)	tactic/proof search	formal theorem	closed proof and benchmark result	possible downstream proof-search layer
DeepSeek-Prover (Xin et al. 2024a, 2025)	formal proof generation	formal or translated problems	benchmark proof acceptance	complementary to statement curation
Draft, Sketch, Prove (Jiang et al. 2023)	autoformalization plus proof	informal proof/problem	formal statement and discharged obligations	related translation workflow at theorem scale
PhysLean (Tooby-Smith 2024)	human-led physics library	domain mathematics	reviewed definitions/theorems in Lean	closest library-building analogue
FEP Lean	catalogue curation, semantic audit, and evidence separation	contested domain claims	source parity + native receipt + semantic disposition	formalization-surface map for FEP

The distinctive contribution is the coupling of breadth with claim calibration: every row has an exact source, declaration inventory, primary theorem, assumption review, non-vacuity note, and disposition. This is not evidence that the approach outperforms proof-search systems on their benchmarks.

9.6.1 Manual vs Hermes-Assisted Formalization

No controlled human-time experiment is included, so the manuscript does not report speedups, first-pass error rates, or "weeks versus hours" estimates. The defensible comparison is functional:

Activity	Researcher-owned path	Optional Hermes contribution	Acceptance owner
Select domain claim and scope	author/reviewer	may summarize	semantic review
Write canonical theorem and imports	author/reviewer	may suggest a revision in a full run	canonical source review
Establish type/proof correctness	invoke Lean	none	Lean kernel + native receipt
Explain proof strategy	write prose	may draft commentary	author/reviewer
Assert model/time/cost results	inspect full run	generates observed fields	validated full receipt

This allocation is intentionally conservative. Uniform model commentary can reduce blank-page effort, but it also creates new review work: semantic drift, invented library names, missing code fences, and provider-dependent output must be checked. A future comparison should randomize topics, preregister editing-time and semantic-fidelity metrics, and distinguish initial draft, compile repair, and expert review.

9.6.2 Comparison to Similar Projects

PhysLean demonstrates the value of a human-reviewed, domain-specific Lean library. FEP Lean shares the commitment to a pinned formal substrate but pursues a different shape: 155 deliberately bounded domain instances, 125 checked cross-topic witnesses with derivational and non-implicational pairing kinds kept distinct, a reusable formal kernel, and a machine-readable capability/blocker graph rather than a single mature mathematical hierarchy. Its deepest strands include native Bayesian kernels; posterior VFE and EFE on shared finite carriers; finite-dimensional Fisher geometry; concrete blanket factorization and dynamics; path-space thermodynamics; collective inference; and asymptotic and finite-sample learning laws. Its breadth still contains intentionally small topic-scoped results. LeanDojo, LEGO-Prover, DeepSeek-Prover, Lean Copilot (Song, Yang, and Anandkumar 2025), and Draft, Sketch, Prove address premise selection, tactic generation, proof completion, or autoformalization. Those capabilities could assist further strengthening, but they do not decide which formal statement resolves a contested FEP interpretation.

The repository makes no priority claim about being the first FEP formalization or the largest thermodynamics corpus. Such claims require a systematic, dated literature review and can become stale. Its inspectable claim is narrower: this release contains the exact catalogue and formalism coverage recorded by its generated reports and bound native receipt.

9.6.3 State-Space Models, Domain-Specific Languages, and Generalized Notation

Executable Active-Inference tools and notations such as pymdp (Heins, Millidge, et al. 2022) or GNN (Smékal and Friedman 2023) occupy a different layer. They specify and run model instances; Lean states and proves invariants. A useful bridge would translate a typed model representation into a common Lean probability/kernel structure, generate proof obligations for normalization and conditional independence, and retain a provenance link back to executable parameters. The current catalogue does not implement that bridge.

This distinction matters for validation. A numerical implementation can be tested on data while a theorem can be checked for deductive correctness; neither subsumes the other. End-to-end assurance needs both, plus a proof that the executable representation refines the formal one.

9.6.4 Our Approach vs GPT-4-Class Direct Lean Generation

The project has not run a controlled direct-generation baseline, so it does not claim a higher compile rate than a named frontier model. Its architecture embodies a testable hypothesis: separating deterministic theorem ownership from stochastic review should reduce run-to-run drift and make provenance easier to audit.

A valid future baseline would hold the topic statement, toolchain, Mathlib pin, prompt budget, number of repair attempts, and human-review budget fixed. It would report at least:

- syntactic extraction rate;
- warning-free and `sorry`-free compile rate;
- theorem-statement preservation;
- semantic disposition under blinded expert review;
- time and token cost; and
- reproducibility across multiple model seeds or provider runs.

Without those controls, comparing a curated catalogue's native compile rate to one-shot generated code would conflate authorship with verification.

9.6.5 Quality Metrics

The publication metrics are deliberately split:

Metric family	Current source
roster, areas, maturity tags	generated `topics.yaml`
semantic breadth/depth	`docs/formalism-coverage.md`
declaration/import counts	parsed canonical Lean source
native compile/warning/`sorry` outcomes	validated `output/native-verification.json`
Hermes success, model, retries, tokens, latency	validated full report only

The native compile rate rendered for this source state is `155/155 (100.0%)`; its evidence kind is `native-lean`, warning count `0`, and claim-ready predicate `true`. Full Hermes/OpenGauss claim readiness is separately `false`. These values are not interchangeable, and a false predicate is a result rather than a placeholder for a preferred headline.

9.6.6 Time Comparison

Native duration is measured by the receipt (`556.106` seconds for its exact roster and environment). No cross-person manual baseline is available. Full-model duration is reported only when a current full receipt is claim-ready. This avoids turning hardware-, cache-, network-, and provider-specific observations into general productivity claims.

9.6.7 Implications for Active Inference Practitioners

The catalogue and finite kernel already provide invariants that executable systems can use as specifications: support-aware policy normalization, exact transition–observation posterior reconstruction, Bellman recursion, matrix–vector message propagation, finite policy minimizers, chronological rollout, action-law pushforward, and both EFE decompositions. They do not prove that pymdp, SPM, or another implementation satisfies those statements, because no refinement mapping from implementation state to Lean objects has been supplied.

The practical next step is therefore not to label the implementation "verified," but to define that mapping and prove focused conformance lemmas. This would connect formal breadth to software behavior while preserving the manuscript's central distinction among a compiling theorem, a semantically adequate model, and an empirically validated system.

9.7 Broader Impact: Reproducible Science and Digital Mathematics

The strongest broader contribution is reviewability. A disputed informal formula can be translated into an exact proposition, checked by a small trusted kernel, and discussed together with its assumptions and semantic disposition. The resulting artifact is more useful than a binary "formalized" label because it makes disagreement local: readers can contest the translation, the assumptions, the proof, or the scientific model separately.

The package also demonstrates a publication pattern for computational mathematics: maintained authoring sources produce deterministic catalogue and coverage projections; execution evidence is content-addressed; and manuscript rendering fails before publication when a value or theorem reference is stale. These principles generalize beyond the FEP without implying that the current catalogue is training data of sufficient quality, a safety proof for active-inference systems, or a completed digital theory.

9.8 Limitations and Threats to Validity

Threat	Consequence	Current control
Semantic distance	A true scoped theorem may still be weaker than an informal universal reading	Per-topic disposition, assumptions, and non-vacuity review
Fragmentation	125 witnessed relations still do not form one end-to-end FEP theory	Derivational edges, non-implicational formal pairings, conceptual edges, and blockers remain distinct
Model choice	The formal object may not capture the intended scientific system	Narrow claims and explicit capability boundaries
Toolchain drift	A later Lean/Mathlib release may alter APIs or warnings	Pins, generation checks, and native receipts bound to source/toolchain
External nondeterminism	Provider output and availability vary	Full mode is optional and has a separate claim predicate
Catalogue selection	155 curated rows are not a census of FEP mathematics	Stable scope and explicit extension criteria

Threat	Consequence	Current control
Publication drift	Stale variables or identifiers can preserve obsolete claims	Fail-closed renderer and reference audits

The most important limitation is compositional depth. The catalogue contains individually checkable fragments plus **125** witnessed cross-topic relations—only **20** of them derivational and **105** explicitly non-implicational—not an end-to-end derivation from stochastic dynamics to perception, action, and thermodynamic behavior. The **136** `formalized` rows are deliberately narrowed: among them are KL identities, generic conditional-independence and invariant-kernel laws, binary entropy optimality, strict two-point Jensen, and one concrete contraction-convergence theorem. They do not establish the general FEP.

The second limitation is that the remaining frontiers belong largely to this project's model choices rather than to confirmed Mathlib absence. Native KL, posterior kernels, Gaussian measures, a shared finite active-inference carrier with cumulative open-loop EFE, arbitrary finite-depth policy trees, finite and native blanket laws, generalized-coordinate corrections, finite path thermodynamics, an exact two-state continuous-time chain, categorical and scalar exponential-family geometry, collective inference, concentration, finite Laplace/Brier-risk transfer, and model-evidence algebra are integrated. The open obligations are broader: probability laws or learning over policy trees, generic blanket existence and causal identification, multidimensional smooth information geometry with dual connections, continuous-state stochastic processes, and posterior contraction, minimax or empirical calibration on a shared sampling model.

9.8.1 Model-Quality Ceiling

Hermes is an advisory commentary and drafting layer. A stronger model might produce better explanations or proof suggestions, but it cannot change the evidence boundary: the Lean kernel decides compilation, the semantic audit decides claim alignment, and report validation decides provenance. No model benchmark is inferred from this catalogue.

9.8.2 Scope Limitations: 155 Topics of Many

The stable roster samples five areas but omits many important targets: generic CTMC and continuous-state SDE/PDE existence, infinite-horizon or continuous-belief active inference, probability laws and learning over policy trees, blanket existence and arbitrary-mixture invariance, causal identification, multidimensional Fisher geometry and dual connections, posterior contraction and empirical calibration, and quantitative equivalence among competing FEP or EFE formulations. New topics should be added only with a clear invariant, carrier delta, explicit assumptions, non-vacuity plan, required composition bridge, and acceptance probe; topic count alone is not progress.

9.8.3 Ethical Considerations: Authorship and AI-Assisted Proof

Responsibility attaches to the people who choose the statement, accept its assumptions, review the proof, and make the scientific claim. Model identity and full-run provenance should be retained when LLM output contributes to a publication, but kernel checking does not settle authorship or semantic responsibility. Generated prose remains draft material until reviewed.

9.8.4 Future Work: From Topic Bodies to End-to-End Proofs

Priority should follow dependency and scientific value:

1. connect finite Laplace risk transfer and the learning family's concentration theorems to posterior contraction, minimax or empirical calibration, or a named empirical marginal-likelihood objective on one sampling model;
2. add probability laws or learning over finite policy trees and prove carefully scoped equivalences among EFE formulations;
3. extend the native finite blanket transfer to blanket existence, invariance, and arbitrary-mixture closure in one stochastic dynamics;
4. generalize the finite scalar exponential-family and categorical Fisher carriers to multidimensional smooth families, dual connections, and curvature;
5. extend a manifested composition leaf only when reviewed theorems genuinely share objects, and require every formal graph edge to name its witness;
6. generalize the exact two-state continuous-time chain and select one continuous-state stochastic model for a deep end-to-end case study before broadening the roster.

Each step should change semantic disposition only after mathematical review and new native evidence.

10 Conclusion and Future Work

This work turns a broad FEP-facing catalogue into an auditable Lean package with explicit semantic maturity, composition, provenance, and visual diagnostics. Its principal result is inspectable semantic reach: 136 of 155 rows directly formalize their deliberately bounded claims, while the remaining rows retain explicit conditional or structural classifications rather than inheriting strength from compilation. The 125 authored cross-topic theorems make reuse inspectable, while the graph distinguishes 20 derivations or identifications from 105 checked pairings.

Capability status totals remain visible in the same generated graph. A manifested reusable kernel spans normalized finite and measure-theoretic probability, Bayesian inversion, variational and expected free energy, controlled and temporal inference, finite policy trees, causal interventions, native finite blankets, predictive coding, finite path and two-state continuous-time thermodynamics, categorical and scalar exponential-family geometry, collective inference, finite risk, and learning/model evidence. It still does not supply generic blanket existence, unrestricted continuous-state active inference, multidimensional smooth-manifold geometry with dual connections, or empirical validation of a biological model. This is not a proof of a universal FEP; it is a deep, inspectable library of exact finite, measure-theoretic, geometric, dynamical, learning-theoretic, and thermodynamic instances.

10.1 Theoretical Synthesis: What Machine-Checked FEP Establishes

10.1.1 Formal Adequacy as a Distinct Dimension of Theory Evaluation

Formal adequacy asks whether a proposition is well-typed, proved, explicit about assumptions, and non-vacuous at the intended scope. It is distinct from empirical adequacy and explanatory value. Lean can establish the first for an exact statement; it cannot infer that the statement is the right model of a living system.

10.1.2 Typology of Results: Definitional, Structural, Quantitative

The catalogue contains definitions, structural laws, and quantitative inequalities, but these categories do not form an automatic ladder of scientific strength. A scalar inequality may be mathematically complete and scientifically remote; a definition may expose the precise object needed for a later deep theorem. The per-topic semantic review is therefore more informative than classifying declarations by syntax alone.

10.1.3 Definitional Commitment via the Type System

The fep-005 row illustrates both the value and limit of definitional commitment. `fep005_partitionCover`, `fep005_disjoint`, and `fep005_existsUnique_block` give an exact finite labeled partition with unique membership. Those row declarations do not establish conditional independence, stationarity, or a dynamical Markov blanket. The maintained blanket foundation then introduces the missing probabilistic objects explicitly and proves finite positive-mass factorization, zero conditional mutual information, and a typed nontrivial one-step dynamic; fep-135–141 embed that carrier into native measures and prove one `CondIndepFun` instance plus rowwise preservation. The remaining boundary is therefore generic existence, invariance under arbitrary mixtures, and biological interpretation—not the absence of a concrete blanket model or every native bridge.

10.1.4 The Catalogue as Formal Review Infrastructure

The catalogue is most useful as review infrastructure. A proposed extension must identify its primary invariant, theorem statement, assumptions, non-vacuity evidence, and acceptance probe. Generated coverage and receipt checks then make it difficult for a narrow compiler success to acquire a stronger title through prose drift.

10.2 Summary of Contributions

The package contributes:

1. a real `fep_lean` import namespace and isolated-wheel contract;
2. one typed generation graph for metadata, semantic review, Lean bodies, YAML, and package data;
3. native Mathlib posterior, KL, Gaussian, conditional-independence, and Markov-kernel formalizations;
4. a manifested reusable kernel of foundations and leaf compositions for probability, information, variational and expected free energy, finite policy trees, temporal and causal inference, native blanket transfer, predictive coding, stochastic thermodynamics, scalar exponential-family geometry, collective inference, finite risk, and learning;
5. exact Bernoulli Fisher–Rao, Hellinger, stochastic-dynamics, entropy-production, and Landauer surfaces in the topic catalogue;
6. manifested composition leaves with **125** declaration-witnessed cross-topic relations, typed as derivational edges or non-implicational formal pairings, and an import-only aggregate;
7. a warning-free aggregate plus declaration-resolution and axiom-audit acceptance target;
8. source-bound native and full-report evidence contracts;
9. generated formalism coverage, an offline SVG/HTML atlas whose import dependencies remain distinct from scientific edges, and a deterministic numerical formal-kernel dashboard with typed per-check relations and tolerances, plus fail-closed manuscript and theorem-reference audits.

10.3 Implications for the Active Inference Community

The practical implication is a change in review questions. Instead of asking only whether an equation looks familiar, readers can ask: which Lean object represents it, which theorem is primary, what hypotheses are used, is the hypothesis inhabited, does the theorem match the title, and what receipt proves that the current bytes compiled? These questions sharpen rather than settle the ongoing scientific debate.

10.4 Engineering Outcomes and Lessons

10.4.1 Compilation Headline

The rendered native rate is **`155/155 (100.0%)`** from evidence kind `native-lean`. Claim readiness is `true`, with 0 warnings and 0 admitted proofs. Catalogue generation alone cannot populate this headline.

10.4.2 Mathlib Integration Lessons

The most important correction was source-grounded: the pinned Mathlib already has native KL definitions and chain rules, posterior kernels, Gaussian laws, conditional independence, kernel invariance, and a real strong law. Directly compiling minimal probes prevented the manuscript from repeating obsolete ecosystem claims and enabled stronger theorem-level constructions. At the same time, the pin does not justify claiming a general data-processing theorem, multidimensional smooth statistical-manifold geometry, blanket existence, or continuous-state stochastic dynamics; the exact two-state continuous-time chain is deliberately narrower than the last target.

10.4.3 LLM-ITP Synergy

LLMs can help translate, explain, and review candidate statements. Their output remains proposal material. Compiler acceptance, semantic adequacy, and publication provenance are independent gates, and a baseline fallback must not be presented as validation of an LLM-refined proof.

10.5 Future Work

The highest-value next step for the empirical-prior strand is to connect finite Laplace risk transfer and the concentration machinery to posterior contraction, minimax or empirical calibration on one sampling model, and to formalize a named empirical marginal-likelihood objective. Beyond that, depth should proceed through probability laws or learning over finite policy trees and EFE equivalence conditions, blanket existence and arbitrary-mixture invariance for the native finite model, multidimensional smooth exponential-family geometry with dual connections, a general constrained maximum-entropy theorem, generic finite CTMC infrastructure, and one continuous-state diffusion case study. Shared library extraction should follow demonstrated cross-topic proof reuse rather than precede it.

10.6 Reproducibility Statement

The Python environment is lockfile-controlled; Lean and Mathlib are pinned in the Lake workspace; generated artifacts have drift checks; and native evidence can be reproduced with:

```
uv sync --locked
uv run fep-lean setup
uv run python scripts/_maint_build_topics_catalogue.py --check
uv run python scripts/_maint_build_fep_all_lean.py --check
uv run python scripts/_maint_build_formal_modules.py --check
uv run python scripts/theorem_maturity_audit.py
uv run python scripts/build_formalism_coverage.py --check
uv run fep-lean atlas --check
uv run python scripts/build_formal_kernel_dashboard.py --check
uv run python scripts/audit_formalisms.py --receipt output/formalism-audit.json
uv run fep-lean verify --fail-on-warnings --receipt output/native-verification.json
uv run python scripts/render_manuscript.py --check
uv run python docs/theorem_ref_audit.py
```

Full Hermes/OpenGauss reproduction additionally requires the declared external capabilities and a validated full report. It is not required to reproduce native Lean results.

10.7 Data Availability

Canonical sources, generated catalogue data, topic and maintained foundation/composition Lean modules, semantic review records, coverage/atlas projections, and validation scripts are distributed with the repository. Provider credentials, local caches, and transient receipts are not source artifacts.

10.8 Closing Remark

Formalization is most valuable here not as a seal of approval but as a disciplined way to discover exactly what has and has not been proved. The strengthened catalogue now makes that distinction visible at the level of theorem, package, receipt, and manuscript.

11 Bibliography

References are in `manuscript/references.bib`. Inline `[@key]` resolved by Pandoc citeproc during rendering. See `docs/_generated/canonical_facts.md` for pipeline status.

Adams, Rick A., Stewart Shipp, and Karl J. Friston. 2013. "Predictions Not Commands: Active Inference in the Motor System." *Brain Structure and Function* 218 (3): 611–43. https://doi.org/10.1007/s00429-012-0475-5.

Aguilera, Miguel, Beren Millidge, Alexander Tschantz, and Christopher L. Buckley. 2022. "How Particular Is the Physics of the Free Energy Principle?" *Physics of Life Reviews* 40: 24–56. https://doi.org/10.1016/j.plrev.2021.11.001.

AlphaProof team, Google DeepMind. 2024. "AlphaProof and AlphaGeometry 2: Solving IMO Problems at Silver-Medal Level." https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/.

Amari, Shun-ichi. 1983. "A Foundation of Information Geometry." *Electronics and Communications in Japan (Part I: Communications)* 66 (6): 1–10. https://doi.org/10.1002/ecja.4400660602.

———. 1998. "Natural Gradient Works Efficiently in Learning." *Neural Computation* 10 (2): 251–76. https://doi.org/10.1162/089976698300017746.

———. 2016. *Information Geometry and Its Applications*. Vol. 194. Applied Mathematical Sciences. Springer. https://doi.org/10.1007/978-4-431-55978-8.

Andrews, Mel. 2021. "The Math Is Not the Territory: Navigating the Free Energy Principle." *Biology & Philosophy* 36: 30. https://doi.org/10.1007/s10539-021-09807-0.

Ao, Ping. 2004. "Potential in Stochastic Differential Equations: Novel Construction." *Journal of Physics A: Mathematical and General* 37 (3): L25–30. https://doi.org/10.1088/0305-4470/37/3/L01.

Avigad, Jeremy, Johannes Hölzl, and Luke Serafin. 2017. "A Formally Verified Proof of the Central Limit Theorem." *Journal of Automated Reasoning* 59 (4): 389–423. https://doi.org/10.1007/s10817-017-9404-x.

Baum, Leonard E., Ted Petrie, George Soules, and Norman Weiss. 1970. "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains." *The Annals of Mathematical Statistics* 41 (1): 164–71. https://doi.org/10.1214/aoms/1177697196.

Beni, Majid D., Mark Solms, Krzysztof Dołęga, Wanja Wiese, and Karl Friston. 2023. "Brains Blog Roundtable: Free Energy Principle, Consciousness, Realism and Illusionism." https://philosophyofbrains.com/2023/05/09/brains-blog-roundtable-free-energy-principle-consciousness-realism-and-illusionism.aspx.

Biehl, Martin, Felix A. Pollock, and Ryota Kanai. 2021. "A Technical Critique of Some Parts of the Free Energy Principle." *Entropy* 23 (3): 293. https://doi.org/10.3390/e23030293.

Blei, David M., Alp Kucukelbir, and Jon D. McAuliffe. 2017. "Variational Inference: A Review for Statisticians." *Journal of the American Statistical Association* 112 (518): 859–77. https://doi.org/10.1080/01621459.2017.1285773.

Brier, Glenn W. 1950. "Verification of Forecasts Expressed in Terms of Probability." *Monthly Weather Review* 78 (1): 1–3. https://doi.org/10.1175/1520-0493(1950)078%3C0001:VOFEIT%3E2.0.CO;2.

Burda, Yuri, Roger Grosse, and Ruslan Salakhutdinov. 2016. "Importance Weighted Autoencoders." In *International Conference on Learning Representations*. https://arxiv.org/abs/1509.00519.

Buzzard, Kevin, Johan Commelin, and Patrick Massot. 2020. "Formalising Perfectoid Spaces." *Proceedings of the ACM on Programming Languages* 4 (POPL): 1–29. https://doi.org/10.1145/3371163.

Champion, Théophile, Howard Bowman, Dimitrije Marković, and Marek Grześ. 2026. "Reframing the Expected Free Energy: Four Formulations and a Unification." *Neural Computation* 38 (3): 439–69. https://doi.org/10.1162/NECO.a.1491.

Crooks, Gavin E. 1999. "Entropy Production Fluctuation Theorem and the Nonequilibrium Work Relation for Free Energy Differences." *Physical Review E* 60 (3): 2721–26. https://doi.org/10.1103/PhysRevE.60.2721.

Da Costa, Lancelot, Karl Friston, Conor Heins, and Grigorios A. Pavliotis. 2021. "Bayesian Mechanics for Stationary Processes." *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 477 (2256): 20210518. https://doi.org/10.1098/rspa.2021.0518.

———. 2024. "Bayesian Mechanics of Synaptic Learning Under the Free-Energy Principle." *arXiv Preprint arXiv:2310.16556*. https://doi.org/10.48550/arXiv.2310.16556.

Da Costa, Lancelot, Thomas Parr, Nadia Sajid, Sabine Veselic, Víctor Neberáth, and Karl Friston. 2023. "Active Inference on Discrete State-Spaces: A Synthesis." *Journal of Mathematical Psychology* 99: 102447. https://doi.org/10.1016/j.jmp.2020.102447.

Donsker, Monroe D., and S. R. Srinivasa Varadhan. 1975. "Asymptotic Evaluation of Certain Markov Process Expectations for Large Time, I." *Communications on Pure and Applied Mathematics* 28 (1): 1–47. https://doi.org/10.1002/cpa.3160280102.

First, Emily, Markus N. Rabe, Talia Ringer, and Yuriy Brun. 2023. "Baldur: Whole-Proof Generation and Repair with Large Language Models." In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 1229–41. https://doi.org/10.1145/3611643.3616243.

Friston, Karl. 2010. "The Free-Energy Principle: A Unified Brain Theory?" *Nature Reviews Neuroscience* 11 (2): 127–38. https://doi.org/10.1038/nrn2787.

———. 2019. "A Free Energy Principle for a Particular Physics." *arXiv Preprint arXiv:1906.10184*. https://doi.org/10.48550/arXiv.1906.10184.

Friston, Karl J. 2005. "A Theory of Cortical Responses." *Philosophical Transactions of the Royal Society B* 360 (1456): 815–36. https://doi.org/10.1098/rstb.2005.1622.

Friston, Karl J., Thomas Parr, Conor Heins, Axel Constant, Daniel A. Friedman, Takuya Isomura, Chris Fields, et al. 2024. "Federated

Inference and Belief Sharing." *Neuroscience and Biobehavioral Reviews* 156: 105500. https://doi.org/10.1016/j.neubiorev.2023.105500.
Friston, Karl, Lancelot Da Costa, Danijar Hafner, Casper Hesp, and Thomas Parr. 2020. "Sophisticated Inference." https://doi.org/10.48550/arXiv.2006.04120.
Friston, Karl, Lancelot Da Costa, Dalton A. R. Sakthivadivel, Conor Heins, Grigorios A. Pavliotis, Maxwell Ramstead, and Thomas Parr. 2023. "Path Integrals, Particular Kinds, and Strange Things." *Physics of Life Reviews* 47: 35–62. https://doi.org/10.1016/j.plrev.2023.08.016.
Friston, Karl, Erik D. Fagerholm, T. Sabesan Zarghami, Thomas Parr, Rosalyn J. Moran, Matteo Frigo, and Christopher L. Buckley. 2021. "Stochastic Chaos and Markov Blankets." *Entropy* 23 (1): 14. https://doi.org/10.3390/e23010014.
Friston, Karl, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, John P. O'Doherty, and Giovanni Pezzulo. 2016. "Active Inference and Learning." *Neuroscience & Biobehavioral Reviews* 68: 862–79. https://doi.org/10.1016/j.neubiorev.2016.06.022.
Friston, Karl, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, and Giovanni Pezzulo. 2017. "Active Inference: A Process Theory." *Neural Computation* 29 (1): 1–49. https://doi.org/10.1162/neco_a_00912.
Friston, Karl, and Stefan Kiebel. 2009. "Predictive Coding Under the Free-Energy Principle." *Philosophical Transactions of the Royal Society B: Biological Sciences* 364 (1521): 1211–21. https://doi.org/10.1098/rstb.2008.0300.
Friston, Karl, James Kilner, and Lee Harrison. 2006. "A Free Energy Principle for the Brain." *Journal of Physiology-Paris* 100 (1–3): 70–87. https://doi.org/10.1016/j.jphysparis.2006.10.001.
Friston, Karl, Jérémie Mattout, Nelson Trujillo-Barreto, John Ashburner, and Will Penny. 2007. "Variational Free Energy and the Laplace Approximation." *NeuroImage* 34 (1): 220–34. https://doi.org/10.1016/j.neuroimage.2006.08.035.
Friston, Karl, Thomas Parr, and Bert de Vries. 2018. "The Graphical Brain: Belief Propagation and Active Inference." *Network Neuroscience* 1 (4): 381–414. https://doi.org/10.1162/NETN_a_00018.
Friston, Karl, Francesco Rigoli, Dimitri Ognibene, Christoph Mathys, Thomas Fitzgerald, and Giovanni Pezzulo. 2015. "Active Inference and Epistemic Value." *Cognitive Neuroscience* 6 (4): 187–214. https://doi.org/10.1080/17588928.2015.1020053.
Friston, Karl, Nelson Trujillo-Barreto, and Jean Daunizeau. 2008. "DEM: A Variational Treatment of Dynamic Systems." *NeuroImage* 41 (3): 849–85. https://doi.org/10.1016/j.neuroimage.2008.02.054.
Gibson, James J. 1979. *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin.
Heins, Conor, Brennan Klein, Daphne Demekas, Miguel Aguilera, and Christopher L. Buckley. 2022. "Spin Glass Systems as Collective Active Inference." https://doi.org/10.48550/arXiv.2207.06970.
Heins, Conor, Beren Millidge, Daphne Demekas, Brennan Klein, Karl Friston, Alec W. Corcoran, and Alexander Tschantz. 2022. "Pymdp: A Python Library for Active Inference in Discrete State Spaces." *Journal of Open Source Software* 7 (73): 4098. https://doi.org/10.21105/joss.04098.
Hoeffding, Wassily. 1963. "Probability Inequalities for Sums of Bounded Random Variables." *Journal of the American Statistical Association* 58 (301): 13–30. https://doi.org/10.1080/01621459.1963.10500830.
Jarzynski, Christopher. 1997. "Nonequilibrium Equality for Free Energy Differences." *Physical Review Letters* 78 (14): 2690–93. https://doi.org/10.1103/PhysRevLett.78.2690.
Jaynes, Edwin T. 1957. "Information Theory and Statistical Mechanics." *Physical Review* 106 (4): 620–30. https://doi.org/10.1103/PhysRev.106.620.
Jiang, Albert Q., Sean Welleck, Jin Peng Zhou, Wenda Li, Jiacheng Liu, Mateja Jamnik, Timothée Lacroix, Yuhuai Wu, and Guillaume Lample. 2023. "Draft, Sketch, and Prove: Guiding Formal Theorem Provers with Informal Proofs." In *International Conference on Learning Representations (ICLR)*.
Kappen, Hilbert J. 2005. "Path Integrals and Symmetry Breaking for Optimal Control Theory." *Journal of Statistical Mechanics: Theory and Experiment* 2005 (11): P11011. https://doi.org/10.1088/1742-5468/2005/11/P11011.
Landauer, Rolf. 1961. "Irreversibility and Heat Generation in the Computing Process." *IBM Journal of Research and Development* 5 (3): 183–91. https://doi.org/10.1147/rd.53.0183.
Lean FRO. 2026a. "Lean 4.33.0 Release Notes." Lean Language Reference. https://lean-lang.org/doc/reference/latest/releases/v4.33.0/.
———. 2026b. "Lean 4.33.1 Release." GitHub release. https://github.com/leanprover/lean4/releases/tag/v4.33.1.
Leroy, Xavier. 2009. "Formal Verification of a Realistic Compiler." *Communications of the ACM* 52 (7): 107–15. https://doi.org/10.1145/1538788.1538814.
McAllester, David A. 1999. "Some PAC-Bayesian Theorems." *Machine Learning* 37 (3): 355–63. https://doi.org/10.1023/A:1007618624809.
Mehta, Ravi, Reynald Affeldt, Jacques Garrigue, and Kazuhiko Sakaguchi. 2021. "A Library for Formalised Information Theory in Isabelle/HOL." In *Interactive Theorem Proving (ITP)*.
Millidge, Beren, Alexander Tschantz, and Christopher L. Buckley. 2021. "Whence the Expected Free Energy?" *Neural Computation* 33 (2): 447–82. https://doi.org/10.1162/neco_a_01354.
Moura, Leonardo de, and Sebastian Ullrich. 2021. "The Lean 4 Theorem Prover and Programming Language." In *Automated Deduction – CADE 28*, 12699:625–35. Lecture Notes in Computer Science. Springer. https://doi.org/10.1007/978-3-030-79876-5_37.
Namjoshi, Sanjeev V. 2026. *Fundamentals of Active Inference: Principles, Algorithms, and Applications of the Free Energy Principle for Engineers*. The MIT Press.
Olfati-Saber, Reza, J. Alex Fax, and Richard M. Murray. 2007. "Consensus and Cooperation in Networked Multi-Agent Systems." *Proceedings of the IEEE* 95 (1): 215–33. https://doi.org/10.1109/JPROC.2006.887293.
Parr, Thomas, Lancelot Da Costa, and Karl J. Friston. 2018. "Markov Blankets, Information Geometry and Stochastic Thermodynamics." *Philosophical Transactions of the Royal Society A* 378 (2164): 20190159. https://doi.org/10.1098/rsta.2019.0159.
Parr, Thomas, and Karl J. Friston. 2019. "Generalised Free Energy and Active Inference." *Biological Cybernetics* 113 (5–6): 495–513. https://doi.org/10.1007/s00422-019-00805-w.

Parr, Thomas, Giovanni Pezzulo, and Karl J. Friston. 2022. *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. MIT Press. https://doi.org/10.7551/mitpress/14088.001.0001.

Paulson, Lawrence C. 2022. "Formalised Statistical Mechanics in Isabelle/HOL." Archive of Formal Proofs.

Pavliotis, Grigorios A. 2014. *Stochastic Processes and Applications*. Vol. 60. Texts in Applied Mathematics. Springer. https://doi.org/10.1007/978-1-4939-1323-7.

"Polynomial Freiman–Ruzsa in Lean 4." 2023. https://github.com/teorth/pfr.

Prigogine, Ilya, and Grégoire Nicolis. 1977. *Self-Organization in Nonequilibrium Systems*. New York: Wiley-Interscience.

Rokhlin, Vladimir A. 1949. "On the Fundamental Ideas of Measure Theory." *Matematicheskii Sbornik* 25(67) (1): 107–50. https://www.mathnet.ru/eng/sm5995.

Rosas, Fernando E., Pedro A. M. Mediano, Martin Biehl, Shamil Chandaria, and Daniel Polani. 2020. "Causal Blankets: Theory and Algorithmic Framework." *arXiv Preprint arXiv:2008.12568*. https://doi.org/10.48550/arXiv.2008.12568.

Sajid, Nadia, Philip J. Ball, Thomas Parr, and Karl J. Friston. 2021. "Active Inference: Demystified and Compared." *Neural Computation* 33 (3): 674–712. https://doi.org/10.1162/neco_a_01357.

Sakthivadivel, Dalton A. R. 2023. "On Bayesian Mechanics: A Physics of and by Beliefs." *Interface Focus* 13 (3): 20220029. https://doi.org/10.1098/rsfs.2022.0029.

Scholze, Peter, and Johan Commelin. 2022. "Liquid Tensor Experiment: Lean Formalization of Condensed Mathematics." https://leanprover-community.github.io/lt2021/.

Smékal, Jakub, and Daniel A. Friedman. 2023. "Generalized Notation Notation for Active Inference Models." *Active Inference Journal*. https://doi.org/10.5281/zenodo.7803328.

Song, Peiyang, Kaiyu Yang, and Anima Anandkumar. 2025. "Lean Copilot: Large Language Models as Copilots for Theorem Proving in Lean." In *International Conference on Neuro-Symbolic Systems*, 288:144–69. Proceedings of Machine Learning Research. PMLR. https://proceedings.mlr.press/v288/song25a.html.

The mathlib Community. 2020. "The Lean Mathematical Library." https://leanprover-community.github.io/mathlib4/.

———. 2026. "Mathlib v4.33.1 Release." GitHub release. https://github.com/leanprover-community/mathlib4/releases/tag/v4.33.1.

Todorov, Emanuel. 2006. "Linearly-Solvable Markov Decision Problems." In *Advances in Neural Information Processing Systems 19*. https://proceedings.neurips.cc/paper/2006/hash/d806ca13ca3449af72a1ea5aedbed26a-Abstract.html.

Tooby-Smith, Joseph. 2024. "Formalization of Physics Index Notation in Lean 4 (HEPLean)." https://github.com/HEPLean/PhysLean.

Trinh, Trieu H., Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. 2024. "AlphaGeometry: Solving Olympiad Geometry Without Human Demonstrations." *Nature* 625 (7995): 476–82. https://doi.org/10.1038/s41586-023-06747-5.

Xin, Huajian et al. 2025. "DeepSeek-Prover-V2: Advancing Formal Mathematical Reasoning via Reinforcement Learning for Subgoal Decomposition." *arXiv Preprint arXiv:2504.21801*.

Xin, Huajian, Daya Guo, Zhihong Shao, Zhizhou Ren, Qihao Zhu, Bo Liu, Chong Ruan, Wenda Li, and Xiaodan Liang. 2024a. "DeepSeek-Prover: Advancing Theorem Proving in LLMs Through Large-Scale Synthetic Data." *arXiv Preprint arXiv:2405.14333*.

———. 2024b. "LEGO-Prover: Neural Theorem Proving with Growing Libraries." In *International Conference on Learning Representations (ICLR)*.

Yang, Kaiyu, Aidan M. Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. 2024. "LeanDojo: Theorem Proving with Retrieval-Augmented Language Models." *Advances in Neural Information Processing Systems (NeurIPS)* 36.

12 Appendix A: Formalisms Overview

Appendices B and C (material) are one auto-generated file, `09z_unified_formalism_catalogue.md`, produced during Manuscript Artifacts. It juxtaposes, per `fep-NNN` topic, the fenced Lean body and the typeset display-math blocks (former appendices B and C). Stable descriptive metadata comes from `config/catalogue_metadata.yaml`, semantic review from `config/theorem_maturity.yaml`, canonical Lean bodies from family modules under `src/fep_lean/catalogue/bodies/`, and equation signatures from `src/fep_lean/catalogue/latex.py`; `scripts/_maint_build_topics_catalogue.py` joins them into the generated catalogue. Each topic has one display-math **block** per `theorem` (typically `aligned`), with a unique stable id `eq:fep-NNN-k`. Counts and validated evidence projections come from `manuscript_vars.yaml`. See `docs/_generated/canonical_facts.md` for status.

12.1 Complete Topic Catalogue

The **per-topic index** (id, human title, area, primary Mathlib path, and full `lean_sketch`) is not duplicated here: it appears in the **unified formalism appendix** (§13), with stable per-topic anchors `#sec:catalogue-fep-NNN` (Lean sketch) and `#sec:eqs-fep-NNN` (typeset LaTeX equations) for each `fep-NNN`. That file is regenerated from `config/topics.yaml` on every Manuscript Artifacts pass, so titles and bodies cannot drift from the committed catalogue.

Native compilation status for the full roster is **`155/155 (100.0%)`** against Mathlib **`v4.33.1`** / Lean **`leanprover/lean4:v4.33.1`**. `manuscript_vars.yaml` prefers an independently validated, live-source-bound native receipt and accepts a full-run verification manifest only through its separate claim-ready report contract. Diagnostics and verifier fields are summarized in §5.5.

Summary: All 155 rows are `mathlib_status: real` in `topics.yaml`. Per-area rates are **`41/41 (100.0%)`** (FEP core), **`31/31 (100.0%)`** (Active Inference), **`21/21 (100.0%)`** (Information Geometry), **`41/41 (100.0%)`** (Bayesian Mechanics), and **`21/21 (100.0%)`** (Thermodynamics); see §5.5.1.

12.2 Area Breakdown

Area (topics)	Native compile (verifier)	Mathlib domains
FEP core (41)	`41/41 (100.0%)`	Measures, real logarithms
Active Inference (31)	`31/31 (100.0%)`	Finite sets, big operators, order
Information Geometry (21)	`21/21 (100.0%)`	Inner products, metric spaces
Bayesian Mechanics (41)	`41/41 (100.0%)`	Matrices, probability measures
Thermodynamics (21)	`21/21 (100.0%)`	Real logarithms and exponentials
Total (155)	**`155/155 (100.0%)`**	—

Rates come from the selected validated evidence projected into `manuscript_vars.yaml`:

- Receipt: `native-566a22fce86c`.
- Native verifier ran: `true`.
- Clean / failed / recorded topics: `155` / `0` / `155`.

After any toolchain or sketch change, regenerate a source-bound receipt with:

```
uv run fep-lean verify \
  --fail-on-warnings \
  --receipt output/native-verification.json
```

Then refresh the catalogue and manuscript projections.

12.3 Representative topics (pointers only)

To avoid duplicating Lean that can drift from the SSOT, this appendix does **not** paste catalogue fences. The pointers below cover the variational-bound backbone (fep-001), Boltzmann–Gibbs weights (fep-031), and stick-breaking bookkeeping (fep-046).

Topic	Appendix B (Lean)	Appendix C (display math)
fep-001	§13.1	§14.1
fep-031	§13.31	§14.31
fep-046	§13.46	§14.46

12.4 Mathlib4 Imports Used Across the Catalogue

Every shipped row in Appendix B declares its own Mathlib imports rather than relying on an implicit global prelude. The project uses measure/probability kernels and conditional independence, finite sums and sets, real logarithm/exponential and calculus, finite-dimensional matrices and metrics, information theory, contraction limits, and the real strong law. Pedagogical snippets may use broader imports for exposition only; they are not catalogue sources.

The generated formalism coverage report is the exact import and declaration roster. It owns both the topic-to-Mathlib incidence table and the maintained formal-module dependency graph. The atlas renders those import dependencies separately from scientific edges, while formal relations require named declarations. This avoids a static appendix roster that would drift whenever a sketch narrows an import.

12.5 Formalization Epistemology: Realism vs. Illusionism

The catalogue offers a structural way to inspect claims that appear in debates about FEP and consciousness (Solms, Dołęga, Wiese, 2023–2025) (Beni et al. 2023). Its typed measure-theoretic rows illustrate how candidate belief objects can receive explicit domains and assumptions, and its finite kernel composes one bounded generative-model instance through posterior, VFE, EFE, policy, and rollout layers. It does not formalize consciousness, establish that the finite model describes a living system, or adjudicate realist and illusionist positions.

For machine-checked native compilation, run:

```
uv run fep-lean verify \
  --fail-on-warnings \
  --receipt output/native-verification.json
uv run fep-lean catalogue
```

Validate the receipt against the live tree before using its result. Full-run manifests under `output/reports/run_*/` are a separate Hermes/OpenGauss evidence class and cannot substitute for the native receipt.

13 Appendix B: Full Lean Catalogue

13.1 fep-001 — Variational Free Energy Bound

Surprisal plus Mathlib's native measure KL upper-bounds surprisal; for finite measures and finite surprisal, equality holds exactly when the approximate and exact posterior laws coincide. Assumption scope: The inequality is unconditional in ENNReal. Gap separation requires finite measures, and bound equality additionally requires surprisal below infinity so additive cancellation is sound. Semantic disposition: `formalized`.

13.1.1 Lean sketch

```
import Mathlib.InformationTheory.KullbackLeibler.Basic

namespace FEP001

variable {α : Type*} [MeasurableSpace α]

open MeasureTheory
open scoped ENNReal

/-- Native variational gap between an approximate and exact posterior law. -/
noncomputable def fep001_variationalGap
    (approximation posterior : Measure α) : ENNReal :=
  InformationTheory.klDiv approximation posterior

/-- Surprisal plus the native KL variational gap. -/
noncomputable def fep001_variationalUpperBound
    (approximation posterior : Measure α) (surprisal : ENNReal) : ENNReal :=
  surprisal + fep001_variationalGap approximation posterior

/-- The KL remainder makes the variational quantity an upper bound. -/
theorem fep001_variationalUpperBound_ge
    (approximation posterior : Measure α) (surprisal : ENNReal) :
    surprisal ≤
      fep001_variationalUpperBound approximation posterior surprisal := by
  exact le_add_right (le_refl _)

/-- For finite measures, the variational gap vanishes exactly at the posterior. -/
theorem fep001_variationalGap_eq_zero_iff
    (approximation posterior : Measure α)
    [IsFiniteMeasure approximation] [IsFiniteMeasure posterior] :
    fep001_variationalGap approximation posterior = 0 ↔
      approximation = posterior := by
  exact InformationTheory.klDiv_eq_zero_iff

/-- At finite surprisal, the variational bound is exact exactly when the
approximation equals the posterior. -/
theorem fep001_variationalUpperBound_eq_iff
    (approximation posterior : Measure α)
    [IsFiniteMeasure approximation] [IsFiniteMeasure posterior]
    {surprisal : ENNReal} (hsurprisal : surprisal ≠ ∞) :
    fep001_variationalUpperBound approximation posterior surprisal = surprisal ↔
      approximation = posterior := by
  constructor
  · intro hbound
    have hadd : surprisal + fep001_variationalGap approximation posterior =
        surprisal + 0 := by
      simpa [fep001_variationalUpperBound] using hbound
```

```
    have hgap : fep001_variationalGap approximation posterior = 0 :=
      (ENNReal.add_right_inj hsurprisal).mp hadd
    exact (fep001_variationalGap_eq_zero_iff approximation posterior).mp hgap
  · intro hequal
    subst posterior
    simp [fep001_variationalUpperBound, fep001_variationalGap,
      InformationTheory.klDiv_self]

end FEP001
```

13.1.2 Typeset statement signatures

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{approximation\ posterior} : \mathsf{Measure}\ \alpha)\ (\mathsf{surprisal} : \mathbb{R}_{\geq 0}^{\infty})$$
$$\mathsf{surprisal} \leq \mathsf{fep001\_variationalUpperBound\ approximation\ posterior\ surprisal}$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{approximation\ posterior} : \mathsf{Measure}\ \alpha)\ [\mathsf{IsFiniteMeasure\ approximation}]$$
$$[\mathsf{IsFiniteMeasure\ posterior}]$$
$$\mathsf{fep001\_variationalGap\ approximation\ posterior} = 0 \leftrightarrow \mathsf{approximation} = \mathsf{posterior}$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{approximation\ posterior} : \mathsf{Measure}\ \alpha)\ [\mathsf{IsFiniteMeasure\ approximation}]$$
$$[\mathsf{IsFiniteMeasure\ posterior}]\ \{\mathsf{surprisal} : \mathbb{R}_{\geq 0}^{\infty}\}\ (\mathsf{hsurprisal} : \mathsf{surprisal} \neq \infty)$$
$$\mathsf{fep001\_variationalUpperBound\ approximation\ posterior\ surprisal} = \mathsf{surprisal}$$
$$\leftrightarrow \mathsf{approximation} = \mathsf{posterior}$$

13.2 fep-002 — Variational Evidence Bound via KL Divergence

Variational free energy, defined as surprisal plus Mathlib KL divergence, upper-bounds surprisal and is exact at the posterior. Assumption scope: The bound is unconditional over measures in a measurable space; exactness uses Mathlib's SigmaFinite assumption for KL self-divergence. The codomain is ENNReal, so infinite divergences are represented without a finiteness side condition. Semantic disposition: `formalized`.

13.2.1 Lean sketch

```
import Mathlib.InformationTheory.KullbackLeibler.Basic
import Mathlib.MeasureTheory.Measure.Typeclasses.Probability

namespace FEP002

variable {α : Type*} [MeasurableSpace α]

open MeasureTheory
open scoped ENNReal

-- [proof strategy: IsProbabilityMeasure.measure_univ + measure_compl for complement;
 ↪  linarith for ELBO]
/-- A probability measure assigns unit total mass. -/
theorem fep002_prob_measure_univ (μ : Measure α) [IsProbabilityMeasure μ] :
    μ Set.univ = 1 :=
  IsProbabilityMeasure.measure_univ

/-- Complement: μ(sᶜ) = 1 - μ(s) for probability measures on measurable sets. -/
theorem fep002_prob_compl (μ : Measure α) [IsProbabilityMeasure μ] {s : Set α}
    (hs : MeasurableSet s) (hfin : μ s ≠ ⊤) :
    μ sᶜ = 1 - μ s := by
  rw [measure_compl hs hfin, IsProbabilityMeasure.measure_univ]

/-- A scalar ELBO decomposition: free energy ≤ log marginal likelihood when the
```

```
KL remainder is nonnegative. -/
theorem fep002_elbo_bound (logEvidence freeEnergy klDiv : ℝ)
    (h_decomp : logEvidence = freeEnergy + klDiv)
    (h_kl_nonneg : 0 ≤ klDiv) :
    freeEnergy ≤ logEvidence := by linarith

/-- Variational free energy in Mathlib's native nonnegative extended-real KL
codomain: surprisal plus the divergence from an approximate posterior to the
exact posterior. -/
noncomputable def fep002_variationalFreeEnergy
    (q posterior : Measure α) (surprisal : ENNReal) : ENNReal :=
  surprisal + InformationTheory.klDiv q posterior

/-- The KL remainder makes variational free energy an upper bound on
surprisal. -/
theorem fep002_vfe_ge_surprisal (q posterior : Measure α) (surprisal : ENNReal) :
    surprisal ≤ fep002_variationalFreeEnergy q posterior surprisal := by
  simp only [fep002_variationalFreeEnergy]
  exact le_add_right (le_refl surprisal)

/-- The variational bound is exact when the approximation is the posterior
itself. -/
theorem fep002_vfe_exact_at_posterior (posterior : Measure α)
    [SigmaFinite posterior] (surprisal : ENNReal) :
    fep002_variationalFreeEnergy posterior posterior surprisal = surprisal := by
  simp [fep002_variationalFreeEnergy, InformationTheory.klDiv_self]

end FEP002
```

13.2.2 Typeset statement signatures

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu : \mathsf{Measure}\ \alpha)\,[\mathsf{IsProbabilityMeasure}\ \mu]$$
$$\mu\,\Omega = 1$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu : \mathsf{Measure}\ \alpha)\,[\mathsf{IsProbabilityMeasure}\ \mu]\,\{s : \mathsf{Set}\ \alpha\}\,(\mathsf{hs} : \mathsf{MeasurableSet} s)$$
$$(\mathsf{hfin} : \mu\, s \neq \top)$$
$$\mu\, s^{\mathsf{c}} = 1 - \mu\, s$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{logEvidence}\ \mathsf{freeEnergy}\ \mathsf{klDiv} : \mathbb{R})\,(\mathsf{h\_decomp} : \mathsf{logEvidence} = \mathsf{freeEnergy} + \mathsf{klDiv})$$
$$(\mathsf{h\_kl\_nonneg} : 0 \leq \mathsf{klDiv})$$
$$\mathsf{freeEnergy} \leq \mathsf{logEvidence}$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(q\ \mathsf{posterior} : \mathsf{Measure}\ \alpha)\,(\mathsf{surprisal} : \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{surprisal} \leq \mathsf{fep002\_variationalFreeEnergy}\ q\ \mathsf{posterior}\ \mathsf{surprisal}$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{posterior} : \mathsf{Measure}\ \alpha)\,[\mathsf{SigmaFinite}\ \mathsf{posterior}]\,(\mathsf{surprisal} : \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{fep002\_variationalFreeEnergy}\ \mathsf{posterior}\ \mathsf{posterior}\ \mathsf{surprisal} = \mathsf{surprisal}$$

13.3 fep-003 — Discounted Pragmatic Cost

Finite-horizon pragmatic cost is the discounted ENNReal sum of stage costs; horizon extension exposes the exact terminal increment, costs are pointwise monotone, and horizon extension is monotone. Assumption scope: All costs and the discount live in ENNReal, so nonnegativity and infinity are represented directly. This is the pragmatic-cost input to EFE; epistemic value and its sign are owned by fep-021. Semantic disposition: `formalized`.

13.3.1 Lean sketch

```
import Mathlib.Algebra.Order.BigOperators.Group.Finset
import Mathlib.Data.ENNReal.Inv

namespace FEP003

open scoped ENNReal

/-- Discounted pragmatic expected-cost input to the EFE convention. -/
noncomputable def fep003_pragmaticCost
    (discount : ENNReal) (stageCost : ℕ → ENNReal) (horizon : ℕ) : ENNReal :=
  ∑ t ∈ Finset.range horizon, discount ^ t * stageCost t

/-- A zero planning horizon has zero pragmatic cost. -/
theorem fep003_pragmaticCost_zero
    (discount : ENNReal) (stageCost : ℕ → ENNReal) :
    fep003_pragmaticCost discount stageCost 0 = 0 := by
  simp [fep003_pragmaticCost]

/-- Extending the horizon exposes the exact discounted terminal increment. -/
theorem fep003_pragmaticCost_succ
    (discount : ENNReal) (stageCost : ℕ → ENNReal) (horizon : ℕ) :
    fep003_pragmaticCost discount stageCost (horizon + 1) =
      fep003_pragmaticCost discount stageCost horizon +
        discount ^ horizon * stageCost horizon := by
  simp [fep003_pragmaticCost, Finset.sum_range_succ]

/-- Pointwise larger stage costs produce larger pragmatic cost. -/
theorem fep003_pragmaticCost_mono
    (discount : ENNReal) {cost₁ cost₂ : ℕ → ENNReal}
    (hcost : ∀ t, cost₁ t ≤ cost₂ t) (horizon : ℕ) :
    fep003_pragmaticCost discount cost₁ horizon ≤
      fep003_pragmaticCost discount cost₂ horizon := by
  exact Finset.sum_le_sum fun t _ => mul_le_mul_right (hcost t) _

/-- Every additional nonnegative stage weakly increases finite-horizon cost. -/
theorem fep003_pragmaticCost_horizon_succ
    (discount : ENNReal) (stageCost : ℕ → ENNReal) (horizon : ℕ) :
    fep003_pragmaticCost discount stageCost horizon ≤
      fep003_pragmaticCost discount stageCost (horizon + 1) := by
  rw [fep003_pragmaticCost_succ]
  exact le_add_right (le_refl _)

end FEP003
```

13.3.2 Typeset statement signatures

$$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\,(\mathsf{stageCost} : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{fep003\_pragmaticCost}\ \mathsf{discount}\ \mathsf{stageCost}\ 0 = 0$$

$$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\,(\mathsf{stageCost} : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0})\,(\mathsf{horizon} : \mathbb{N})$$
$$\mathsf{fep003\_pragmaticCost}\ \mathsf{discount}\ \mathsf{stageCost}\ (\mathsf{horizon} + 1) = \mathsf{fep003\_pragmaticCost}$$
$$\mathsf{discount}\ \mathsf{stageCost}\ \mathsf{horizon} + \mathsf{discount}^{\mathsf{horizon}} \cdot \mathsf{stageCost}\ \mathsf{horizon}$$

$$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\,\{cost_1\ cost_2 : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0}\}$$
$$(\mathsf{hcost} : \forall t, cost_1\ t \leq cost_2\ t)\,(\mathsf{horizon} : \mathbb{N})$$
$$\mathsf{fep003\_pragmaticCost}\ \mathsf{discount}\ cost_1\ \mathsf{horizon} \leq \mathsf{fep003\_pragmaticCost}\ \mathsf{discount}$$
$$cost_2\ \mathsf{horizon}$$

$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{stageCost} : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{horizon} : \mathbb{N})$

$\mathsf{fep003\_pragmaticCost\ discount\ stageCost\ horizon} \leq \mathsf{fep003\_pragmaticCost\ discount}$

$\mathsf{stageCost}\ (\mathsf{horizon} + 1)$

13.4 fep-004 — Finite Diagonal Fisher Metric

A finite diagonal Fisher metric is symmetric and positive semidefinite under nonnegative information weights, and becomes positive definite under strictly positive weights. Assumption scope: The information weights are explicit inputs rather than derived from a statistical family in this row. Positive definiteness requires every coordinate weight to be strictly positive; fep-038 computes the one-dimensional Bernoulli weight and the composed theorem identifies that specialization. Semantic disposition: `formalized`.

13.4.1 Lean sketch

```
import Mathlib.Algebra.Order.BigOperators.Group.Finset
import Mathlib.Tactic

namespace FEP004

open Finset

/-- A finite diagonal Fisher metric with explicitly supplied information
weights.  Statistical models such as fep-038 compute these weights from
expected squared scores. -/
def fep004_fisherMetric {ι : Type*} [Fintype ι]
    (information u v : ι → ℝ) : ℝ :=
  ∑ i, information i * u i * v i

/-- The finite Fisher metric is symmetric. -/
theorem fep004_fisherMetric_symm {ι : Type*} [Fintype ι]
    (information u v : ι → ℝ) :
    fep004_fisherMetric information u v =
      fep004_fisherMetric information v u := by
  apply Finset.sum_congr rfl
  intro i _
  ring

/-- Nonnegative information weights make every tangent self-pairing
nonnegative. -/
theorem fep004_fisherMetric_nonneg {ι : Type*} [Fintype ι]
    (information v : ι → ℝ) (hinformation : ∀ i, 0 ≤ information i) :
    0 ≤ fep004_fisherMetric information v v := by
  exact Finset.sum_nonneg fun i _ => by
    simpa [mul_assoc] using
      mul_nonneg (hinformation i) (mul_self_nonneg (v i))

/-- Strictly positive information weights make the metric positive definite:
zero tangent norm is equivalent to the zero tangent vector. -/
theorem fep004_fisherMetric_eq_zero_iff {ι : Type*} [Fintype ι]
    (information v : ι → ℝ) (hinformation : ∀ i, 0 < information i) :
    fep004_fisherMetric information v v = 0 ↔ ∀ i, v i = 0 := by
  constructor
  · intro hzero i
    have hterm : information i * v i * v i = 0 :=
      (Finset.sum_eq_zero_iff_of_nonneg
        (fun j _ => by
          simpa [mul_assoc] using
            mul_nonneg (hinformation j).le (mul_self_nonneg (v j)))).mp
        hzero i (Finset.mem_univ i)
    have hvv : v i * v i = 0 := by
      apply (mul_eq_zero.mp (by simpa [mul_assoc] using hterm)).resolve_left
      exact ne_of_gt (hinformation i)
```

```
    exact mul_self_eq_zero.mp hvv
  · intro hv
    simp [fep004_fisherMetric, hv]

end FEP004
```

13.4.2 Typeset statement signatures

$$\{\iota : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\iota]\,(\mathsf{information}\,u\,v : \iota \Rightarrow \mathbb{R})$$
$$\mathsf{fep004\_fisherMetric}\,\mathsf{information}\,u\,v = \mathsf{fep004\_fisherMetric}\,\mathsf{information}\,v\,u$$

$$\{\iota : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\iota]\,(\mathsf{information}\,v : \iota \Rightarrow \mathbb{R})$$
$$(\mathsf{hinformation} : \forall i, 0 \le \mathsf{information}\,i)$$
$$0 \le \mathsf{fep004\_fisherMetric}\,\mathsf{information}\,v\,v$$

$$\{\iota : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\iota]\,(\mathsf{information}\,v : \iota \Rightarrow \mathbb{R})$$
$$(\mathsf{hinformation} : \forall i, 0 < \mathsf{information}\,i)$$
$$\mathsf{fep004\_fisherMetric}\,\mathsf{information}\,v\,v = 0 \leftrightarrow \forall i, v\,i = 0$$

13.5 fep-005 — Four-Block State Partition

A function-valued assignment partitions a 20-state finite space into four pairwise-disjoint blocks, and every state belongs to exactly one block. Assumption scope: Unconditional for any assignment Fin 20 → Fin 4. This establishes an exact finite partition only; it does not infer conditional independence or identify any block as a learned Markov blanket. Semantic disposition: `formalized`.

13.5.1 Lean sketch

```
import Mathlib.Data.Finset.Basic
import Mathlib.Data.Finset.Filter
import Mathlib.Data.Fintype.Basic

namespace FEP005

open Finset

abbrev BlkPart := Fin 4  -- internal(0), sensory(1), active(2), active/external(3)

/-- Markov blanket partition: four components cover the full state space exactly. -/
def fep005_partitionCover (assign : Fin 20 → BlkPart) (k : BlkPart) : Finset (Fin 20)
 ↪  :=
  Finset.univ.filter (fun s => assign s = k)

/-- The four partition blocks are pairwise disjoint (functional determinism). -/
theorem fep005_disjoint (assign : Fin 20 → BlkPart) (i j : BlkPart) (hij : i ≠ j) :
    Disjoint (fep005_partitionCover assign i) (fep005_partitionCover assign j) := by
  refine Finset.disjoint_left.mpr ?_
  intro x hx hy
  simp only [fep005_partitionCover, Finset.mem_filter, Finset.mem_univ, true_and] at
 ↪  hx hy
  exact hij ((Eq.symm hx).trans hy)

/-- Every state belongs to exactly one partition block (totality of assignment). -/
theorem fep005_total_cover (assign : Fin 20 → BlkPart) :
    ∀ s : Fin 20, ∃ k : BlkPart, s ∈ fep005_partitionCover assign k := by
  intro s
  refine ⟨assign s, Finset.mem_filter.mpr ⟨Finset.mem_univ s, rfl⟩⟩

/-- Membership is decidable: a state is in block k iff its assignment equals k. -/
theorem fep005_mem_iff (assign : Fin 20 → BlkPart) (s : Fin 20) (k : BlkPart) :
    s ∈ fep005_partitionCover assign k ↔ assign s = k := by
```

```
  dsimp [fep005_partitionCover]
  exact Iff.intro (fun h => (Finset.mem_filter.mp h).2)
    fun hk => Finset.mem_filter.mpr ⟨Finset.mem_univ s, hk⟩

/-- Every state lies in one and only one block. -/
theorem fep005_existsUnique_block (assign : Fin 20 → BlkPart) (s : Fin 20) :
    ∃! k : BlkPart, s ∈ fep005_partitionCover assign k := by
  refine ⟨assign s, (fep005_mem_iff assign s (assign s)).2 rfl, ?_⟩
  intro k hk
  exact ((fep005_mem_iff assign s k).1 hk).symm

end FEP005
```

13.5.2 Typeset statement signatures

$$(\mathsf{assign} : \mathsf{Fin}\,20 \Rightarrow \mathsf{BlkPart})\,(i\,j : \mathsf{BlkPart})\,(\mathsf{hij} : i \neq j)$$
$$\mathsf{Disjoint}\,(\mathsf{fep005\_partitionCover\ assign}\ i)\,(\mathsf{fep005\_partitionCover\ assign}\ j)$$

$$(\mathsf{assign} : \mathsf{Fin}\,20 \Rightarrow \mathsf{BlkPart})$$
$$\forall s : \mathsf{Fin}\,20, \exists k : \mathsf{BlkPart}, s \in \mathsf{fep005\_partitionCover\ assign}\ k$$

$$(\mathsf{assign} : \mathsf{Fin}\,20 \Rightarrow \mathsf{BlkPart})\,(s : \mathsf{Fin}\,20)\,(k : \mathsf{BlkPart})$$
$$s \in \mathsf{fep005\_partitionCover\ assign}\ k \leftrightarrow \mathsf{assign}\ s = k$$

$$(\mathsf{assign} : \mathsf{Fin}\,20 \Rightarrow \mathsf{BlkPart})\,(s : \mathsf{Fin}\,20)$$
$$\exists! k : \mathsf{BlkPart}, s \in \mathsf{fep005\_partitionCover\ assign}\ k$$

13.6 fep-006 — Measurable Discrete-Time Flow

Iteration of one measurable state update defines a measurable discrete-time flow with identity at time zero, an exact additive semigroup law, and persistence of fixed points. Assumption scope: Measurability of each time slice requires the one-step map to be measurable. The algebraic semigroup and fixed-point laws require no topology, invariant measure, differentiability, or continuous-time interpolation. Semantic disposition: `formalized`.

13.6.1 Lean sketch

```
import Mathlib.Dynamics.FixedPoints.Basic
import Mathlib.MeasureTheory.MeasurableSpace.Basic

namespace FEP006

variable {α : Type*}

/-- The discrete-time flow generated by repeatedly applying one state update. -/
def fep006_iterateFlow (step : α → α) (time : ℕ) : α → α :=
  step^[time]

/-- Every time slice of a measurable discrete flow is measurable. -/
theorem fep006_iterateFlow_measurable
    [MeasurableSpace α] {step : α → α} (hstep : Measurable step) (time : ℕ) :
    Measurable (fep006_iterateFlow step time) := by
  exact hstep.iterate time

/-- The time-zero flow is the identity. -/
theorem fep006_iterateFlow_zero (step : α → α) :
    fep006_iterateFlow step 0 = id := by
  rfl

/-- Discrete flow obeys the exact additive semigroup law. -/
theorem fep006_iterateFlow_add (step : α → α) (m n : ℕ) (state : α) :
    fep006_iterateFlow step (m + n) state =
```

```
      fep006_iterateFlow step m (fep006_iterateFlow step n state) := by
  exact Function.iterate_add_apply step m n state

/-- A fixed point remains fixed at every discrete time. -/
theorem fep006_iterateFlow_fixed
    (step : α → α) {state : α} (hfixed : Function.IsFixedPt step state)
    (time : ℕ) :
    fep006_iterateFlow step time state = state := by
  exact hfixed.iterate time

end FEP006
```

13.6.2 Typeset statement signatures

$$\{\alpha : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ \{\mathsf{step} : \alpha \Rightarrow \alpha\}\ (\mathsf{hstep} : \mathsf{Measurable}\ \mathsf{step})\ (\mathsf{time} : \mathbb{N})$$
$$\mathsf{Measurable}\ (\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ \mathsf{time})$$

$$\{\alpha : \mathsf{Type}^*\}$$
$$(\mathsf{step} : \alpha \Rightarrow \alpha)$$
$$\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ 0 = \mathsf{id}$$

$$\{\alpha : \mathsf{Type}^*\}$$
$$(\mathsf{step} : \alpha \Rightarrow \alpha)\ (m\ n : \mathbb{N})\ (\mathsf{state} : \alpha)$$
$$\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ (m + n)\ \mathsf{state} = \mathsf{fep006\_iterateFlow}\ \mathsf{step}\ m$$
$$(\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ n\ \mathsf{state})$$

$$\{\alpha : \mathsf{Type}^*\}$$
$$(\mathsf{step} : \alpha \Rightarrow \alpha)\ \{\mathsf{state} : \alpha\}\ (\mathsf{hfixed} : \mathsf{Function.IsFixedPt}\ \mathsf{step}\ \mathsf{state})\ (\mathsf{time} : \mathbb{N})$$
$$\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ \mathsf{time}\ \mathsf{state} = \mathsf{state}$$

13.7 fep-007 — Normalized Finite Sum-Product Message

Strictly positive local factors and incoming messages over a nonempty finite neighbor support define a nonnegative normalized sum-product message with support mass exactly one. Assumption scope: Normalization requires a nonempty selected support and strict positivity of every factor and incoming message on it. The message is zero outside that support. This is one exact local update, not loopy-belief-propagation convergence or global marginal correctness. Semantic disposition: `formalized`.

13.7.1 Lean sketch

```
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.Algebra.BigOperators.Field
import Mathlib.Algebra.Order.BigOperators.Group.Finset
import Mathlib.Data.Real.Basic

namespace FEP007

open Finset

-- [proof strategy: mul_nonneg / Finset.sum_nonneg for positivity; Finset.sum_le_sum
 ↪  for monotone updates]

abbrev Node := Fin 8

/-- Belief propagation: local potential products stay nonnegative when factors are. -/
theorem fep007_factorProduct_nonneg (ψ : Node → Node → ℝ) (i j : Node)
    (hψ : 0 ≤ ψ i j) (hj : 0 ≤ ψ j i) : 0 ≤ ψ i j * ψ j i := by
  exact mul_nonneg hψ hj
```

```
/-- Message aggregation: sum of factor-weighted messages stays nonneg. -/
theorem fep007_message_agg_nonneg (ψ : Node → Node → ℝ) (msg : Node → ℝ)
    (N : Finset Node) (i : Node)
    (hψ : ∀ j, 0 ≤ ψ i j) (hm : ∀ j, 0 ≤ msg j) :
    0 ≤ ∑ j ∈ N, ψ i j * msg j :=
  Finset.sum_nonneg fun j _ => mul_nonneg (hψ j) (hm j)

/-- Monotonicity: stronger messages → stronger aggregate. -/
theorem fep007_message_agg_mono (ψ : Node → Node → ℝ) (m₁ m₂ : Node → ℝ)
    (N : Finset Node) (i : Node)
    (hψ : ∀ j, 0 ≤ ψ i j) (h : ∀ j ∈ N, m₁ j ≤ m₂ j) :
    ∑ j ∈ N, ψ i j * m₁ j ≤ ∑ j ∈ N, ψ i j * m₂ j :=
  Finset.sum_le_sum fun j hj => mul_le_mul_of_nonneg_left (h j hj) (hψ j)

/-- Zero messages yield zero aggregate (absorbing behavior). -/
theorem fep007_zero_msg (ψ : Node → Node → ℝ) (N : Finset Node) (i : Node) :
    ∑ j ∈ N, ψ i j * (0 : ℝ) = 0 := by simp

/-- Positive unnormalized sum-product weight sent from neighbor `j` to node
`i`. -/
def fep007_unnormalizedMessage
    (ψ : Node → Node → ℝ) (incoming : Node → ℝ)
    (i j : Node) : ℝ :=
  ψ i j * incoming j

/-- Normalizing constant over a selected finite neighbor support. -/
def fep007_messageNormalizer
    (ψ : Node → Node → ℝ) (incoming : Node → ℝ)
    (neighbors : Finset Node) (i : Node) : ℝ :=
  ∑ j ∈ neighbors, fep007_unnormalizedMessage ψ incoming i j

/-- A normalized finite belief-propagation message, extended by zero outside
the selected support. -/
noncomputable def fep007_normalizedMessage
    (ψ : Node → Node → ℝ) (incoming : Node → ℝ)
    (neighbors : Finset Node) (i j : Node) : ℝ :=
  if j ∈ neighbors then
    fep007_unnormalizedMessage ψ incoming i j /
      fep007_messageNormalizer ψ incoming neighbors i
  else 0

/-- Strictly positive factors and incoming messages give a positive
normalizer on every nonempty support. -/
theorem fep007_messageNormalizer_pos
    (ψ : Node → Node → ℝ) (incoming : Node → ℝ)
    (neighbors : Finset Node) (i : Node)
    (hne : neighbors.Nonempty)
    (hψ : ∀ j ∈ neighbors, 0 < ψ i j)
    (hincoming : ∀ j ∈ neighbors, 0 < incoming j) :
    0 < fep007_messageNormalizer ψ incoming neighbors i := by
  exact Finset.sum_pos
    (fun j hj => mul_pos (hψ j hj) (hincoming j hj)) hne

/-- Normalized messages are nonnegative everywhere. -/
theorem fep007_normalizedMessage_nonneg
    (ψ : Node → Node → ℝ) (incoming : Node → ℝ)
    (neighbors : Finset Node) (i j : Node)
    (hne : neighbors.Nonempty)
    (hψ : ∀ k ∈ neighbors, 0 < ψ i k)
    (hincoming : ∀ k ∈ neighbors, 0 < incoming k) :
    0 ≤ fep007_normalizedMessage ψ incoming neighbors i j := by
  classical
```

```
  by_cases hj : j ∈ neighbors
  · rw [fep007_normalizedMessage, if_pos hj]
    exact div_nonneg
      (mul_nonneg (hψ j hj).le (hincoming j hj).le)
      (fep007_messageNormalizer_pos
        ψ incoming neighbors i hne hψ hincoming).le
  · simp [fep007_normalizedMessage, hj]

/-- A normalized message has total mass one on its selected support. -/
theorem fep007_normalizedMessage_sum_one
    (ψ : Node → Node → ℝ) (incoming : Node → ℝ)
    (neighbors : Finset Node) (i : Node)
    (hne : neighbors.Nonempty)
    (hψ : ∀ j ∈ neighbors, 0 < ψ i j)
    (hincoming : ∀ j ∈ neighbors, 0 < incoming j) :
    Σ j ∈ neighbors,
      fep007_normalizedMessage ψ incoming neighbors i j = 1 := by
  classical
  have hden : fep007_messageNormalizer ψ incoming neighbors i ≠ 0 :=
    ne_of_gt (fep007_messageNormalizer_pos
      ψ incoming neighbors i hne hψ hincoming)
  calc
    Σ j ∈ neighbors,
        fep007_normalizedMessage ψ incoming neighbors i j =
        Σ j ∈ neighbors,
          fep007_unnormalizedMessage ψ incoming i j /
            fep007_messageNormalizer ψ incoming neighbors i := by
      apply Finset.sum_congr rfl
      intro j hj
      simp [fep007_normalizedMessage, hj]
    _ = fep007_messageNormalizer ψ incoming neighbors i /
        fep007_messageNormalizer ψ incoming neighbors i := by
      rw [← Finset.sum_div]
      rfl
    _ = 1 := div_self hden

end FEP007
```

13.7.2 Typeset statement signatures

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (i\ j : \mathsf{Node})\ (h\psi : 0 \le \psi\ i\ j)\ (\mathsf{hj} : 0 \le \psi\ j\ i)$$
$$0 \le \psi\ i\ j \cdot \psi\ j\ i$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{msg} : \mathsf{Node} \Rightarrow \mathbb{R})\ (N : \mathsf{Finset\ Node})\ (i : \mathsf{Node})$$
$$(h\psi : \forall j, 0 \le \psi\ i\ j)\ (\mathsf{hm} : \forall j, 0 \le \mathsf{msg}\ j)$$
$$0 \le \sum j \in N, \psi\ i\ j \cdot \mathsf{msg}\ j$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (m_1\ m_2 : \mathsf{Node} \Rightarrow \mathbb{R})\ (N : \mathsf{Finset\ Node})\ (i : \mathsf{Node})$$
$$(h\psi : \forall j, 0 \le \psi\ i\ j)\ (h : \forall j \in N, m_1\ j \le m_2\ j)$$
$$\sum j \in N, \psi\ i\ j \cdot m_1\ j \le \sum j \in N, \psi\ i\ j \cdot m_2\ j$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (N : \mathsf{Finset\ Node})\ (i : \mathsf{Node})$$
$$\sum j \in N, \psi\ i\ j \cdot (0 : \mathbb{R}) = 0$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{incoming} : \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{neighbors} : \mathsf{Finset\ Node})\ (i : \mathsf{Node})$$
$$(\mathsf{hne} : \mathsf{neighbors.Nonempty})\ (h\psi : \forall j \in \mathsf{neighbors}, 0 < \psi\ i\ j)$$
$$(\mathsf{hincoming} : \forall j \in \mathsf{neighbors}, 0 < \mathsf{incoming}\ j)$$
$$0 < \mathsf{fep007\_messageNormalizer}\ \psi\ \mathsf{incoming\ neighbors}\ i$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{incoming} : \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{neighbors} : \mathsf{Finset}\ \mathsf{Node})$$
$$(i\ j : \mathsf{Node})\ (\mathsf{hne} : \mathsf{neighbors.Nonempty})\ (h\psi : \forall k \in \mathsf{neighbors}, 0 < \psi\ i\ k)$$
$$(\mathsf{hincoming} : \forall k \in \mathsf{neighbors}, 0 < \mathsf{incoming}\ k)$$
$$0 \leq \mathsf{fep007\_normalizedMessage}\ \psi\ \mathsf{incoming}\ \mathsf{neighbors}\ i\ j$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{incoming} : \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{neighbors} : \mathsf{Finset}\ \mathsf{Node})\ (i : \mathsf{Node})$$
$$(\mathsf{hne} : \mathsf{neighbors.Nonempty})\ (h\psi : \forall j \in \mathsf{neighbors}, 0 < \psi\ i\ j)$$
$$(\mathsf{hincoming} : \forall j \in \mathsf{neighbors}, 0 < \mathsf{incoming}\ j)$$
$$\sum j \in \mathsf{neighbors}, \mathsf{fep007\_normalizedMessage}\ \psi\ \mathsf{incoming}\ \mathsf{neighbors}\ i\ j = 1$$

13.8 fep-008 — Finite Policy Objective Minimizer

A real-valued objective attains a minimum on a finite nonempty policy set. Assumption scope: Existence requires only a finite nonempty policy set and an explicit real-valued objective. The theorem does not claim uniqueness of the policy; it separately proves that all minimizers agree in value. Semantic disposition: `formalized`.

13.8.1 Lean sketch

```
import Mathlib.Data.Finset.Basic
import Mathlib.Data.Finset.Max
import Mathlib.Data.Real.Basic
import Mathlib.Order.Bounds.Basic

namespace FEP008

abbrev Policy := Fin 14

/-- Discrete active inference: some policy minimizes expected free energy on a finite
 ↪  set. -/
theorem fep008_exists_minG (policies : Finset Policy) (hne : policies.Nonempty) (G :
 ↪  Policy → ℝ) :
    ∃ p ∈ policies, ∀ p' ∈ policies, G p ≤ G p' :=
  Finset.exists_min_image policies G hne

/-- Any two policy minimizers attain the same value (uniqueness of the minimum). -/
theorem fep008_min_agrees_on_value (policies : Finset Policy) (_hne :
 ↪  policies.Nonempty) (G : Policy → ℝ)
    (p p' : Policy) (hp : p ∈ policies) (hp' : p' ∈ policies)
    (hmin : ∀ p'' ∈ policies, G p ≤ G p'') (hmin' : ∀ p'' ∈ policies, G p' ≤ G p'') :
    G p = G p' :=
  le_antisymm (hmin p' hp') (hmin' p hp)

/-- Dual: some policy maximizes expected value on a finite nonempty set. -/
theorem fep008_exists_maxG (policies : Finset Policy) (hne : policies.Nonempty) (G :
 ↪  Policy → ℝ) :
    ∃ p ∈ policies, ∀ p' ∈ policies, G p' ≤ G p :=
  Finset.exists_max_image policies G hne

/-- Minimum is ≤ any evaluation in the policy set. -/
theorem fep008_min_is_lb (policies : Finset Policy) (G : Policy → ℝ)
    (p : Policy) (_hp : p ∈ policies)
    (hmin : ∀ p' ∈ policies, G p ≤ G p') :
    ∀ p' ∈ policies, G p ≤ G p' := hmin

end FEP008
```

13.8.2 Typeset statement signatures

$$(\text{policies} : \text{Finset Policy})\ (\text{hne} : \text{policies.Nonempty})\ (G : \text{Policy} \Rightarrow \mathbb{R})$$
$$\exists p \in \text{policies}, \forall p' \in \text{policies}, G\,p \le G\,p'$$

$$(\text{policies} : \text{Finset Policy})\ ({}_h ne : \text{policies.Nonempty})\ (G : \text{Policy} \Rightarrow \mathbb{R})$$
$$(p\,p' : \text{Policy})\ (\text{hp} : p \in \text{policies})\ (\text{hp}' : p' \in \text{policies})$$
$$(\text{hmin} : \forall p'' \in \text{policies}, G\,p \le G\,p'')\ (\text{hmin}' : \forall p'' \in \text{policies}, G\,p' \le G\,p'')$$
$$G\,p = G\,p'$$

$$(\text{policies} : \text{Finset Policy})\ (\text{hne} : \text{policies.Nonempty})\ (G : \text{Policy} \Rightarrow \mathbb{R})$$
$$\exists p \in \text{policies}, \forall p' \in \text{policies}, G\,p' \le G\,p$$

$$(\text{policies} : \text{Finset Policy})\ (G : \text{Policy} \Rightarrow \mathbb{R})\ (p : \text{Policy})\ ({}_h p : p \in \text{policies})$$
$$(\text{hmin} : \forall p' \in \text{policies}, G\,p \le G\,p')$$
$$\forall p' \in \text{policies}, G\,p \le G\,p'$$

13.9 fep-009 — Conditional Independence Laws

Conditional independence of two σ-algebras given a third is symmetric under Mathlib's conditional-expectation-kernel definition. Assumption scope: Conditional on a standard Borel measurable space, a finite measure, a conditioning σ-algebra below the ambient one, and the stated conditional-independence relation. The theorem is a law of conditional independence, not a learned Markov-blanket factorization. Semantic disposition: `formalized`.

13.9.1 Lean sketch

```
import Mathlib.Probability.Independence.Conditional

namespace FEP009

open MeasureTheory ProbabilityTheory

variable {Ω : Type*} [mΩ : MeasurableSpace Ω] [StandardBorelSpace Ω]

/-- Conditional independence of two σ-algebras is symmetric. Mathlib's
definition is built from a conditional-expectation kernel for a finite measure. -/
theorem fep009_condIndep_symm
    (m' m₁ m₂ : MeasurableSpace Ω) (hm' : m' ≤ mΩ)
    (μ : @Measure Ω mΩ) [IsFiniteMeasure μ]
    (h : CondIndep m' m₁ m₂ hm' μ) :
    CondIndep m' m₂ m₁ hm' μ :=
  h.symm

/-- Conditional independence is inhabited: any σ-algebra is conditionally
independent of the trivial σ-algebra. -/
theorem fep009_condIndep_bot_right
    (m' m₁ : MeasurableSpace Ω) (hm' : m' ≤ mΩ)
    (μ : @Measure Ω mΩ) [IsFiniteMeasure μ] :
    CondIndep m' m₁ ⊥ hm' μ :=
  condIndep_bot_right m₁

variable {α β : Type*} [MeasurableSpace α] [MeasurableSpace β]

-- [proof strategy: zero_le for ENNReal; measure_mono for likelihood ordering]

/-- Generative model: joint mass factors as product of marginals (independence). -/
theorem fep009_joint_product_nonneg (μ : Measure α) (ν : Measure β) (s : Set α) (t :
 ↪ Set β) :
    0 ≤ μ s * ν t :=
  bot_le
```

```
/-- Likelihood monotonicity: larger sets yield larger likelihoods. -/
theorem fep009_likelihood_mono {μ : Measure α} {s t : Set α} (h : s ⊆ t) :
    μ s ≤ μ t :=
  measure_mono h

/-- Marginalization via measure.map preserves non-negativity. -/
theorem fep009_map_nonneg (μ : Measure α) {f : α → β} (_hf : Measurable f) (s : Set β) :
    0 ≤ μ.map f s :=
  bot_le

/-- Likelihood on empty event is zero. -/
theorem fep009_empty_zero (μ : Measure α) : μ ∅ = 0 :=
  measure_empty

/-- Union bound on likelihoods (subadditivity of generative model mass). -/
theorem fep009_union_le (μ : Measure α) (s t : Set α) :
    μ (s ∪ t) ≤ μ s + μ t :=
  measure_union_le s t

end FEP009
```

13.9.2 Typeset statement signatures

$$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$$
$$(m'm_1\,m_2 : \mathsf{MeasurableSpace}\Omega)\ (\mathsf{hm}' : m' \le m\Omega)\ (\mu : @\mathsf{Measure}\Omega\,m\Omega)$$
$$[\mathsf{IsFiniteMeasure}\,\mu]\ (h : \mathsf{CondIndep}\,m'm_1\,m_2\,\mathsf{hm}'\mu)$$
$$\mathsf{CondIndep}\,m'm_2\,m_1\,\mathsf{hm}'\mu$$

$$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$$
$$(m'm_1 : \mathsf{MeasurableSpace}\Omega)\ (\mathsf{hm}' : m' \le m\Omega)\ (\mu : @\mathsf{Measure}\Omega\,m\Omega)$$
$$[\mathsf{IsFiniteMeasure}\,\mu]$$
$$\mathsf{CondIndep}\,m'm_1\bot\mathsf{hm}'\mu$$

$$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\nu : \mathsf{Measure}\beta)\ (s : \mathsf{Set}\ \alpha)\ (t : \mathsf{Set}\ \beta)$$
$$0 \le \mu\,s \cdot \nu\,t$$

$$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$$
$$\{\mu : \mathsf{Measure}\ \alpha\}\ \{s\,t : \mathsf{Set}\ \alpha\}\ (h : s \subseteq t)$$
$$\mu\,s \le \mu\,t$$

$$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ \{f : \alpha \Rightarrow \beta\}\ (_hf : \mathsf{Measurable}\ f)\ (s : \mathsf{Set}\ \beta)$$
$$0 \le \mu.\mathsf{map}\ f\ s$$

$$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$$
$$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$$
$$(\mu : \mathsf{Measure}\ \alpha)$$
$$\mu\emptyset = 0$$

$\{\Omega : \mathsf{Type}^*\}\,[m\Omega : \mathsf{MeasurableSpace}\Omega]\,[\mathsf{StandardBorelSpace}\,\Omega]\,\{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\,\alpha]\,[\mathsf{MeasurableSpace}\beta]$
$(\mu : \mathsf{Measure}\,\alpha)\,(s\,t : \mathsf{Set}\,\alpha)$
$\mu\,(s \cup t) \le \mu\,s + \mu\,t$

13.10 fep-010 — Detailed Balance Implies Invariance

A Markov kernel satisfying measure-theoretic detailed balance leaves its reference measure invariant. Assumption scope: Conditional on Mathlib's IsMarkovKernel instance and IsReversible flow equality over all measurable set pairs. This proves stationarity, not a path-space fluctuation theorem or convergence to equilibrium. Semantic disposition: `formalized`.

13.10.1 Lean sketch

```
import Mathlib.Probability.Kernel.Invariance

namespace FEP010

open MeasureTheory ProbabilityTheory

variable {α : Type*} [MeasurableSpace α]

/-- Detailed balance in Mathlib's measure-kernel sense implies stationarity:
a reversible Markov kernel leaves its reference measure invariant. -/
theorem fep010_reversible_invariant (κ : Kernel α α) (π : Measure α)
    [IsMarkovKernel κ] (hrev : Kernel.IsReversible κ π) :
    Kernel.Invariant κ π :=
  hrev.invariant

/-- The identity Markov kernel is a concrete reversible witness for every
measure: flow through two measurable sets is symmetric. -/
theorem fep010_identity_reversible (π : Measure α) :
    Kernel.IsReversible (Kernel.id : Kernel α α) π := by
  intro A B hA hB
  simp [Kernel.id_apply, hA, hB, Set.inter_comm]

/-- The identity-kernel witness leaves every measure invariant. -/
theorem fep010_identity_invariant (π : Measure α) :
    Kernel.Invariant (Kernel.id : Kernel α α) π :=
  fep010_reversible_invariant Kernel.id π (fep010_identity_reversible π)

/-- Invariance is exactly preservation by measure-kernel bind. -/
theorem fep010_invariant_def (κ : Kernel α α) (π : Measure α)
    (hinv : Kernel.Invariant κ π) :
    π.bind κ = π :=
  hinv.def

/-- Composition preserves a common invariant measure. -/
theorem fep010_invariant_comp (κ η : Kernel α α) (π : Measure α)
    (hκ : Kernel.Invariant κ π) (hη : Kernel.Invariant η π) :
    Kernel.Invariant (κ ∘ₖ η) π :=
  hκ.comp hη

end FEP010
```

13.10.2 Typeset statement signatures

$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\,\alpha]$
$(\kappa : \mathsf{Kernel}\,\alpha\,\alpha)\,(\pi : \mathsf{Measure}\,\alpha)\,[\mathsf{IsMarkovKernel}\,\kappa]$
$(\mathsf{hrev} : \mathsf{Kernel.IsReversible}\,\kappa\,\pi)$
$\mathsf{Kernel.Invariant}\,\kappa\,\pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\pi : \mathsf{Measure}\ \alpha)$
$\mathsf{Kernel.IsReversible}\ (\mathsf{Kernel.id} : \mathsf{Kernel}\ \alpha\ \alpha)\ \pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\pi : \mathsf{Measure}\ \alpha)$
$\mathsf{Kernel.Invariant}\ (\mathsf{Kernel.id} : \mathsf{Kernel}\ \alpha\ \alpha)\ \pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\kappa : \mathsf{Kernel}\ \alpha\ \alpha)\ (\pi : \mathsf{Measure}\ \alpha)\ (\mathsf{hinv} : \mathsf{Kernel.Invariant}\ \kappa\ \pi)$
$\pi.\mathsf{bind}\ \kappa = \pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\kappa\ \eta : \mathsf{Kernel}\ \alpha\ \alpha)\ (\pi : \mathsf{Measure}\ \alpha)\ (h\kappa : \mathsf{Kernel.Invariant}\ \kappa\ \pi)$
$(h\eta : \mathsf{Kernel.Invariant}\ \eta\ \pi)$
$\mathsf{Kernel.Invariant}\ (\kappa \circ_k\ \eta)\ \pi$

13.11 fep-011 — Surprise and Self-Information

Negative-log self-information is nonnegative on positive probabilities at most one, additive on positive products, strictly antitone, and zero exactly at certainty. Assumption scope: Positivity excludes the logarithmic boundary, and the probability interpretation additionally uses p≤1. Additivity is scalar and corresponds to product probabilities; no independence relation is inferred by the theorem. Semantic disposition: `formalized`.

13.11.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Log.Basic

namespace FEP011

open Real

/-- Shannon self-information (surprise) of a positive scalar probability. -/
noncomputable def fep011_surprise (probability : ℝ) : ℝ :=
  -Real.log probability

/-- Self-information: log is nonnegative on ``[1, ∞)``. -/
theorem fep011_log_nonneg_on_ge_one {x : ℝ} (hx : 1 ≤ x) : 0 ≤ Real.log x :=
  Real.log_nonneg hx

/-- Coding cost ``-log u`` is nonnegative for probabilities ``u ∈ (0, 1]``. -/
theorem fep011_surprise_nonneg {u : ℝ} (hu : 0 < u) (hu1 : u ≤ 1) :
    0 ≤ fep011_surprise u :=
  neg_nonneg.mpr (Real.log_nonpos (le_of_lt hu) hu1)

/-- Surprise is additive for independent observations: -log(ab) = -log a + -log b. -/
theorem fep011_surprise_additive {a b : ℝ} (ha : 0 < a) (hb : 0 < b) :
    fep011_surprise (a * b) = fep011_surprise a + fep011_surprise b := by
  simp only [fep011_surprise]
  rw [Real.log_mul (ne_of_gt ha) (ne_of_gt hb), neg_add]

/-- Surprise of certain event (u = 1) is zero. -/
theorem fep011_surprise_cert : fep011_surprise 1 = 0 := by
  simp [fep011_surprise]

/-- Surprise is monotone decreasing in probability: larger p → lower surprise. -/
theorem fep011_surprise_mono {p q : ℝ} (hp : 0 < p) (h : p ≤ q) :
    fep011_surprise q ≤ fep011_surprise p :=
  neg_le_neg (Real.log_le_log hp h)
```

```
/-- On positive probabilities, only a certain event has zero surprise. -/
theorem fep011_surprise_eq_zero_iff {p : ℝ} (hp : 0 < p) :
    fep011_surprise p = 0 ↔ p = 1 := by
  constructor
  · intro hzero
    exact Real.eq_one_of_pos_of_log_eq_zero hp (neg_eq_zero.mp hzero)
  · rintro rfl
    exact fep011_surprise_cert

/-- Surprise is strictly antitone on positive probabilities. -/
theorem fep011_surprise_strictAnti {p q : ℝ} (hp : 0 < p) (hpq : p < q) :
    fep011_surprise q < fep011_surprise p := by
  exact neg_lt_neg (Real.log_lt_log hp hpq)

end FEP011
```

13.11.2 Typeset statement signatures

$$\{x : \mathbb{R}\}\,(\mathsf{hx} : 1 \le x)$$
$$0 \le \mathsf{Real.log}\,x$$

$$\{u : \mathbb{R}\}\,(\mathsf{hu} : 0 < u)\,(\mathsf{hu1} : u \le 1)$$
$$0 \le \mathsf{fep011\_surprise}\,u$$

$$\{a\,b : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)\,(\mathsf{hb} : 0 < b)$$
$$\mathsf{fep011\_surprise}\,(a \cdot b) = \mathsf{fep011\_surprise}\,a + \mathsf{fep011\_surprise}\,b$$

$$\mathsf{fep011\_surprise}\,1 = 0$$

$$\{p\,q : \mathbb{R}\}\,(\mathsf{hp} : 0 < p)\,(h : p \le q)$$
$$\mathsf{fep011\_surprise}\,q \le \mathsf{fep011\_surprise}\,p$$

$$\{p : \mathbb{R}\}\,(\mathsf{hp} : 0 < p)$$
$$\mathsf{fep011\_surprise}\,p = 0 \leftrightarrow p = 1$$

$$\{p\,q : \mathbb{R}\}\,(\mathsf{hp} : 0 < p)\,(\mathsf{hpq} : p < q)$$
$$\mathsf{fep011\_surprise}\,q < \mathsf{fep011\_surprise}\,p$$

13.12 fep-012 — Finite Policy Shannon Entropy Regularization

Finite Shannon policy entropy is nonnegative on unit-interval weights, vanishes for point policies, and at nonnegative temperature lowers expected cost by exactly its entropy reward. Assumption scope: Entropy nonnegativity only requires each weight in [0,1]; normalization is supplied separately by a policy law such as fep-028 softmax. The theorem compares objectives and does not claim that softmax is the unique optimizer. Semantic disposition: `formalized`.

13.12.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Exp
import Mathlib.Analysis.SpecialFunctions.Log.Basic
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.Algebra.Order.BigOperators.Group.Finset

namespace FEP012

open Real Finset

-- [proof strategy: Real.exp_nonneg / exp_pos combined with Finset.sum_nonneg /
 ↪  sum_pos]
```

```
abbrev Policy := Fin 10

/-- Shannon entropy of a finite policy-weight vector.  The probability-simplex
assumptions are stated by the theorems that use it. -/
noncomputable def fep012_policyEntropy (q : Policy → ℝ) : ℝ :=
  Σ p, q p * (-Real.log (q p))

/-- Expected policy cost under finite weights. -/
def fep012_expectedCost (q cost : Policy → ℝ) : ℝ :=
  Σ p, q p * cost p

/-- Entropy-regularized policy objective, with temperature `τ`. -/
noncomputable def fep012_entropyRegularizedCost
    (q cost : Policy → ℝ) (τ : ℝ) : ℝ :=
  fep012_expectedCost q cost - τ * fep012_policyEntropy q

/-- Policy entropy is nonnegative for weights in the unit interval. -/
theorem fep012_policyEntropy_nonneg (q : Policy → ℝ)
    (hq : ∀ p, q p ∈ Set.Icc (0 : ℝ) 1) :
    0 ≤ fep012_policyEntropy q := by
  exact Finset.sum_nonneg fun p _ =>
    mul_nonneg (hq p).1
      (neg_nonneg.mpr (Real.log_nonpos (hq p).1 (hq p).2))

/-- A deterministic point policy has zero Shannon entropy. -/
theorem fep012_policyEntropy_pointMass (chosen : Policy) :
    fep012_policyEntropy (fun p => if p = chosen then 1 else 0) = 0 := by
  classical
  simp [fep012_policyEntropy]

/-- Nonnegative entropy temperature can only lower the regularized objective
relative to the unregularized expected cost. -/
theorem fep012_entropyRegularizedCost_le_expectedCost
    (q cost : Policy → ℝ) (τ : ℝ)
    (hτ : 0 ≤ τ) (hq : ∀ p, q p ∈ Set.Icc (0 : ℝ) 1) :
    fep012_entropyRegularizedCost q cost τ ≤ fep012_expectedCost q cost := by
  exact sub_le_self _ (mul_nonneg hτ (fep012_policyEntropy_nonneg q hq))

/-- At zero temperature entropy regularization vanishes exactly. -/
theorem fep012_entropyRegularizedCost_zeroTemperature
    (q cost : Policy → ℝ) :
    fep012_entropyRegularizedCost q cost 0 = fep012_expectedCost q cost := by
  simp [fep012_entropyRegularizedCost]

/-- Gibbs partition sum ``Σ exp(-G)`` is nonnegative (Boltzmann weights). -/
theorem fep012_gibbsPartition_nonneg (G : Policy → ℝ) (s : Finset Policy) :
    0 ≤ Σ p ∈ s, Real.exp (-G p) :=
  Finset.sum_nonneg fun _ _ => Real.exp_nonneg _

/-- Gibbs partition sum is strictly positive when policy set is nonempty. -/
theorem fep012_gibbsPartition_pos (G : Policy → ℝ) (s : Finset Policy) (hne :
 ↪ s.Nonempty) :
    0 < Σ p ∈ s, Real.exp (-G p) :=
  Finset.sum_pos (fun _ _ => Real.exp_pos _) hne

/-- Monotonicity: lower cost → larger Boltzmann weight at temperature β = 1. -/
theorem fep012_gibbs_mono (G₁ G₂ : Policy) (G : Policy → ℝ) (h : G G₁ ≤ G G₂) :
    Real.exp (-G G₂) ≤ Real.exp (-G G₁) :=
  Real.exp_le_exp.mpr (by linarith)

end FEP012
```

13.12.2 Typeset statement signatures

$$(q : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{hq} : \forall p, q\,p \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$0 \le \mathsf{fep012\_policyEntropy}\ q$$

$$(\mathsf{chosen} : \mathsf{Policy})$$
$$\mathsf{fep012\_policyEntropy}\ (\lambda\, p \mapsto \mathsf{if}\, p = \mathsf{chosen}\ \mathsf{then}\ 1\ \mathsf{else}\ 0) = 0$$

$$(q\,\mathsf{cost} : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\tau : \mathbb{R})\ (h\tau : 0 \le \tau)\ (\mathsf{hq} : \forall p, q\,p \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$\mathsf{fep012\_entropyRegularizedCost}\ q\,\mathsf{cost}\,\tau \le \mathsf{fep012\_expectedCost}\ q\,\mathsf{cost}$$

$$(q\,\mathsf{cost} : \mathsf{Policy} \Rightarrow \mathbb{R})$$
$$\mathsf{fep012\_entropyRegularizedCost}\ q\,\mathsf{cost}\,0 = \mathsf{fep012\_expectedCost}\ q\,\mathsf{cost}$$

$$(G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (s : \mathsf{Finset\ Policy})$$
$$0 \le \sum p \in s, \mathsf{Real.exp}\,(-G\,p)$$

$$(G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (s : \mathsf{Finset\ Policy})\ (\mathsf{hne} : \mathsf{s.Nonempty})$$
$$0 < \sum p \in s, \mathsf{Real.exp}\,(-G\,p)$$

$$(G_1\,G_2 : \mathsf{Policy})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (h : G\,G_1 \le G\,G_2)$$
$$\mathsf{Real.exp}\,(-G\,G_2) \le \mathsf{Real.exp}\,(-G\,G_1)$$

13.13 fep-013 — Equilibrium Helmholtz Differential Identities

For differentiable U(T) and S(T), Helmholtz free energy has derivative U'−S−TS'; under the equilibrium first-law identity U'=TS', this reduces exactly to dF/dT=−S. Assumption scope: The derivative law is conditional on pointwise differentiability and the explicitly stated equilibrium identity. It is a thermodynamic calculus identity, not a derivation of the first law or an identification with variational free energy. Semantic disposition: `formalized`.

13.13.1 Lean sketch

```
import Mathlib.Analysis.Calculus.Deriv.Add
import Mathlib.Analysis.Calculus.Deriv.Mul
import Mathlib.Tactic

namespace FEP013

-- [proof strategy: simp / ring for algebraic rewrites; linarith for entropy-monotone
 ↪  bounds]

/-- Helmholtz free energy: F = U - TS defines the thermodynamic potential. -/
noncomputable def fep013_helmholtz (U T S : ℝ) : ℝ := U - T * S

/-- At zero temperature, Helmholtz free energy equals internal energy. -/
theorem fep013_helmholtz_at_zero_temp (U S : ℝ) : fep013_helmholtz U 0 S = U := by
  simp [fep013_helmholtz]

/-- Helmholtz free energy is monotone decreasing in entropy at positive temperature. -/
theorem fep013_helmholtz_mono_entropy (U T : ℝ) (S₁ S₂ : ℝ) (hT : 0 < T) (h : S₁ ≤ S₂) :
    fep013_helmholtz U T S₂ ≤ fep013_helmholtz U T S₁ := by
  simp only [fep013_helmholtz]
  linarith [mul_le_mul_of_nonneg_left h (le_of_lt hT)]

/-- Helmholtz free energy is monotone increasing in internal energy. -/
theorem fep013_helmholtz_mono_U (U₁ U₂ T S : ℝ) (h : U₁ ≤ U₂) :
    fep013_helmholtz U₁ T S ≤ fep013_helmholtz U₂ T S := by
  simp only [fep013_helmholtz]; linarith
```

```
/-- Helmholtz difference: ΔF = ΔU - TΔS. -/
theorem fep013_delta_F (U₁ U₂ T S₁ S₂ : ℝ) :
    fep013_helmholtz U₂ T S₂ - fep013_helmholtz U₁ T S₁ = (U₂ - U₁) - T * (S₂ - S₁) :=
 ↪ by
  simp only [fep013_helmholtz]; ring

/-- Exact temperature derivative of `F(T) = U(T) - T S(T)`. -/
theorem fep013_helmholtz_hasDerivAt
    {U S : ℝ → ℝ} {U' S' T : ℝ}
    (hU : HasDerivAt U U' T) (hS : HasDerivAt S S' T) :
    HasDerivAt (fun t => fep013_helmholtz (U t) t (S t))
      (U' - (S T + T * S')) T := by
  convert HasDerivAt.sub hU (HasDerivAt.mul (hasDerivAt_id T) hS) using 1
  all_goals
    first
    | exact AddCommGroup.ext rfl
    | exact Module.ext rfl
    | rfl
    | simp

/-- Under the equilibrium first-law identity `U'(T) = T S'(T)`, the
Helmholtz derivative is exactly minus entropy. -/
theorem fep013_helmholtz_derivative_eq_neg_entropy
    {U S : ℝ → ℝ} {U' S' T : ℝ}
    (hU : HasDerivAt U U' T) (hS : HasDerivAt S S' T)
    (hfirstLaw : U' = T * S') :
    HasDerivAt (fun t => fep013_helmholtz (U t) t (S t)) (-S T) T := by
  convert fep013_helmholtz_hasDerivAt hU hS using 1
  rw [hfirstLaw]
  ring

end FEP013
```

13.13.2 Typeset statement signatures

$$(U\ S : \mathbb{R})$$
$$\mathsf{fep013\_helmholtz}\ U\ 0\ S = U$$

$$(U\ T : \mathbb{R})\ (S_1\ S_2 : \mathbb{R})\ (\mathsf{hT} : 0 < T)\ (h : S_1 \le S_2)$$
$$\mathsf{fep013\_helmholtz}\ U\ T\ S_2 \le \mathsf{fep013\_helmholtz}\ U\ T\ S_1$$

$$(U_1\ U_2\ T\ S : \mathbb{R})\ (h : U_1 \le U_2)$$
$$\mathsf{fep013\_helmholtz}\ U_1\ T\ S \le \mathsf{fep013\_helmholtz}\ U_2\ T\ S$$

$$(U_1\ U_2\ T\ S_1\ S_2 : \mathbb{R})$$
$$\mathsf{fep013\_helmholtz}\ U_2\ T\ S_2 - \mathsf{fep013\_helmholtz}\ U_1\ T\ S_1 = (U_2 - U_1) - T \cdot (S_2 - S_1)$$

$$\{U\ S : \mathbb{R} \Rightarrow \mathbb{R}\}\ \{U'\ S'\ T : \mathbb{R}\}\ (\mathsf{hU} : \mathsf{HasDerivAt}\ U\ U'\ T)\ (\mathsf{hS} : \mathsf{HasDerivAt}\ S\ S'\ T)$$
$$\mathsf{HasDerivAt}\ (\lambda\, t \mapsto \mathsf{fep013\_helmholtz}\ (U\ t)\ t\ (S\ t))\ (U' - (S\ T + T \cdot S'))\ T$$

$$\{U\ S : \mathbb{R} \Rightarrow \mathbb{R}\}\ \{U'\ S'\ T : \mathbb{R}\}\ (\mathsf{hU} : \mathsf{HasDerivAt}\ U\ U'\ T)\ (\mathsf{hS} : \mathsf{HasDerivAt}\ S\ S'\ T)$$
$$(\mathsf{hfirstLaw} : U' = T \cdot S')$$
$$\mathsf{HasDerivAt}\ (\lambda\, t \mapsto \mathsf{fep013\_helmholtz}\ (U\ t)\ t\ (S\ t))\ (-S\ T)\ T$$

13.14 fep-014 — KL Divergence: Non-Negativity, Identity, and Chain Rule

Mathlib KL divergence is nonnegative, vanishes exactly on equal finite measures, and obeys the composition-product chain rule for Markov kernels. Assumption scope: Self-divergence uses SigmaFinite; zero-characterization and the chain rule use finite measures; the chain rule additionally requires Markov kernels. No data-processing theorem is claimed because the pinned Mathlib revision does not expose one. Semantic disposition: `formalized`.

13.14.1 Lean sketch

```
import Mathlib.InformationTheory.KullbackLeibler.ChainRule

namespace FEP014

variable {α : Type*} [MeasurableSpace α]

open MeasureTheory ProbabilityTheory
open scoped ENNReal

/-- KL divergence is nonnegative in its native extended-real codomain. -/
theorem fep014_kl_nonneg (μ ν : Measure α) :
    0 ≤ InformationTheory.klDiv μ ν :=
  bot_le

/-- KL divergence of a sigma-finite measure from itself is zero. -/
theorem fep014_kl_self (μ : Measure α) [SigmaFinite μ] :
    InformationTheory.klDiv μ μ = 0 :=
  InformationTheory.klDiv_self μ

/-- For finite measures, zero KL divergence characterizes equality. -/
theorem fep014_kl_eq_zero_iff (μ ν : Measure α)
    [IsFiniteMeasure μ] [IsFiniteMeasure ν] :
    InformationTheory.klDiv μ ν = 0 ↔ μ = ν :=
  InformationTheory.klDiv_eq_zero_iff

variable {β : Type*} [MeasurableSpace β]

/-- Mathlib's composition-product chain rule for KL divergence. -/
theorem fep014_kl_chain_rule (μ ν : Measure α) (κ η : Kernel α β)
    [IsFiniteMeasure μ] [IsFiniteMeasure ν]
    [IsMarkovKernel κ] [IsMarkovKernel η] :
    InformationTheory.klDiv (μ ⊗ₘ κ) (ν ⊗ₘ η) =
      InformationTheory.klDiv μ ν +
        InformationTheory.klDiv (μ ⊗ₘ κ) (μ ⊗ₘ η) :=
  InformationTheory.klDiv_compProd_eq_add μ ν κ η

end FEP014
```

13.14.2 Typeset statement signatures

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu\,\nu : \mathsf{Measure}\ \alpha)$$
$$0 \le \mathsf{InformationTheory.klDiv}\ \mu\,\nu$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu : \mathsf{Measure}\ \alpha)\,[\mathsf{SigmaFinite}\ \mu]$$
$$\mathsf{InformationTheory.klDiv}\ \mu\,\mu = 0$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu\,\nu : \mathsf{Measure}\ \alpha)\,[\mathsf{IsFiniteMeasure}\ \mu]\,[\mathsf{IsFiniteMeasure}\ \nu]$$
$$\mathsf{InformationTheory.klDiv}\ \mu\,\nu = 0 \leftrightarrow \mu = \nu$$

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu\,\nu : \mathsf{Measure}\ \alpha)\,(\kappa\,\eta : \mathsf{Kernel}\ \alpha\,\beta)\,[\mathsf{IsFiniteMeasure}\ \mu]\,[\mathsf{IsFiniteMeasure}\ \nu]$$
$$[\mathsf{IsMarkovKernel}\ \kappa]\,[\mathsf{IsMarkovKernel}\ \eta]$$
$$\mathsf{InformationTheory.klDiv}\ (\mu \otimes_m\ \kappa)\,(\nu \otimes_m\ \eta) = \mathsf{InformationTheory.klDiv}\ \mu\,\nu$$
$$+\ \mathsf{InformationTheory.klDiv}\ (\mu \otimes_m\ \kappa)\,(\mu \otimes_m\ \eta)$$

13.15 fep-015 — Measurable Variational Integrand

A pointwise variational integrand built from measurable energy, approximate log density, and generative log density is measurable and remains so under measurable reparameterization and baseline shifts. Assumption scope: All three real-valued ingredients and any reparameterization are assumed measurable. Measurability does not imply integrability, finite expectation, or existence of an optimizer, none of which is claimed. Semantic disposition: `formalized`.

13.15.1 Lean sketch

```
import Mathlib.MeasureTheory.Constructions.BorelSpace.Basic

namespace FEP015

variable {α : Type*} [MeasurableSpace α]

/-- Pointwise variational-free-energy integrand: energy plus approximate
log-density minus generative log-density. -/
def fep015_variationalIntegrand
    (energy logApproximation logGenerative : α → ℝ) (state : α) : ℝ :=
  energy state + logApproximation state - logGenerative state

/-- Measurability of all three scientific ingredients is sufficient for
measurability of the complete variational integrand. -/
theorem fep015_variationalIntegrand_measurable
    {energy logApproximation logGenerative : α → ℝ}
    (henergy : Measurable energy)
    (happroximation : Measurable logApproximation)
    (hgenerative : Measurable logGenerative) :
    Measurable
      (fep015_variationalIntegrand energy logApproximation logGenerative) := by
  exact (henergy.add happroximation).sub hgenerative

/-- Reparameterizing a measurable variational integrand by a measurable state
map preserves measurability. -/
theorem fep015_variationalIntegrand_comp_measurable
    {β : Type*} [MeasurableSpace β]
    {energy logApproximation logGenerative : α → ℝ}
    {reparameterize : β → α}
    (henergy : Measurable energy)
    (happroximation : Measurable logApproximation)
    (hgenerative : Measurable logGenerative)
    (hreparameterize : Measurable reparameterize) :
    Measurable
      (fep015_variationalIntegrand energy logApproximation logGenerative ∘
        reparameterize) :=
  (fep015_variationalIntegrand_measurable
    henergy happroximation hgenerative).comp hreparameterize

/-- Adding a measurable scalar baseline preserves variational-integrand
measurability. -/
theorem fep015_variationalIntegrand_add_const_measurable
    {energy logApproximation logGenerative : α → ℝ}
    (henergy : Measurable energy)
    (happroximation : Measurable logApproximation)
    (hgenerative : Measurable logGenerative) (baseline : ℝ) :
    Measurable (fun state =>
      fep015_variationalIntegrand energy logApproximation logGenerative state +
        baseline) :=
  (fep015_variationalIntegrand_measurable
    henergy happroximation hgenerative).add measurable_const

end FEP015
```

13.15.2 Typeset statement signatures

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$\{\mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative} : \alpha \Rightarrow \mathbb{R}\}\ (\mathsf{henergy} : \mathsf{Measurable}\ \mathsf{energy})$
$(\mathsf{happroximation} : \mathsf{Measurable}\ \mathsf{logApproximation})$
$(\mathsf{hgenerative} : \mathsf{Measurable}\ \mathsf{logGenerative})$
$\mathsf{Measurable}\ (\mathsf{fep015\_variationalIntegrand}\ \mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative})$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$\{\beta : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\beta]\ \{\mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative} : \alpha \Rightarrow \mathbb{R}\}$
$\{\mathsf{reparameterize} : \beta \Rightarrow \alpha\}\ (\mathsf{henergy} : \mathsf{Measurable}\ \mathsf{energy})$
$(\mathsf{happroximation} : \mathsf{Measurable}\ \mathsf{logApproximation})$
$(\mathsf{hgenerative} : \mathsf{Measurable}\ \mathsf{logGenerative})$
$(\mathsf{hreparameterize} : \mathsf{Measurable}\ \mathsf{reparameterize})$
$\mathsf{Measurable}\ (\mathsf{fep015\_variationalIntegrand}\ \mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative} \circ$
$\mathsf{reparameterize})$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$\{\mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative} : \alpha \Rightarrow \mathbb{R}\}\ (\mathsf{henergy} : \mathsf{Measurable}\ \mathsf{energy})$
$(\mathsf{happroximation} : \mathsf{Measurable}\ \mathsf{logApproximation})$
$(\mathsf{hgenerative} : \mathsf{Measurable}\ \mathsf{logGenerative})\ (\mathsf{baseline} : \mathbb{R})$
$\mathsf{Measurable}\ (\lambda\ \mathsf{state} \mapsto \mathsf{fep015\_variationalIntegrand}\ \mathsf{energy}\ \mathsf{logApproximation}$
$\mathsf{logGenerative}\ \mathsf{state} + \mathsf{baseline})$

13.16 fep-016 — Exact Scalar Quadratic Laplace Kernel

A positive-precision scalar quadratic exponential is integrable and has the exact Gaussian normalizing integral sqrt(π/(precision/2)). Assumption scope: Precision is strictly positive for the scientific integrability result. This is the exact scalar quadratic case underlying Laplace approximation; no approximation-error or higher-dimensional determinant theorem is claimed. Semantic disposition: `formalized`.

13.16.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Gaussian.GaussianIntegral

namespace FEP016

/-- Laplace approximation: quadratic forms have a global minimum at zero. -/
theorem fep016_quadratic_min (x : ℝ) : 0 ≤ x ^ 2 :=
  sq_nonneg x

/-- The quadratic (x - μ)² achieves its minimum value of 0 at x = μ. -/
theorem fep016_quadratic_at_mode (μ : ℝ) : (μ - μ) ^ 2 = 0 := by ring

/-- Precision-weighted quadratic: prec * (x - μ)² is nonneg when prec ≥ 0. -/
theorem fep016_precision_weighted (prec x μ : ℝ) (hp : 0 ≤ prec) :
    0 ≤ prec * (x - μ) ^ 2 :=
  mul_nonneg hp (sq_nonneg _)

/-- Symmetry of quadratic loss: (x - μ)² = (μ - x)². -/
theorem fep016_quadratic_sym (x μ : ℝ) : (x - μ) ^ 2 = (μ - x) ^ 2 := by ring

/-- Expansion: (x - μ)² = x² - 2μx + μ² (second-order Taylor anchor). -/
theorem fep016_quadratic_expand (x μ : ℝ) :
    (x - μ) ^ 2 = x ^ 2 - 2 * μ * x + μ ^ 2 := by ring

/-- The centered exponential kernel produced by a scalar quadratic Laplace
model with precision `precision`. -/
noncomputable def fep016_laplaceKernel (precision x : ℝ) : ℝ :=
```

```
  Real.exp (-(precision / 2) * x ^ 2)

/-- Positive precision makes the quadratic Laplace kernel integrable. -/
theorem fep016_laplaceKernel_integrable
    {precision : ℝ} (hprecision : 0 < precision) :
    MeasureTheory.Integrable (fep016_laplaceKernel precision) := by
  exact integrable_exp_neg_mul_sq (by positivity : 0 < precision / 2)

/-- Mathlib's Gaussian integral gives the exact scalar Laplace normalizer. -/
theorem fep016_laplaceKernel_integral
    {precision : ℝ} (hprecision : 0 < precision) :
    (∫ x : ℝ, fep016_laplaceKernel precision x) =
      Real.sqrt (Real.pi / (precision / 2)) := by
  have _hintegrable :
      MeasureTheory.Integrable (fep016_laplaceKernel precision) :=
    fep016_laplaceKernel_integrable hprecision
  simpa [fep016_laplaceKernel] using
    integral_gaussian (precision / 2)

end FEP016
```

13.16.2 Typeset statement signatures

$$(x : \mathbb{R})$$
$$0 \le x^2$$

$$(\mu : \mathbb{R})$$
$$(\mu - \mu)^2 = 0$$

$$(\mathsf{prec}\; x\; \mu : \mathbb{R})\; (\mathsf{hp} : 0 \le \mathsf{prec})$$
$$0 \le \mathsf{prec} \cdot (x - \mu)^2$$

$$(x\; \mu : \mathbb{R})$$
$$(x - \mu)^2 = (\mu - x)^2$$

$$(x\; \mu : \mathbb{R})$$
$$(x - \mu)^2 = x^2 - 2 \cdot \mu \cdot x + \mu^2$$

$$\{\mathsf{precision} : \mathbb{R}\}\; (\mathsf{hprecision} : 0 < \mathsf{precision})$$
$$\mathsf{MeasureTheory.Integrable}\; (\mathsf{fep016\_laplaceKernel}\; \mathsf{precision})$$

$$\{\mathsf{precision} : \mathbb{R}\}\; (\mathsf{hprecision} : 0 < \mathsf{precision})$$
$$(\int x : \mathbb{R}, \mathsf{fep016\_laplaceKernel}\; \mathsf{precision}\; x) = \mathsf{Real.sqrt}$$
$$(\mathsf{Real.pi}/(\mathsf{precision}/2))$$

13.17 fep-017 — Posterior Kernel Reconstruction

Mathlib's posterior is a Markov kernel whose composition with the predictive law reconstructs the swapped prior-likelihood joint law; for a Markov likelihood it also recovers the prior. Assumption scope: Conditional on standard-Borel and nonempty latent structure plus finite prior and likelihood kernel. The Radon–Nikodym Bayes formula additionally uses countability of the latent space and standard-Borel observation structure. This is a regular posterior-kernel theorem, not a claim about learned or approximate posteriors. Semantic disposition: `formalized`.

13.17.1 Lean sketch

```
import Mathlib.Probability.Kernel.Posterior

namespace FEP017

open MeasureTheory ProbabilityTheory
open scoped ENNReal ProbabilityTheory

variable {Ω 𝒳 : Type*} [MeasurableSpace Ω] [MeasurableSpace 𝒳]

/-- The native posterior kernel of a likelihood kernel with respect to a prior. -/
noncomputable def fep017_posterior
    (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel κ] :
    Kernel 𝒳 Ω :=
  ProbabilityTheory.posterior κ μ

/-- Every posterior fibre is a probability measure. -/
theorem fep017_posterior_mass_one
    (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel κ]
    (x : 𝒳) :
    fep017_posterior κ μ x Set.univ = 1 := by
  change (ProbabilityTheory.posterior κ μ x) Set.univ = 1
  exact measure_univ

/-- Posterior reconstruction: predictive mass followed by the posterior
recovers the swapped prior-likelihood joint law. -/
theorem fep017_posterior_joint_reconstruction
    (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel κ] :
    (κ ∘ₘ μ) ⊗ₘ fep017_posterior κ μ = (μ ⊗ₘ κ).map Prod.swap := by
  exact ProbabilityTheory.compProd_posterior_eq_map_swap

/-- Applying a Markov likelihood and then its posterior recovers the prior. -/
theorem fep017_posterior_recovers_prior
    (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsMarkovKernel κ] :
    fep017_posterior κ μ ∘ₘ κ ∘ₘ μ = μ := by
  exact ProbabilityTheory.posterior_comp_self

/-- On a countable latent space, the posterior is the prior weighted by the
Radon--Nikodym likelihood ratio, almost everywhere under the predictive law. -/
theorem fep017_posterior_bayes_density
    [Countable Ω] [StandardBorelSpace Ω] [Nonempty Ω]
    [StandardBorelSpace 𝒳] [Nonempty 𝒳]
    (κ : Kernel Ω 𝒳) (μ : Measure Ω) [IsFiniteMeasure μ] [IsFiniteKernel κ] :
    ∀ᵐ x ∂(κ ∘ₘ μ),
      fep017_posterior κ μ x =
        μ.withDensity (fun ω ↦ (κ ω).rnDeriv (κ ∘ₘ μ) x) := by
  exact ProbabilityTheory.posterior_eq_withDensity_of_countable κ μ

end FEP017
```

13.17.2 Typeset statement signatures

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\,\Omega]\ [\mathsf{Nonempty}\,\Omega]$$
$$[\mathsf{IsFiniteMeasure}\,\mu]\ [\mathsf{IsFiniteKernel}\,\kappa]\ (x : \mathcal{X})$$
$$\mathsf{fep017\_posterior}\,\kappa\,\mu\,x\,\Omega = 1$$

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\,\Omega]\ [\mathsf{Nonempty}\,\Omega]$$
$$[\mathsf{IsFiniteMeasure}\,\mu]\ [\mathsf{IsFiniteKernel}\,\kappa]$$
$$(\kappa \circ_m\ \mu) \otimes_m\ \mathsf{fep017\_posterior}\,\kappa\,\mu = (\mu \otimes_m\ \kappa).\mathsf{map}\ \mathsf{Prod.swap}$$

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\,\Omega]\ [\mathsf{Nonempty}\,\Omega]$$
$$[\mathsf{IsFiniteMeasure}\,\mu]\ [\mathsf{IsMarkovKernel}\,\kappa]$$
$$\mathsf{fep017\_posterior}\,\kappa\,\mu \circ_m\ \kappa \circ_m\ \mu = \mu$$

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$[\mathsf{Countable}\,\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ [\mathsf{Nonempty}\,\Omega]\ [\mathsf{StandardBorelSpace}\,\mathcal{X}]$$
$$[\mathsf{Nonempty}\,\mathcal{X}]\ (\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{IsFiniteMeasure}\,\mu]$$
$$[\mathsf{IsFiniteKernel}\,\kappa]$$
$$\forall^{\text{a.e.}}\, x\, \mathrm{d}\,(\kappa \circ_m\ \mu), \mathsf{fep017\_posterior}\,\kappa\,\mu\,x = \mu.\mathsf{withDensity}$$
$$(\lambda\,\omega \mapsto (\kappa\,\omega).\mathsf{rnDeriv}\,(\kappa \circ_m\ \mu)\,x)$$

13.18 fep-018 — Bernoulli Fisher–Rao Coordinate Distance

The Bernoulli Fisher–Rao coordinate distance is nonnegative, symmetric, separating on the probability interval, and satisfies the triangle inequality. Assumption scope: Separation is restricted to Bernoulli parameters in [0,1], where square root and arcsine are injective. This is the closed-form one-dimensional Fisher–Rao distance, not a general geodesic-existence theorem for arbitrary statistical manifolds. Semantic disposition: `formalized`.

13.18.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Trigonometric.Inverse
import Mathlib.Analysis.Real.Sqrt
import Mathlib.Tactic

namespace FEP018

/-- Variance-stabilizing Fisher--Rao coordinate for the Bernoulli family. -/
noncomputable def fep018_fisherCoordinate (p : ℝ) : ℝ :=
  2 * Real.arcsin (Real.sqrt p)

/-- Closed-form coordinate distance on the Bernoulli probability interval. -/
noncomputable def fep018_fisherRaoDistance (p q : ℝ) : ℝ :=
  |fep018_fisherCoordinate p - fep018_fisherCoordinate q|

/-- Fisher--Rao coordinate distance is nonnegative. -/
theorem fep018_fisherRaoDistance_nonneg (p q : ℝ) :
    0 ≤ fep018_fisherRaoDistance p q :=
  abs_nonneg _

/-- Fisher--Rao coordinate distance is symmetric. -/
theorem fep018_fisherRaoDistance_symm (p q : ℝ) :
    fep018_fisherRaoDistance p q = fep018_fisherRaoDistance q p := by
  exact abs_sub_comm _ _

/-- The coordinate distance satisfies the triangle inequality. -/
```

```
theorem fep018_fisherRaoDistance_triangle (p q r : ℝ) :
    fep018_fisherRaoDistance p r ≤
      fep018_fisherRaoDistance p q + fep018_fisherRaoDistance q r := by
  calc
    |fep018_fisherCoordinate p - fep018_fisherCoordinate r| =
        |(fep018_fisherCoordinate p - fep018_fisherCoordinate q) +
          (fep018_fisherCoordinate q - fep018_fisherCoordinate r)| := by
            congr 1
            ring
    _ ≤ |fep018_fisherCoordinate p - fep018_fisherCoordinate q| +
          |fep018_fisherCoordinate q - fep018_fisherCoordinate r| :=
      abs_add_le _ _

/-- On the probability interval, zero Fisher--Rao distance characterizes
equality of Bernoulli parameters. -/
theorem fep018_fisherRaoDistance_eq_zero_iff
    {p q : ℝ} (hp : p ∈ Set.Icc (0 : ℝ) 1) (hq : q ∈ Set.Icc (0 : ℝ) 1) :
    fep018_fisherRaoDistance p q = 0 ↔ p = q := by
  constructor
  · intro h
    have hcoord : fep018_fisherCoordinate p = fep018_fisherCoordinate q :=
      sub_eq_zero.mp (abs_eq_zero.mp h)
    have harcsin : Real.arcsin (Real.sqrt p) = Real.arcsin (Real.sqrt q) := by
      simp only [fep018_fisherCoordinate] at hcoord
      linarith
    have hsqrt : Real.sqrt p = Real.sqrt q :=
      (Real.arcsin_inj
        (by linarith [Real.sqrt_nonneg p]) (Real.sqrt_le_one.mpr hp.2)
        (by linarith [Real.sqrt_nonneg q]) (Real.sqrt_le_one.mpr hq.2)).mp harcsin
    exact (Real.sqrt_inj hp.1 hq.1).mp hsqrt
  · rintro rfl
    simp [fep018_fisherRaoDistance]

end FEP018
```

13.18.2 Typeset statement signatures

$$(p\,q : \mathbb{R})$$
$$0 \le \mathsf{fep018\_fisherRaoDistance}\,p\,q$$

$$(p\,q : \mathbb{R})$$
$$\mathsf{fep018\_fisherRaoDistance}\,p\,q = \mathsf{fep018\_fisherRaoDistance}\,q\,p$$

$$(p\,q\,r : \mathbb{R})$$
$$\mathsf{fep018\_fisherRaoDistance}\,p\,r \le \mathsf{fep018\_fisherRaoDistance}\,p\,q + \mathsf{fep018\_fisherRaoDistance}\,q\,r$$

$$\{p\,q : \mathbb{R}\}\,(\mathsf{hp} : p \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)\,(\mathsf{hq} : q \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$\mathsf{fep018\_fisherRaoDistance}\,p\,q = 0 \leftrightarrow p = q$$

13.19 fep-019 — Prior-Predictive Kernel Composition

The native prior-predictive law preserves total mass for a Markov likelihood and sequential prediction equals measure composition by the composed kernels. Assumption scope: Mass preservation requires a Markov likelihood; probability mass one additionally requires a probability prior. The associativity law is structural measure-kernel composition and does not assert model calibration or identifiability. Semantic disposition: `formalized`.

13.19.1 Lean sketch

```
import Mathlib.Probability.Kernel.Composition.MeasureComp

namespace FEP019

open MeasureTheory ProbabilityTheory
open scoped ENNReal

variable {Ω 𝒳 𝒴 : Type*}
variable [MeasurableSpace Ω] [MeasurableSpace 𝒳] [MeasurableSpace 𝒴]

/-- The prior-predictive law is native measure-kernel composition. -/
noncomputable def fep019_priorPredictive
    (κ : Kernel Ω 𝒳) (μ : Measure Ω) : Measure 𝒳 :=
  κ ∘ₘ μ

/-- A Markov likelihood preserves total prior mass in the predictive law. -/
theorem fep019_priorPredictive_mass
    (κ : Kernel Ω 𝒳) (μ : Measure Ω) [IsMarkovKernel κ] :
    fep019_priorPredictive κ μ Set.univ = μ Set.univ := by
  exact Measure.comp_apply_univ

/-- A probability prior and Markov likelihood produce a probability predictive law. -/
theorem fep019_priorPredictive_mass_one
    (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [IsProbabilityMeasure μ] [IsMarkovKernel κ] :
    fep019_priorPredictive κ μ Set.univ = 1 := by
  change (κ ∘ₘ μ) Set.univ = 1
  exact measure_univ

/-- Sequential prediction agrees with composition of the two kernels. -/
theorem fep019_priorPredictive_assoc
    (κ : Kernel Ω 𝒳) (η : Kernel 𝒳 𝒴) (μ : Measure Ω) :
    η ∘ₘ fep019_priorPredictive κ μ =
      fep019_priorPredictive (η ∘ₖ κ) μ := by
  exact Measure.comp_assoc

end FEP019
```

13.19.2 Typeset statement signatures

$$\{\Omega\ \mathcal{X}\ \mathcal{Y} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\,\Omega]\ [\mathsf{MeasurableSpace}\,\mathcal{X}]\ [\mathsf{MeasurableSpace}\,\mathcal{Y}]$$
$$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\,\Omega)\ [\mathsf{IsMarkovKernel}\,\kappa]$$
$$\mathsf{fep019\_priorPredictive}\,\kappa\,\mu\,\Omega = \mu\,\Omega$$

$$\{\Omega\ \mathcal{X}\ \mathcal{Y} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\,\Omega]\ [\mathsf{MeasurableSpace}\,\mathcal{X}]\ [\mathsf{MeasurableSpace}\,\mathcal{Y}]$$
$$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\,\Omega)\ [\mathsf{IsProbabilityMeasure}\,\mu]\ [\mathsf{IsMarkovKernel}\,\kappa]$$
$$\mathsf{fep019\_priorPredictive}\,\kappa\,\mu\,\Omega = 1$$

$$\{\Omega\ \mathcal{X}\ \mathcal{Y} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\,\Omega]\ [\mathsf{MeasurableSpace}\,\mathcal{X}]\ [\mathsf{MeasurableSpace}\,\mathcal{Y}]$$
$$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\eta : \mathsf{Kernel}\,\mathcal{X}\,\mathcal{Y})\ (\mu : \mathsf{Measure}\,\Omega)$$
$$\eta \circ_m \mathsf{fep019\_priorPredictive}\,\kappa\,\mu = \mathsf{fep019\_priorPredictive}\,(\eta \circ_k \kappa)\,\mu$$

13.20 fep-020 — Two-State Markov Relaxation

A normalized symmetric two-state transition has the uniform stationary law; its exact n-step deviation is (1-2α)^n times the initial deviation and converges to zero for 0 < α < 1. Assumption scope: Transition nonnegativity uses α in [0,1], while convergence uses the strict interior so |1-2α| < 1. This is an exact finite-state Markov relaxation model, not a discretized Langevin diffusion. Semantic disposition: `formalized`.

13.20.1 Lean sketch

```
import Mathlib.Analysis.SpecificLimits.Normed
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.Data.Bool.Basic
import Mathlib.Tactic

namespace FEP020

/-- Symmetric two-state Markov transition with switching probability `α`. -/
def fep020_transition (α : ℝ) : Bool → Bool → ℝ
  | false, false => 1 - α
  | false, true => α
  | true, false => α
  | true, true => 1 - α

/-- The transition entries are nonnegative when `α` is a probability. -/
theorem fep020_transition_nonneg
    {α : ℝ} (hα0 : 0 ≤ α) (hα1 : α ≤ 1) (source target : Bool) :
    0 ≤ fep020_transition α source target := by
  cases source <;> cases target <;> simp [fep020_transition] <;> linarith

/-- Every transition row sums to one. -/
theorem fep020_transition_sum_one (α : ℝ) (source : Bool) :
    Σ target : Bool, fep020_transition α source target = 1 := by
  cases source <;> simp [fep020_transition]

/-- Evolution of the probability of state `true` under the transition law. -/
def fep020_evolve (α p : ℝ) : ℝ :=
  (1 - p) * fep020_transition α false true +
    p * fep020_transition α true true

/-- The two-state master equation is an affine relaxation map. -/
theorem fep020_evolve_affine (α p : ℝ) :
    fep020_evolve α p = α + (1 - 2 * α) * p := by
  simp [fep020_evolve, fep020_transition]
  ring

/-- The uniform law is stationary under the symmetric transition. -/
theorem fep020_uniform_stationary (α : ℝ) :
    fep020_evolve α (1 / 2 : ℝ) = 1 / 2 := by
  rw [fep020_evolve_affine]
  ring

/-- One transition scales deviation from equilibrium by `1 - 2α`. -/
theorem fep020_deviation_step (α p : ℝ) :
    fep020_evolve α p - 1 / 2 = (1 - 2 * α) * (p - 1 / 2) := by
  rw [fep020_evolve_affine]
  ring

/-- Exact deviation after `n` Markov steps. -/
theorem fep020_iterate_deviation (α p : ℝ) (n : ℕ) :
    ((fep020_evolve α)^[n]) p - 1 / 2 =
      (1 - 2 * α) ^ n * (p - 1 / 2) := by
  induction n with
  | zero => simp
  | succ n ih =>
      rw [Function.iterate_succ_apply']
      rw [fep020_deviation_step, ih, pow_succ]
      ring

/-- With a strictly interior switching probability, every initial mass
```

```
converges to the stationary uniform law. -/
theorem fep020_tendsto_uniform
    {α : ℝ} (hα0 : 0 < α) (hα1 : α < 1) (p : ℝ) :
    Filter.Tendsto (fun n => ((fep020_evolve α)^[n]) p)
      Filter.atTop (nhds (1 / 2 : ℝ)) := by
  have habs : |1 - 2 * α| < 1 := by
    rw [abs_lt]
    constructor <;> linarith
  have hpow : Filter.Tendsto (fun n : ℕ => (1 - 2 * α) ^ n)
      Filter.atTop (nhds 0) :=
    tendsto_pow_atTop_nhds_zero_of_abs_lt_one habs
  have hdev : Filter.Tendsto
      (fun n : ℕ => ((fep020_evolve α)^[n]) p - 1 / 2)
      Filter.atTop (nhds 0) := by
    simpa only [fep020_iterate_deviation, zero_mul] using
      hpow.mul_const (p - 1 / 2)
  simpa only [sub_add_cancel, zero_add] using hdev.add_const (1 / 2 : ℝ)

end FEP020
```

13.20.2 Typeset statement signatures

$$\{\alpha : \mathbb{R}\}\,(h\alpha 0 : 0 \le \alpha)\,(h\alpha 1 : \alpha \le 1)\,(\mathsf{source}\ \mathsf{target} : \mathsf{Bool})$$
$$0 \le \mathsf{fep020\_transition}\ \alpha\ \mathsf{source}\ \mathsf{target}$$

$$(\alpha : \mathbb{R})\,(\mathsf{source} : \mathsf{Bool})$$
$$\sum \mathsf{target} : \mathsf{Bool}, \mathsf{fep020\_transition}\ \alpha\ \mathsf{source}\ \mathsf{target} = 1$$

$$(\alpha\ p : \mathbb{R})$$
$$\mathsf{fep020\_evolve}\ \alpha\ p = \alpha + (1 - 2 \cdot \alpha) \cdot p$$

$$(\alpha : \mathbb{R})$$
$$\mathsf{fep020\_evolve}\ \alpha\ (1/2 : \mathbb{R}) = 1/2$$

$$(\alpha\ p : \mathbb{R})$$
$$\mathsf{fep020\_evolve}\ \alpha\ p - 1/2 = (1 - 2 \cdot \alpha) \cdot (p - 1/2)$$

$$(\alpha\ p : \mathbb{R})\,(n : \mathbb{N})$$
$$((\mathsf{fep020\_evolve}\ \alpha)^{[n]})\ p - 1/2 = (1 - 2 \cdot \alpha)^n \cdot (p - 1/2)$$

$$\{\alpha : \mathbb{R}\}\,(h\alpha 0 : 0 < \alpha)\,(h\alpha 1 : \alpha < 1)\,(p : \mathbb{R})$$
$$\mathsf{Filter.Tendsto}\ (\lambda\, n \mapsto ((\mathsf{fep020\_evolve}\ \alpha)^{[n]})\ p)\ \mathsf{Filter.atTop}$$
$$(\mathsf{nhds}\ (1/2 : \mathbb{R}))$$

13.21 fep-021 — EFE Epistemic–Pragmatic Balance

Expected free energy is pragmatic expected cost minus epistemic information value in ENNReal; under epistemic value at most cost, adding information back reconstructs cost, while EFE is monotone in cost and antitone in information. Assumption scope: The truncated subtraction convention makes EFE nonnegative and requires an explicit epistemic≤pragmatic premise for exact reconstruction. It does not assert the alternative risk-plus-ambiguity identity without a generative-model derivation. Semantic disposition: `formalized`.

13.21.1 Lean sketch

```
import Mathlib.Data.ENNReal.Inv

namespace FEP021

open scoped ENNReal
```

```
/-- Expected free energy convention: pragmatic expected cost minus epistemic
information value, truncated at zero in the extended nonnegative reals. -/
noncomputable def fep021_expectedFreeEnergy
    (pragmaticCost epistemicValue : ENNReal) : ENNReal :=
  pragmaticCost - epistemicValue

/-- Expected free energy is nonnegative in its native codomain. -/
theorem fep021_efe_nonneg (pragmaticCost epistemicValue : ENNReal) :
    0 ≤ fep021_expectedFreeEnergy pragmaticCost epistemicValue :=
  bot_le

/-- When information value does not exceed pragmatic cost, adding it back
exactly reconstructs that cost. This fixes the epistemic sign convention. -/
theorem fep021_efe_epistemic_balance
    {pragmaticCost epistemicValue : ENNReal}
    (hvalue : epistemicValue ≤ pragmaticCost) :
    fep021_expectedFreeEnergy pragmaticCost epistemicValue + epistemicValue =
      pragmaticCost := by
  exact tsub_add_cancel_of_le hvalue

/-- Higher pragmatic cost cannot reduce EFE at fixed information value. -/
theorem fep021_efe_mono_pragmatic
    {pragmatic₁ pragmatic₂ epistemicValue : ENNReal}
    (hpragmatic : pragmatic₁ ≤ pragmatic₂) :
    fep021_expectedFreeEnergy pragmatic₁ epistemicValue ≤
      fep021_expectedFreeEnergy pragmatic₂ epistemicValue :=
  tsub_le_tsub_right hpragmatic epistemicValue

/-- Higher epistemic value cannot increase EFE at fixed pragmatic cost. -/
theorem fep021_efe_antitone_epistemic
    {pragmaticCost epistemic₁ epistemic₂ : ENNReal}
    (hepistemic : epistemic₁ ≤ epistemic₂) :
    fep021_expectedFreeEnergy pragmaticCost epistemic₂ ≤
      fep021_expectedFreeEnergy pragmaticCost epistemic₁ :=
  tsub_le_tsub_left hepistemic pragmaticCost

/-- Truncated EFE is zero exactly when information value covers pragmatic cost. -/
theorem fep021_efe_eq_zero_iff
    (pragmaticCost epistemicValue : ENNReal) :
    fep021_expectedFreeEnergy pragmaticCost epistemicValue = 0 ↔
      pragmaticCost ≤ epistemicValue := by
  exact tsub_eq_zero_iff_le

end FEP021
```

13.21.2 Typeset statement signatures

$$(\mathsf{pragmaticCost}\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0})$$
$$0 \leq \mathsf{fep021\_expectedFreeEnergy}\ \mathsf{pragmaticCost}\ \mathsf{epistemicValue}$$

$$\{\mathsf{pragmaticCost}\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0}\}$$
$$(\mathsf{hvalue} : \mathsf{epistemicValue} \leq \mathsf{pragmaticCost})$$
$$\mathsf{fep021\_expectedFreeEnergy}\ \mathsf{pragmaticCost}\ \mathsf{epistemicValue} + \mathsf{epistemicValue}$$
$$= \mathsf{pragmaticCost}$$

$$\{pragmatic_1\ pragmatic_2\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0}\}$$
$$(\mathsf{hpragmatic} : pragmatic_1 \leq pragmatic_2)$$
$$\mathsf{fep021\_expectedFreeEnergy}\ pragmatic_1\ \mathsf{epistemicValue} \leq \mathsf{fep021\_expectedFreeEnergy}$$
$$pragmatic_2\ \mathsf{epistemicValue}$$

$\{\mathsf{pragmaticCost}\ epistemic_1\ epistemic_2 : \mathbb{R}^{\infty}_{\geq 0}\}$
$(\mathsf{hepistemic} : epistemic_1 \leq epistemic_2)$
$\mathsf{fep021\_expectedFreeEnergy\ pragmaticCost}\ epistemic_2 \leq \mathsf{fep021\_expectedFreeEnergy}$
$\mathsf{pragmaticCost}\ epistemic_1$

$(\mathsf{pragmaticCost\ epistemicValue} : \mathbb{R}^{\infty}_{\geq 0})$
$\mathsf{fep021\_expectedFreeEnergy\ pragmaticCost\ epistemicValue} = 0 \leftrightarrow \mathsf{pragmaticCost}$
$\leq \mathsf{epistemicValue}$

13.22 fep-022 — Posterior-Predictive Kernel and Brier Properness

Markov posterior and likelihood kernels compose into normalized posterior-predictive fibres, and expected Bernoulli Brier loss is uniquely minimized by the true event probability. Assumption scope: Kernel normalization requires both components to be Markov. Brier properness is an exact real-algebra identity whose probabilistic interpretation restricts true and forecast masses to [0,1]. It is a proper one-event discrepancy, not an empirical calibration-frequency theorem. Semantic disposition: `formalized`.

13.22.1 Lean sketch

```
import Mathlib.Probability.Kernel.Composition.MeasureComp
import Mathlib.Tactic

namespace FEP022

open MeasureTheory ProbabilityTheory
open scoped ENNReal

variable {𝒞 Ω 𝒳 : Type*}
variable [MeasurableSpace 𝒞] [MeasurableSpace Ω] [MeasurableSpace 𝒳]

/-- A posterior-predictive kernel composes a context-indexed posterior with a
latent-state likelihood kernel. -/
noncomputable def fep022_posteriorPredictive
    (posterior : Kernel 𝒞 Ω) (likelihood : Kernel Ω 𝒳) : Kernel 𝒞 𝒳 :=
  likelihood ∘ₖ posterior

/-- Markov posterior and likelihood kernels yield normalized predictive fibres. -/
theorem fep022_posteriorPredictive_mass_one
    (posterior : Kernel 𝒞 Ω) (likelihood : Kernel Ω 𝒳)
    [IsMarkovKernel posterior] [IsMarkovKernel likelihood] (c : 𝒞) :
    fep022_posteriorPredictive posterior likelihood c Set.univ = 1 := by
  change (likelihood ∘ₖ posterior) c Set.univ = 1
  exact measure_univ

/-- Expected Bernoulli Brier loss at true mass `p` and forecast `q`. -/
def fep022_brierScore (p q : ℝ) : ℝ :=
  p * (1 - q) ^ 2 + (1 - p) * q ^ 2

/-- Brier loss is Bayes uncertainty plus squared forecast error. -/
theorem fep022_brier_decomposition (p q : ℝ) :
    fep022_brierScore p q = p * (1 - p) + (q - p) ^ 2 := by
  simp only [fep022_brierScore]
  ring

/-- Expected Brier loss is minimized uniquely by reporting the true mass. -/
theorem fep022_brier_eq_optimum_iff (p q : ℝ) :
    fep022_brierScore p q = p * (1 - p) ↔ q = p := by
  rw [fep022_brier_decomposition]
  constructor
  · intro h
```

```
    nlinarith [sq_nonneg (q - p)]
  · rintro rfl
    ring

/-- Extract the predictive probability assigned to `true` by a binary model. -/
noncomputable def fep022_predictiveTrueMass
    (posterior : Kernel 𝒞 Ω) (likelihood : Kernel Ω Bool) (c : 𝒞) : ℝ :=
  (fep022_posteriorPredictive posterior likelihood c {true}).toReal

/-- The predictive Brier score inherits the exact proper-score decomposition. -/
theorem fep022_predictive_brier_decomposition
    (posterior : Kernel 𝒞 Ω) (likelihood : Kernel Ω Bool) (c : 𝒞) (p : ℝ) :
    fep022_brierScore p (fep022_predictiveTrueMass posterior likelihood c) =
      p * (1 - p) +
        (fep022_predictiveTrueMass posterior likelihood c - p) ^ 2 := by
  exact fep022_brier_decomposition _ _

end FEP022
```

13.22.2 Typeset statement signatures

$\{\mathcal{C}\,\Omega\,\mathcal{X} : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\mathcal{C}]\,[\mathsf{MeasurableSpace}\Omega]\,[\mathsf{MeasurableSpace}\mathcal{X}]$
$(\mathsf{posterior} : \mathsf{Kernel}\,\mathcal{C}\,\Omega)\,(\mathsf{likelihood} : \mathsf{Kernel}\,\Omega\,\mathcal{X})\,[\mathsf{IsMarkovKernel}\,\mathsf{posterior}]$
$[\mathsf{IsMarkovKernel}\,\mathsf{likelihood}]\,(c : \mathcal{C})$
$\mathsf{fep022\_posteriorPredictive}\,\mathsf{posterior}\,\mathsf{likelihood}\,c\,\Omega = 1$

$\{\mathcal{C}\,\Omega\,\mathcal{X} : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\mathcal{C}]\,[\mathsf{MeasurableSpace}\Omega]\,[\mathsf{MeasurableSpace}\mathcal{X}]$
$(p\,q : \mathbb{R})$
$\mathsf{fep022\_brierScore}\,p\,q = p \cdot (1-p) + (q-p)^2$

$\{\mathcal{C}\,\Omega\,\mathcal{X} : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\mathcal{C}]\,[\mathsf{MeasurableSpace}\Omega]\,[\mathsf{MeasurableSpace}\mathcal{X}]$
$(p\,q : \mathbb{R})$
$\mathsf{fep022\_brierScore}\,p\,q = p \cdot (1-p) \leftrightarrow q = p$

$\{\mathcal{C}\,\Omega\,\mathcal{X} : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\mathcal{C}]\,[\mathsf{MeasurableSpace}\Omega]\,[\mathsf{MeasurableSpace}\mathcal{X}]$
$(\mathsf{posterior} : \mathsf{Kernel}\,\mathcal{C}\,\Omega)\,(\mathsf{likelihood} : \mathsf{Kernel}\,\Omega\,\mathsf{Bool})\,(c : \mathcal{C})\,(p : \mathbb{R})$
$\mathsf{fep022\_brierScore}\,p\,(\mathsf{fep022\_predictiveTrueMass}\,\mathsf{posterior}\,\mathsf{likelihood}\,c) = p$
$\cdot\,(1-p) + (\mathsf{fep022\_predictiveTrueMass}\,\mathsf{posterior}\,\mathsf{likelihood}\,c - p)^2$

13.23 fep-023 — Policy-Indexed Reachable Laws

The distributional affordance set is the image of available policies under their outcome-law map; policy-set inclusion is monotone and normalization of every policy law transfers to every reachable law. Assumption scope: Reachability is policy-indexed one-step image semantics. Normalization is an explicit premise on available policy laws; no controllability, multi-step support closure, or topology on the law space is claimed. Semantic disposition: `formalized`.

13.23.1 Lean sketch

```
import Mathlib.MeasureTheory.Measure.MeasureSpace

namespace FEP023

open MeasureTheory

variable {Policy Outcome : Type*} [MeasurableSpace Outcome]

/-- Distributional affordance set: outcome laws induced by available policies. -/
def fep023_reachableLaws
    (policies : Set Policy) (law : Policy → Measure Outcome) :
```

```
    Set (Measure Outcome) :=
  law '' policies

/-- Every available policy induces a reachable outcome law. -/
theorem fep023_policy_reachable
    (policies : Set Policy) (law : Policy → Measure Outcome)
    {policy : Policy} (hpolicy : policy ∈ policies) :
    law policy ∈ fep023_reachableLaws policies law :=
  ⟨policy, hpolicy, rfl⟩

/-- Expanding the policy set can only enlarge the reachable-law set. -/
theorem fep023_reachable_mono
    {policies₁ policies₂ : Set Policy} (law : Policy → Measure Outcome)
    (hpolicies : policies₁ ⊆ policies₂) :
    fep023_reachableLaws policies₁ law ⊆
      fep023_reachableLaws policies₂ law := by
  rintro _ ⟨policy, hpolicy, rfl⟩
  exact ⟨policy, hpolicies hpolicy, rfl⟩

/-- If each policy law is normalized, every reachable law is normalized. -/
theorem fep023_reachable_normalized
    (policies : Set Policy) (law : Policy → Measure Outcome)
    (hlaw : ∀ policy ∈ policies, law policy Set.univ = 1)
    {μ : Measure Outcome} (hμ : μ ∈ fep023_reachableLaws policies law) :
    μ Set.univ = 1 := by
  rcases hμ with ⟨policy, hpolicy, rfl⟩
  exact hlaw policy hpolicy

/-- No policy means no reachable law. -/
theorem fep023_reachable_empty (law : Policy → Measure Outcome) :
    fep023_reachableLaws (∅ : Set Policy) law = ∅ := by
  simp [fep023_reachableLaws]

end FEP023
```

13.23.2 Typeset statement signatures

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$(\text{policies} : \text{Set Policy})\ (\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})\ \{\text{policy} : \text{Policy}\}$$
$$(\text{hpolicy} : \text{policy} \in \text{policies})$$
$$\text{law policy} \in \text{fep023\_reachableLaws policies law}$$

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$\{policies_1\ policies_2 : \text{Set Policy}\}\ (\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})$$
$$(\text{hpolicies} : policies_1 \subseteq policies_2)$$
$$\text{fep023\_reachableLaws}\ policies_1\ \text{law} \subseteq \text{fep023\_reachableLaws}\ policies_2\ \text{law}$$

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$(\text{policies} : \text{Set Policy})\ (\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})$$
$$(\text{hlaw} : \forall \text{policy} \in \text{policies}, \text{law policy}\ \Omega = 1)\ \{\mu : \text{MeasureOutcome}\}$$
$$(h\mu : \mu \in \text{fep023\_reachableLaws policies law})$$
$$\mu\ \Omega = 1$$

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$(\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})$$
$$\text{fep023\_reachableLaws}\ (\emptyset : \text{Set Policy})\ \text{law} = \emptyset$$

13.24 fep-024 — KL Regularization in Objectives

An ENNReal objective regularized by a nonnegative weight times native measure KL upper-bounds its base, is monotone in weight, recovers the base at zero weight, and is exact when approximation equals a sigma-finite prior. Assumption scope: The codomain explicitly admits infinite objectives and KL values. Exactness at the prior uses sigma-finiteness; no differentiability or optimizer-existence claim is made. Semantic disposition: `formalized`.

13.24.1 Lean sketch

```
import Mathlib.InformationTheory.KullbackLeibler.Basic

namespace FEP024

open MeasureTheory
open scoped ENNReal

variable {α : Type*} [MeasurableSpace α]

/-- Base objective plus a weighted native measure-valued KL regularizer. -/
noncomputable def fep024_klRegularizedObjective
    (base weight : ENNReal) (approximation prior : Measure α) : ENNReal :=
  base + weight * InformationTheory.klDiv approximation prior

/-- A KL-regularized objective upper-bounds its base objective. -/
theorem fep024_klRegularizedObjective_ge
    (base weight : ENNReal) (approximation prior : Measure α) :
    base ≤ fep024_klRegularizedObjective base weight approximation prior := by
  exact le_add_right (le_refl _)

/-- Zero regularization weight recovers the base objective exactly. -/
theorem fep024_klRegularizedObjective_zeroWeight
    (base : ENNReal) (approximation prior : Measure α) :
    fep024_klRegularizedObjective base 0 approximation prior = base := by
  simp [fep024_klRegularizedObjective]

/-- The regularized objective is monotone in its nonnegative weight. -/
theorem fep024_klRegularizedObjective_monoWeight
    (base : ENNReal) {weight₁ weight₂ : ENNReal}
    (hweight : weight₁ ≤ weight₂) (approximation prior : Measure α) :
    fep024_klRegularizedObjective base weight₁ approximation prior ≤
      fep024_klRegularizedObjective base weight₂ approximation prior := by
  simpa [fep024_klRegularizedObjective, mul_comm, add_comm] using
    add_le_add_left
      (mul_le_mul_right hweight (InformationTheory.klDiv approximation prior)) base

/-- Matching a sigma-finite prior removes the KL regularizer. -/
theorem fep024_klRegularizedObjective_exact
    (base weight : ENNReal) (prior : Measure α) [SigmaFinite prior] :
    fep024_klRegularizedObjective base weight prior prior = base := by
  simp [fep024_klRegularizedObjective, InformationTheory.klDiv_self]

end FEP024
```

13.24.2 Typeset statement signatures

$$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$$

$$(\mathsf{base}\ \mathsf{weight} : \mathbb{R}^{\infty}_{\geq 0})\,(\mathsf{approximation}\ \mathsf{prior} : \mathsf{Measure}\ \alpha)$$

$$\mathsf{base} \leq \mathsf{fep024\_klRegularizedObjective}\ \mathsf{base}\ \mathsf{weight}\ \mathsf{approximation}\ \mathsf{prior}$$

$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$
$(\mathsf{base} : \mathbb{R}^\infty_{\geq 0})\,(\mathsf{approximation\ prior} : \mathsf{Measure}\ \alpha)$
$\mathsf{fep024\_klRegularizedObjective\ base}\ 0\ \mathsf{approximation\ prior} = \mathsf{base}$

$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$
$(\mathsf{base} : \mathbb{R}^\infty_{\geq 0})\,\{weight_1\ weight_2 : \mathbb{R}^\infty_{\geq 0}\}\,(\mathsf{hweight} : weight_1 \leq weight_2)$
$(\mathsf{approximation\ prior} : \mathsf{Measure}\ \alpha)$
$\mathsf{fep024\_klRegularizedObjective\ base}\ weight_1\ \mathsf{approximation\ prior}$
$\leq \mathsf{fep024\_klRegularizedObjective\ base}\ weight_2\ \mathsf{approximation\ prior}$

$\{\alpha : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\ \alpha]$
$(\mathsf{base\ weight} : \mathbb{R}^\infty_{\geq 0})\,(\mathsf{prior} : \mathsf{Measure}\ \alpha)\,[\mathsf{SigmaFinite\ prior}]$
$\mathsf{fep024\_klRegularizedObjective\ base\ weight\ prior\ prior} = \mathsf{base}$

13.25 fep-025 — Conserved Nonequilibrium Probability Currents

For flux $\pi_i P_{ij}$, forward-minus-reverse current is antisymmetric; row normalization plus π-stationarity derives zero divergence. A directed three-state transition with uniform stationary law has a nonzero divergence-free current. Assumption scope: The general stationarity theorem explicitly requires normalized transition rows and a stationary real mass vector. The concrete witness is a finite deterministic cycle; no continuous-state NESS existence or thermodynamic force law is inferred here. Semantic disposition: `formalized`.

13.25.1 Lean sketch

```
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.LinearAlgebra.Matrix.Notation
import Mathlib.Tactic

namespace FEP025

open Finset

/-- Net probability current is forward flow minus reverse flow. -/
def fep025_probabilityCurrent {n : ℕ}
    (flow : Matrix (Fin n) (Fin n) ℝ) : Matrix (Fin n) (Fin n) ℝ :=
  fun i j => flow i j - flow j i

/-- Probability current is antisymmetric under edge reversal. -/
theorem fep025_probabilityCurrent_antisymm {n : ℕ}
    (flow : Matrix (Fin n) (Fin n) ℝ) (i j : Fin n) :
    fep025_probabilityCurrent flow i j =
      -fep025_probabilityCurrent flow j i := by
  simp only [fep025_probabilityCurrent]
  ring

/-- A probability current has no diagonal self-current. -/
theorem fep025_probabilityCurrent_diag_zero {n : ℕ}
    (flow : Matrix (Fin n) (Fin n) ℝ) (i : Fin n) :
    fep025_probabilityCurrent flow i i = 0 := by
  simp [fep025_probabilityCurrent]

/-- Net current leaving a state. Stationarity is zero divergence at each state. -/
def fep025_divergence {n : ℕ}
    (current : Matrix (Fin n) (Fin n) ℝ) (i : Fin n) : ℝ :=
  Σ j, current i j

/-- Antisymmetry conserves total probability: divergences sum to zero. -/
theorem fep025_total_divergence_zero {n : ℕ}
    (current : Matrix (Fin n) (Fin n) ℝ)
```

```
    (hcurrent : ∀ i j, current i j = -current j i) :
    Σ i, fep025_divergence current i = 0 := by
  have hneg : (Σ i, Σ j, current i j) = -(Σ i, Σ j, current i j) := by
    calc
      (Σ i, Σ j, current i j) = Σ j, Σ i, current i j :=
        Finset.sum_comm
      _ = Σ j, Σ i, -current j i := by
        apply Finset.sum_congr rfl
        intro j _
        apply Finset.sum_congr rfl
        intro i _
        exact hcurrent i j
      _ = -(Σ j, Σ i, current j i) := by simp
      _ = -(Σ i, Σ j, current i j) := rfl
  simp only [fep025_divergence]
  linarith

/-- Current induced by a distribution and transition matrix. -/
def fep025_transitionCurrent {n : ℕ}
    (stationary : Fin n → ℝ) (transition : Matrix (Fin n) (Fin n) ℝ) :
    Matrix (Fin n) (Fin n) ℝ :=
  fep025_probabilityCurrent (fun i j => stationary i * transition i j)

/-- Row normalization and distribution stationarity imply zero current
divergence at every state. -/
theorem fep025_transitionCurrent_stationary {n : ℕ}
    (stationary : Fin n → ℝ) (transition : Matrix (Fin n) (Fin n) ℝ)
    (hrow : ∀ i, Σ j, transition i j = 1)
    (hstationary : ∀ j, Σ i, stationary i * transition i j = stationary j)
    (i : Fin n) :
    fep025_divergence (fep025_transitionCurrent stationary transition) i = 0 := by
  simp only [fep025_divergence, fep025_transitionCurrent,
    fep025_probabilityCurrent, Finset.sum_sub_distrib, ← Finset.mul_sum]
  rw [hrow i, mul_one, hstationary i, sub_self]

/-- Uniform stationary law for a directed three-state cycle. -/
noncomputable def fep025_cycleStationary : Fin 3 → ℝ := ![1 / 3, 1 / 3, 1 / 3]

/-- Deterministic clockwise transition on three states. -/
def fep025_cycleTransition : Matrix (Fin 3) (Fin 3) ℝ :=
  !![0, 1, 0; 0, 0, 1; 1, 0, 0]

/-- The probability current generated by the stationary cycle model. -/
noncomputable def fep025_cycleCurrent : Matrix (Fin 3) (Fin 3) ℝ :=
  fep025_transitionCurrent fep025_cycleStationary fep025_cycleTransition

theorem fep025_cycleTransition_row (i : Fin 3) :
    Σ j, fep025_cycleTransition i j = 1 := by
  fin_cases i <;> norm_num [fep025_cycleTransition, Fin.sum_univ_succ]

theorem fep025_cycleStationary_invariant (j : Fin 3) :
    Σ i, fep025_cycleStationary i * fep025_cycleTransition i j =
      fep025_cycleStationary j := by
  fin_cases j <;> norm_num [fep025_cycleStationary, fep025_cycleTransition,
    Fin.sum_univ_succ]

/-- The three-cycle current is divergence-free at every state. -/
theorem fep025_cycleCurrent_stationary (i : Fin 3) :
    fep025_divergence fep025_cycleCurrent i = 0 := by
  exact fep025_transitionCurrent_stationary fep025_cycleStationary
    fep025_cycleTransition fep025_cycleTransition_row
    fep025_cycleStationary_invariant i
```

```
/-- Stationarity does not force detailed balance: the cycle current is nonzero. -/
theorem fep025_cycleCurrent_nonzero : fep025_cycleCurrent 0 1 ≠ 0 := by
  norm_num [fep025_cycleCurrent, fep025_transitionCurrent,
    fep025_probabilityCurrent, fep025_cycleStationary, fep025_cycleTransition]

end FEP025
```

13.25.2 Typeset statement signatures

$$\{n : \mathbb{N}\}\ (\mathsf{flow} : \mathsf{Matrix}\ (\mathsf{Fin}\ n)\ (\mathsf{Fin}\ n)\ \mathbb{R})\ (i\ j : \mathsf{Fin}\ n)$$
$$\mathsf{fep025\_probabilityCurrent}\ \mathsf{flow}\ i\ j = -\mathsf{fep025\_probabilityCurrent}\ \mathsf{flow}\ j\ i$$

$$\{n : \mathbb{N}\}\ (\mathsf{flow} : \mathsf{Matrix}\ (\mathsf{Fin}\ n)\ (\mathsf{Fin}\ n)\ \mathbb{R})\ (i : \mathsf{Fin}\ n)$$
$$\mathsf{fep025\_probabilityCurrent}\ \mathsf{flow}\ i\ i = 0$$

$$\{n : \mathbb{N}\}\ (\mathsf{current} : \mathsf{Matrix}\ (\mathsf{Fin}\ n)\ (\mathsf{Fin}\ n)\ \mathbb{R})$$
$$(\mathsf{hcurrent} : \forall i\ j, \mathsf{current}\ i\ j = -\mathsf{current}\ j\ i)$$
$$\sum i, \mathsf{fep025\_divergence}\ \mathsf{current}\ i = 0$$

$$\{n : \mathbb{N}\}\ (\mathsf{stationary} : \mathsf{Fin}\ n \Rightarrow \mathbb{R})\ (\mathsf{transition} : \mathsf{Matrix}\ (\mathsf{Fin}\ n)\ (\mathsf{Fin}\ n)\ \mathbb{R})$$
$$(\mathsf{hrow} : \forall i, \sum j, \mathsf{transition}\ i\ j = 1)\ (\mathsf{hstationary} : \forall j, \sum i, \mathsf{stationary}\ i$$
$$\cdot\ \mathsf{transition}\ i\ j = \mathsf{stationary}\ j)\ (i : \mathsf{Fin}\ n)$$
$$\mathsf{fep025\_divergence}\ (\mathsf{fep025\_transitionCurrent}\ \mathsf{stationary}\ \mathsf{transition})\ i = 0$$

$$(i : \mathsf{Fin}\ 3)$$
$$\sum j, \mathsf{fep025\_cycleTransition}\ i\ j = 1$$

$$(j : \mathsf{Fin}\ 3)$$
$$\sum i, \mathsf{fep025\_cycleStationary}\ i \cdot \mathsf{fep025\_cycleTransition}\ i\ j$$
$$= \mathsf{fep025\_cycleStationary}\ j$$

$$(i : \mathsf{Fin}\ 3)$$
$$\mathsf{fep025\_divergence}\ \mathsf{fep025\_cycleCurrent}\ i = 0$$

$$\mathsf{fep025\_cycleCurrent}\ 0\ 1 \neq 0$$

13.26 fep-026 — Negative-Log Prior Complexity

Negative-log prior complexity is additive on positive product densities, strictly antitone, and zero exactly at unit density. Assumption scope: The input is a positive scalar prior-density value, not a normalized prior measure. Product additivity is exact; measure-valued relative complexity is owned by fep-014 and fep-024. Semantic disposition: `formalized`.

13.26.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Log.Basic

namespace FEP026

open Real

/-- Negative-log prior complexity of a positive prior density value. -/
noncomputable def fep026_priorComplexity (priorDensity : ℝ) : ℝ :=
  -Real.log priorDensity

/-- Complexity penalties via log: log is monotone on positive reals. -/
```

```
theorem fep026_log_monotone {a b : ℝ} (ha : 0 < a) (hab : a ≤ b) :
    Real.log a ≤ Real.log b :=
  Real.log_le_log ha hab

/-- Quotient rule for real logarithms (``log(a/b) = log a - log b``). -/
theorem fep026_log_div {a b : ℝ} (ha : 0 < a) (hb : 0 < b) :
    Real.log (a / b) = Real.log a - Real.log b :=
  Real.log_div ha.ne' hb.ne'

/-- Complexity penalty: -log p(θ) increases as the prior p(θ) decreases toward zero. -/
theorem fep026_complexity_increases {p q : ℝ} (hp : 0 < p) (_hq : 0 < q) (h : p ≤ q) :
    fep026_priorComplexity q ≤ fep026_priorComplexity p :=
  neg_le_neg (Real.log_le_log hp h)

/-- Complexity at prior equal to 1 is zero. -/
theorem fep026_complexity_zero_at_one : fep026_priorComplexity 1 = 0 := by
  simp [fep026_priorComplexity]

/-- Complexity of product is sum of complexities. -/
theorem fep026_complexity_additive {a b : ℝ} (ha : 0 < a) (hb : 0 < b) :
    fep026_priorComplexity (a * b) =
      fep026_priorComplexity a + fep026_priorComplexity b := by
  simp only [fep026_priorComplexity]
  rw [Real.log_mul ha.ne' hb.ne', neg_add]

/-- For positive prior density, zero complexity occurs exactly at unit
density. -/
theorem fep026_priorComplexity_eq_zero_iff {p : ℝ} (hp : 0 < p) :
    fep026_priorComplexity p = 0 ↔ p = 1 := by
  constructor
  · intro hzero
    apply Real.eq_one_of_pos_of_log_eq_zero hp
    exact neg_eq_zero.mp (by simpa [fep026_priorComplexity] using hzero)
  · rintro rfl
    exact fep026_complexity_zero_at_one

/-- Negative-log complexity strictly decreases as positive prior density
increases. -/
theorem fep026_priorComplexity_strictAnti
    {p q : ℝ} (hp : 0 < p) (hpq : p < q) :
    fep026_priorComplexity q < fep026_priorComplexity p := by
  exact neg_lt_neg (Real.log_lt_log hp hpq)

end FEP026
```

13.26.2 Typeset statement signatures

$$\{a\,b : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)\,(\mathsf{hab} : a \le b)$$
$$\mathsf{Real.log}\,a \le \mathsf{Real.log}\,b$$

$$\{a\,b : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)\,(\mathsf{hb} : 0 < b)$$
$$\mathsf{Real.log}\,(a/b) = \mathsf{Real.log}\,a - \mathsf{Real.log}\,b$$

$$\{p\,q : \mathbb{R}\}\,(\mathsf{hp} : 0 < p)\,(\_hq : 0 < q)\,(h : p \le q)$$
$$\mathsf{fep026\_priorComplexity}\,q \le \mathsf{fep026\_priorComplexity}\,p$$

$$\mathsf{fep026\_priorComplexity}\,1 = 0$$

$$\{a\, b : \mathbb{R}\}\, (\mathsf{ha} : 0 < a)\, (\mathsf{hb} : 0 < b)$$
$$\mathsf{fep026\_priorComplexity}\, (a \cdot b) = \mathsf{fep026\_priorComplexity}\, a$$
$$+ \mathsf{fep026\_priorComplexity}\, b$$

$$\{p : \mathbb{R}\}\, (\mathsf{hp} : 0 < p)$$
$$\mathsf{fep026\_priorComplexity}\, p = 0 \leftrightarrow p = 1$$

$$\{p\, q : \mathbb{R}\}\, (\mathsf{hp} : 0 < p)\, (\mathsf{hpq} : p < q)$$
$$\mathsf{fep026\_priorComplexity}\, q < \mathsf{fep026\_priorComplexity}\, p$$

13.27 fep-027 — Hierarchical Kernel Factorization

A parent measure and child kernel form a normalized hierarchical joint whose marginals recover the parent and prior predictive, and whose three-level factorization is associative up to product reassociation. Assumption scope: Joint normalization requires a probability parent and Markov child; first- and second-marginal identities require the stated finite-kernel classes. The associativity theorem is exact for Mathlib composition products and does not impose causal identifiability. Semantic disposition: `formalized`.

13.27.1 Lean sketch

```
import Mathlib.Probability.Kernel.Composition.MeasureComp

namespace FEP027

open MeasureTheory ProbabilityTheory
open scoped ENNReal

variable {α β γ : Type*}
variable [MeasurableSpace α] [MeasurableSpace β] [MeasurableSpace γ]

/-- A normalized hierarchical joint is the native composition product of a
parent law and a child kernel. -/
noncomputable def fep027_hierarchicalJoint
    (μ : Measure α) (κ : Kernel α β) : Measure (α × β) :=
  μ ⊗ₘ κ

/-- Probability parents and Markov children yield a probability joint law. -/
theorem fep027_hierarchical_mass_one
    (μ : Measure α) (κ : Kernel α β)
    [IsProbabilityMeasure μ] [IsMarkovKernel κ] :
    fep027_hierarchicalJoint μ κ Set.univ = 1 := by
  change (μ ⊗ₘ κ) Set.univ = 1
  exact measure_univ

/-- The first marginal of the hierarchical joint recovers the parent law. -/
theorem fep027_hierarchical_fst
    (μ : Measure α) (κ : Kernel α β) [SFinite μ] [IsMarkovKernel κ] :
    (fep027_hierarchicalJoint μ κ).fst = μ := by
  exact Measure.fst_compProd μ κ

/-- The second marginal is the prior-predictive measure-kernel composition. -/
theorem fep027_hierarchical_snd
    (μ : Measure α) (κ : Kernel α β) [SFinite μ] [IsSFiniteKernel κ] :
    (fep027_hierarchicalJoint μ κ).snd = κ ∘ₘ μ := by
  exact Measure.snd_compProd μ κ

/-- Three-level hierarchical factorization is associative up to the canonical
measurable equivalence between product bracketings. -/
theorem fep027_hierarchical_assoc
    (μ : Measure α) (κ : Kernel α β) (η : Kernel (α × β) γ) :
    (fep027_hierarchicalJoint μ κ ⊗ₘ η).map MeasurableEquiv.prodAssoc =
```

```
      fep027_hierarchicalJoint μ (κ ⊗ₖ η) := by
  exact Measure.compProd_assoc'

end FEP027
```

13.27.2 Typeset statement signatures

$$\{\alpha\,\beta\,\gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\,\alpha\,\beta)\ [\mathsf{IsProbabilityMeasure}\ \mu]\ [\mathsf{IsMarkovKernel}\ \kappa]$$
$$\mathsf{fep027\_hierarchicalJoint}\ \mu\,\kappa\,\Omega = 1$$

$$\{\alpha\,\beta\,\gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\,\alpha\,\beta)\ [\mathsf{SFinite}\ \mu]\ [\mathsf{IsMarkovKernel}\ \kappa]$$
$$(\mathsf{fep027\_hierarchicalJoint}\ \mu\,\kappa).\mathsf{fst} = \mu$$

$$\{\alpha\,\beta\,\gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\,\alpha\,\beta)\ [\mathsf{SFinite}\ \mu]\ [\mathsf{IsSFiniteKernel}\ \kappa]$$
$$(\mathsf{fep027\_hierarchicalJoint}\ \mu\,\kappa).\mathsf{snd} = \kappa \circ_m\ \mu$$

$$\{\alpha\,\beta\,\gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\,\alpha\,\beta)\ (\eta : \mathsf{Kernel}\ (\alpha \times \beta)\ \gamma)$$
$$(\mathsf{fep027\_hierarchicalJoint}\ \mu\,\kappa \otimes_m\ \eta).\mathsf{map}\ \mathsf{MeasurableEquiv.prodAssoc}$$
$$= \mathsf{fep027\_hierarchicalJoint}\ \mu\ (\kappa \otimes_k\ \eta)$$

13.28 fep-028 — Support-Aware Finite Softmax Policy

Exponentiated policy scores on a nonempty selected support define a nonnegative full finite weight vector, vanish outside support, are pointwise at most one, and sum to one globally. Assumption scope: Only support nonemptiness is required; inverse temperature and costs may be arbitrary reals. This proves an exact stochastic policy law but not optimality, sampling behavior, or a temperature limit. Semantic disposition: `formalized`.

13.28.1 Lean sketch

```
import Mathlib.Data.Finset.Basic
import Mathlib.Analysis.SpecialFunctions.Exp

namespace FEP028

open Real Finset

-- [proof strategy: Real.exp_pos + Finset.sum_pos + div identities for softmax
 ↪  normalization]

abbrev Policy := Fin 10

noncomputable def fep028_softmax (γ : ℝ) (G : Policy → ℝ) (policies : Finset Policy)
 ↪  (p : Policy) :
    ℝ :=
  if p ∈ policies then
    Real.exp (-γ * G p) / Σ p' ∈ policies, Real.exp (-γ * G p')
  else 0

/-- Softmax probabilities are nonneg over nonempty policy sets. -/
theorem fep028_softmax_nonneg (γ : ℝ) (G : Policy → ℝ) (policies : Finset Policy) (p :
 ↪  Policy)
    (hne : policies.Nonempty) : 0 ≤ fep028_softmax γ G policies p := by
  classical
  have hsum : 0 < Σ p' ∈ policies, Real.exp (-γ * G p') :=
    Finset.sum_pos (fun _ _ => Real.exp_pos _) hne
```

```
  by_cases hp : p ∈ policies
  · rw [fep028_softmax, if_pos hp]
    exact div_nonneg (Real.exp_nonneg _) hsum.le
  · simp [fep028_softmax, hp]

/-- Every support-aware softmax weight is at most one. -/
theorem fep028_softmax_le_one
    (γ : ℝ) (G : Policy → ℝ) (policies : Finset Policy) (p : Policy)
    (hne : policies.Nonempty) :
    fep028_softmax γ G policies p ≤ 1 := by
  classical
  have hsum : 0 < ∑ p' ∈ policies, Real.exp (-γ * G p') :=
    Finset.sum_pos (fun _ _ => Real.exp_pos _) hne
  by_cases hp : p ∈ policies
  · rw [fep028_softmax, if_pos hp]
    apply (div_le_one hsum).2
    exact Finset.single_le_sum
      (fun q _ => Real.exp_nonneg (-γ * G q)) hp
  · simp [fep028_softmax, hp]

/-- Softmax probabilities over a nonempty finite policy set sum to one. -/
theorem fep028_softmax_probs_sum_one (γ : ℝ) (G : Policy → ℝ) (policies : Finset
 ↪ Policy)
    (hne : policies.Nonempty) :
    ∑ p ∈ policies, fep028_softmax γ G policies p = 1 := by
  classical
  have hden :
      (∑ p' ∈ policies, Real.exp (-γ * G p')) ≠ 0 :=
    ne_of_gt (Finset.sum_pos (fun _ _ => Real.exp_pos _) hne)
  calc
    ∑ p ∈ policies, fep028_softmax γ G policies p =
        ∑ p ∈ policies,
          Real.exp (-γ * G p) /
            ∑ p' ∈ policies, Real.exp (-γ * G p') := by
      apply Finset.sum_congr rfl
      intro p hp
      simp [fep028_softmax, hp]
    _ = 1 := by
      simp_rw [div_eq_mul_inv]
      rw [← Finset.sum_mul, mul_inv_cancel₀ hden]

/-- Policies outside the declared support receive exactly zero probability. -/
theorem fep028_softmax_support
    (γ : ℝ) (G : Policy → ℝ) (policies : Finset Policy) {p : Policy}
    (hp : p ∉ policies) :
    fep028_softmax γ G policies p = 0 := by
  simp [fep028_softmax, hp]

/-- The support-aware softmax is a normalized weight vector on the complete
finite policy type. -/
theorem fep028_softmax_sum_univ
    (γ : ℝ) (G : Policy → ℝ) (policies : Finset Policy)
    (hne : policies.Nonempty) :
    ∑ p : Policy, fep028_softmax γ G policies p = 1 := by
  classical
  have hsubset :
      (∑ p ∈ policies, fep028_softmax γ G policies p) =
        ∑ p : Policy, fep028_softmax γ G policies p := by
    apply Finset.sum_subset (Finset.subset_univ policies)
    intro p _ hp
    exact fep028_softmax_support γ G policies hp
  rw [← hsubset]
```

```
  exact fep028_softmax_probs_sum_one γ G policies hne

/-- Softmax numerator is strictly positive. -/
theorem fep028_numerator_pos (γ : ℝ) (G : Policy → ℝ) (p : Policy) :
    0 < Real.exp (-γ * G p) :=
  Real.exp_pos _

/-- Softmax denominator is strictly positive over a nonempty set. -/
theorem fep028_denominator_pos (γ : ℝ) (G : Policy → ℝ) (policies : Finset Policy)
    (hne : policies.Nonempty) :
    0 < ∑ p' ∈ policies, Real.exp (-γ * G p') :=
  Finset.sum_pos (fun _ _ => Real.exp_pos _) hne

end FEP028
```

13.28.2 Typeset statement signatures

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(\mathsf{policies} : \mathsf{Finset}\,\mathsf{Policy})\,(p : \mathsf{Policy})$$
$$(\mathsf{hne} : \mathsf{policies.Nonempty})$$
$$0 \le \mathsf{fep028\_softmax}\,\gamma\,G\,\mathsf{policies}\,p$$

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(\mathsf{policies} : \mathsf{Finset}\,\mathsf{Policy})\,(p : \mathsf{Policy})$$
$$(\mathsf{hne} : \mathsf{policies.Nonempty})$$
$$\mathsf{fep028\_softmax}\,\gamma\,G\,\mathsf{policies}\,p \le 1$$

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(\mathsf{policies} : \mathsf{Finset}\,\mathsf{Policy})\,(\mathsf{hne} : \mathsf{policies.Nonempty})$$
$$\sum p \in \mathsf{policies}, \mathsf{fep028\_softmax}\,\gamma\,G\,\mathsf{policies}\,p = 1$$

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(\mathsf{policies} : \mathsf{Finset}\,\mathsf{Policy})\,\{p : \mathsf{Policy}\}$$
$$(\mathsf{hp} : p \notin \mathsf{policies})$$
$$\mathsf{fep028\_softmax}\,\gamma\,G\,\mathsf{policies}\,p = 0$$

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(\mathsf{policies} : \mathsf{Finset}\,\mathsf{Policy})\,(\mathsf{hne} : \mathsf{policies.Nonempty})$$
$$\sum p : \mathsf{Policy}, \mathsf{fep028\_softmax}\,\gamma\,G\,\mathsf{policies}\,p = 1$$

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(p : \mathsf{Policy})$$
$$0 < \mathsf{Real.exp}\,(-\gamma \cdot G\,p)$$

$$(\gamma : \mathbb{R})\,(G : \mathsf{Policy} \Rightarrow \mathbb{R})\,(\mathsf{policies} : \mathsf{Finset}\,\mathsf{Policy})\,(\mathsf{hne} : \mathsf{policies.Nonempty})$$
$$0 < \sum p' \in \mathsf{policies}, \mathsf{Real.exp}\,(-\gamma \cdot G\,p')$$

13.29 fep-029 — Quadratic Bregman Divergence

The scalar Bregman divergence generated by $\varphi(x)=x^2$ equals $(x-y)^2$, is nonnegative, and vanishes exactly when x=y. Assumption scope: Unconditional over real x and y for the explicit quadratic generator; no general differentiable convex function, projection theorem, or mirror-descent convergence is claimed. Semantic disposition: `formalized`.

13.29.1 Lean sketch

```
import Mathlib.Analysis.Convex.Basic
import Mathlib.Tactic

namespace FEP029

/-- Bregman divergence prerequisite: convex functions satisfy the secant inequality. -/
theorem fep029_secant_ineq (a b t : ℝ) (ht0 : 0 ≤ t) (_ht1 : t ≤ 1) (hab : a ≤ b) :
    (1 - t) * a + t * b ≥ a := by
```

```
  nlinarith

/-- Weighted midpoint lies between endpoints (convex combination). -/
theorem fep029_convex_combo_bound (a b t : ℝ) (_ht0 : 0 ≤ t) (ht1 : t ≤ 1) (hab : a ≤
 ↪ b) :
    (1 - t) * a + t * b ≤ b := by
  nlinarith

/-- Endpoints of convex combination: t = 0 gives a. -/
theorem fep029_combo_t_zero (a b : ℝ) : (1 - 0) * a + 0 * b = a := by ring

/-- Endpoints of convex combination: t = 1 gives b. -/
theorem fep029_combo_t_one (a b : ℝ) : (1 - 1) * a + 1 * b = b := by ring

/-- The Bregman divergence generated by ψ(x) = x², whose derivative at y is 2y. -/
def fep029_quadraticBregman (x y : ℝ) : ℝ :=
  x ^ 2 - y ^ 2 - (2 * y) * (x - y)

/-- The quadratic Bregman divergence is exactly squared Euclidean distance. -/
theorem fep029_quadraticBregman_eq_sq (x y : ℝ) :
    fep029_quadraticBregman x y = (x - y) ^ 2 := by
  simp [fep029_quadraticBregman]
  ring

/-- The quadratic Bregman divergence is nonnegative. -/
theorem fep029_bregman_quadratic_nonneg (x y : ℝ) :
    0 ≤ fep029_quadraticBregman x y := by
  rw [fep029_quadraticBregman_eq_sq]
  exact sq_nonneg _

/-- The quadratic Bregman divergence separates points. -/
theorem fep029_quadraticBregman_eq_zero_iff (x y : ℝ) :
    fep029_quadraticBregman x y = 0 ↔ x = y := by
  rw [fep029_quadraticBregman_eq_sq, sq_eq_zero_iff, sub_eq_zero]

end FEP029
```

13.29.2 Typeset statement signatures

$$(a\,b\,t : \mathbb{R})\,(\mathsf{ht0} : 0 \le t)\,({}_h t1 : t \le 1)\,(\mathsf{hab} : a \le b)$$
$$(1-t)\cdot a + t\cdot b \ge a$$

$$(a\,b\,t : \mathbb{R})\,({}_h t0 : 0 \le t)\,(\mathsf{ht1} : t \le 1)\,(\mathsf{hab} : a \le b)$$
$$(1-t)\cdot a + t\cdot b \le b$$

$$(a\,b : \mathbb{R})$$
$$(1-0)\cdot a + 0\cdot b = a$$

$$(a\,b : \mathbb{R})$$
$$(1-1)\cdot a + 1\cdot b = b$$

$$(x\,y : \mathbb{R})$$
$$\mathsf{fep029\_quadraticBregman}\,x\,y = (x-y)^2$$

$$(x\,y : \mathbb{R})$$
$$0 \le \mathsf{fep029\_quadraticBregman}\,x\,y$$

$$(x\,y : \mathbb{R})$$
$$\mathsf{fep029\_quadraticBregman}\,x\,y = 0 \leftrightarrow x = y$$

13.30 fep-030 — Maximum Entropy Principle

Binary Shannon entropy in nats is globally at most log 2, with equality exactly at probability one half. Assumption scope: The native Mathlib theorem is unconditional over real inputs because Real.binEntropy has a total extension; the scientific probability domain is [0,1]. This is the Bernoulli maximum-entropy principle, not the general constrained Jaynes problem. Semantic disposition: `formalized`.

13.30.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.BinaryEntropy
import Mathlib.Algebra.BigOperators.Ring.Finset
import Mathlib.Data.Real.Basic
import Mathlib.Tactic

namespace FEP030

open Real Finset

-- [proof strategy: native binary-entropy maximum plus explicit finite-uniform
 ↪  calculation]

/-- Binary Shannon entropy (in nats) is globally bounded above by log 2. -/
theorem fep030_binaryEntropy_max (p : ℝ) :
    Real.binEntropy p ≤ Real.log 2 :=
  Real.binEntropy_le_log_two

/-- The binary entropy maximum is attained uniquely at probability one half. -/
theorem fep030_binaryEntropy_eq_max_iff (p : ℝ) :
    Real.binEntropy p = Real.log 2 ↔ p = (2 : ℝ)⁻¹ :=
  Real.binEntropy_eq_log_two

/-- A uniform scalar weight is nonnegative. -/
theorem fep030_uniform_nonneg (n : ℕ) (_hn : 0 < n) : 0 ≤ (1 : ℝ) / n :=
  div_nonneg zero_le_one (Nat.cast_nonneg' (n := n))

/-- Uniform weights on a positive finite range sum to one. -/
theorem fep030_uniform_sum_one (n : ℕ) (hn : 0 < n) :
    ∑ _ ∈ Finset.range n, (1 : ℝ) / n = 1 := by
  rw [Finset.sum_const, Finset.card_range, nsmul_eq_mul]
  have hn0 : (n : ℝ) ≠ 0 := Nat.cast_ne_zero.mpr (Nat.ne_of_gt hn)
  field_simp [hn0]

/-- The logarithm of a positive natural cardinality is nonnegative. -/
theorem fep030_log_card_nonneg (n : ℕ) (hn : 1 ≤ n) : 0 ≤ Real.log n := by
  apply Real.log_nonneg
  exact_mod_cast hn

/-- The explicit uniform-weight entropy expression equals log n; maximality is not
 ↪  proved. -/
theorem fep030_entropy_eq_log (n : ℕ) (hn : 0 < n) :
    -∑ _ ∈ Finset.range n, (1 : ℝ) / n * Real.log ((1 : ℝ) / n)
    = Real.log n := by
  have hnR : (n : ℝ) ≠ 0 := Nat.cast_ne_zero.mpr (Nat.ne_of_gt hn)
  have hlog : Real.log ((1 : ℝ) / n) = - Real.log n := by
    rw [Real.log_div (one_ne_zero' ℝ) hnR, Real.log_one, zero_sub]
  have hsum : ∑ _ ∈ Finset.range n, (1 : ℝ) / n * Real.log ((1 : ℝ) / n) = - Real.log n
 ↪  := by
    calc
      ∑ _ ∈ Finset.range n, (1 : ℝ) / n * Real.log ((1 : ℝ) / n)
          = ∑ _ ∈ Finset.range n, (1 : ℝ) / n * (- Real.log n) :=
            Finset.sum_congr rfl fun _ _ => by rw [hlog]
      _ = (∑ _ ∈ Finset.range n, (1 : ℝ) / n) * (- Real.log n) := by rw [←
 ↪  Finset.sum_mul]
```

```
      _ = 1 * (- Real.log n) := by rw [fep030_uniform_sum_one n hn]
      _ = - Real.log n := by ring
  rw [hsum, neg_neg]

end FEP030
```

13.30.2 Typeset statement signatures

$$(p : \mathbb{R})$$
$$\mathsf{Real.binEntropy}\, p \le \mathsf{Real.log}\, 2$$

$$(p : \mathbb{R})$$
$$\mathsf{Real.binEntropy}\, p = \mathsf{Real.log}\, 2 \leftrightarrow p = (2 : \mathbb{R})^{-1}$$

$$(n : \mathbb{N})\, ({}_h n : 0 < n)$$
$$0 \le (1 : \mathbb{R})/n$$

$$(n : \mathbb{N})\, (\mathsf{hn} : 0 < n)$$
$$\sum\nolimits_{\_} \in \mathsf{Finset.range}\, n, (1 : \mathbb{R})/n = 1$$

$$(n : \mathbb{N})\, (\mathsf{hn} : 1 \le n)$$
$$0 \le \mathsf{Real.log}\, n$$

$$(n : \mathbb{N})\, (\mathsf{hn} : 0 < n)$$
$$-\sum\nolimits_{\_} \in \mathsf{Finset.range}\, n, (1 : \mathbb{R})/n \cdot \mathsf{Real.log}\, ((1 : \mathbb{R})/n) = \mathsf{Real.log}\, n$$

13.31 fep-031 — Finite Boltzmann–Gibbs Weights

Positive exponential Gibbs weights normalize to a nonnegative mass function summing to one on any nonempty finite support. Assumption scope: Conditional only on a nonempty selected support; β and energies are arbitrary reals. The row does not derive Gibbs weights from maximum entropy or bundle the result as a measure. Semantic disposition: `formalized`.

13.31.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Exp

namespace FEP031

/-- Boltzmann weight exp(-βE) is strictly positive. -/
theorem fep031_gibbs_weight_pos (β E : ℝ) : 0 < Real.exp (-β * E) :=
  Real.exp_pos _

/-- Gibbs weight monotonicity: lower energy → higher weight at positive temperature (β
 ↪  > 0). -/
theorem fep031_gibbs_mono (β E₁ E₂ : ℝ) (hβ : 0 < β) (hE : E₁ ≤ E₂) :
    Real.exp (-β * E₂) ≤ Real.exp (-β * E₁) :=
  Real.exp_le_exp.mpr (by nlinarith [mul_le_mul_of_nonneg_left hE (le_of_lt hβ)])

/-- Partition function is strictly positive over any nonempty finite state space. -/
theorem fep031_partition_pos (β : ℝ) (n : ℕ) (E : Fin n → ℝ)
    (S : Finset (Fin n)) (hS : S.Nonempty) :
    0 < ∑ i ∈ S, Real.exp (-β * E i) :=
  Finset.sum_pos (fun _ _ => Real.exp_pos _) hS

/-- Gibbs probability on a selected finite support, normalized by its partition sum. -/
noncomputable def fep031_gibbsProbability (β : ℝ) (n : ℕ) (E : Fin n → ℝ)
    (S : Finset (Fin n)) (i : Fin n) : ℝ :=
  Real.exp (-β * E i) / ∑ j ∈ S, Real.exp (-β * E j)
```

```
/-- Each normalized Gibbs probability is nonnegative on nonempty support. -/
theorem fep031_gibbsProbability_nonneg (β : ℝ) (n : ℕ) (E : Fin n → ℝ)
    (S : Finset (Fin n)) (hS : S.Nonempty) (i : Fin n) :
    0 ≤ fep031_gibbsProbability β n E S i :=
  div_nonneg (le_of_lt (Real.exp_pos _))
    (le_of_lt (fep031_partition_pos β n E S hS))

/-- Normalized Gibbs probabilities sum to one on their selected support. -/
theorem fep031_gibbsProbability_sum_one (β : ℝ) (n : ℕ) (E : Fin n → ℝ)
    (S : Finset (Fin n)) (hS : S.Nonempty) :
    ∑ i ∈ S, fep031_gibbsProbability β n E S i = 1 := by
  have hden : (∑ j ∈ S, Real.exp (-β * E j)) ≠ 0 :=
    ne_of_gt (fep031_partition_pos β n E S hS)
  simp_rw [fep031_gibbsProbability, div_eq_mul_inv]
  rw [← Finset.sum_mul, mul_inv_cancel₀ hden]

end FEP031
```

13.31.2 Typeset statement signatures

$$(\beta\, E : \mathbb{R})$$
$$0 < \mathsf{Real.exp}\,(-\beta \cdot E)$$

$$(\beta\, E_1\, E_2 : \mathbb{R})\,(h\beta : 0 < \beta)\,(\mathsf{hE} : E_1 \le E_2)$$
$$\mathsf{Real.exp}\,(-\beta \cdot E_2) \le \mathsf{Real.exp}\,(-\beta \cdot E_1)$$

$$(\beta : \mathbb{R})\,(n : \mathbb{N})\,(E : \mathsf{Fin}\, n \Rightarrow \mathbb{R})\,(S : \mathsf{Finset}\,(\mathsf{Fin}\, n))\,(\mathsf{hS} : \mathsf{S.Nonempty})$$
$$0 < \sum i \in S, \mathsf{Real.exp}\,(-\beta \cdot E\, i)$$

$$(\beta : \mathbb{R})\,(n : \mathbb{N})\,(E : \mathsf{Fin}\, n \Rightarrow \mathbb{R})\,(S : \mathsf{Finset}\,(\mathsf{Fin}\, n))\,(\mathsf{hS} : \mathsf{S.Nonempty})$$
$$(i : \mathsf{Fin}\, n)$$
$$0 \le \mathsf{fep031\_gibbsProbability}\,\beta\, n\, E\, S\, i$$

$$(\beta : \mathbb{R})\,(n : \mathbb{N})\,(E : \mathsf{Fin}\, n \Rightarrow \mathbb{R})\,(S : \mathsf{Finset}\,(\mathsf{Fin}\, n))\,(\mathsf{hS} : \mathsf{S.Nonempty})$$
$$\sum i \in S, \mathsf{fep031\_gibbsProbability}\,\beta\, n\, E\, S\, i = 1$$

13.32 fep-032 — Exact Quadratic Gradient-Descent Dynamics

Gradient descent on the centered scalar quadratic has an exact n-step iterate, nonincreasing energy for step sizes in [0,2], and converges globally to the center for steps in (0,2). Assumption scope: The closed form is unconditional. Energy descent uses $0 \le \eta \le 2$ and convergence uses the strict stability interval $0 < \eta < 2$. This is an exact quadratic model, not a theorem for arbitrary smooth or convex objectives. Semantic disposition: `formalized`.

13.32.1 Lean sketch

```
import Mathlib.Analysis.SpecificLimits.Normed
import Mathlib.Tactic

namespace FEP032

/-- Gradient-descent update for the centered quadratic energy
`(x - center)² / 2`. -/
def fep032_quadraticUpdate (step center x : ℝ) : ℝ :=
  x - step * (x - center)

/-- One descent step scales displacement from the minimizer exactly. -/
theorem fep032_quadraticUpdate_deviation (step center x : ℝ) :
    fep032_quadraticUpdate step center x - center =
      (1 - step) * (x - center) := by
```

```
  simp only [fep032_quadraticUpdate]
  ring

/-- The minimizer is a fixed point for every step size. -/
theorem fep032_quadraticUpdate_fixed (step center : ℝ) :
    fep032_quadraticUpdate step center center = center := by
  simp [fep032_quadraticUpdate]

/-- Closed form for every finite gradient-descent iterate. -/
theorem fep032_quadraticUpdate_iterate (step center x : ℝ) (n : ℕ) :
    ((fep032_quadraticUpdate step center)^[n]) x - center =
      (1 - step) ^ n * (x - center) := by
  induction n with
  | zero => simp
  | succ n ih =>
      rw [Function.iterate_succ_apply']
      rw [fep032_quadraticUpdate_deviation, ih, pow_succ]
      ring

/-- Step sizes in `[0,2]` cannot increase the centered quadratic energy. -/
theorem fep032_quadraticEnergy_descent
    {step center x : ℝ} (hstep0 : 0 ≤ step) (hstep2 : step ≤ 2) :
    (fep032_quadraticUpdate step center x - center) ^ 2 ≤
      (x - center) ^ 2 := by
  rw [fep032_quadraticUpdate_deviation]
  have hfactor : (1 - step) ^ 2 ≤ 1 := by nlinarith
  have hscaled :=
    mul_le_mul_of_nonneg_right hfactor (sq_nonneg (x - center))
  nlinarith

/-- Every strictly stable step size converges to the unique quadratic
minimizer from every initial state. -/
theorem fep032_quadraticUpdate_tendsto
    {step : ℝ} (hstep0 : 0 < step) (hstep2 : step < 2)
    (center x : ℝ) :
    Filter.Tendsto
      (fun n => ((fep032_quadraticUpdate step center)^[n]) x)
      Filter.atTop (nhds center) := by
  have habs : |1 - step| < 1 := by
    rw [abs_lt]
    constructor <;> linarith
  have hpow : Filter.Tendsto (fun n : ℕ => (1 - step) ^ n)
      Filter.atTop (nhds 0) :=
    tendsto_pow_atTop_nhds_zero_of_abs_lt_one habs
  have hdeviation : Filter.Tendsto
      (fun n : ℕ =>
        ((fep032_quadraticUpdate step center)^[n]) x - center)
      Filter.atTop (nhds 0) := by
    simpa only [fep032_quadraticUpdate_iterate, zero_mul] using
      hpow.mul_const (x - center)
  simpa only [sub_add_cancel, zero_add] using hdeviation.add_const center

end FEP032
```

13.32.2 Typeset statement signatures

$$(\mathsf{step}\ \mathsf{center}\ x : \mathbb{R})$$
$$\mathsf{fep032\_quadraticUpdate}\ \mathsf{step}\ \mathsf{center}\ x - \mathsf{center} = (1 - \mathsf{step}) \cdot (x - \mathsf{center})$$

$$(\mathsf{step}\ \mathsf{center} : \mathbb{R})$$
$$\mathsf{fep032\_quadraticUpdate}\ \mathsf{step}\ \mathsf{center}\ \mathsf{center} = \mathsf{center}$$

$$(\mathsf{step}\ \mathsf{center}\ x : \mathbb{R})\ (n : \mathbb{N})$$
$$((\mathsf{fep032\_quadraticUpdate}\ \mathsf{step}\ \mathsf{center})^{\lfloor n \rfloor})\ x - \mathsf{center} = (1 - \mathsf{step})^n$$
$$\cdot\ (x - \mathsf{center})$$

$$\{\mathsf{step}\ \mathsf{center}\ x : \mathbb{R}\}\ (\mathsf{hstep0} : 0 \le \mathsf{step})\ (\mathsf{hstep2} : \mathsf{step} \le 2)$$
$$(\mathsf{fep032\_quadraticUpdate}\ \mathsf{step}\ \mathsf{center}\ x - \mathsf{center})^2 \le (x - \mathsf{center})^2$$

$$\{\mathsf{step} : \mathbb{R}\}\ (\mathsf{hstep0} : 0 < \mathsf{step})\ (\mathsf{hstep2} : \mathsf{step} < 2)\ (\mathsf{center}\ x : \mathbb{R})$$
$$\mathsf{Filter.Tendsto}\ (\lambda\, n \mapsto ((\mathsf{fep032\_quadraticUpdate}\ \mathsf{step}\ \mathsf{center})^{\lfloor n \rfloor})\ x)$$
$$\mathsf{Filter.atTop}\ (\mathsf{nhds}\ \mathsf{center})$$

13.33 fep-033 — Finite-Horizon Bellman Recursion

A deterministic transition-aware finite-horizon value obeys an exact Bellman recursion with stage cost, discount, and terminal cost; it is monotone in both cost functions and has exact zero-cost and zero-discount boundaries. Assumption scope: The transition is deterministic and the codomain is ENNReal. No stochastic policy optimization, discount-convergence limit, or infinite-horizon fixed point is claimed. Semantic disposition: `formalized`.

13.33.1 Lean sketch

```
import Mathlib.Data.ENNReal.Inv

namespace FEP033

open scoped ENNReal

/-- Transition-aware finite-horizon value with stage cost, discount, and
terminal cost. -/
noncomputable def fep033_value {State : Type*}
    (discount : ENNReal) (stageCost : State → ENNReal)
    (step : State → State) (terminalCost : State → ENNReal) :
    ℕ → State → ENNReal
  | 0, state => terminalCost state
  | horizon + 1, state =>
      stageCost state +
        discount * fep033_value discount stageCost step terminalCost horizon (step
 ↪  state)

/-- Bellman recursion is the defining finite-horizon transition law. -/
theorem fep033_bellman {State : Type*}
    (discount : ENNReal) (stageCost : State → ENNReal)
    (step : State → State) (terminalCost : State → ENNReal)
    (horizon : ℕ) (state : State) :
    fep033_value discount stageCost step terminalCost (horizon + 1) state =
      stageCost state + discount *
        fep033_value discount stageCost step terminalCost horizon (step state) :=
  rfl

/-- Pointwise larger stage and terminal costs produce a larger value. -/
theorem fep033_value_mono {State : Type*}
    (discount : ENNReal) {stage₁ stage₂ terminal₁ terminal₂ : State → ENNReal}
    (step : State → State) (hstage : ∀ state, stage₁ state ≤ stage₂ state)
    (hterminal : ∀ state, terminal₁ state ≤ terminal₂ state)
    (horizon : ℕ) (state : State) :
    fep033_value discount stage₁ step terminal₁ horizon state ≤
      fep033_value discount stage₂ step terminal₂ horizon state := by
  induction horizon generalizing state with
  | zero => exact hterminal state
  | succ horizon ih =>
```

```
      exact add_le_add (hstage state)
        (mul_le_mul_right (ih (step state)) discount)

/-- Zero stage and terminal costs yield zero value at every horizon. -/
theorem fep033_zeroCost {State : Type*}
    (discount : ENNReal) (step : State → State) (horizon : ℕ) (state : State) :
    fep033_value discount (fun _ => 0) step (fun _ => 0) horizon state = 0 := by
  induction horizon generalizing state with
  | zero => rfl
  | succ horizon ih => simp [fep033_value, ih]

/-- At zero discount, every positive-horizon value is exactly immediate cost. -/
theorem fep033_zeroDiscount {State : Type*}
    (stageCost : State → ENNReal) (step : State → State)
    (terminalCost : State → ENNReal) (horizon : ℕ) (state : State) :
    fep033_value 0 stageCost step terminalCost (horizon + 1) state =
      stageCost state := by
  simp [fep033_value]

end FEP033
```

13.33.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ (\mathsf{discount} : \mathbb{R}^\infty_{\geq 0})\ (\mathsf{stageCost} : \mathsf{State} \Rightarrow \mathbb{R}^\infty_{\geq 0})$$
$$(\mathsf{step} : \mathsf{State} \Rightarrow \mathsf{State})\ (\mathsf{terminalCost} : \mathsf{State} \Rightarrow \mathbb{R}^\infty_{\geq 0})\ (\mathsf{horizon} : \mathbb{N})$$
$$(\mathsf{state} : \mathsf{State})$$
$$\mathsf{fep033\_value}\ \mathsf{discount}\ \mathsf{stageCost}\ \mathsf{step}\ \mathsf{terminalCost}\ (\mathsf{horizon} + 1)\ \mathsf{state}$$
$$= \mathsf{stageCost}\ \mathsf{state} + \mathsf{discount} \cdot \mathsf{fep033\_value}\ \mathsf{discount}\ \mathsf{stageCost}\ \mathsf{step}\ \mathsf{terminalCost}$$
$$\mathsf{horizon}\ (\mathsf{step}\ \mathsf{state})$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ (\mathsf{discount} : \mathbb{R}^\infty_{\geq 0})$$
$$\{stage_1\ stage_2\ terminal_1\ terminal_2 : \mathsf{State} \Rightarrow \mathbb{R}^\infty_{\geq 0}\}\ (\mathsf{step} : \mathsf{State} \Rightarrow \mathsf{State})$$
$$(\mathsf{hstage} : \forall \mathsf{state}, stage_1\ \mathsf{state} \leq stage_2\ \mathsf{state})$$
$$(\mathsf{hterminal} : \forall \mathsf{state}, terminal_1\ \mathsf{state} \leq terminal_2\ \mathsf{state})\ (\mathsf{horizon} : \mathbb{N})$$
$$(\mathsf{state} : \mathsf{State})$$
$$\mathsf{fep033\_value}\ \mathsf{discount}\ stage_1\ \mathsf{step}\ terminal_1\ \mathsf{horizon}\ \mathsf{state} \leq \mathsf{fep033\_value}$$
$$\mathsf{discount}\ stage_2\ \mathsf{step}\ terminal_2\ \mathsf{horizon}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ (\mathsf{discount} : \mathbb{R}^\infty_{\geq 0})\ (\mathsf{step} : \mathsf{State} \Rightarrow \mathsf{State})\ (\mathsf{horizon} : \mathbb{N})$$
$$(\mathsf{state} : \mathsf{State})$$
$$\mathsf{fep033\_value}\ \mathsf{discount}\ (\lambda\ \_ \mapsto 0)\ \mathsf{step}\ (\lambda\ \_ \mapsto 0)\ \mathsf{horizon}\ \mathsf{state} = 0$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ (\mathsf{stageCost} : \mathsf{State} \Rightarrow \mathbb{R}^\infty_{\geq 0})\ (\mathsf{step} : \mathsf{State} \Rightarrow \mathsf{State})$$
$$(\mathsf{terminalCost} : \mathsf{State} \Rightarrow \mathbb{R}^\infty_{\geq 0})\ (\mathsf{horizon} : \mathbb{N})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{fep033\_value}\ 0\ \mathsf{stageCost}\ \mathsf{step}\ \mathsf{terminalCost}\ (\mathsf{horizon} + 1)\ \mathsf{state} = \mathsf{stageCost}\ \mathsf{state}$$

13.34 fep-034 — Normalized Bayesian Filtering

A transition-composed predictive prior and native observation posterior reconstruct the swapped predicted-state/observation joint law, with probability mass one in every posterior fibre. Assumption scope: Conditional on standard-Borel nonempty latent structure, a finite prior, and finite transition and observation kernels; prediction preserves total mass when the transition is Markov, and posterior recovery additionally requires a Markov observation kernel. This is a one-step exact filter, not a convergence or learned-state theorem. Semantic disposition: `formalized`.

13.34.1 Lean sketch

```
import Mathlib.Probability.Kernel.Posterior

namespace FEP034

open MeasureTheory ProbabilityTheory
open scoped ENNReal ProbabilityTheory

variable {Ω 𝒳 : Type*} [MeasurableSpace Ω] [MeasurableSpace 𝒳]

/-- The one-step predictive prior obtained by composing a transition kernel
with the previous latent-state law. -/
noncomputable def fep034_predictivePrior
    (τ : Kernel Ω Ω) (μ : Measure Ω) : Measure Ω :=
  τ ∘ₘ μ

/-- A Markov transition preserves total prior mass. -/
theorem fep034_predictive_mass
    (τ : Kernel Ω Ω) (μ : Measure Ω) [IsMarkovKernel τ] :
    fep034_predictivePrior τ μ Set.univ = μ Set.univ := by
  exact Measure.comp_apply_univ

/-- The normalized filtering kernel is the posterior of the observation
kernel with respect to the transition-predicted prior. -/
noncomputable def fep034_filter
    (τ : Kernel Ω Ω) (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel τ] [IsFiniteKernel κ] : Kernel 𝒳 Ω :=
  ProbabilityTheory.posterior κ (τ ∘ₘ μ)

/-- Every observation-indexed filtering posterior has total mass one. -/
theorem fep034_filter_mass_one
    (τ : Kernel Ω Ω) (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel τ] [IsFiniteKernel κ] (x : 𝒳) :
    fep034_filter τ κ μ x Set.univ = 1 := by
  change (ProbabilityTheory.posterior κ (τ ∘ₘ μ) x) Set.univ = 1
  exact measure_univ

/-- The predictive observation law and filtering posterior reconstruct the
swapped predicted-state/observation joint law. -/
theorem fep034_filter_joint_reconstruction
    (τ : Kernel Ω Ω) (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel τ] [IsFiniteKernel κ] :
    (κ ∘ₘ fep034_predictivePrior τ μ) ⊗ₘ fep034_filter τ κ μ =
      (fep034_predictivePrior τ μ ⊗ₘ κ).map Prod.swap := by
  change
    (κ ∘ₘ (τ ∘ₘ μ)) ⊗ₘ ProbabilityTheory.posterior κ (τ ∘ₘ μ) =
      ((τ ∘ₘ μ) ⊗ₘ κ).map Prod.swap
  exact ProbabilityTheory.compProd_posterior_eq_map_swap

/-- With a Markov observation kernel, conditioning after prediction recovers
the transition-predicted prior. -/
theorem fep034_filter_recovers_prediction
    (τ : Kernel Ω Ω) (κ : Kernel Ω 𝒳) (μ : Measure Ω)
    [StandardBorelSpace Ω] [Nonempty Ω]
    [IsFiniteMeasure μ] [IsFiniteKernel τ] [IsMarkovKernel κ] :
    fep034_filter τ κ μ ∘ₘ κ ∘ₘ fep034_predictivePrior τ μ =
      fep034_predictivePrior τ μ := by
  change
```

```
    ProbabilityTheory.posterior κ (τ ∘ₘ μ) ∘ₘ κ ∘ₘ (τ ∘ₘ μ) =
      τ ∘ₘ μ
  exact ProbabilityTheory.posterior_comp_self

end FEP034
```

13.34.2 Typeset statement signatures

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\, [\mathsf{MeasurableSpace}\Omega]\, [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\tau : \mathsf{Kernel}\, \Omega\, \Omega)\, (\mu : \mathsf{Measure}\Omega)\, [\mathsf{IsMarkovKernel}\, \tau]$$
$$\mathsf{fep034\_predictivePrior}\, \tau\, \mu\, \Omega = \mu\, \Omega$$

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\, [\mathsf{MeasurableSpace}\Omega]\, [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\tau : \mathsf{Kernel}\, \Omega\, \Omega)\, (\kappa : \mathsf{Kernel}\, \Omega\, \mathcal{X})\, (\mu : \mathsf{Measure}\Omega)\, [\mathsf{StandardBorelSpace}\, \Omega]$$
$$[\mathsf{Nonempty}\, \Omega]\, [\mathsf{IsFiniteMeasure}\, \mu]\, [\mathsf{IsFiniteKernel}\, \tau]\, [\mathsf{IsFiniteKernel}\, \kappa]\, (x : \mathcal{X})$$
$$\mathsf{fep034\_filter}\, \tau\, \kappa\, \mu\, x\, \Omega = 1$$

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\, [\mathsf{MeasurableSpace}\Omega]\, [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\tau : \mathsf{Kernel}\, \Omega\, \Omega)\, (\kappa : \mathsf{Kernel}\, \Omega\, \mathcal{X})\, (\mu : \mathsf{Measure}\Omega)\, [\mathsf{StandardBorelSpace}\, \Omega]$$
$$[\mathsf{Nonempty}\, \Omega]\, [\mathsf{IsFiniteMeasure}\, \mu]\, [\mathsf{IsFiniteKernel}\, \tau]\, [\mathsf{IsFiniteKernel}\, \kappa]$$
$$(\kappa \circ_m \mathsf{fep034\_predictivePrior}\, \tau\, \mu) \otimes_m \mathsf{fep034\_filter}\, \tau\, \kappa\, \mu$$
$$= (\mathsf{fep034\_predictivePrior}\, \tau\, \mu \otimes_m \kappa).\mathsf{map}\, \mathsf{Prod.swap}$$

$$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\, [\mathsf{MeasurableSpace}\Omega]\, [\mathsf{MeasurableSpace}\mathcal{X}]$$
$$(\tau : \mathsf{Kernel}\, \Omega\, \Omega)\, (\kappa : \mathsf{Kernel}\, \Omega\, \mathcal{X})\, (\mu : \mathsf{Measure}\Omega)\, [\mathsf{StandardBorelSpace}\, \Omega]$$
$$[\mathsf{Nonempty}\, \Omega]\, [\mathsf{IsFiniteMeasure}\, \mu]\, [\mathsf{IsFiniteKernel}\, \tau]\, [\mathsf{IsMarkovKernel}\, \kappa]$$
$$\mathsf{fep034\_filter}\, \tau\, \kappa\, \mu \circ_m \kappa \circ_m \mathsf{fep034\_predictivePrior}\, \tau\, \mu = \mathsf{fep034\_predictivePrior}$$
$$\tau\, \mu$$

13.35 fep-035 — Jensen's Inequality for Log

For distinct positive inputs and strictly positive weights summing to one, the weighted average of logarithms is strictly below the logarithm of the weighted average. Assumption scope: Conditional on x,y > 0, x ≠ y, a,b > 0, and a+b=1. This is the exact two-point strict Jensen law; finite-family and measure-integral variants are not claimed. Semantic disposition: `formalized`.

13.35.1 Lean sketch

```
import Mathlib.Analysis.Convex.SpecificFunctions.Basic

namespace FEP035

open Real

-- [proof strategy: Mathlib strict concavity plus logarithm identities]

/-- The real logarithm is strictly concave on the positive half-line. -/
theorem fep035_log_strictConcave :
    StrictConcaveOn ℝ (Set.Ioi 0) Real.log :=
  strictConcaveOn_log_Ioi

/-- Strict two-point Jensen inequality for distinct positive inputs and
strictly positive weights summing to one. -/
theorem fep035_log_jensen_two_strict
    {x y a b : ℝ} (hx : 0 < x) (hy : 0 < y) (hxy : x ≠ y)
    (ha : 0 < a) (hb : 0 < b) (hab : a + b = 1) :
    a * Real.log x + b * Real.log y < Real.log (a * x + b * y) := by
  simpa only [smul_eq_mul] using
    strictConcaveOn_log_Ioi.2 hx hy hxy ha hb hab
```

```
/-- Logarithm turns products into sums (``log(ab)=log a+log b``). -/
theorem fep035_log_mul {a b : ℝ} (ha : 0 < a) (hb : 0 < b) :
    Real.log (a * b) = Real.log a + Real.log b :=
  Real.log_mul (ne_of_gt ha) (ne_of_gt hb)

/-- Power rule for logarithms on ``(0,∞)``. -/
theorem fep035_log_pow {a : ℝ} (_ha : 0 < a) (n : ℕ) : Real.log (a ^ n) = n * Real.log
 ↪  a :=
  Real.log_pow a n

/-- Jensen anchor: log(exp x) = x (log-exp inverse). -/
theorem fep035_log_exp (x : ℝ) : Real.log (Real.exp x) = x :=
  Real.log_exp x

/-- Concavity anchor: log on (0,∞) is monotone. -/
theorem fep035_log_mono {a b : ℝ} (ha : 0 < a) (h : a ≤ b) :
    Real.log a ≤ Real.log b :=
  Real.log_le_log ha h

/-- log 1 = 0. -/
theorem fep035_log_one : Real.log (1 : ℝ) = 0 := Real.log_one

end FEP035
```

13.35.2 Typeset statement signatures

$$\mathsf{StrictConcaveOn}\,\mathbb{R}\,(\mathsf{Set.Ioi}\,0)\,\mathsf{Real.log}$$

$$\{x\,y\,a\,b : \mathbb{R}\}\,(\mathsf{hx} : 0 < x)\,(\mathsf{hy} : 0 < y)\,(\mathsf{hxy} : x \neq y)\,(\mathsf{ha} : 0 < a)\,(\mathsf{hb} : 0 < b)$$
$$(\mathsf{hab} : a + b = 1)$$
$$a \cdot \mathsf{Real.log}\,x + b \cdot \mathsf{Real.log}\,y < \mathsf{Real.log}\,(a \cdot x + b \cdot y)$$

$$\{a\,b : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)\,(\mathsf{hb} : 0 < b)$$
$$\mathsf{Real.log}\,(a \cdot b) = \mathsf{Real.log}\,a + \mathsf{Real.log}\,b$$

$$\{a : \mathbb{R}\}\,(_h a : 0 < a)\,(n : \mathbb{N})$$
$$\mathsf{Real.log}\,(a^n) = n \cdot \mathsf{Real.log}\,a$$

$$(x : \mathbb{R})$$
$$\mathsf{Real.log}\,(\mathsf{Real.exp}\,x) = x$$

$$\{a\,b : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)\,(h : a \leq b)$$
$$\mathsf{Real.log}\,a \leq \mathsf{Real.log}\,b$$

$$\mathsf{Real.log}\,(1 : \mathbb{R}) = 0$$

13.36 fep-036 — Laplace-Smoothed Empirical Bernoulli Prior

A Mathlib finite binomial sampling law supplies admissible success counts; Laplace smoothing maps every outcome to an interior Bernoulli prior, is monotone in successes, and preserves the limit of any convergent empirical-frequency sequence. Assumption scope: The finite sampling law requires p in [0,1]. Interiority uses the intrinsic count bound, and consistency transfer explicitly assumes convergence of the raw empirical frequency. The row does not prove the binomial law of large numbers, a finite-sample risk bound, or marginal-likelihood hyperparameter optimality. Semantic disposition: `formalized`.

13.36.1 Lean sketch

```
import Mathlib.Analysis.SpecificLimits.Basic
import Mathlib.Data.Nat.Cast.Field
import Mathlib.Probability.Distributions.Binomial
import Mathlib.Tactic

namespace FEP036

/-- Laplace-smoothed empirical success rate from `successes` of `trials`. -/
noncomputable def fep036_smoothedRate (successes trials : ℕ) : ℝ :=
  ((successes + 1 : ℕ) : ℝ) / ((trials + 2 : ℕ) : ℝ)

/-- Convert a bounded nonnegative parameter into Mathlib's unit interval. -/
noncomputable def fep036_unitInterval
    (p : NNReal) (hp : p ≤ 1) : unitInterval :=
  ⟨p, p.2, by exact_mod_cast hp⟩

/-- Binomial sampling measure for a cohort success count. -/
noncomputable def fep036_binomialModel
    (p : NNReal) (hp : p ≤ 1) (trials : ℕ) : MeasureTheory.Measure ℕ :=
  ProbabilityTheory.binomial trials (fep036_unitInterval p hp)

/-- Data-dependent Bernoulli prior obtained from one admissible cohort count. -/
noncomputable def fep036_empiricalPrior
    (trials : ℕ) (sample : Fin (trials + 1)) : ℝ :=
  fep036_smoothedRate sample trials

/-- The project sampling law is exactly Mathlib's binomial measure. -/
theorem fep036_binomialModel_apply (p : NNReal) (hp : p ≤ 1) (trials : ℕ)
    (sample : ℕ) :
    (fep036_binomialModel p hp trials).real {sample} =
      (trials.choose sample : ℝ) * (p : ℝ) ^ sample *
        (1 - (p : ℝ)) ^ (trials - sample) := by
  simpa [fep036_binomialModel, fep036_unitInterval] using
    ProbabilityTheory.binomial_real_singleton trials sample
      (fep036_unitInterval p hp)

/-- Laplace smoothing makes the empirical prior parameter strictly positive. -/
theorem fep036_smoothedRate_pos (successes trials : ℕ) :
    0 < fep036_smoothedRate successes trials := by
  simp only [fep036_smoothedRate]
  positivity

/-- A valid success count yields a smoothed parameter strictly below one. -/
theorem fep036_smoothedRate_lt_one
    {successes trials : ℕ} (h : successes ≤ trials) :
    fep036_smoothedRate successes trials < 1 := by
  have hden : (0 : ℝ) < (trials + 2 : ℕ) := by positivity
  apply (div_lt_one hden).2
  exact_mod_cast (show successes + 1 < trials + 2 by omega)

/-- The smoothed empirical prior lies in the interior probability interval. -/
theorem fep036_smoothedRate_mem_Ioo
    {successes trials : ℕ} (h : successes ≤ trials) :
    fep036_smoothedRate successes trials ∈ Set.Ioo (0 : ℝ) 1 :=
  ⟨fep036_smoothedRate_pos successes trials,
    fep036_smoothedRate_lt_one h⟩

/-- Every possible binomial cohort outcome produces an interior prior. -/
theorem fep036_empiricalPrior_mem_Ioo
    (trials : ℕ) (sample : Fin (trials + 1)) :
```

```
    fep036_empiricalPrior trials sample ∈ Set.Ioo (0 : ℝ) 1 := by
  apply fep036_smoothedRate_mem_Ioo
  exact Nat.le_of_lt_succ sample.isLt

/-- At fixed trial count, the smoothed empirical rate is monotone in successes. -/
theorem fep036_smoothedRate_mono
    {s₁ s₂ trials : ℕ} (h : s₁ ≤ s₂) :
    fep036_smoothedRate s₁ trials ≤ fep036_smoothedRate s₂ trials := by
  simp only [fep036_smoothedRate]
  gcongr

/-- Laplace smoothing is the raw empirical rate multiplied by a shrinkage
factor, plus a vanishing pseudocount offset. -/
theorem fep036_smoothedRate_eq_shrunkEmpirical
    {successes trials : ℕ} (htrials : 0 < trials) :
    fep036_smoothedRate successes trials =
      ((successes : ℝ) / trials) * ((trials : ℝ) / (trials + 2)) +
        1 / ((trials : ℝ) + 2) := by
  have hn : (trials : ℝ) ≠ 0 :=
    Nat.cast_ne_zero.mpr (Nat.ne_of_gt htrials)
  simp only [fep036_smoothedRate, Nat.cast_add, Nat.cast_one, Nat.cast_ofNat]
  field_simp [hn]

/-- Any consistent empirical-frequency sequence remains consistent after
Laplace smoothing.  The sampling-law convergence premise is explicit. -/
theorem fep036_smoothedRate_tendsto_of_empiricalRate
    (successes : ℕ → ℕ) (p : ℝ)
    (hraw : Filter.Tendsto (fun n : ℕ => (successes n : ℝ) / n)
      Filter.atTop (nhds p)) :
    Filter.Tendsto (fun n : ℕ => fep036_smoothedRate (successes n) n)
      Filter.atTop (nhds p) := by
  have hshrink : Filter.Tendsto (fun n : ℕ => (n : ℝ) / (n + 2))
      Filter.atTop (nhds 1) :=
    tendsto_natCast_div_add_atTop (2 : ℝ)
  have hoffset : Filter.Tendsto (fun n : ℕ => (1 : ℝ) / ((n : ℝ) + 2))
      Filter.atTop (nhds 0) := by
    simpa [add_comm, div_eq_mul_inv] using
      (tendsto_add_mul_div_add_mul_atTop_nhds
        (1 : ℝ) 2 0 (d := 1) one_ne_zero)
  have hcombined := (hraw.mul hshrink).add hoffset
  have hcombined' : Filter.Tendsto
      (fun n : ℕ => ((successes n : ℝ) / n) * ((n : ℝ) / (n + 2)) +
        1 / ((n : ℝ) + 2)) Filter.atTop (nhds p) := by
    simpa using hcombined
  refine hcombined'.congr' ?_
  filter_upwards [Filter.eventually_atTop.2 ⟨1, fun _ hn => hn⟩] with n hn
  exact
    (fep036_smoothedRate_eq_shrunkEmpirical (Nat.zero_lt_of_lt hn)).symm

end FEP036
```

13.36.2 Typeset statement signatures

$$(p : \mathbb{R}_{\geq 0})\,(\mathsf{hp} : p \leq 1)\,(\mathsf{trials} : \mathbb{N})\,(\mathsf{sample} : \mathbb{N})$$
$$(\mathsf{fep036\_binomialModel}\,p\,\mathsf{hp}\,\mathsf{trials}).\mathsf{real}\,\{\mathsf{sample}\} = (\mathsf{trials.choose}\,\mathsf{sample} : \mathbb{R})$$
$$\cdot\,(p : \mathbb{R})^{\mathsf{sample}} \cdot (1 - (p : \mathbb{R}))^{(\mathsf{trials} - \mathsf{sample})}$$

$$(\mathsf{successes}\,\mathsf{trials} : \mathbb{N})$$
$$0 < \mathsf{fep036\_smoothedRate}\,\mathsf{successes}\,\mathsf{trials}$$

$$\{\text{successes trials} : \mathbb{N}\}\ (h : \text{successes} \le \text{trials})$$
$$\text{fep036\_smoothedRate successes trials} < 1$$

$$\{\text{successes trials} : \mathbb{N}\}\ (h : \text{successes} \le \text{trials})$$
$$\text{fep036\_smoothedRate successes trials} \in \text{Set.Ioo}\ (0 : \mathbb{R})\ 1$$

$$(\text{trials} : \mathbb{N})\ (\text{sample} : \text{Fin}\ (\text{trials} + 1))$$
$$\text{fep036\_empiricalPrior trials sample} \in \text{Set.Ioo}\ (0 : \mathbb{R})\ 1$$

$$\{s_1\ s_2\ \text{trials} : \mathbb{N}\}\ (h : s_1 \le s_2)$$
$$\text{fep036\_smoothedRate}\ s_1\ \text{trials} \le \text{fep036\_smoothedRate}\ s_2\ \text{trials}$$

$$\{\text{successes trials} : \mathbb{N}\}\ (\text{htrials} : 0 < \text{trials})$$
$$\text{fep036\_smoothedRate successes trials} = ((\text{successes} : \mathbb{R})/\text{trials})$$
$$\cdot\ ((\text{trials} : \mathbb{R})/(\text{trials} + 2)) + 1/((\text{trials} : \mathbb{R}) + 2)$$

$$(\text{successes} : \mathbb{N} \Rightarrow \mathbb{N})\ (p : \mathbb{R})\ (\text{hraw} : \text{Filter.Tendsto}\ (\lambda\, n : \mathbb{N} \mapsto (\text{successes}\, n :$$
$$\mathbb{R})/n)\ \text{Filter.atTop}\ (\text{nhds}\, p))$$
$$\text{Filter.Tendsto}\ (\lambda\, n : \mathbb{N} \mapsto \text{fep036\_smoothedRate}\ (\text{successes}\, n)\, n)\ \text{Filter.atTop}$$
$$(\text{nhds}\, p)$$

13.37 fep-037 — Two-State Correlation–Response Law

For the symmetric two-state relaxation eigenvalue 1-2α, stationary autocorrelation obeys an exact recurrence and decays for 0<α<1; the discrete fluctuation–response kernel equals 2βα variance times the same geometric mode and also decays. Assumption scope: Response is defined as inverse temperature times the one-step autocorrelation loss. Nonnegativity additionally restricts α≤1/2 and nonnegative inverse temperature and variance. This is an exact discrete two-state response law, not a microscopic Kubo derivation for general dynamics. Semantic disposition: `formalized`.

13.37.1 Lean sketch

```
import Mathlib.Analysis.SpecificLimits.Normed
import Mathlib.Tactic

namespace FEP037

/-- Relaxation eigenvalue of the symmetric two-state Markov transition. -/
def fep037_relaxation (α : ℝ) : ℝ := 1 - 2 * α

/-- Stationary autocorrelation at lag `n` with equilibrium variance `variance`. -/
def fep037_autocorrelation (α variance : ℝ) (n : ℕ) : ℝ :=
  variance * fep037_relaxation α ^ n

/-- The autocorrelation obeys the Markov relaxation recurrence. -/
theorem fep037_autocorrelation_recurrence (α variance : ℝ) (n : ℕ) :
    fep037_autocorrelation α variance (n + 1) =
      fep037_relaxation α * fep037_autocorrelation α variance n := by
  simp only [fep037_autocorrelation, pow_succ]
  ring

/-- For a strictly interior switching probability, equilibrium correlations
decay to zero. -/
theorem fep037_autocorrelation_tendsto_zero
    {α : ℝ} (hα0 : 0 < α) (hα1 : α < 1) (variance : ℝ) :
    Filter.Tendsto (fep037_autocorrelation α variance)
      Filter.atTop (nhds 0) := by
  have habs : |fep037_relaxation α| < 1 := by
    rw [fep037_relaxation, abs_lt]
```

```
    constructor <;> linarith
  have hpow : Filter.Tendsto (fun n : ℕ => fep037_relaxation α ^ n)
      Filter.atTop (nhds 0) :=
    tendsto_pow_atTop_nhds_zero_of_abs_lt_one habs
  change Filter.Tendsto (fun n : ℕ => variance * fep037_relaxation α ^ n)
    Filter.atTop (nhds 0)
  simpa only [mul_zero] using hpow.const_mul variance

/-- Discrete fluctuation--response law: inverse temperature times the
one-step loss of stationary autocorrelation. -/
def fep037_response (β α variance : ℝ) (n : ℕ) : ℝ :=
  β * (fep037_autocorrelation α variance n -
    fep037_autocorrelation α variance (n + 1))

/-- Closed form of the discrete response kernel. -/
theorem fep037_response_eq (β α variance : ℝ) (n : ℕ) :
    fep037_response β α variance n =
      2 * β * α * variance * fep037_relaxation α ^ n := by
  rw [fep037_response, fep037_autocorrelation_recurrence]
  simp only [fep037_autocorrelation, fep037_relaxation]
  ring

/-- For nonnegative inverse temperature and variance, and switching no faster
than one half, the response kernel is nonnegative. -/
theorem fep037_response_nonneg
    {β α variance : ℝ} (hβ : 0 ≤ β) (hα0 : 0 ≤ α) (hαhalf : α ≤ 1 / 2)
    (hvariance : 0 ≤ variance) (n : ℕ) :
    0 ≤ fep037_response β α variance n := by
  rw [fep037_response_eq]
  have hrelax : 0 ≤ fep037_relaxation α := by
    rw [fep037_relaxation]
    linarith
  positivity

/-- The discrete response also relaxes to zero. -/
theorem fep037_response_tendsto_zero
    {α : ℝ} (hα0 : 0 < α) (hα1 : α < 1) (β variance : ℝ) :
    Filter.Tendsto (fun n => fep037_response β α variance n)
      Filter.atTop (nhds 0) := by
  have habs : |fep037_relaxation α| < 1 := by
    rw [fep037_relaxation, abs_lt]
    constructor <;> linarith
  have hpow : Filter.Tendsto (fun n : ℕ => fep037_relaxation α ^ n)
      Filter.atTop (nhds 0) :=
    tendsto_pow_atTop_nhds_zero_of_abs_lt_one habs
  simpa only [fep037_response_eq, mul_zero] using
    hpow.const_mul (2 * β * α * variance)

end FEP037
```

13.37.2 Typeset statement signatures

$$(\alpha\ \mathsf{variance} : \mathbb{R})\ (n : \mathbb{N})$$
$$\mathsf{fep037\_autocorrelation}\ \alpha\ \mathsf{variance}\ (n+1) = \mathsf{fep037\_relaxation}\ \alpha \cdot \mathsf{fep037\_autocorrelation}\ \alpha\ \mathsf{variance}\ n$$

$$\{\alpha : \mathbb{R}\}\ (h\alpha 0 : 0 < \alpha)\ (h\alpha 1 : \alpha < 1)\ (\mathsf{variance} : \mathbb{R})$$
$$\mathsf{Filter.Tendsto}\ (\mathsf{fep037\_autocorrelation}\ \alpha\ \mathsf{variance})\ \mathsf{Filter.atTop}\ (\mathsf{nhds}\ 0)$$

$$(\beta\ \alpha\ \mathsf{variance} : \mathbb{R})\ (n : \mathbb{N})$$
$$\mathsf{fep037\_response}\ \beta\ \alpha\ \mathsf{variance}\ n = 2 \cdot \beta \cdot \alpha \cdot \mathsf{variance} \cdot \mathsf{fep037\_relaxation}\ \alpha^n$$

$$\{\beta\,\alpha\,\mathsf{variance} : \mathbb{R}\}\,(h\beta : 0 \le \beta)\,(h\alpha 0 : 0 \le \alpha)\,(h\alpha half : \alpha \le 1/2)$$
$$(\mathsf{hvariance} : 0 \le \mathsf{variance})\,(n : \mathbb{N})$$
$$0 \le \mathsf{fep037\_response}\,\beta\,\alpha\,\mathsf{variance}\,n$$

$$\{\alpha : \mathbb{R}\}\,(h\alpha 0 : 0 < \alpha)\,(h\alpha 1 : \alpha < 1)\,(\beta\,\mathsf{variance} : \mathbb{R})$$
$$\mathsf{Filter.Tendsto}\,(\lambda\, n \mapsto \mathsf{fep037\_response}\,\beta\,\alpha\,\mathsf{variance}\,n)\,\mathsf{Filter.atTop}\,(\mathsf{nhds}\,0)$$

13.38 fep-038 — Bernoulli Fisher Information and Natural Gradient

The differentiated Bernoulli family has zero expected score and Fisher information 1/(p(1-p)); its inverse metric defines a dual natural gradient and is the pullback of the Euclidean metric along the Fisher–Rao coordinate. Assumption scope: Metric positivity, inverse-metric duality, and the coordinate law require an interior Bernoulli parameter 0 < p < 1. This is a complete one-dimensional Bernoulli information geometry, not a general finite- or infinite-dimensional statistical manifold. Semantic disposition: `formalized`.

13.38.1 Lean sketch

```
import Mathlib.Analysis.Calculus.Deriv.Add
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.Data.Bool.Basic
import Mathlib.Analysis.Real.Sqrt
import Mathlib.Tactic

namespace FEP038

/-- Bernoulli probability mass with success parameter `p`. -/
def fep038_bernoulliMass (p : ℝ) : Bool → ℝ
  | false => 1 - p
  | true => p

/-- Pointwise parameter derivative of the Bernoulli mass. -/
def fep038_bernoulliMassDeriv : Bool → ℝ
  | false => -1
  | true => 1

/-- The Bernoulli family is differentiable in its scalar parameter. -/
theorem fep038_bernoulliMass_hasDerivAt (p : ℝ) (b : Bool) :
    HasDerivAt (fun q => fep038_bernoulliMass q b)
      (fep038_bernoulliMassDeriv b) p := by
  cases b
  · convert (hasDerivAt_const p (1 : ℝ)).sub (hasDerivAt_id p) using 1
    all_goals
      first
      | exact AddCommGroup.ext rfl
      | exact Module.ext rfl
      | rfl
      | norm_num [fep038_bernoulliMass, fep038_bernoulliMassDeriv]
  · convert hasDerivAt_id p using 1
    all_goals rfl

/-- Bernoulli masses are normalized for every scalar parameter. Their
probabilistic interpretation additionally restricts `p` to `[0,1]`. -/
theorem fep038_bernoulliMass_sum_one (p : ℝ) :
    ∑ b : Bool, fep038_bernoulliMass p b = 1 := by
  simp [fep038_bernoulliMass]

/-- Bernoulli score `∂ₚ log Pₚ(b)`, represented as `(∂ₚ Pₚ(b)) / Pₚ(b)`. -/
noncomputable def fep038_score (p : ℝ) (b : Bool) : ℝ :=
  fep038_bernoulliMassDeriv b / fep038_bernoulliMass p b

/-- At an interior parameter the expected Bernoulli score vanishes. -/
```

```
theorem fep038_expectedScore_zero {p : ℝ} (hp0 : 0 < p) (hp1 : p < 1) :
    ∑ b : Bool, fep038_bernoulliMass p b * fep038_score p b = 0 := by
  have hp : p ≠ 0 := ne_of_gt hp0
  have h1p : 1 - p ≠ 0 := ne_of_gt (sub_pos.mpr hp1)
  simp [fep038_bernoulliMass, fep038_score, fep038_bernoulliMassDeriv, hp]
  field_simp [h1p]
  ring

/-- Fisher information is the expected squared score. -/
noncomputable def fep038_fisherInformation (p : ℝ) : ℝ :=
  ∑ b : Bool,
    fep038_bernoulliMass p b * fep038_score p b ^ 2

/-- The Bernoulli Fisher information is `1 / (p(1-p))` in the interior. -/
theorem fep038_fisherInformation_eq {p : ℝ} (hp0 : 0 < p) (hp1 : p < 1) :
    fep038_fisherInformation p = 1 / (p * (1 - p)) := by
  have hp : p ≠ 0 := ne_of_gt hp0
  have h1p : 1 - p ≠ 0 := ne_of_gt (sub_pos.mpr hp1)
  simp [fep038_fisherInformation, fep038_bernoulliMass, fep038_score,
    fep038_bernoulliMassDeriv]
  field_simp [hp, h1p]
  ring

/-- The one-dimensional Fisher metric applied to tangent coordinates. -/
noncomputable def fep038_fisherMetric (p v w : ℝ) : ℝ :=
  fep038_fisherInformation p * v * w

/-- The Bernoulli Fisher metric is positive definite in the interior. -/
theorem fep038_fisherMetric_pos {p v : ℝ} (hp0 : 0 < p) (hp1 : p < 1)
    (hv : v ≠ 0) :
    0 < fep038_fisherMetric p v v := by
  rw [fep038_fisherMetric, fep038_fisherInformation_eq hp0 hp1]
  have hden : 0 < p * (1 - p) := mul_pos hp0 (sub_pos.mpr hp1)
  have hmetric : 0 < (1 / (p * (1 - p))) * (v * v) :=
    mul_pos (one_div_pos.mpr hden) (mul_self_pos.mpr hv)
  nlinarith

/-- Inverse-metric natural gradient for the Bernoulli parameter. -/
def fep038_naturalGradient (p gradient : ℝ) : ℝ :=
  p * (1 - p) * gradient

/-- Applying the Fisher metric to its natural gradient recovers the covector. -/
theorem fep038_naturalGradient_duality {p gradient : ℝ}
    (hp0 : 0 < p) (hp1 : p < 1) :
    fep038_fisherInformation p * fep038_naturalGradient p gradient = gradient := by
  rw [fep038_fisherInformation_eq hp0 hp1]
  have hp : p ≠ 0 := ne_of_gt hp0
  have h1p : 1 - p ≠ 0 := ne_of_gt (sub_pos.mpr hp1)
  simp only [fep038_naturalGradient]
  field_simp [hp, h1p]

/-- Jacobian of the Bernoulli Fisher--Rao variance-stabilizing coordinate. -/
noncomputable def fep038_coordinateJacobian (p : ℝ) : ℝ :=
  1 / Real.sqrt (p * (1 - p))

/-- The Fisher information is the square of the Fisher--Rao coordinate
Jacobian, i.e. the metric is the pullback of the Euclidean metric. -/
theorem fep038_fisherMetric_coordinate {p : ℝ} (hp0 : 0 < p) (hp1 : p < 1) :
    fep038_coordinateJacobian p ^ 2 = fep038_fisherInformation p := by
  rw [fep038_fisherInformation_eq hp0 hp1]
  have hprod : 0 ≤ p * (1 - p) := (mul_pos hp0 (sub_pos.mpr hp1)).le
  simp [fep038_coordinateJacobian, Real.sq_sqrt hprod]
```

```
/-- Tangent-vector form of the coordinate pullback law. -/
theorem fep038_fisherMetric_pullback {p v w : ℝ}
    (hp0 : 0 < p) (hp1 : p < 1) :
    fep038_fisherMetric p v w =
      (fep038_coordinateJacobian p * v) *
        (fep038_coordinateJacobian p * w) := by
  rw [fep038_fisherMetric]
  have hcoord := fep038_fisherMetric_coordinate hp0 hp1
  rw [← hcoord]
  ring

end FEP038
```

13.38.2 Typeset statement signatures

$$(p : \mathbb{R})\ (b : \mathsf{Bool})$$
$$\mathsf{HasDerivAt}\ (\lambda\, q \mapsto \mathsf{fep038\_bernoulliMass}\, q\, b)\ (\mathsf{fep038\_bernoulliMassDeriv}\, b)\ p$$

$$(p : \mathbb{R})$$
$$\sum b : \mathsf{Bool}, \mathsf{fep038\_bernoulliMass}\, p\, b = 1$$

$$\{p : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$$
$$\sum b : \mathsf{Bool}, \mathsf{fep038\_bernoulliMass}\, p\, b \cdot \mathsf{fep038\_score}\, p\, b = 0$$

$$\{p : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$$
$$\mathsf{fep038\_fisherInformation}\, p = 1/(p \cdot (1 - p))$$

$$\{p\, v : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)\ (\mathsf{hv} : v \neq 0)$$
$$0 < \mathsf{fep038\_fisherMetric}\, p\, v\, v$$

$$\{p\, \mathsf{gradient} : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$$
$$\mathsf{fep038\_fisherInformation}\, p \cdot \mathsf{fep038\_naturalGradient}\, p\, \mathsf{gradient} = \mathsf{gradient}$$

$$\{p : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$$
$$\mathsf{fep038\_coordinateJacobian}\, p^2 = \mathsf{fep038\_fisherInformation}\, p$$

$$\{p\, v\, w : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$$
$$\mathsf{fep038\_fisherMetric}\, p\, v\, w = (\mathsf{fep038\_coordinateJacobian}\, p \cdot v)$$
$$\cdot\ (\mathsf{fep038\_coordinateJacobian}\, p \cdot w)$$

13.39 fep-039 — Four-Block Additive Free Energy

Four local free-energy contributions aggregate additively and monotonically; under local nonnegativity, their global sum vanishes exactly when every local term vanishes. Assumption scope: Additivity is unconditional over four real components. Zero separation requires each local term to be nonnegative. The composed partition theorem ties those four components to fep-005's exact state blocks. Semantic disposition: `formalized`.

13.39.1 Lean sketch

```
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.Algebra.Order.BigOperators.Group.Finset
import Mathlib.Data.Real.Basic

namespace FEP039

open Finset
```

```
-- [proof strategy: Finset.sum_nonneg + sum_le_sum + sum_const for global-vs-local FE]

/-- Global free energy: Σ local contributions over a partition. -/
def fep039_global_fe (local_fe : Fin 4 → ℝ) : ℝ :=
  Σ i : Fin 4, local_fe i

/-- Global FE is nonneg when all local contributions are nonneg. -/
theorem fep039_global_nonneg (local_fe : Fin 4 → ℝ) (h : ∀ i, 0 ≤ local_fe i) :
    0 ≤ fep039_global_fe local_fe :=
  Finset.sum_nonneg fun i _ => h i

/-- Global FE monotone: improving one local term improves the global. -/
theorem fep039_global_mono (f g : Fin 4 → ℝ) (h : ∀ i, f i ≤ g i) :
    fep039_global_fe f ≤ fep039_global_fe g :=
  Finset.sum_le_sum fun i _ => h i

/-- Zero local FE everywhere → zero global FE. -/
theorem fep039_global_zero : fep039_global_fe (fun _ => 0) = 0 := by
  simp [fep039_global_fe]

/-- Global FE additive in pointwise sums. -/
theorem fep039_global_add (f g : Fin 4 → ℝ) :
    fep039_global_fe (fun i => f i + g i) = fep039_global_fe f + fep039_global_fe g :=
 ↪ by
  simp [fep039_global_fe, Finset.sum_add_distrib]

/-- For nonnegative local energies, global free energy vanishes exactly when
every local contribution vanishes. -/
theorem fep039_global_eq_zero_iff (local_fe : Fin 4 → ℝ)
    (hnonneg : ∀ i, 0 ≤ local_fe i) :
    fep039_global_fe local_fe = 0 ↔ ∀ i, local_fe i = 0 := by
  constructor
  · intro hzero i
    exact (Finset.sum_eq_zero_iff_of_nonneg
      (fun j _ => hnonneg j)).mp hzero i (Finset.mem_univ i)
  · intro hlocal
    simp [fep039_global_fe, hlocal]

end FEP039
```

13.39.2 Typeset statement signatures

$$(\mathsf{local\_fe} : \mathsf{Fin}\,4 \Rightarrow \mathbb{R})\,(h : \forall i, 0 \le \mathsf{local\_fe}\,i)$$
$$0 \le \mathsf{fep039\_global\_fe}\,\mathsf{local\_fe}$$

$$(f\,g : \mathsf{Fin}\,4 \Rightarrow \mathbb{R})\,(h : \forall i, f\,i \le g\,i)$$
$$\mathsf{fep039\_global\_fe}\,f \le \mathsf{fep039\_global\_fe}\,g$$

$$\mathsf{fep039\_global\_fe}\,(\lambda \_ \mapsto 0) = 0$$

$$(f\,g : \mathsf{Fin}\,4 \Rightarrow \mathbb{R})$$
$$\mathsf{fep039\_global\_fe}\,(\lambda\,i \mapsto f\,i + g\,i) = \mathsf{fep039\_global\_fe}\,f + \mathsf{fep039\_global\_fe}\,g$$

$$(\mathsf{local\_fe} : \mathsf{Fin}\,4 \Rightarrow \mathbb{R})\,(\mathsf{hnonneg} : \forall i, 0 \le \mathsf{local\_fe}\,i)$$
$$\mathsf{fep039\_global\_fe}\,\mathsf{local\_fe} = 0 \leftrightarrow \forall i, \mathsf{local\_fe}\,i = 0$$

13.40 fep-040 — Native Gaussian Law and Thermal Entropy Response

Mathlib's real Gaussian law is normalized with its stated mean and variance; the one-dimensional closed-form entropy is strictly variance-monotone, differentiable with derivative 1/(2v), and under v=κT has derivative 1/(2T) and heat capacity 1/2. Assumption scope: The native probability, mean, and variance theorems include the zero-variance Dirac boundary. Entropy monotonicity and derivatives require positive variance, and the thermal relation requires κ,T>0. The closed-form entropy is specified directly rather than rederived here as an integral of negative log density. Semantic disposition: `formalized`.

13.40.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.Log.Deriv
import Mathlib.Probability.Distributions.Gaussian.Real
import Mathlib.Tactic

namespace FEP040

open MeasureTheory ProbabilityTheory

/-- Mathlib's normalized real Gaussian law. -/
noncomputable def fep040_gaussianLaw (mean : ℝ) (variance : NNReal) : Measure ℝ :=
  ProbabilityTheory.gaussianReal mean variance

/-- Every Gaussian law, including the zero-variance Dirac boundary, has mass one. -/
theorem fep040_gaussian_mass_one (mean : ℝ) (variance : NNReal) :
    fep040_gaussianLaw mean variance Set.univ = 1 := by
  change ProbabilityTheory.gaussianReal mean variance Set.univ = 1
  exact measure_univ

/-- The native Gaussian's expectation is its mean parameter. -/
theorem fep040_gaussian_mean (mean : ℝ) (variance : NNReal) :
    ∫ x, x ∂fep040_gaussianLaw mean variance = mean := by
  change ∫ x, x ∂ProbabilityTheory.gaussianReal mean variance = mean
  exact ProbabilityTheory.integral_id_gaussianReal

/-- The native Gaussian's variance is its variance parameter. -/
theorem fep040_gaussian_variance (mean : ℝ) (variance : NNReal) :
    ProbabilityTheory.variance id (fep040_gaussianLaw mean variance) = variance := by
  change ProbabilityTheory.variance id
    (ProbabilityTheory.gaussianReal mean variance) = variance
  exact ProbabilityTheory.variance_id_gaussianReal

/-- Closed-form differential entropy of a one-dimensional Gaussian with
positive variance `v`. -/
noncomputable def fep040_gaussianEntropy (v : ℝ) : ℝ :=
  (1 / 2 : ℝ) * Real.log (2 * Real.pi * Real.exp 1 * v)

/-- Gaussian entropy is strictly increasing with positive variance. -/
theorem fep040_gaussianEntropy_strictMono
    {v₁ v₂ : ℝ} (hv₁ : 0 < v₁) (hv : v₁ < v₂) :
    fep040_gaussianEntropy v₁ < fep040_gaussianEntropy v₂ := by
  unfold fep040_gaussianEntropy
  have hconstant : 0 < 2 * Real.pi * Real.exp 1 := by positivity
  gcongr

/-- The entropy derivative with respect to variance is `1/(2v)`. -/
theorem fep040_gaussianEntropy_hasDerivAt
    {v : ℝ} (hv : 0 < v) :
    HasDerivAt fep040_gaussianEntropy (1 / (2 * v)) v := by
  let constant : ℝ := 2 * Real.pi * Real.exp 1
  have hconstant : constant ≠ 0 := by
    dsimp [constant]
    positivity
```

```
  have hargument : constant * v ≠ 0 := mul_ne_zero hconstant (ne_of_gt hv)
  have hlog : HasDerivAt (fun x => Real.log (constant * x))
      (constant / (constant * v)) v := by
    simpa using ((hasDerivAt_id v).const_mul constant).log hargument
  have hscaled := hlog.const_mul (1 / 2 : ℝ)
  have hcoefficient :
      (1 / 2 : ℝ) * (constant / (constant * v)) = 1 / (2 * v) := by
    field_simp [hconstant, ne_of_gt hv]
  rw [← hcoefficient]
  convert hscaled using 1
  all_goals
    first
    | exact AddCommGroup.ext rfl
    | exact Module.ext rfl
    | rfl

/-- Thermal variance scale `κT`. -/
def fep040_thermalVariance (κ T : ℝ) : ℝ := κ * T

/-- Gaussian entropy as a function of absolute temperature. -/
noncomputable def fep040_thermalEntropy (κ T : ℝ) : ℝ :=
  fep040_gaussianEntropy (fep040_thermalVariance κ T)

/-- For positive scale and temperature, thermal entropy has derivative `1/(2T)`. -/
theorem fep040_thermalEntropy_hasDerivAt
    {κ T : ℝ} (hκ : 0 < κ) (hT : 0 < T) :
    HasDerivAt (fep040_thermalEntropy κ) (1 / (2 * T)) T := by
  have hvariance : 0 < fep040_thermalVariance κ T :=
    mul_pos hκ hT
  have houter := fep040_gaussianEntropy_hasDerivAt hvariance
  have hinner : HasDerivAt (fep040_thermalVariance κ) κ T := by
    convert (hasDerivAt_id T).const_mul κ using 1
    all_goals
      first
      | exact AddCommGroup.ext rfl
      | exact Module.ext rfl
      | rfl
      | simp
  have hcomp := houter.comp T hinner
  have hcoefficient :
      (1 / (2 * fep040_thermalVariance κ T)) * κ = 1 / (2 * T) := by
    simp only [fep040_thermalVariance]
    field_simp [ne_of_gt hκ, ne_of_gt hT]
  rw [← hcoefficient]
  convert hcomp using 1
  all_goals
    first
    | exact AddCommGroup.ext rfl
    | exact Module.ext rfl
    | rfl

/-- Dimensionless constant-volume heat capacity `T ∂H/∂T`. -/
noncomputable def fep040_heatCapacity (T : ℝ) : ℝ :=
  T * (1 / (2 * T))

/-- The one-dimensional Gaussian thermal model has heat capacity `1/2`. -/
theorem fep040_heatCapacity_eq_half {T : ℝ} (hT : 0 < T) :
    fep040_heatCapacity T = 1 / 2 := by
  unfold fep040_heatCapacity
  field_simp [ne_of_gt hT]

end FEP040
```

13.40.2 Typeset statement signatures

$$(\mathsf{mean} : \mathbb{R})\,(\mathsf{variance} : \mathbb{R}_{\geq 0})$$
$$\mathsf{fep040\_gaussianLaw}\ \mathsf{mean}\ \mathsf{variance}\ \Omega = 1$$

$$(\mathsf{mean} : \mathbb{R})\,(\mathsf{variance} : \mathbb{R}_{\geq 0})$$
$$\int x, x\ \mathrm{d}\ \mathsf{fep040\_gaussianLaw}\ \mathsf{mean}\ \mathsf{variance} = \mathsf{mean}$$

$$(\mathsf{mean} : \mathbb{R})\,(\mathsf{variance} : \mathbb{R}_{\geq 0})$$
$$\mathsf{ProbabilityTheory.variance}\ \mathsf{id}\ (\mathsf{fep040\_gaussianLaw}\ \mathsf{mean}\ \mathsf{variance}) = \mathsf{variance}$$

$$\{v_1\ v_2 : \mathbb{R}\}\,(hv_1 : 0 < v_1)\,(\mathsf{hv} : v_1 < v_2)$$
$$\mathsf{fep040\_gaussianEntropy}\ v_1 < \mathsf{fep040\_gaussianEntropy}\ v_2$$

$$\{v : \mathbb{R}\}\,(\mathsf{hv} : 0 < v)$$
$$\mathsf{HasDerivAt}\ \mathsf{fep040\_gaussianEntropy}\ (1/(2 \cdot v))\ v$$

$$\{\kappa\ T : \mathbb{R}\}\,(h\kappa : 0 < \kappa)\,(\mathsf{hT} : 0 < T)$$
$$\mathsf{HasDerivAt}\ (\mathsf{fep040\_thermalEntropy}\ \kappa)\ (1/(2 \cdot T))\ T$$

$$\{T : \mathbb{R}\}\,(\mathsf{hT} : 0 < T)$$
$$\mathsf{fep040\_heatCapacity}\ T = 1/2$$

13.41 fep-041 — Expected Information Gain via Measure KL

Information gain is native measure-valued KL divergence: it is nonnegative, separates finite posterior and prior measures, and its predictive expectation vanishes when observations leave the posterior unchanged almost everywhere. Assumption scope: KL separation requires finite measures. The zero expected-information theorem requires a sigma-finite prior and almost-everywhere posterior equality; it does not claim that every informative experiment has strictly positive expected gain. Semantic disposition: `formalized`.

13.41.1 Lean sketch

```
import Mathlib.InformationTheory.KullbackLeibler.Basic
import Mathlib.MeasureTheory.Integral.Lebesgue.Countable

namespace FEP041

open MeasureTheory
open scoped ENNReal

variable {Obs State : Type*}
variable [MeasurableSpace Obs] [MeasurableSpace State]

/-- Information gain is Mathlib's measure-valued Kullback--Leibler divergence
from a posterior law to its prior law. -/
noncomputable def fep041_informationGain
    (posterior prior : Measure State) : ℝ≥0∞ :=
  InformationTheory.klDiv posterior prior

/-- Information gain is nonnegative because KL divergence is extended
nonnegative-real valued. -/
theorem fep041_informationGain_nonneg (posterior prior : Measure State) :
    0 ≤ fep041_informationGain posterior prior :=
  bot_le

/-- For finite measures, zero information gain characterizes equality of the
posterior and prior laws. -/
theorem fep041_informationGain_eq_zero_iff
```

```
    (posterior prior : Measure State)
    [IsFiniteMeasure posterior] [IsFiniteMeasure prior] :
    fep041_informationGain posterior prior = 0 ↔ posterior = prior := by
  exact InformationTheory.klDiv_eq_zero_iff

/-- Expected information gain under a predictive observation law. -/
noncomputable def fep041_expectedInformationGain
    (predictive : Measure Obs)
    (posterior : Obs → Measure State)
    (prior : Measure State) : ℝ≥0∞ :=
  ∫⁻ observation,
    fep041_informationGain (posterior observation) prior ∂predictive

/-- If observations leave the posterior equal to the prior almost everywhere,
then expected information gain is exactly zero. -/
theorem fep041_expectedInformationGain_zero
    (predictive : Measure Obs)
    (posterior : Obs → Measure State)
    (prior : Measure State) [SigmaFinite prior]
    (hposterior : ∀ᵐ observation ∂predictive, posterior observation = prior) :
    fep041_expectedInformationGain predictive posterior prior = 0 := by
  rw [fep041_expectedInformationGain]
  calc
    (∫⁻ observation,
        fep041_informationGain (posterior observation) prior ∂predictive) =
        ∫⁻ _observation, 0 ∂predictive := by
          apply lintegral_congr_ae
          filter_upwards [hposterior] with observation hobservation
          simp [fep041_informationGain, hobservation,
            InformationTheory.klDiv_self]
    _ = 0 := lintegral_zero

end FEP041
```

13.41.2 Typeset statement signatures

$$\{\mathsf{Obs}\ \mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpaceObs}]\ [\mathsf{MeasurableSpaceState}]$$
$$(\mathsf{posterior}\ \mathsf{prior} : \mathsf{MeasureState})$$
$$0 \le \mathsf{fep041\_informationGain}\ \mathsf{posterior}\ \mathsf{prior}$$

$$\{\mathsf{Obs}\ \mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpaceObs}]\ [\mathsf{MeasurableSpaceState}]$$
$$(\mathsf{posterior}\ \mathsf{prior} : \mathsf{MeasureState})\ [\mathsf{IsFiniteMeasure}\ \mathsf{posterior}]$$
$$[\mathsf{IsFiniteMeasure}\ \mathsf{prior}]$$
$$\mathsf{fep041\_informationGain}\ \mathsf{posterior}\ \mathsf{prior} = 0 \leftrightarrow \mathsf{posterior} = \mathsf{prior}$$

$$\{\mathsf{Obs}\ \mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpaceObs}]\ [\mathsf{MeasurableSpaceState}]$$
$$(\mathsf{predictive} : \mathsf{MeasureObs})\ (\mathsf{posterior} : \mathsf{Obs} \Rightarrow \mathsf{MeasureState})$$
$$(\mathsf{prior} : \mathsf{MeasureState})\ [\mathsf{SigmaFinite}\ \mathsf{prior}]\ (\mathsf{hposterior} : \forall^{\mathrm{a.e.}}\ \mathsf{observation}$$
$$\mathrm{d}\,\mathsf{predictive}, \mathsf{posterior}\ \mathsf{observation} = \mathsf{prior})$$
$$\mathsf{fep041\_expectedInformationGain}\ \mathsf{predictive}\ \mathsf{posterior}\ \mathsf{prior} = 0$$

13.42 fep-042 — Bernoulli Sufficient-Statistic Factorization

The Bernoulli sequence likelihood factorizes exactly through success and failure counts, so samples with equal count statistics have identical likelihood for every parameter. Assumption scope: The theorem is finite-sample and algebraic over a Boolean observation list. Its probabilistic interpretation restricts p to [0,1]; no completeness, asymptotic efficiency, or non-Bernoulli family is claimed. Semantic disposition: `formalized`.

13.42.1 Lean sketch

```
import Mathlib.Data.List.Count
import Mathlib.Tactic

namespace FEP042

/-- Number of observed Bernoulli successes. -/
def fep042_successCount (data : List Bool) : ℕ :=
  data.count true

/-- Number of observed Bernoulli failures. -/
def fep042_failureCount (data : List Bool) : ℕ :=
  data.count false

/-- The two-count statistic used by the Bernoulli likelihood factorization. -/
def fep042_sufficientStatistic (data : List Bool) : ℕ × ℕ :=
  (fep042_successCount data, fep042_failureCount data)

/-- Bernoulli sequence likelihood, with `true` mass `p`. -/
def fep042_bernoulliLikelihood (p : ℝ) : List Bool → ℝ
  | [] => 1
  | b :: data => (if b then p else 1 - p) * fep042_bernoulliLikelihood p data

/-- Fisher--Neyman factorization through success and failure counts. -/
theorem fep042_likelihood_factorizes (p : ℝ) (data : List Bool) :
    fep042_bernoulliLikelihood p data =
      p ^ fep042_successCount data * (1 - p) ^ fep042_failureCount data := by
  induction data with
  | nil => simp [fep042_bernoulliLikelihood, fep042_successCount, fep042_failureCount]
  | cons b data ih =>
      cases b <;>
        simp [fep042_bernoulliLikelihood, fep042_successCount,
          fep042_failureCount, ih, pow_succ] <;>
        ring

/-- Equal sufficient statistics imply equal likelihoods for every parameter. -/
theorem fep042_likelihood_eq_of_stat_eq
    (p : ℝ) (xs ys : List Bool)
    (h : fep042_sufficientStatistic xs = fep042_sufficientStatistic ys) :
    fep042_bernoulliLikelihood p xs = fep042_bernoulliLikelihood p ys := by
  have hs : fep042_successCount xs = fep042_successCount ys :=
    congrArg Prod.fst h
  have hf : fep042_failureCount xs = fep042_failureCount ys :=
    congrArg Prod.snd h
  rw [fep042_likelihood_factorizes, fep042_likelihood_factorizes, hs, hf]

end FEP042
```

13.42.2 Typeset statement signatures

$$(p : \mathbb{R})\ (\mathsf{data} : \mathsf{List\ Bool})$$
$$\mathsf{fep042\_bernoulliLikelihood}\ p\ \mathsf{data} = p^{\mathsf{fep042\_successCount}\ \mathsf{data}} \cdot (1-p)^{\mathsf{fep042\_failureCount}\ \mathsf{data}}$$

$$(p : \mathbb{R})\ (\mathsf{xs}\ \mathsf{ys} : \mathsf{List\ Bool})\ (h : \mathsf{fep042\_sufficientStatistic}\ \mathsf{xs} = \mathsf{fep042\_sufficientStatistic}\ \mathsf{ys})$$
$$\mathsf{fep042\_bernoulliLikelihood}\ p\ \mathsf{xs} = \mathsf{fep042\_bernoulliLikelihood}\ p\ \mathsf{ys}$$

13.43 fep-043 — Fermat Criticality and Quadratic Curvature

Fermat's theorem makes every local minimum critical; for an exact positive-curvature quadratic, the derivative is the stated gradient, the center is the unique critical point and global minimizer, and the scalar Hessian is strictly positive. Assumption scope: The general derivative-zero theorem uses Mathlib's local-minimum notion. Uniqueness and positive second derivative are proved only for the explicit scalar quadratic with a>0, not inferred for arbitrary differentiable objectives. Semantic disposition: `formalized`.

13.43.1 Lean sketch

```
import Mathlib.Analysis.Calculus.Deriv.Mul
import Mathlib.Analysis.Calculus.Deriv.Pow
import Mathlib.Analysis.Calculus.LocalExtr.Basic
import Mathlib.Tactic

namespace FEP043

/-- Fermat's theorem: the derivative vanishes at every local minimum. -/
theorem fep043_localMin_deriv_zero
    {f : ℝ → ℝ} {x₀ : ℝ} (hmin : IsLocalMin f x₀) :
    deriv f x₀ = 0 :=
  hmin.deriv_eq_zero

/-- Exact scalar quadratic free-energy landscape. -/
def fep043_quadraticFreeEnergy (a x₀ c x : ℝ) : ℝ :=
  a * (x - x₀) ^ 2 + c

/-- Its gradient field. -/
def fep043_quadraticGradient (a x₀ x : ℝ) : ℝ :=
  2 * a * (x - x₀)

/-- Exact first derivative of the quadratic free energy. -/
theorem fep043_quadratic_hasDerivAt (a x₀ c x : ℝ) :
    HasDerivAt (fep043_quadraticFreeEnergy a x₀ c)
      (fep043_quadraticGradient a x₀ x) x := by
  convert (((hasDerivAt_id x).sub_const x₀).pow 2).const_mul a |>.add_const c using 1
  all_goals
    first
    | exact AddCommGroup.ext rfl
    | exact Module.ext rfl
    | rfl
    | (simp [fep043_quadraticGradient] ; ring)

/-- Positive curvature makes the center the unique global minimizer. -/
theorem fep043_quadratic_unique_min
    {a x₀ c x : ℝ} (ha : 0 < a) :
    fep043_quadraticFreeEnergy a x₀ c x =
        fep043_quadraticFreeEnergy a x₀ c x₀ ↔
      x = x₀ := by
  constructor
  · intro h
    have hcenter : fep043_quadraticFreeEnergy a x₀ c x₀ = c := by
      simp [fep043_quadraticFreeEnergy]
    rw [hcenter] at h
    rw [fep043_quadraticFreeEnergy] at h
    have hsquare : (x - x₀) ^ 2 = 0 := by nlinarith
    nlinarith
  · rintro rfl
    rfl

/-- For positive curvature, the unique critical point is the unique minimum. -/
theorem fep043_quadratic_critical_iff
    {a x₀ c x : ℝ} (ha : 0 < a) :
```

```
    deriv (fep043_quadraticFreeEnergy a x₀ c) x = 0 ↔ x = x₀ := by
  rw [(fep043_quadratic_hasDerivAt a x₀ c x).deriv]
  simp only [fep043_quadraticGradient]
  constructor <;> intro h
  · nlinarith
  · rw [h, sub_self, mul_zero]

/-- The gradient derivative is the constant scalar Hessian `2a`. -/
theorem fep043_quadraticGradient_hasDerivAt (a x₀ x : ℝ) :
    HasDerivAt (fep043_quadraticGradient a x₀) (2 * a) x := by
  convert ((hasDerivAt_id x).sub_const x₀).const_mul (2 * a) using 1
  all_goals
    first
    | exact AddCommGroup.ext rfl
    | exact Module.ext rfl
    | rfl
    | simp

/-- Positive quadratic curvature gives a strictly positive Hessian. -/
theorem fep043_quadratic_hessian_pos {a : ℝ} (ha : 0 < a) :
    0 < 2 * a := by positivity

end FEP043
```

13.43.2 Typeset statement signatures

$$\{f : \mathbb{R} \Rightarrow \mathbb{R}\}\,\{x_0 : \mathbb{R}\}\,(\mathsf{hmin} : \mathsf{IsLocalMin}\,f\,x_0)$$
$$\mathsf{deriv}\,f\,x_0 = 0$$

$$(a\,x_0\,c\,x : \mathbb{R})$$
$$\mathsf{HasDerivAt}\,(\mathsf{fep043\_quadraticFreeEnergy}\,a\,x_0\,c)\,(\mathsf{fep043\_quadraticGradient}\,a\,x_0\,x)$$
$$x$$

$$\{a\,x_0\,c\,x : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)$$
$$\mathsf{fep043\_quadraticFreeEnergy}\,a\,x_0\,c\,x = \mathsf{fep043\_quadraticFreeEnergy}\,a\,x_0\,c\,x_0 \leftrightarrow x$$
$$= x_0$$

$$\{a\,x_0\,c\,x : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)$$
$$\mathsf{deriv}\,(\mathsf{fep043\_quadraticFreeEnergy}\,a\,x_0\,c)\,x = 0 \leftrightarrow x = x_0$$

$$(a\,x_0\,x : \mathbb{R})$$
$$\mathsf{HasDerivAt}\,(\mathsf{fep043\_quadraticGradient}\,a\,x_0)\,(2 \cdot a)\,x$$

$$\{a : \mathbb{R}\}\,(\mathsf{ha} : 0 < a)$$
$$0 < 2 \cdot a$$

13.44 fep-044 — Bernoulli Squared Hellinger Divergence

The squared Hellinger divergence between Bernoulli laws is nonnegative, symmetric, invariant under success/failure relabeling, and separates parameters on [0,1]. Assumption scope: Nonnegativity and symmetry are unconditional for the real-valued formula; the probability-divergence interpretation and separation theorem restrict both parameters to [0,1]. This is the Hellinger member of the alpha-divergence family, not a theorem for every alpha. Semantic disposition: `formalized`.

13.44.1 Lean sketch

```
import Mathlib.Analysis.Real.Sqrt
import Mathlib.Tactic

namespace FEP044
```

```
/-- Squared Hellinger divergence between Bernoulli laws, including the
conventional factor `1/2`. -/
noncomputable def fep044_hellingerSq (p q : ℝ) : ℝ :=
  ((Real.sqrt p - Real.sqrt q) ^ 2 +
      (Real.sqrt (1 - p) - Real.sqrt (1 - q)) ^ 2) / 2

/-- Squared Hellinger divergence is nonnegative. -/
theorem fep044_hellingerSq_nonneg (p q : ℝ) :
    0 ≤ fep044_hellingerSq p q := by
  unfold fep044_hellingerSq
  positivity

/-- Squared Hellinger divergence is symmetric. -/
theorem fep044_hellingerSq_symm (p q : ℝ) :
    fep044_hellingerSq p q = fep044_hellingerSq q p := by
  unfold fep044_hellingerSq
  ring

/-- On the probability interval, zero squared Hellinger divergence
characterizes equality of Bernoulli parameters. -/
theorem fep044_hellingerSq_eq_zero_iff
    {p q : ℝ} (hp : p ∈ Set.Icc (0 : ℝ) 1) (hq : q ∈ Set.Icc (0 : ℝ) 1) :
    fep044_hellingerSq p q = 0 ↔ p = q := by
  constructor
  · intro hzero
    rw [fep044_hellingerSq] at hzero
    have hfirst_sq : (Real.sqrt p - Real.sqrt q) ^ 2 = 0 := by
      nlinarith [sq_nonneg (Real.sqrt (1 - p) - Real.sqrt (1 - q))]
    have hfirst : Real.sqrt p - Real.sqrt q = 0 := by
      nlinarith
    have hsqrt : Real.sqrt p = Real.sqrt q := sub_eq_zero.mp hfirst
    exact (Real.sqrt_inj hp.1 hq.1).mp hsqrt
  · rintro rfl
    simp [fep044_hellingerSq]

/-- Relabeling Bernoulli success and failure leaves Hellinger divergence
unchanged. -/
theorem fep044_hellingerSq_complement (p q : ℝ) :
    fep044_hellingerSq (1 - p) (1 - q) = fep044_hellingerSq p q := by
  unfold fep044_hellingerSq
  congr 1
  ring_nf

end FEP044
```

13.44.2 Typeset statement signatures

$$(p\,q : \mathbb{R})$$
$$0 \le \mathsf{fep044\_hellingerSq}\,p\,q$$

$$(p\,q : \mathbb{R})$$
$$\mathsf{fep044\_hellingerSq}\,p\,q = \mathsf{fep044\_hellingerSq}\,q\,p$$

$$\{p\,q : \mathbb{R}\}\,(\mathsf{hp} : p \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)\,(\mathsf{hq} : q \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$\mathsf{fep044\_hellingerSq}\,p\,q = 0 \leftrightarrow p = q$$

$$(p\,q : \mathbb{R})$$
$$\mathsf{fep044\_hellingerSq}\,(1-p)\,(1-q) = \mathsf{fep044\_hellingerSq}\,p\,q$$

13.45 fep-045 — Finite Bernoulli Conjugate Closure

For a nonzero evidence normalizer, likelihood-times-Bernoulli-prior mass normalizes pointwise to another Bernoulli law at the Bayes-updated parameter. Assumption scope: Closure is exact for a binary latent hypothesis and binary evidence likelihood. Interior prior and positive likelihood assumptions prove positive evidence and keep the updated parameter in [0,1]. This is a finite Bernoulli conjugate family, not Beta-density conjugacy. Semantic disposition: `formalized`.

13.45.1 Lean sketch

```
import Mathlib.Algebra.BigOperators.Group.Finset.Basic
import Mathlib.Tactic

namespace FEP045

/-- Bernoulli mass function with `true` parameter `p`. -/
def fep045_bernoulliMass (p : ℝ) : Bool → ℝ
  | false => 1 - p
  | true => p

/-- Binary evidence likelihood, indexed by the latent Boolean hypothesis. -/
def fep045_binaryLikelihood (l₀ l₁ : ℝ) : Bool → ℝ
  | false => l₀
  | true => l₁

/-- Evidence normalizer for a Bernoulli prior and binary likelihood. -/
def fep045_evidence (p l₀ l₁ : ℝ) : ℝ :=
  p * l₁ + (1 - p) * l₀

/-- Updated Bernoulli parameter after observing the binary evidence. -/
noncomputable def fep045_posteriorParameter (p l₀ l₁ : ℝ) : ℝ :=
  p * l₁ / fep045_evidence p l₀ l₁

/-- Interior priors and positive likelihoods have positive evidence. -/
theorem fep045_evidence_pos
    {p l₀ l₁ : ℝ} (hp₀ : 0 < p) (hp₁ : p < 1)
    (hl₀ : 0 < l₀) (hl₁ : 0 < l₁) :
    0 < fep045_evidence p l₀ l₁ := by
  exact add_pos (mul_pos hp₀ hl₁) (mul_pos (sub_pos.mpr hp₁) hl₀)

/-- The updated Bernoulli parameter remains in the unit interval. -/
theorem fep045_posteriorParameter_mem_unit
    {p l₀ l₁ : ℝ} (hp₀ : 0 < p) (hp₁ : p < 1)
    (hl₀ : 0 < l₀) (hl₁ : 0 < l₁) :
    0 ≤ fep045_posteriorParameter p l₀ l₁ ∧
      fep045_posteriorParameter p l₀ l₁ ≤ 1 := by
  have hZ : 0 < fep045_evidence p l₀ l₁ :=
    fep045_evidence_pos hp₀ hp₁ hl₀ hl₁
  constructor
  · exact div_nonneg (mul_nonneg hp₀.le hl₁.le) hZ.le
  · apply (div_le_one hZ).2
    simp only [fep045_evidence]
    exact le_add_of_nonneg_right (mul_nonneg (sub_nonneg.mpr hp₁.le) hl₀.le)

/-- Pointwise Bayes closure: normalized likelihood-times-prior mass is exactly
the Bernoulli family at the updated parameter. -/
theorem fep045_bernoulli_posterior_closed
    {p l₀ l₁ : ℝ} (hZ : fep045_evidence p l₀ l₁ ≠ 0) (b : Bool) :
    fep045_bernoulliMass (fep045_posteriorParameter p l₀ l₁) b =
      fep045_bernoulliMass p b * fep045_binaryLikelihood l₀ l₁ b /
        fep045_evidence p l₀ l₁ := by
  cases b
  · simp only [fep045_bernoulliMass, fep045_binaryLikelihood,
```

```
      fep045_posteriorParameter]
    field_simp [fep045_evidence]
    simp only [fep045_evidence]
    ring
  · simp [fep045_bernoulliMass, fep045_binaryLikelihood,
      fep045_posteriorParameter]

/-- The conjugate posterior Bernoulli mass function remains normalized. -/
theorem fep045_posterior_mass_one (p l₀ l₁ : ℝ) :
    ∑ b : Bool, fep045_bernoulliMass (fep045_posteriorParameter p l₀ l₁) b = 1 := by
  simp [fep045_bernoulliMass]

end FEP045
```

13.45.2 Typeset statement signatures

$$\{p\,l_0\,l_1 : \mathbb{R}\}\,(hp_0 : 0 < p)\,(hp_1 : p < 1)\,(hl_0 : 0 < l_0)\,(hl_1 : 0 < l_1)$$
$$0 < \mathsf{fep045\_evidence}\,p\,l_0\,l_1$$

$$\{p\,l_0\,l_1 : \mathbb{R}\}\,(hp_0 : 0 < p)\,(hp_1 : p < 1)\,(hl_0 : 0 < l_0)\,(hl_1 : 0 < l_1)$$
$$0 \le \mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1 \wedge \mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1 \le 1$$

$$\{p\,l_0\,l_1 : \mathbb{R}\}\,(\mathsf{hZ} : \mathsf{fep045\_evidence}\,p\,l_0\,l_1 \neq 0)\,(b : \mathsf{Bool})$$
$$\mathsf{fep045\_bernoulliMass}\,(\mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1)\,b$$
$$= \mathsf{fep045\_bernoulliMass}\,p\,b \cdot \mathsf{fep045\_binaryLikelihood}\,l_0\,l_1\,b / \mathsf{fep045\_evidence}\,p$$
$$l_0\,l_1$$

$$(p\,l_0\,l_1 : \mathbb{R})$$
$$\sum b : \mathsf{Bool}, \mathsf{fep045\_bernoulliMass}\,(\mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1)\,b = 1$$

13.46 fep-046 — Finite Stick-Breaking Mass Conservation

For every finite stick-breaking sequence, allocated weights plus residual mass equal exactly one; unit-interval breaks keep the residual in [0,1]. Assumption scope: Mass conservation is a purely algebraic identity with no bounds. Residual nonnegativity and the unit upper bound require every break fraction in [0,1]. No infinite-series exhaustion or random-process law is claimed. Semantic disposition: `formalized`.

13.46.1 Lean sketch

```
import Mathlib.Algebra.BigOperators.Ring.List
import Mathlib.Tactic

namespace FEP046

/-- Finite stick-breaking weights.  The head receives fraction `v`; all later
weights are scaled by the retained fraction `1-v`. -/
def fep046_stickWeights : List ℝ → List ℝ
  | [] => []
  | v :: breaks =>
      v :: (fep046_stickWeights breaks).map (fun weight => (1 - v) * weight)

/-- Mass remaining after every listed break. -/
def fep046_remainder : List ℝ → ℝ
  | [] => 1
  | v :: breaks => (1 - v) * fep046_remainder breaks

/-- Allocated weights plus residual mass equal exactly one after any finite
sequence of breaks.  This algebraic conservation law needs no inequalities. -/
theorem fep046_mass_conservation (breaks : List ℝ) :
    (fep046_stickWeights breaks).sum + fep046_remainder breaks = 1 := by
  induction breaks with
```

```
  | nil => simp [fep046_stickWeights, fep046_remainder]
  | cons v breaks ih =>
      have hscaled :
          ((fep046_stickWeights breaks).map
            (fun weight => (1 - v) * weight)).sum =
            (1 - v) * (fep046_stickWeights breaks).sum := by
        simpa using
          (List.sum_map_mul_left
            (l := fep046_stickWeights breaks)
            (f := id) (r := 1 - v))
      simp only [fep046_stickWeights, fep046_remainder, List.sum_cons]
      rw [hscaled]
      calc
        v + (1 - v) * (fep046_stickWeights breaks).sum +
            (1 - v) * fep046_remainder breaks =
            v + (1 - v) *
              ((fep046_stickWeights breaks).sum +
                fep046_remainder breaks) := by ring
        _ = v + (1 - v) * 1 := by rw [ih]
        _ = 1 := by ring

/-- Unit-interval break fractions leave nonnegative residual mass. -/
theorem fep046_remainder_nonneg (breaks : List ℝ)
    (hbreaks : ∀ v ∈ breaks, v ∈ Set.Icc (0 : ℝ) 1) :
    0 ≤ fep046_remainder breaks := by
  induction breaks with
  | nil => simp [fep046_remainder]
  | cons v breaks ih =>
      rw [fep046_remainder]
      exact mul_nonneg (sub_nonneg.mpr (hbreaks v (by simp)).2)
        (ih (fun w hw => hbreaks w (by simp [hw])))

/-- The residual mass never exceeds one for unit-interval breaks. -/
theorem fep046_remainder_le_one (breaks : List ℝ)
    (hbreaks : ∀ v ∈ breaks, v ∈ Set.Icc (0 : ℝ) 1) :
    fep046_remainder breaks ≤ 1 := by
  induction breaks with
  | nil => simp [fep046_remainder]
  | cons v breaks ih =>
      rw [fep046_remainder]
      have hv := hbreaks v (by simp)
      have htail : ∀ w ∈ breaks, w ∈ Set.Icc (0 : ℝ) 1 :=
        fun w hw => hbreaks w (by simp [hw])
      have hrem0 := fep046_remainder_nonneg breaks htail
      have hrem1 := ih htail
      nlinarith [mul_nonneg hv.1 hrem0]

end FEP046
```

13.46.2 Typeset statement signatures

$$(\mathsf{breaks} : \mathsf{List}\,\mathbb{R})$$
$$(\mathsf{fep046\_stickWeights}\,\mathsf{breaks}).\mathsf{sum} + \mathsf{fep046\_remainder}\,\mathsf{breaks} = 1$$

$$(\mathsf{breaks} : \mathsf{List}\,\mathbb{R})\,(\mathsf{hbreaks} : \forall v \in \mathsf{breaks}, v \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$0 \le \mathsf{fep046\_remainder}\,\mathsf{breaks}$$

$$(\mathsf{breaks} : \mathsf{List}\,\mathbb{R})\,(\mathsf{hbreaks} : \forall v \in \mathsf{breaks}, v \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$\mathsf{fep046\_remainder}\,\mathsf{breaks} \le 1$$

13.47 fep-047 — Compositional Finite Sum-Product Propagation

Finite sum-product forward propagation is matrix–vector multiplication: it preserves nonnegativity and order under nonnegative factors, has zero and identity laws, and composes exactly by matrix multiplication. Assumption scope: Nonnegativity and monotonicity require entry-wise nonnegative factors and inputs. Identity and composition are unconditional algebraic laws. The row proves exact finite propagation, not normalization or loopy fixed-point convergence. Semantic disposition: `formalized`.

13.47.1 Lean sketch

```
import Mathlib.Data.Matrix.Mul
import Mathlib.Data.Real.Basic

namespace FEP047

open Finset

abbrev State := Fin 7
abbrev Factor := Matrix State State ℝ

/-- A full finite sum-product forward pass is matrix--vector multiplication. -/
def fep047_forward (factor : Factor) (incoming : State → ℝ) : State → ℝ :=
  Matrix.mulVec factor incoming

/-- Forward message is nonneg when factors and incoming messages are nonneg. -/
theorem fep047_forward_nonneg
    (factor : Factor) (incoming : State → ℝ)
    (hfactor : ∀ x y, 0 ≤ factor x y)
    (hincoming : ∀ y, 0 ≤ incoming y) (x : State) :
    0 ≤ fep047_forward factor incoming x := by
  simp only [fep047_forward, Matrix.mulVec, dotProduct]
  exact Finset.sum_nonneg fun y _ =>
    mul_nonneg (hfactor x y) (hincoming y)

/-- Message-passing monotonicity: larger incoming messages → larger output. -/
theorem fep047_forward_mono
    (factor : Factor) (incoming₁ incoming₂ : State → ℝ)
    (hfactor : ∀ x y, 0 ≤ factor x y)
    (hincoming : ∀ y, incoming₁ y ≤ incoming₂ y) (x : State) :
    fep047_forward factor incoming₁ x ≤
      fep047_forward factor incoming₂ x := by
  simp only [fep047_forward, Matrix.mulVec, dotProduct]
  exact Finset.sum_le_sum fun y _ =>
    mul_le_mul_of_nonneg_left (hincoming y) (hfactor x y)

/-- Zero incoming messages → zero forward output. -/
theorem fep047_zero_in (factor : Factor) :
    fep047_forward factor (fun _ => 0) = 0 := by
  ext x
  change (Σ y : State, factor x y * 0) = (0 : ℝ)
  simp

/-- Identity factor leaves every incoming message unchanged. -/
theorem fep047_identity (incoming : State → ℝ) :
    fep047_forward (1 : Factor) incoming = incoming := by
  exact Matrix.one_mulVec incoming

/-- Two consecutive sum-product passes are exactly one pass through the
matrix product of their factors. -/
theorem fep047_forward_compose
    (outer inner : Factor) (incoming : State → ℝ) :
    fep047_forward outer (fep047_forward inner incoming) =
      fep047_forward (outer * inner) incoming := by
```

```
  exact Matrix.mulVec_mulVec incoming outer inner

end FEP047
```

13.47.2 Typeset statement signatures

$$(\mathsf{factor} : \mathsf{Factor})\ (\mathsf{incoming} : \mathsf{State} \Rightarrow \mathbb{R})\ (\mathsf{hfactor} : \forall x\, y, 0 \le \mathsf{factor}\, x\, y)$$
$$(\mathsf{hincoming} : \forall y, 0 \le \mathsf{incoming}\, y)\ (x : \mathsf{State})$$
$$0 \le \mathsf{fep047\_forward}\ \mathsf{factor}\ \mathsf{incoming}\ x$$

$$(\mathsf{factor} : \mathsf{Factor})\ (\mathit{incoming}_1\ \mathit{incoming}_2 : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hfactor} : \forall x\, y, 0 \le \mathsf{factor}\, x\, y)\ (\mathsf{hincoming} : \forall y, \mathit{incoming}_1\, y \le \mathit{incoming}_2\, y)$$
$$(x : \mathsf{State})$$
$$\mathsf{fep047\_forward}\ \mathsf{factor}\ \mathit{incoming}_1\ x \le \mathsf{fep047\_forward}\ \mathsf{factor}\ \mathit{incoming}_2\ x$$

$$(\mathsf{factor} : \mathsf{Factor})$$
$$\mathsf{fep047\_forward}\ \mathsf{factor}\ (\lambda \_ \mapsto 0) = 0$$

$$(\mathsf{incoming} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\mathsf{fep047\_forward}\ (1 : \mathsf{Factor})\ \mathsf{incoming} = \mathsf{incoming}$$

$$(\mathsf{outer}\ \mathsf{inner} : \mathsf{Factor})\ (\mathsf{incoming} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\mathsf{fep047\_forward}\ \mathsf{outer}\ (\mathsf{fep047\_forward}\ \mathsf{inner}\ \mathsf{incoming}) = \mathsf{fep047\_forward}$$
$$(\mathsf{outer} \cdot \mathsf{inner})\ \mathsf{incoming}$$

13.48 fep-048 — Contractive Policy Update Uniqueness

Mathlib's global ContractingWith contract gives a unique fixed point on $\mathbb{R}$, and iterations of the concrete halving update converge to zero. Assumption scope: ContractingWith packages a nonnegative Lipschitz coefficient strictly below one. Completeness and nonemptiness of $\mathbb{R}$ discharge the Banach fixed-point assumptions; no claim is made for arbitrary noncontractive policy updates. Semantic disposition: `formalized`.

13.48.1 Lean sketch

```
import Mathlib.Order.Monotone.Basic
import Mathlib.Topology.MetricSpace.Contracting
import Mathlib.Tactic

namespace FEP048

open Topology

-- [proof strategy: native Mathlib ContractingWith and fixed-point API]

/-- Sync update: both components advance together, total is nonneg. -/
theorem fep048_sync_nonneg (a b : ℝ) (ha : 0 ≤ a) (hb : 0 ≤ b) : 0 ≤ a + b :=
  add_nonneg ha hb

/-- Async update: sequential composition preserves monotonicity. -/
theorem fep048_async_mono (f g : ℝ → ℝ) (hf : Monotone f) (hg : Monotone g) :
    Monotone (g ∘ f) :=
  hg.comp hf

/-- Mathlib's global contraction contract gives fixed-point uniqueness. -/
theorem fep048_contraction_unique (f : ℝ → ℝ) (c : NNReal)
    (hcontract : ContractingWith c f) (x y : ℝ)
    (hfx : Function.IsFixedPt f x) (hfy : Function.IsFixedPt f y) : x = y :=
  hcontract.fixedPoint_unique' hfx hfy
```

```
/-- A concrete nontrivial contraction used as an executable witness. -/
noncomputable def fep048_halfUpdate (x : ℝ) : ℝ := x / 2

/-- Halving contracts distances by exactly one half. -/
theorem fep048_halfUpdate_contracts :
    ContractingWith (1 / 2 : NNReal) fep048_halfUpdate := by
  refine ⟨by norm_num, LipschitzWith.of_dist_le_mul ?_⟩
  intro a b
  simp only [fep048_halfUpdate, Real.dist_eq]
  rw [← sub_div, abs_div]
  norm_num [div_eq_mul_inv]
  rw [mul_comm]

/-- The concrete halving contraction has zero as its unique fixed point. -/
theorem fep048_halfUpdate_unique_fixed_point (x : ℝ)
    (hx : Function.IsFixedPt fep048_halfUpdate x) : x = 0 := by
  exact fep048_halfUpdate_contracts.fixedPoint_unique' hx (by
    change fep048_halfUpdate 0 = 0
    norm_num [fep048_halfUpdate])

/-- Iterating the halving update from any real initial state converges to zero. -/
theorem fep048_halfUpdate_iterates_converge (x : ℝ) :
    Filter.Tendsto (fun n ↦ fep048_halfUpdate^[n] x) Filter.atTop (𝓝 0) := by
  have hzero : Function.IsFixedPt fep048_halfUpdate 0 := by
    change fep048_halfUpdate 0 = 0
    norm_num [fep048_halfUpdate]
  have hfixed : ContractingWith.fixedPoint fep048_halfUpdate
      fep048_halfUpdate_contracts = 0 :=
    (fep048_halfUpdate_contracts.fixedPoint_unique hzero).symm
  rw [← hfixed]
  exact fep048_halfUpdate_contracts.tendsto_iterate_fixedPoint x

/-- Identity update is monotone (trivial sync). -/
theorem fep048_id_mono : Monotone (id : ℝ → ℝ) := fun _ _ h => h

/-- Sum of monotone scalar updates is monotone. -/
theorem fep048_add_mono (f g : ℝ → ℝ) (hf : Monotone f) (hg : Monotone g) :
    Monotone (fun x => f x + g x) :=
  fun _ _ h => add_le_add (hf h) (hg h)

end FEP048
```

13.48.2 Typeset statement signatures

$$(a\,b : \mathbb{R})\,(\mathsf{ha} : 0 \le a)\,(\mathsf{hb} : 0 \le b)$$
$$0 \le a + b$$

$$(f\,g : \mathbb{R} \Rightarrow \mathbb{R})\,(\mathsf{hf} : \mathsf{Monotone}\,f)\,(\mathsf{hg} : \mathsf{Monotone}\,g)$$
$$\mathsf{Monotone}\,(g \circ f)$$

$$(f : \mathbb{R} \Rightarrow \mathbb{R})\,(c : \mathbb{R}_{\ge 0})\,(\mathsf{hcontract} : \mathsf{ContractingWith}\,c\,f)\,(x\,y : \mathbb{R})$$
$$(\mathsf{hfx} : \mathsf{Function.IsFixedPt}\,f\,x)\,(\mathsf{hfy} : \mathsf{Function.IsFixedPt}\,f\,y)$$
$$x = y$$

$$\mathsf{ContractingWith}\,(1/2 : \mathbb{R}_{\ge 0})\,\mathsf{fep048\_halfUpdate}$$

$$(x : \mathbb{R})\,(\mathsf{hx} : \mathsf{Function.IsFixedPt}\,\mathsf{fep048\_halfUpdate}\,x)$$
$$x = 0$$

$$(x : \mathbb{R})$$

$$\mathsf{Filter.Tendsto}\ (\lambda\, n \mapsto \mathsf{fep048\_halfUpdate}^{\lfloor n \rfloor}\, x)\ \mathsf{Filter.atTop}\ (\mathcal{N}\ 0)$$

$$\mathsf{Monotone}\ (\mathsf{id} : \mathbb{R} \Rightarrow \mathbb{R})$$

$$(f\ g : \mathbb{R} \Rightarrow \mathbb{R})\ (\mathsf{hf} : \mathsf{Monotone}\ f)\ (\mathsf{hg} : \mathsf{Monotone}\ g)$$
$$\mathsf{Monotone}\ (\lambda\, x \mapsto f\, x + g\, x)$$

13.49 fep-049 — Linear Constitutive Entropy Production

Under J_i=$L_i X_i$, the flux–force pairing is a nonnegative quadratic form that separates equilibrium for L_i>0. Independently, positive forward/reverse edge fluxes have production (f-r)log(f/r)≥0, with equality exactly at detailed balance. Assumption scope: The finite constitutive law is diagonal and assumes nonnegative conductances. The edge theorem assumes strictly positive bidirectional fluxes so affinity is finite. Cross-coupled Onsager matrices and microscopic derivations remain outside the claim. Semantic disposition: `formalized`.

13.49.1 Lean sketch

```
import Mathlib.Algebra.Order.BigOperators.Group.Finset
import Mathlib.Analysis.SpecialFunctions.Log.Basic
import Mathlib.Tactic

namespace FEP049

open Finset

/-- Diagonal linear constitutive law `J_i = L_i X_i`. -/
def fep049_linearFlux {ι : Type*}
    (conductance force : ι → ℝ) (i : ι) : ℝ :=
  conductance i * force i

/-- Entropy production generated by diagonal conductances and forces. -/
def fep049_entropyProduction {ι : Type*} [Fintype ι]
    (conductance force : ι → ℝ) : ℝ :=
  ∑ i, conductance i * force i ^ 2

/-- The flux--force pairing is exactly the entropy-production quadratic form. -/
theorem fep049_flux_force_identity {ι : Type*} [Fintype ι]
    (conductance force : ι → ℝ) :
    (∑ i, force i * fep049_linearFlux conductance force i) =
      fep049_entropyProduction conductance force := by
  apply Finset.sum_congr rfl
  intro i _
  simp only [fep049_linearFlux]
  ring

/-- Nonnegative conductances imply nonnegative entropy production. -/
theorem fep049_entropyProduction_nonneg {ι : Type*} [Fintype ι]
    (conductance force : ι → ℝ)
    (hconductance : ∀ i, 0 ≤ conductance i) :
    0 ≤ fep049_entropyProduction conductance force := by
  exact Finset.sum_nonneg fun i _ =>
    mul_nonneg (hconductance i) (sq_nonneg (force i))

/-- With strictly positive conductances, entropy production vanishes exactly
at thermodynamic equilibrium (all forces zero). -/
theorem fep049_entropyProduction_eq_zero_iff {ι : Type*} [Fintype ι]
    (conductance force : ι → ℝ)
    (hconductance : ∀ i, 0 < conductance i) :
    fep049_entropyProduction conductance force = 0 ↔
```

```
      ∀ i, force i = 0 := by
  constructor
  · intro hzero i
    have hterm : conductance i * force i ^ 2 = 0 :=
      (Finset.sum_eq_zero_iff_of_nonneg
        (fun j _ => mul_nonneg (hconductance j).le (sq_nonneg (force j)))).mp
        hzero i (Finset.mem_univ i)
    have hsquare : force i ^ 2 = 0 :=
      (mul_eq_zero.mp hterm).resolve_left (ne_of_gt (hconductance i))
    nlinarith
  · intro hforce
    simp [fep049_entropyProduction, hforce]

/-- Thermodynamic affinity of a directed edge from positive forward and
reverse stationary fluxes. -/
noncomputable def fep049_edgeAffinity (forward reverse : ℝ) : ℝ :=
  Real.log (forward / reverse)

/-- Edge entropy production is current times affinity. -/
noncomputable def fep049_edgeProduction (forward reverse : ℝ) : ℝ :=
  (forward - reverse) * fep049_edgeAffinity forward reverse

/-- Positive bidirectional fluxes produce nonnegative edge entropy. -/
theorem fep049_edgeProduction_nonneg
    {forward reverse : ℝ} (hforward : 0 < forward) (hreverse : 0 < reverse) :
    0 ≤ fep049_edgeProduction forward reverse := by
  rw [fep049_edgeProduction, fep049_edgeAffinity,
    Real.log_div hforward.ne' hreverse.ne']
  rcases le_total reverse forward with h | h
  · exact mul_nonneg (sub_nonneg.mpr h)
      (sub_nonneg.mpr (Real.log_le_log hreverse h))
  · exact mul_nonneg_of_nonpos_of_nonpos (sub_nonpos.mpr h)
      (sub_nonpos.mpr (Real.log_le_log hforward h))

/-- Under positive bidirectional fluxes, zero edge production is equivalent
to detailed balance on that edge. -/
theorem fep049_edgeProduction_eq_zero_iff
    {forward reverse : ℝ} (hforward : 0 < forward) (hreverse : 0 < reverse) :
    fep049_edgeProduction forward reverse = 0 ↔ forward = reverse := by
  constructor
  · intro hzero
    rcases mul_eq_zero.mp hzero with hcurrent | haffinity
    · exact sub_eq_zero.mp hcurrent
    · have hratio : forward / reverse = 1 :=
        Real.eq_one_of_pos_of_log_eq_zero (div_pos hforward hreverse) haffinity
      have hmul : forward = 1 * reverse :=
        (div_eq_iff hreverse.ne').mp hratio
      simpa using hmul
  · rintro rfl
    simp [fep049_edgeProduction]

end FEP049
```

13.49.2 Typeset statement signatures

$$\{\iota : \mathsf{Type}^{*}\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{conductance}\ \mathsf{force} : \iota \Rightarrow \mathbb{R})$$
$$(\sum i, \mathsf{force}\ i \cdot \mathsf{fep049\_linearFlux}\ \mathsf{conductance}\ \mathsf{force}\ i)$$
$$= \mathsf{fep049\_entropyProduction}\ \mathsf{conductance}\ \mathsf{force}$$

$$\{\iota : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\iota]\,(\mathsf{conductance}\,\mathsf{force} : \iota \Rightarrow \mathbb{R})$$
$$(\mathsf{hconductance} : \forall i, 0 \le \mathsf{conductance}\,i)$$
$$0 \le \mathsf{fep049\_entropyProduction}\,\mathsf{conductance}\,\mathsf{force}$$

$$\{\iota : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\iota]\,(\mathsf{conductance}\,\mathsf{force} : \iota \Rightarrow \mathbb{R})$$
$$(\mathsf{hconductance} : \forall i, 0 < \mathsf{conductance}\,i)$$
$$\mathsf{fep049\_entropyProduction}\,\mathsf{conductance}\,\mathsf{force} = 0 \leftrightarrow \forall i, \mathsf{force}\,i = 0$$

$$\{\mathsf{forward}\ \mathsf{reverse} : \mathbb{R}\}\,(\mathsf{hforward} : 0 < \mathsf{forward})\,(\mathsf{hreverse} : 0 < \mathsf{reverse})$$
$$0 \le \mathsf{fep049\_edgeProduction}\,\mathsf{forward}\,\mathsf{reverse}$$

$$\{\mathsf{forward}\ \mathsf{reverse} : \mathbb{R}\}\,(\mathsf{hforward} : 0 < \mathsf{forward})\,(\mathsf{hreverse} : 0 < \mathsf{reverse})$$
$$\mathsf{fep049\_edgeProduction}\,\mathsf{forward}\,\mathsf{reverse} = 0 \leftrightarrow \mathsf{forward} = \mathsf{reverse}$$

13.50 fep-050 — Logical Erasure and the Landauer Bound

A noninjective reset maps an unbiased bit to a pure state and loses exactly kB log 2 entropy; nonnegative total entropy at positive temperature derives the corresponding heat bound, and work above heat derives the work bound. Assumption scope: The derivation explicitly assumes the second-law total-entropy inequality, positive environmental temperature, and work at least as large as dissipated heat. It does not model a microscopic Hamiltonian or finite-time erasure protocol. Semantic disposition: `formalized`.

13.50.1 Lean sketch

```
import Mathlib.Analysis.SpecialFunctions.BinaryEntropy
import Mathlib.Tactic

namespace FEP050

/-- Logical reset of a binary memory to `false`. -/
def fep050_erasure (_bit : Bool) : Bool := false

/-- Reset is logically irreversible because it is not injective. -/
theorem fep050_erasure_not_injective :
    ¬Function.Injective fep050_erasure := by
  intro hinjective
  have hfalse_true : false = true := hinjective (by rfl)
  simp at hfalse_true

/-- Thermodynamic bit entropy in nats with Boltzmann constant `kB`. -/
noncomputable def fep050_bitEntropy (kB p : ℝ) : ℝ :=
  kB * Real.binEntropy p

/-- Entropy lost when an unbiased bit is reset to a pure state. -/
noncomputable def fep050_erasureEntropyLoss (kB : ℝ) : ℝ :=
  fep050_bitEntropy kB (2 : ℝ)⁻¹ - fep050_bitEntropy kB 0

/-- Resetting an unbiased bit loses exactly `kB log 2` entropy. -/
theorem fep050_erasureEntropyLoss_eq (kB : ℝ) :
    fep050_erasureEntropyLoss kB = kB * Real.log 2 := by
  simp [fep050_erasureEntropyLoss, fep050_bitEntropy]

/-- Environmental entropy gain from heat `Q` delivered at temperature `T`. -/
noncomputable def fep050_environmentEntropyChange (Q T : ℝ) : ℝ := Q / T

/-- Total entropy change for one-bit erasure. -/
noncomputable def fep050_totalEntropyChange (Q T kB : ℝ) : ℝ :=
  fep050_environmentEntropyChange Q T - fep050_erasureEntropyLoss kB

/-- Landauer threshold `kB T log 2`. -/
```

```
noncomputable def fep050_landauerBound (kB T : ℝ) : ℝ :=
  kB * T * Real.log 2

/-- The second-law entropy balance derives the Landauer heat bound; the
threshold inequality is a conclusion rather than an input. -/
theorem fep050_landauer_heat_bound
    {Q T kB : ℝ} (hT : 0 < T)
    (hsecondLaw : 0 ≤ fep050_totalEntropyChange Q T kB) :
    fep050_landauerBound kB T ≤ Q := by
  have hratio : kB * Real.log 2 ≤ Q / T := by
    rw [fep050_totalEntropyChange, fep050_environmentEntropyChange,
      fep050_erasureEntropyLoss_eq] at hsecondLaw
    exact sub_nonneg.mp hsecondLaw
  have hheat : (kB * Real.log 2) * T ≤ Q :=
    (le_div_iff₀ hT).mp hratio
  rw [fep050_landauerBound]
  nlinarith

/-- If supplied work dominates dissipated heat, the heat bound yields the
Landauer work bound. -/
theorem fep050_landauer_work_bound
    {W Q T kB : ℝ} (hT : 0 < T)
    (hsecondLaw : 0 ≤ fep050_totalEntropyChange Q T kB)
    (hworkHeat : Q ≤ W) :
    fep050_landauerBound kB T ≤ W :=
  (fep050_landauer_heat_bound hT hsecondLaw).trans hworkHeat

/-- The Landauer threshold is positive for positive `kB` and temperature. -/
theorem fep050_landauerBound_pos
    {kB T : ℝ} (hkB : 0 < kB) (hT : 0 < T) :
    0 < fep050_landauerBound kB T := by
  exact mul_pos (mul_pos hkB hT)
    (Real.log_pos (by norm_num : (1 : ℝ) < 2))

end FEP050
```

13.50.2 Typeset statement signatures

$$\neg\mathsf{Function.Injective}\ \mathsf{fep050\_erasure}$$

$$(\mathsf{kB} : \mathbb{R})$$
$$\mathsf{fep050\_erasureEntropyLoss}\ \mathsf{kB} = \mathsf{kB} \cdot \mathsf{Real.log}\ 2$$

$$\{Q\ T\ \mathsf{kB} : \mathbb{R}\}\ (\mathsf{hT} : 0 < T)\ (\mathsf{hsecondLaw} : 0 \le \mathsf{fep050\_totalEntropyChange}\ Q\ T\ \mathsf{kB})$$
$$\mathsf{fep050\_landauerBound}\ \mathsf{kB}\ T \le Q$$

$$\{W\ Q\ T\ \mathsf{kB} : \mathbb{R}\}\ (\mathsf{hT} : 0 < T)\ (\mathsf{hsecondLaw} : 0 \le \mathsf{fep050\_totalEntropyChange}\ Q\ T\ \mathsf{kB})$$
$$(\mathsf{hworkHeat} : Q \le W)$$
$$\mathsf{fep050\_landauerBound}\ \mathsf{kB}\ T \le W$$

$$\{\mathsf{kB}\ T : \mathbb{R}\}\ (\mathsf{hkB} : 0 < \mathsf{kB})\ (\mathsf{hT} : 0 < T)$$
$$0 < \mathsf{fep050\_landauerBound}\ \mathsf{kB}\ T$$

13.51 fep-051 — Likelihood-Ratio Reconstruction under Absolute Continuity

A native Radon–Nikodym derivative reconstructs a target measure from its reference exactly when the target is absolutely continuous with respect to that reference. Assumption scope: Both results require a measurable carrier and a HaveLebesgueDecomposition instance. The forward reconstruction additionally assumes target ≪ reference; the equivalence proves that this is the exact support condition and does not erase a singular component. Semantic disposition: `formalized`.

13.51.1 Lean sketch

```
import FepSketches.measure_bayes

namespace FEP051

open MeasureTheory

variable {Ω : Type*} [MeasurableSpace Ω]

/-- Absolute continuity is sufficient for reconstruction by the native
Radon--Nikodym likelihood ratio. -/
theorem fep051_likelihoodRatio_reconstruction
    (target reference : Measure Ω)
    [target.HaveLebesgueDecomposition reference]
    (h_ac : target ≪ reference) :
    reference.withDensity (target.rnDeriv reference) = target :=
  FEP.MeasureBayes.likelihoodRatio_reconstruction target reference h_ac

/-- The same equality characterizes absolute continuity, exposing the exact
failure boundary when a singular component is present. -/
theorem fep051_reconstruction_iff_absoluteContinuous
    (target reference : Measure Ω)
    [target.HaveLebesgueDecomposition reference] :
    reference.withDensity (target.rnDeriv reference) = target ↔
      target ≪ reference :=
  (FEP.MeasureBayes.likelihoodRatio_reconstruction_iff
    target reference).symm

end FEP051
```

13.51.2 Typeset statement signatures

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\,\Omega]$$
$$(\mathsf{target\ reference} : \mathsf{Measure}\,\Omega)\ [\mathsf{target.HaveLebesgueDecomposition\ reference}]$$
$$(\mathsf{h\_ac} : \mathsf{target} \ll \mathsf{reference})$$
$$\mathsf{reference.withDensity}\,(\mathsf{target.rnDeriv\ reference}) = \mathsf{target}$$

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\,\Omega]$$
$$(\mathsf{target\ reference} : \mathsf{Measure}\,\Omega)\ [\mathsf{target.HaveLebesgueDecomposition\ reference}]$$
$$\mathsf{reference.withDensity}\,(\mathsf{target.rnDeriv\ reference}) = \mathsf{target} \leftrightarrow \mathsf{target} \ll \mathsf{reference}$$

13.52 fep-052 — Posterior Density as a Likelihood Tilt

A posterior kernel is a prior measure tilted by the likelihood-to-predictive Radon–Nikodym derivative for predictive-almost-every observation. Assumption scope: The native result assumes a finite prior, a finite likelihood kernel, a nonempty standard-Borel parameter space, countable generation between the parameter and observation spaces, and prior-almost-every likelihood domination. The countable specialization discharges domination but remains an almost-everywhere identity, not a pointwise formula on null observations. Semantic disposition: `conditional_proxy`.

13.52.1 Lean sketch

```
import FepSketches.measure_bayes

namespace FEP052

open Filter MeasureTheory ProbabilityTheory
open scoped MeasureTheory ProbabilityTheory

variable {Parameter Observation : Type*}
  [MeasurableSpace Parameter] [MeasurableSpace Observation]
```

```
  [StandardBorelSpace Parameter] [Nonempty Parameter]
  [MeasurableSpace.CountableOrCountablyGenerated Parameter Observation]

/-- The posterior density is a likelihood-ratio tilt only almost everywhere
under the predictive observation law. -/
theorem fep052_posterior_density_tilt
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    (likelihood : Kernel Parameter Observation) [IsFiniteKernel likelihood]
    (h_ac : ∀ᵐ parameter ∂prior,
      likelihood parameter ≪ likelihood ∘ₘ prior) :
    ∀ᵐ observation ∂likelihood ∘ₘ prior,
      (likelihood†prior) observation =
        prior.withDensity (fun parameter =>
          likelihood.rnDeriv
            (Kernel.const Parameter (likelihood ∘ₘ prior))
            parameter observation) :=
  FEP.MeasureBayes.posterior_density_as_likelihood_tilt h_ac

/-- For countable parameter spaces Mathlib supplies the required domination,
but the resulting density identity remains predictive-almost-everywhere. -/
theorem fep052_countable_posterior_density_tilt
    {Parameter : Type*} [Countable Parameter] [MeasurableSpace Parameter]
    [Nonempty Parameter] [StandardBorelSpace Parameter]
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    (likelihood : Kernel Parameter Observation) [IsFiniteKernel likelihood] :
    ∀ᵐ observation ∂likelihood ∘ₘ prior,
      (likelihood†prior) observation =
        prior.withDensity (fun parameter =>
          (likelihood parameter).rnDeriv (likelihood ∘ₘ prior) observation) :=
  posterior_eq_withDensity_of_countable likelihood prior

end FEP052
```

13.52.2 Typeset statement signatures

$$\{\text{Parameter Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{MeasureParameter})\ [\text{IsFiniteMeasure prior}]$$
$$(\text{likelihood} : \text{Kernel Parameter Observation})\ [\text{IsFiniteKernel likelihood}]\ (\text{h\_ac} :$$
$$\forall^{\text{a.e.}}\ \text{parameter} \,\partial\, \text{prior},\ \text{likelihood parameter} \ll \text{likelihood} \circ_m \text{prior})$$
$$\forall^{\text{a.e.}}\ \text{observation} \,\partial\, \text{likelihood} \circ_m \text{prior},\ (\text{likelihood}^\dagger \text{prior})\ \text{observation}$$
$$= \text{prior.withDensity}\ (\lambda\ \text{parameter} \mapsto \text{likelihood.rnDeriv}\ (\text{Kernel.const Parameter}$$
$$(\text{likelihood} \circ_m \text{prior}))\ \text{parameter observation})$$

$$\{\text{Parameter Observation} : \text{Type}^*\}$$
$$\{\text{Parameter} : \text{Type}^*\}\ [\text{Countable Parameter}]\ [\text{MeasurableSpaceParameter}]$$
$$[\text{Nonempty Parameter}]\ [\text{StandardBorelSpace Parameter}]\ (\text{prior} : \text{MeasureParameter})$$
$$[\text{IsFiniteMeasure prior}]\ (\text{likelihood} : \text{Kernel Parameter Observation})$$
$$[\text{IsFiniteKernel likelihood}]$$
$$\forall^{\text{a.e.}}\ \text{observation} \,\partial\, \text{likelihood} \circ_m \text{prior},\ (\text{likelihood}^\dagger \text{prior})\ \text{observation}$$
$$= \text{prior.withDensity}\ (\lambda\ \text{parameter} \mapsto (\text{likelihood parameter}).\text{rnDeriv}\ (\text{likelihood}$$
$$\circ_m \text{prior})\ \text{observation})$$

13.53 fep-053 — Kernel Bayesian-Inversion Reconstruction

Bayesian inversion reconstructs the swapped prior-likelihood joint measure, and at every positive finite evidence atom posterior mass times evidence equals the corresponding joint atom. Assumption scope: The measure theorem requires a finite prior and likelihood on a nonempty standard-Borel parameter space. The finite atom theorem additionally exposes strictly positive predictive evidence; it makes no normalized posterior claim at a zero-evidence atom. Semantic disposition: `formalized`.

13.53.1 Lean sketch

```
import FepSketches.measure_bayes

namespace FEP053

open FEP MeasureTheory ProbabilityTheory
open scoped MeasureTheory ProbabilityTheory

variable {Parameter Observation : Type*}
  [MeasurableSpace Parameter] [MeasurableSpace Observation]
  [StandardBorelSpace Parameter] [Nonempty Parameter]

/-- A posterior kernel reconstructs the complete swapped joint law, not only
one posterior density. -/
theorem fep053_kernelBayes_joint_reconstruction
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    (likelihood : Kernel Parameter Observation) [IsFiniteKernel likelihood] :
    (likelihood ∘ₘ prior) ⊗ₘ likelihood†prior =
      (prior ⊗ₘ likelihood).map Prod.swap :=
  FEP.MeasureBayes.posterior_joint_reconstruction

/-- At a positive finite evidence atom, posterior times evidence reconstructs
the corresponding joint atom exactly. -/
theorem fep053_finiteBayes_atom_reconstruction
    {FiniteParameter FiniteEvidence : Type*}
    [Fintype FiniteParameter] [Fintype FiniteEvidence]
    (prior : FiniteLaw FiniteParameter)
    (likelihood : FiniteKernel FiniteParameter FiniteEvidence)
    (evidence : FiniteEvidence)
    (hEvidence : 0 < likelihood.predictive prior evidence)
    (parameter : FiniteParameter) :
    likelihood.posterior prior evidence hEvidence parameter *
        likelihood.predictive prior evidence =
      prior parameter * likelihood parameter evidence :=
  FEP.MeasureBayes.finite_posterior_reconstruction
    prior likelihood evidence hEvidence parameter

end FEP053
```

13.53.2 Typeset statement signatures

$$\{\mathsf{Parameter}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{prior} : \mathsf{MeasureParameter})\ [\mathsf{IsFiniteMeasure}\ \mathsf{prior}]$$
$$(\mathsf{likelihood} : \mathsf{Kernel}\ \mathsf{Parameter}\ \mathsf{Observation})\ [\mathsf{IsFiniteKernel}\ \mathsf{likelihood}]$$
$$(\mathsf{likelihood} \circ_m \mathsf{prior}) \otimes_m \mathsf{likelihood}^\dagger \mathsf{prior} = (\mathsf{prior} \otimes_m \mathsf{likelihood}).\mathsf{map}\ \mathsf{Prod.swap}$$

$$\{\mathsf{Parameter}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$\{\mathsf{FiniteParameter}\ \mathsf{FiniteEvidence} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{FiniteParameter}]$$
$$[\mathsf{Fintype}\ \mathsf{FiniteEvidence}]\ (\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{FiniteParameter})\ (\mathsf{likelihood} :$$
$$\mathsf{FiniteKernel}\ \mathsf{FiniteParameter}\ \mathsf{FiniteEvidence})\ (\mathsf{evidence} : \mathsf{FiniteEvidence})$$
$$(\mathsf{hEvidence} : 0 < \mathsf{likelihood.predictive}\ \mathsf{prior}\ \mathsf{evidence})$$
$$(\mathsf{parameter} : \mathsf{FiniteParameter})$$
$$\mathsf{likelihood.posterior}\ \mathsf{prior}\ \mathsf{evidence}\ \mathsf{hEvidence}\ \mathsf{parameter} \cdot \mathsf{likelihood.predictive}$$
$$\mathsf{prior}\ \mathsf{evidence} = \mathsf{prior}\ \mathsf{parameter} \cdot \mathsf{likelihood}\ \mathsf{parameter}\ \mathsf{evidence}$$

13.54 fep-054 — Bayes Involution for Finite Joint Laws

Inverting a finite Markov likelihood twice returns the original likelihood prior-almost-everywhere, and one inversion followed by prediction reconstructs the prior exactly as a measure. Assumption scope: The likelihood and prior are native finite kernels and measures, not only finite

atom tables. Involution requires both parameter and observation carriers to be nonempty standard Borel and the likelihood to be Markov; its conclusion is almost-everywhere rather than pointwise kernel equality. Semantic disposition: `conditional_proxy`.

13.54.1 Lean sketch

```
import FepSketches.measure_bayes

namespace FEP054

open Filter MeasureTheory ProbabilityTheory
open scoped MeasureTheory ProbabilityTheory

variable {Parameter Observation : Type*}
  [MeasurableSpace Parameter] [MeasurableSpace Observation]
  [StandardBorelSpace Parameter] [Nonempty Parameter]

/-- Inverting a Markov likelihood twice recovers it prior-almost-everywhere,
which is the native Bayes involution law. -/
theorem fep054_bayes_involution
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    (likelihood : Kernel Parameter Observation)
    [StandardBorelSpace Observation] [Nonempty Observation]
    [IsFiniteKernel likelihood] [IsMarkovKernel likelihood] :
    (likelihood†prior)†(likelihood ∘ₘ prior) =ᵐ[prior] likelihood :=
  FEP.MeasureBayes.posterior_involution

/-- The one-sided inversion law is exact as a measure equality. -/
theorem fep054_posterior_reconstructs_prior
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    (likelihood : Kernel Parameter Observation)
    [IsFiniteKernel likelihood] [IsMarkovKernel likelihood] :
    likelihood†prior ∘ₘ likelihood ∘ₘ prior = prior :=
  FEP.MeasureBayes.posterior_reconstructs_prior

end FEP054
```

13.54.2 Typeset statement signatures

$$\{\text{Parameter Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{MeasureParameter})\ [\text{IsFiniteMeasure prior}]$$
$$(\text{likelihood} : \text{Kernel Parameter Observation})\ [\text{StandardBorelSpace Observation}]$$
$$[\text{Nonempty Observation}]\ [\text{IsFiniteKernel likelihood}]\ [\text{IsMarkovKernel likelihood}]$$
$$(\text{likelihood}^\dagger \text{prior})^\dagger (\text{likelihood} \circ_m \text{prior}) \overset{\text{a.e.}}{=} [\text{prior}]\ \text{likelihood}$$

$$\{\text{Parameter Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{MeasureParameter})\ [\text{IsFiniteMeasure prior}]$$
$$(\text{likelihood} : \text{Kernel Parameter Observation})\ [\text{IsFiniteKernel likelihood}]$$
$$[\text{IsMarkovKernel likelihood}]$$
$$\text{likelihood}^\dagger \text{prior} \circ_m \text{likelihood} \circ_m \text{prior} = \text{prior}$$

13.55 fep-055 — Composite-Kernel Bayesian Inversion

Bayesian inversion reverses a two-kernel composition into the corresponding composition of inversions, and the associated predictive law is exactly the chronological two-stage prediction. Assumption scope: The prior and both kernels are finite, and the parameter and intermediate spaces are nonempty standard Borel. The inverse-kernel equality holds only almost everywhere under the final predictive law; predictive associativity itself is an unconditional measure equality under finiteness. Semantic disposition: `conditional_proxy`.

13.55.1 Lean sketch

```
import FepSketches.measure_bayes
```

```
namespace FEP055

open Filter MeasureTheory ProbabilityTheory
open scoped MeasureTheory ProbabilityTheory

variable {Parameter Intermediate Observation : Type*}
  [MeasurableSpace Parameter] [MeasurableSpace Intermediate]
  [MeasurableSpace Observation]

/-- Bayesian inversion reverses sequential kernel composition almost
everywhere under the final predictive law. -/
theorem fep055_compositeKernel_bayesInversion
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    [StandardBorelSpace Parameter] [Nonempty Parameter]
    [StandardBorelSpace Intermediate] [Nonempty Intermediate]
    (earlier : Kernel Parameter Intermediate) [IsFiniteKernel earlier]
    (later : Kernel Intermediate Observation) [IsFiniteKernel later] :
    (later ∘ₖ earlier)†prior =ᵐ[later ∘ₘ earlier ∘ₘ prior]
      earlier†prior ∘ₖ later†(earlier ∘ₘ prior) :=
  FEP.MeasureBayes.posterior_composite later

/-- The predictive law of the composite is the chronological two-kernel
predictive law used as the almost-everywhere reference above. -/
theorem fep055_compositePredictive_associativity
    (prior : Measure Parameter) [IsFiniteMeasure prior]
    (earlier : Kernel Parameter Intermediate) [IsFiniteKernel earlier]
    (later : Kernel Intermediate Observation) [IsFiniteKernel later] :
    (later ∘ₖ earlier) ∘ₘ prior = later ∘ₘ earlier ∘ₘ prior := by
  rw [Measure.comp_assoc]

end FEP055
```

13.55.2 Typeset statement signatures

$$\{\mathsf{Parameter}\ \mathsf{Intermediate}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{prior} : \mathsf{MeasureParameter})\ [\mathsf{IsFiniteMeasure}\ \mathsf{prior}]$$
$$[\mathsf{StandardBorelSpace}\ \mathsf{Parameter}]\ [\mathsf{Nonempty}\ \mathsf{Parameter}]$$
$$[\mathsf{StandardBorelSpace}\ \mathsf{Intermediate}]\ [\mathsf{Nonempty}\ \mathsf{Intermediate}]$$
$$(\mathsf{earlier} : \mathsf{Kernel}\ \mathsf{Parameter}\ \mathsf{Intermediate})\ [\mathsf{IsFiniteKernel}\ \mathsf{earlier}]$$
$$(\mathsf{later} : \mathsf{Kernel}\ \mathsf{Intermediate}\ \mathsf{Observation})\ [\mathsf{IsFiniteKernel}\ \mathsf{later}]$$
$$(\mathsf{later} \circ_k \mathsf{earlier})^\dagger \mathsf{prior} \overset{\text{a.e.}}{=} [\mathsf{later} \circ_m \mathsf{earlier} \circ_m \mathsf{prior}]\ \mathsf{earlier}^\dagger \mathsf{prior} \circ_k \mathsf{later}^\dagger(\mathsf{earlier} \circ_m \mathsf{prior})$$

$$\{\mathsf{Parameter}\ \mathsf{Intermediate}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{prior} : \mathsf{MeasureParameter})\ [\mathsf{IsFiniteMeasure}\ \mathsf{prior}]$$
$$(\mathsf{earlier} : \mathsf{Kernel}\ \mathsf{Parameter}\ \mathsf{Intermediate})\ [\mathsf{IsFiniteKernel}\ \mathsf{earlier}]$$
$$(\mathsf{later} : \mathsf{Kernel}\ \mathsf{Intermediate}\ \mathsf{Observation})\ [\mathsf{IsFiniteKernel}\ \mathsf{later}]$$
$$(\mathsf{later} \circ_k \mathsf{earlier}) \circ_m \mathsf{prior} = \mathsf{later} \circ_m \mathsf{earlier} \circ_m \mathsf{prior}$$

13.56 fep-056 — Standard-Borel Conditional-Kernel Reconstruction

Mathlib's canonical conditional kernel reconstructs a finite joint measure from its first marginal and is normalized at every conditioning value. Assumption scope: The conditioned carrier must be nonempty standard Borel and the joint measure finite. This is the maintained standard-Borel disintegration theorem, not a claim that regular conditional probabilities exist on arbitrary measurable spaces. Semantic disposition: `formalized`.

13.56.1 Lean sketch

```
import FepSketches.measure_bayes

namespace FEP056

open MeasureTheory ProbabilityTheory
open scoped ENNReal MeasureTheory

variable {Conditioning Conditioned : Type*}
  [MeasurableSpace Conditioning] [MeasurableSpace Conditioned]
  [StandardBorelSpace Conditioned] [Nonempty Conditioned]

/-- The canonical conditional kernel reconstructs every finite joint measure
whose conditioned space is standard Borel. -/
theorem fep056_standardBorel_condKernel_reconstruction
    (joint : Measure (Conditioning × Conditioned)) [IsFiniteMeasure joint] :
    joint.fst ⊗ₘ joint.condKernel = joint :=
  FEP.MeasureBayes.conditionalKernel_reconstruction joint

/-- The constructed conditional kernel is normalized at every conditioning
value, including values in a marginal null set. -/
theorem fep056_standardBorel_condKernel_mass_one
    (joint : Measure (Conditioning × Conditioned)) [IsFiniteMeasure joint]
    (conditioning : Conditioning) :
    joint.condKernel conditioning Set.univ = 1 :=
  measure_univ

end FEP056
```

13.56.2 Typeset statement signatures

$$\{\mathsf{Conditioning}\ \mathsf{Conditioned} : \mathsf{Type}^*\}$$
$$(\mathsf{joint} : \mathsf{Measure}(\mathsf{Conditioning} \times \mathsf{Conditioned}))\ [\mathsf{IsFiniteMeasure}\ \mathsf{joint}]$$
$$\mathsf{joint.fst} \otimes_m \mathsf{joint.condKernel} = \mathsf{joint}$$

$$\{\mathsf{Conditioning}\ \mathsf{Conditioned} : \mathsf{Type}^*\}$$
$$(\mathsf{joint} : \mathsf{Measure}(\mathsf{Conditioning} \times \mathsf{Conditioned}))\ [\mathsf{IsFiniteMeasure}\ \mathsf{joint}]$$
$$(\mathsf{conditioning} : \mathsf{Conditioning})$$
$$\mathsf{joint.condKernel}\ \mathsf{conditioning}\ \Omega = 1$$

13.57 fep-057 — Conditional-Expectation Tower through a Kernel

Integrating a conditional expectation against the conditioning marginal equals the original joint expectation, with a parallel extended-nonnegative tower law. Assumption scope: The joint measure is finite and the conditioned carrier nonempty standard Borel. Real-valued observables must be integrable; extended-nonnegative observables need measurability but no separate integrability premise. Semantic disposition: `formalized`.

13.57.1 Lean sketch

```
import FepSketches.measure_bayes

namespace FEP057

open MeasureTheory ProbabilityTheory
open scoped ENNReal MeasureTheory

variable {Conditioning Conditioned : Type*}
  [MeasurableSpace Conditioning] [MeasurableSpace Conditioned]
  [StandardBorelSpace Conditioned] [Nonempty Conditioned]

/-- Conditional expectation followed by marginal expectation equals the
```

```
joint expectation for every integrable real observable. -/
theorem fep057_conditionalExpectation_tower
    (joint : Measure (Conditioning × Conditioned)) [IsFiniteMeasure joint]
    {observable : Conditioning × Conditioned → ℝ}
    (hintegrable : Integrable observable joint) :
    (∫ conditioning,
        ∫ conditioned, observable (conditioning, conditioned)
          ∂joint.condKernel conditioning ∂joint.fst) =
      ∫ pair, observable pair ∂joint :=
  FEP.MeasureBayes.conditionalExpectation_tower joint hintegrable

/-- The nonnegative tower law holds without an integrability premise because
Lebesgue integration uses extended nonnegative values. -/
theorem fep057_conditionalLIntegral_tower
    (joint : Measure (Conditioning × Conditioned)) [IsFiniteMeasure joint]
    {observable : Conditioning × Conditioned → ℝ≥0∞}
    (hmeasurable : Measurable observable) :
    (∫⁻ conditioning,
        ∫⁻ conditioned, observable (conditioning, conditioned)
          ∂joint.condKernel conditioning ∂joint.fst) =
      ∫⁻ pair, observable pair ∂joint :=
  joint.lintegral_condKernel hmeasurable

end FEP057
```

13.57.2 Typeset statement signatures

$$\{\mathsf{Conditioning}\ \mathsf{Conditioned} : \mathsf{Type}^*\}$$
$$(\mathsf{joint} : \mathsf{Measure}(\mathsf{Conditioning} \times \mathsf{Conditioned}))\ [\mathsf{IsFiniteMeasure}\ \mathsf{joint}]$$
$$\{\mathsf{observable} : \mathsf{Conditioning} \times \mathsf{Conditioned} \Rightarrow \mathbb{R}\}$$
$$(\mathsf{hintegrable} : \mathsf{Integrable}\ \mathsf{observable}\ \mathsf{joint})$$
$$\left(\int \mathsf{conditioning}, \int \mathsf{conditioned}, \mathsf{observable}\ (\mathsf{conditioning}, \mathsf{conditioned})\right.$$
$$\left.\mathrm{d}\,\mathsf{joint.condKernel}\ \mathsf{conditioning}\ \mathrm{d}\,\mathsf{joint.fst}\right) = \int \mathsf{pair}, \mathsf{observable}\ \mathsf{pair}\ \mathrm{d}\,\mathsf{joint}$$

$$\{\mathsf{Conditioning}\ \mathsf{Conditioned} : \mathsf{Type}^*\}$$
$$(\mathsf{joint} : \mathsf{Measure}(\mathsf{Conditioning} \times \mathsf{Conditioned}))\ [\mathsf{IsFiniteMeasure}\ \mathsf{joint}]$$
$$\{\mathsf{observable} : \mathsf{Conditioning} \times \mathsf{Conditioned} \Rightarrow \mathbb{R}^{\infty}_{\geq 0}\}$$
$$(\mathsf{hmeasurable} : \mathsf{Measurable}\ \mathsf{observable})$$
$$\left(\int^{-} \mathsf{conditioning}, \int^{-} \mathsf{conditioned}, \mathsf{observable}\ (\mathsf{conditioning}, \mathsf{conditioned})\right.$$
$$\left.\mathrm{d}\,\mathsf{joint.condKernel}\ \mathsf{conditioning}\ \mathrm{d}\,\mathsf{joint.fst}\right) = \int^{-} \mathsf{pair}, \mathsf{observable}\ \mathsf{pair}\ \mathrm{d}\,\mathsf{joint}$$

13.58 fep-058 — Finite Gibbs Variational Principle

Finite Gibbs free energy is bounded below by minus the certified log partition, and the explicit normalized Gibbs law attains that bound. Assumption scope: GibbsCertificate exposes strict support for reference and optimizer laws and the exact pointwise log-density identity. The theorem is finite and does not assert existence of such a certificate for arbitrary continuous potentials. Semantic disposition: `formalized`.

13.58.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP058

open FEP FEP.VariationalDuality
```

```
variable {State : Type*} [Fintype State]

/-- The certified log partition is the exact lower bound of finite Gibbs free
energy. -/
theorem fep058_gibbsVariational_lower_bound
    (certificate : GibbsCertificate State) (candidate : FiniteLaw State) :
    -certificate.logPartition ≤
      gibbsFreeEnergy certificate candidate :=
  neg_logPartition_le_gibbsFreeEnergy certificate candidate

/-- The explicit normalized Gibbs law attains the variational lower bound. -/
theorem fep058_gibbsVariational_optimizer
    (certificate : GibbsCertificate State) :
    gibbsFreeEnergy certificate certificate.optimizer =
      -certificate.logPartition :=
  gibbsFreeEnergy_optimizer certificate

/-- The certificate assumptions are jointly satisfiable: on every nonempty
finite state space, the uniform zero-potential model attains dual value zero. -/
theorem fep058_uniformGibbs_nonvacuity [Nonempty State] :
    dvObjective (uniformZeroPotentialGibbs (α := State))
        (uniformZeroPotentialGibbs (α := State)).optimizer = 0 :=
  uniformZeroPotentialGibbs_objective

end FEP058
```

13.58.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{certificate} : \mathsf{GibbsCertificate}\ \mathsf{State})\ (\mathsf{candidate} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$-\mathsf{certificate.logPartition} \le \mathsf{gibbsFreeEnergy}\ \mathsf{certificate}\ \mathsf{candidate}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{certificate} : \mathsf{GibbsCertificate}\ \mathsf{State})$$
$$\mathsf{gibbsFreeEnergy}\ \mathsf{certificate}\ \mathsf{certificate.optimizer} = -\mathsf{certificate.logPartition}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$[\mathsf{Nonempty}\ \mathsf{State}]$$
$$\mathsf{dvObjective}\ (\mathsf{uniformZeroPotentialGibbs}\ (\alpha := \mathsf{State}))$$
$$(\mathsf{uniformZeroPotentialGibbs}\ (\alpha := \mathsf{State})).\mathsf{optimizer} = 0$$

13.59 fep-059 — Finite Donsker–Varadhan Equality with a Full-Support Gibbs Optimizer

The finite Donsker–Varadhan objective is bounded by the certified log partition, with equality exactly at the supplied full-support Gibbs optimizer. Assumption scope: The result is parameterized by GibbsCertificate, whose strict-support and log-density fields make the finite logarithmic boundary explicit. It does not extend the equality to arbitrary measures, unbounded potentials, or an optimizer whose existence has not been certified. Semantic disposition: `formalized`.

13.59.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP059

open FEP FEP.VariationalDuality

variable {State : Type*} [Fintype State]

/-- A finite Donsker--Varadhan objective is bounded by the certified log
partition. -/
```

```
theorem fep059_donskerVaradhan_upper_bound
    (certificate : GibbsCertificate State) (candidate : FiniteLaw State) :
    dvObjective certificate candidate ≤ certificate.logPartition :=
  dvObjective_le_logPartition certificate candidate

/-- The full-support Gibbs optimizer attains the Donsker--Varadhan equality. -/
theorem fep059_donskerVaradhan_optimizer
    (certificate : GibbsCertificate State) :
    dvObjective certificate certificate.optimizer =
      certificate.logPartition :=
  dvObjective_optimizer certificate

/-- Equality uniquely characterizes that normalized optimizer. -/
theorem fep059_donskerVaradhan_equality_iff
    (certificate : GibbsCertificate State) (candidate : FiniteLaw State) :
    dvObjective certificate candidate = certificate.logPartition ↔
      candidate = certificate.optimizer :=
  dvObjective_eq_logPartition_iff certificate candidate

end FEP059
```

13.59.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{certificate} : \mathsf{GibbsCertificate}\ \mathsf{State})\ (\mathsf{candidate} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$\mathsf{dvObjective}\ \mathsf{certificate}\ \mathsf{candidate} \le \mathsf{certificate.logPartition}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{certificate} : \mathsf{GibbsCertificate}\ \mathsf{State})$$
$$\mathsf{dvObjective}\ \mathsf{certificate}\ \mathsf{certificate.optimizer} = \mathsf{certificate.logPartition}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{certificate} : \mathsf{GibbsCertificate}\ \mathsf{State})\ (\mathsf{candidate} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$\mathsf{dvObjective}\ \mathsf{certificate}\ \mathsf{candidate} = \mathsf{certificate.logPartition} \leftrightarrow \mathsf{candidate} = \mathsf{certificate.optimizer}$$

13.60 fep-060 — Coordinate ELBO Decomposition

A finite joint ELBO separates exactly into a prior-coordinate score minus an expected conditional KL remainder, and that remainder is nonnegative. Assumption scope: The decomposition assumes every reference-prior and reference-kernel atom is strictly positive so all logarithmic rewrites are sound. Conditional KL nonnegativity is proved for the project's totalized finite divergence, but no zero-reference extended-real interpretation is claimed. Semantic disposition: `formalized`.

13.60.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP060

open FEP FEP.FiniteInformation FEP.VariationalDuality

variable {Latent Observation : Type*}
  [Fintype Latent] [Fintype Observation]

/-- Joint ELBO separates into a prior-coordinate score and an expected
conditional KL penalty under explicit reference support. -/
theorem fep060_coordinateELBO_decomposition
    (actualPrior referencePrior : FiniteLaw Latent)
    (actualKernel referenceKernel : FiniteKernel Latent Observation)
    (hprior : ∀ latent, 0 < referencePrior latent)
```

```
    (hkernel : ∀ latent observation,
      0 < referenceKernel latent observation) :
    jointELBO actualPrior referencePrior actualKernel referenceKernel =
      -finiteKL actualPrior referencePrior -
        conditionalKL actualPrior actualKernel referenceKernel :=
  jointELBO_coordinate_decomposition actualPrior referencePrior
    actualKernel referenceKernel hprior hkernel

/-- The conditional remainder in the coordinate decomposition is
nonnegative even under the totalized finite-KL convention. -/
theorem fep060_coordinateKL_nonnegative
    (prior : FiniteLaw Latent)
    (actual reference : FiniteKernel Latent Observation) :
    0 ≤ conditionalKL prior actual reference :=
  conditionalKL_nonneg prior actual reference

end FEP060
```

13.60.2 Typeset statement signatures

$$\{\mathsf{Latent\ Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{actualPrior\ referencePrior} : \mathsf{FiniteLaw\ Latent})\ (\mathsf{actualKernel\ referenceKernel} :$$
$$\mathsf{FiniteKernel\ Latent\ Observation})\ (\mathsf{hprior} : \forall \mathsf{latent}, 0 < \mathsf{referencePrior\ latent})$$
$$(\mathsf{hkernel} : \forall \mathsf{latent\ observation}, 0 < \mathsf{referenceKernel\ latent\ observation})$$
$$\mathsf{jointELBO\ actualPrior\ referencePrior\ actualKernel\ referenceKernel} = -\mathsf{finiteKL}$$
$$\mathsf{actualPrior\ referencePrior} - \mathsf{conditionalKL\ actualPrior\ actualKernel}$$
$$\mathsf{referenceKernel}$$

$$\{\mathsf{Latent\ Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{prior} : \mathsf{FiniteLaw\ Latent})\ (\mathsf{actual\ reference} : \mathsf{FiniteKernel\ Latent\ Observation})$$
$$0 \le \mathsf{conditionalKL\ prior\ actual\ reference}$$

13.61 fep-061 — Mean-Field Coordinate Free-Energy Optimum

Holding one supported mean-field factor fixed reduces the coordinate free energy exactly to finite KL, whose unique zero is the supported target factor. Assumption scope: The fixed and target laws have full support on finite carriers. The theorem is a one-coordinate optimum with the other factor held fixed; it does not prove global convergence of alternating mean-field updates or handle continuous variational families. Semantic disposition: `formalized`.

13.61.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP061

open FEP FEP.VariationalDuality

variable {Fixed Coordinate : Type*}
  [Fintype Fixed] [Fintype Coordinate]

/-- Holding one mean-field factor fixed reduces the supported joint objective
exactly to KL of the coordinate being updated. -/
theorem fep061_meanFieldCoordinate_reduction
    (fixed : FiniteLaw Fixed) (candidate target : FiniteLaw Coordinate)
    (hfixed : ∀ x, 0 < fixed x) (htarget : ∀ y, 0 < target y) :
    meanFieldCoordinateFreeEnergy fixed candidate target =
      FEP.FiniteInformation.finiteKL candidate target :=
  meanFieldCoordinateFreeEnergy_eq fixed candidate target hfixed htarget

/-- The supported target factor is the unique zero-free-energy coordinate
```

```
optimum. -/
theorem fep061_meanFieldCoordinate_optimum_iff
    (fixed : FiniteLaw Fixed) (candidate target : FiniteLaw Coordinate)
    (hfixed : ∀ x, 0 < fixed x) (htarget : ∀ y, 0 < target y) :
    meanFieldCoordinateFreeEnergy fixed candidate target = 0 ↔
      candidate = target :=
  meanFieldCoordinate_optimum_iff fixed candidate target hfixed htarget

end FEP061
```

13.61.2 Typeset statement signatures

$$\{\mathsf{Fixed\ Coordinate} : \mathsf{Type}^*\}$$
$$(\mathsf{fixed} : \mathsf{FiniteLaw\ Fixed})\ (\mathsf{candidate\ target} : \mathsf{FiniteLaw\ Coordinate})$$
$$(\mathsf{hfixed} : \forall x, 0 < \mathsf{fixed}\ x)\ (\mathsf{htarget} : \forall y, 0 < \mathsf{target}\ y)$$
$$\mathsf{meanFieldCoordinateFreeEnergy\ fixed\ candidate\ target}$$
$$= \mathsf{FEP.FiniteInformation.finiteKL\ candidate\ target}$$

$$\{\mathsf{Fixed\ Coordinate} : \mathsf{Type}^*\}$$
$$(\mathsf{fixed} : \mathsf{FiniteLaw\ Fixed})\ (\mathsf{candidate\ target} : \mathsf{FiniteLaw\ Coordinate})$$
$$(\mathsf{hfixed} : \forall x, 0 < \mathsf{fixed}\ x)\ (\mathsf{htarget} : \forall y, 0 < \mathsf{target}\ y)$$
$$\mathsf{meanFieldCoordinateFreeEnergy\ fixed\ candidate\ target} = 0 \leftrightarrow \mathsf{candidate} = \mathsf{target}$$

13.62 fep-062 — Fixed-Sample Importance-Weighted Jensen Bound

For a positive fixed sample count and positive weights, the IID product-law expectation of the log sample-mean weight is at most the log of its expected sample-mean weight. Assumption scope: The carrier is finite, NeZero excludes sample count zero, and every weight is strictly positive before Real.log is used. The positive-expectation witness additionally assumes a nonempty carrier and full-support sampling law. Semantic disposition: `formalized`.

13.62.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP062

open FEP FEP.VariationalDuality Finset
open scoped BigOperators

variable {Sample : Type*} [Fintype Sample]

/-- A positive fixed sample count and the explicit IID product law satisfy the
importance-weighted Jensen bound. -/
theorem fep062_iidProduct_importanceJensen
    (base : FiniteLaw Sample) (sampleCount : ℕ) [NeZero sampleCount]
    (weight : Sample → ℝ) (hweight : ∀ sample, 0 < weight sample) :
    ∑ sampleTuple, iidProductLaw base sampleCount sampleTuple *
        Real.log (sampleMeanWeight sampleCount weight sampleTuple) ≤
      Real.log (∑ sampleTuple, iidProductLaw base sampleCount sampleTuple *
        sampleMeanWeight sampleCount weight sampleTuple) :=
  fixedSampleImportanceJensen base sampleCount weight hweight

/-- For any fixed finite sample law and positive importance weights, expected
log weight is at most the log expected weight. -/
theorem fep062_fixedSample_importanceJensen
    (sampling : FiniteLaw Sample) (weight : Sample → ℝ)
    (hweight : ∀ sample, 0 < weight sample) :
    ∑ sample, sampling sample * Real.log (weight sample) ≤
      Real.log (∑ sample, sampling sample * weight sample) :=
  expected_log_weight_le_log_expected_weight sampling weight hweight
```

```
/-- Strict positivity is visible at the averaging boundary: the weighted
arithmetic mean of positive weights is positive. -/
theorem fep062_expectedImportanceWeight_positive
    [Nonempty Sample]
    (sampling : FiniteLaw Sample) (weight : Sample → ℝ)
    (hsampling : ∀ sample, 0 < sampling sample)
    (hweight : ∀ sample, 0 < weight sample) :
    0 < ∑ sample, sampling sample * weight sample :=
  Finset.sum_pos (fun sample _ =>
    mul_pos (hsampling sample) (hweight sample)) Finset.univ_nonempty

end FEP062
```

13.62.2 Typeset statement signatures

$$\{\mathsf{Sample} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Sample}]$$
$$(\mathsf{base} : \mathsf{FiniteLaw}\ \mathsf{Sample})\ (\mathsf{sampleCount} : \mathbb{N})\ [\mathsf{NeZero}\ \mathsf{sampleCount}]$$
$$(\mathsf{weight} : \mathsf{Sample} \Rightarrow \mathbb{R})\ (\mathsf{hweight} : \forall \mathsf{sample}, 0 < \mathsf{weight}\ \mathsf{sample})$$
$$\sum \mathsf{sampleTuple}, \mathsf{iidProductLaw}\ \mathsf{base}\ \mathsf{sampleCount}\ \mathsf{sampleTuple} \cdot \mathsf{Real.log}$$
$$(\mathsf{sampleMeanWeight}\ \mathsf{sampleCount}\ \mathsf{weight}\ \mathsf{sampleTuple}) \leq \mathsf{Real.log}\ (\sum \mathsf{sampleTuple},$$
$$\mathsf{iidProductLaw}\ \mathsf{base}\ \mathsf{sampleCount}\ \mathsf{sampleTuple} \cdot \mathsf{sampleMeanWeight}\ \mathsf{sampleCount}\ \mathsf{weight}$$
$$\mathsf{sampleTuple})$$

$$\{\mathsf{Sample} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Sample}]$$
$$(\mathsf{sampling} : \mathsf{FiniteLaw}\ \mathsf{Sample})\ (\mathsf{weight} : \mathsf{Sample} \Rightarrow \mathbb{R})$$
$$(\mathsf{hweight} : \forall \mathsf{sample}, 0 < \mathsf{weight}\ \mathsf{sample})$$
$$\sum \mathsf{sample}, \mathsf{sampling}\ \mathsf{sample} \cdot \mathsf{Real.log}\ (\mathsf{weight}\ \mathsf{sample}) \leq \mathsf{Real.log}$$
$$(\sum \mathsf{sample}, \mathsf{sampling}\ \mathsf{sample} \cdot \mathsf{weight}\ \mathsf{sample})$$

$$\{\mathsf{Sample} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Sample}]$$
$$[\mathsf{Nonempty}\ \mathsf{Sample}]\ (\mathsf{sampling} : \mathsf{FiniteLaw}\ \mathsf{Sample})\ (\mathsf{weight} : \mathsf{Sample} \Rightarrow \mathbb{R})$$
$$(\mathsf{hsampling} : \forall \mathsf{sample}, 0 < \mathsf{sampling}\ \mathsf{sample})$$
$$(\mathsf{hweight} : \forall \mathsf{sample}, 0 < \mathsf{weight}\ \mathsf{sample})$$
$$0 < \sum \mathsf{sample}, \mathsf{sampling}\ \mathsf{sample} \cdot \mathsf{weight}\ \mathsf{sample}$$

13.63 fep-063 — Finite-Channel KL Data-Processing Inequality

Passing two finite full-support laws through the same strictly positive finite channel cannot increase their finite KL divergence. Assumption scope: Input is nonempty, both input laws have strict support, and every channel atom is strictly positive; these conditions force reference-predictive support and avoid the project's totalized-log boundary. This is stronger support than the most general measure-theoretic data-processing theorem. Semantic disposition: `conditional_proxy`.

13.63.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP063

open FEP FEP.FiniteInformation FEP.VariationalDuality

variable {Input Output : Type*}
  [Fintype Input] [Fintype Output] [Nonempty Input]

/-- Passing two full-support laws through the same strictly positive finite
channel cannot increase KL divergence. -/
```

```
theorem fep063_finiteChannel_klDataProcessing
    (actual reference : FiniteLaw Input)
    (channel : FiniteKernel Input Output)
    (hactual : ∀ input, 0 < actual input)
    (hreference : ∀ input, 0 < reference input)
    (hchannel : ∀ input output, 0 < channel input output) :
    finiteKL (channel.predictive actual) (channel.predictive reference) ≤
      finiteKL actual reference :=
  finiteChannel_dataProcessing actual reference channel
    hactual hreference hchannel

/-- The same support assumptions make every reference predictive atom
positive, pinning the logarithmic boundary used by the proof. -/
theorem fep063_referencePredictive_fullSupport
    (reference : FiniteLaw Input)
    (channel : FiniteKernel Input Output)
    (hreference : ∀ input, 0 < reference input)
    (hchannel : ∀ input output, 0 < channel input output)
    (output : Output) :
    0 < channel.predictive reference output := by
  simp only [FiniteKernel.predictive_mass]
  exact Finset.sum_pos (fun input _ =>
    mul_pos (hreference input) (hchannel input output))
    Finset.univ_nonempty

end FEP063
```

13.63.2 Typeset statement signatures

$$\{\mathsf{Input}\ \mathsf{Output} : \mathsf{Type}^*\}$$
$$(\mathsf{actual\ reference} : \mathsf{FiniteLaw\ Input})\ (\mathsf{channel} : \mathsf{FiniteKernel\ Input\ Output})$$
$$(\mathsf{hactual} : \forall \mathsf{input}, 0 < \mathsf{actual\ input})$$
$$(\mathsf{hreference} : \forall \mathsf{input}, 0 < \mathsf{reference\ input})$$
$$(\mathsf{hchannel} : \forall \mathsf{input\ output}, 0 < \mathsf{channel\ input\ output})$$
$$\mathsf{finiteKL}\ (\mathsf{channel.predictive\ actual})\ (\mathsf{channel.predictive\ reference}) \le \mathsf{finiteKL}$$
$$\mathsf{actual\ reference}$$

$$\{\mathsf{Input}\ \mathsf{Output} : \mathsf{Type}^*\}$$
$$(\mathsf{reference} : \mathsf{FiniteLaw\ Input})\ (\mathsf{channel} : \mathsf{FiniteKernel\ Input\ Output})$$
$$(\mathsf{hreference} : \forall \mathsf{input}, 0 < \mathsf{reference\ input})$$
$$(\mathsf{hchannel} : \forall \mathsf{input\ output}, 0 < \mathsf{channel\ input\ output})\ (\mathsf{output} : \mathsf{Output})$$
$$0 < \mathsf{channel.predictive\ reference\ output}$$

13.64 fep-064 — Rate–Distortion Lagrangian Weak Duality

Separately certified lower bounds on finite mutual information and expected distortion combine into a lower bound on their rate–distortion Lagrangian for every nonnegative multiplier. Assumption scope: The theorem assumes both component lower bounds rather than deriving a dual function or proving primal or dual optimizer existence. The multiplier is nonnegative, and the zero-multiplier theorem identifies the exact rate-only boundary. Semantic disposition: `structural_proxy`.

13.64.1 Lean sketch

```
import FepSketches.variational_duality

namespace FEP064

open FEP FEP.FiniteInformation FEP.VariationalDuality

variable {Source Code : Type*} [Fintype Source] [Fintype Code]
```

```
/-- Separately certified rate and distortion lower bounds yield a lower bound
on the finite rate--distortion Lagrangian for every nonnegative multiplier. -/
theorem fep064_rateDistortion_weakDuality
    (joint : FiniteLaw (Source × Code))
    (distortion : Source → Code → ℝ)
    (multiplier rateLower distortionLower : ℝ)
    (hmultiplier : 0 ≤ multiplier)
    (hrate : rateLower ≤ mutualInformation joint)
    (hdistortion :
      distortionLower ≤ expectedDistortion joint distortion) :
    rateLower + multiplier * distortionLower ≤
      rateDistortionLagrangian joint distortion multiplier :=
  rateDistortion_weak_duality joint distortion multiplier
    rateLower distortionLower hmultiplier hrate hdistortion

/-- With zero multiplier, the Lagrangian reduces exactly to mutual
information; no optimizer-existence claim is smuggled into weak duality. -/
theorem fep064_zeroMultiplier_boundary
    (joint : FiniteLaw (Source × Code))
    (distortion : Source → Code → ℝ) :
    rateDistortionLagrangian joint distortion 0 =
      mutualInformation joint := by
  simp [rateDistortionLagrangian]

end FEP064
```

13.64.2 Typeset statement signatures

$$\{\mathsf{Source}\ \mathsf{Code} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Source}]\ [\mathsf{Fintype}\ \mathsf{Code}]$$
$$(\mathsf{joint} : \mathsf{FiniteLaw}\ (\mathsf{Source} \times \mathsf{Code}))\ (\mathsf{distortion} : \mathsf{Source} \Rightarrow \mathsf{Code} \Rightarrow \mathbb{R})$$
$$(\mathsf{multiplier}\ \mathsf{rateLower}\ \mathsf{distortionLower} : \mathbb{R})\ (\mathsf{hmultiplier} : 0 \le \mathsf{multiplier})$$
$$(\mathsf{hrate} : \mathsf{rateLower} \le \mathsf{mutualInformation}\ \mathsf{joint})\ (\mathsf{hdistortion} : \mathsf{distortionLower}$$
$$\le \mathsf{expectedDistortion}\ \mathsf{joint}\ \mathsf{distortion})$$
$$\mathsf{rateLower} + \mathsf{multiplier} \cdot \mathsf{distortionLower} \le \mathsf{rateDistortionLagrangian}\ \mathsf{joint}$$
$$\mathsf{distortion}\ \mathsf{multiplier}$$

$$\{\mathsf{Source}\ \mathsf{Code} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Source}]\ [\mathsf{Fintype}\ \mathsf{Code}]$$
$$(\mathsf{joint} : \mathsf{FiniteLaw}\ (\mathsf{Source} \times \mathsf{Code}))\ (\mathsf{distortion} : \mathsf{Source} \Rightarrow \mathsf{Code} \Rightarrow \mathbb{R})$$
$$\mathsf{rateDistortionLagrangian}\ \mathsf{joint}\ \mathsf{distortion}\ 0 = \mathsf{mutualInformation}\ \mathsf{joint}$$

13.65 fep-065 — Controlled-Kernel Normalization

Every action and source state in a finite controlled kernel selects a normalized next-state law. Assumption scope: State and action carriers are finite and ControlledKernel is an action-indexed family of normalized FiniteKernel rows. No policy, controllability, stationarity, or infinite-horizon claim follows from row normalization alone. Semantic disposition: `formalized`.

13.65.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP065

open FEP FEP.ControlledMarkov Finset
open scoped BigOperators

variable {State Action : Type*} [Fintype State] [Fintype Action]

/-- Every action-conditioned Markov row is a normalized finite law. -/
theorem fep065_controlledKernel_normalization
```

```
    (transition : ControlledKernel State Action)
    (action : Action) (state : State) :
    Σ nextState, transition action state nextState = 1 :=
  controlledKernel_sum_one transition action state

end FEP065
```

13.65.2 Typeset statement signatures

$$\{\mathsf{State}\ \mathsf{Action} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]\ [\mathsf{Fintype}\ \mathsf{Action}]$$
$$(\mathsf{transition} : \mathsf{ControlledKernel}\ \mathsf{State}\ \mathsf{Action})\ (\mathsf{action} : \mathsf{Action})\ (\mathsf{state} : \mathsf{State})$$
$$\sum \mathsf{nextState},\ \mathsf{transition}\ \mathsf{action}\ \mathsf{state}\ \mathsf{nextState} = 1$$

13.66 fep-066 — Action-Conditioned Bayesian Belief Update

A positive-evidence action-conditioned prediction and observation update is normalized and reconstructs its predicted state-observation joint mass. Assumption scope: All carriers are finite and the selected action has strictly positive observation evidence. The normalized posterior is not constructed at zero evidence, and the theorem concerns one update rather than policy optimality. Semantic disposition: `formalized`.

13.66.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP066

open FEP FEP.ControlledMarkov Finset
open scoped BigOperators

variable {State Action Observation : Type*}
  [Fintype State] [Fintype Action] [Fintype Observation]

/-- A positive-evidence action-conditioned Bayes update reconstructs its
predicted state-observation joint mass. -/
theorem fep066_actionConditioned_bayes_reconstruction
    (prior : FiniteLaw State) (transition : ControlledKernel State Action)
    (emission : FiniteKernel State Observation) (action : Action)
    (observation : Observation)
    (hEvidence : 0 < actionEvidence prior transition emission action observation)
    (state : State) :
    actionBeliefUpdate prior transition emission action observation hEvidence state *
        actionEvidence prior transition emission action observation =
      actionPrediction prior transition action state * emission state observation :=
  actionBeliefUpdate_reconstruction prior transition emission action observation
    hEvidence state

/-- The same positive-evidence update is normalized. -/
theorem fep066_actionConditioned_update_normalized
    (prior : FiniteLaw State) (transition : ControlledKernel State Action)
    (emission : FiniteKernel State Observation) (action : Action)
    (observation : Observation)
    (hEvidence : 0 < actionEvidence prior transition emission action observation) :
    Σ state,
      actionBeliefUpdate prior transition emission action observation hEvidence state =
        1 :=
  actionBeliefUpdate_sum_one prior transition emission action observation hEvidence

/-- A zero-evidence observation cannot be passed to the normalized update. -/
theorem fep066_zero_evidence_boundary
    (prior : FiniteLaw State) (transition : ControlledKernel State Action)
    (emission : FiniteKernel State Observation) (action : Action)
    (observation : Observation)
```

```
    (hZero : actionEvidence prior transition emission action observation = 0) :
    ¬0 < actionEvidence prior transition emission action observation :=
  actionEvidence_zero_boundary prior transition emission action observation hZero

end FEP066
```

13.66.2 Typeset statement signatures

$$\{\text{State Action Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{ControlledKernel State Action})$$
$$(\text{emission} : \text{FiniteKernel State Observation})\ (\text{action} : \text{Action})$$
$$(\text{observation} : \text{Observation})\ (\text{hEvidence} : 0 < \text{actionEvidence prior transition}$$
$$\text{emission action observation})\ (\text{state} : \text{State})$$
$$\text{actionBeliefUpdate prior transition emission action observation hEvidence state}$$
$$\cdot\ \text{actionEvidence prior transition emission action observation} = \text{actionPrediction}$$
$$\text{prior transition action state} \cdot \text{emission state observation}$$

$$\{\text{State Action Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{ControlledKernel State Action})$$
$$(\text{emission} : \text{FiniteKernel State Observation})\ (\text{action} : \text{Action})$$
$$(\text{observation} : \text{Observation})\ (\text{hEvidence} : 0 < \text{actionEvidence prior transition}$$
$$\text{emission action observation})$$
$$\sum \text{state, actionBeliefUpdate prior transition emission action observation}$$
$$\text{hEvidence state} = 1$$

$$\{\text{State Action Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{ControlledKernel State Action})$$
$$(\text{emission} : \text{FiniteKernel State Observation})\ (\text{action} : \text{Action})$$
$$(\text{observation} : \text{Observation})\ (\text{hZero} : \text{actionEvidence prior transition emission}$$
$$\text{action observation} = 0)$$
$$\neg 0 < \text{actionEvidence prior transition emission action observation}$$

13.67 fep-067 — Reachable Finite-Belief POMDP Reduction

Expanding a finite reachable-belief index into its interpreted finite law preserves every fixed finite-horizon feedback-policy value. Assumption scope: ReachableBeliefPOMDP supplies a finite indexed carrier, an interpretation into FiniteLaw, and sound positive-evidence updates; DecidableEq is required for the belief index. This is not a finite enumeration of the full probability simplex or an infinite-horizon POMDP reduction. Semantic disposition: `structural_proxy`.

13.67.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP067

open FEP FEP.ControlledMarkov

variable {Belief State Action Observation : Type*}
  [Fintype Belief] [Fintype State] [Fintype Action] [Fintype Observation]
  [DecidableEq Belief]

/-- A finite reachable belief index denotes a finite law; expanding that
interpretation preserves every finite-horizon feedback-policy value. -/
theorem fep067_reachableBelief_policyValue_equivalence
    (model : ReachableBeliefPOMDP Belief State Action Observation)
    (stateCost : State → Action → ℝ) (policy : ℕ → Belief → Action)
```

```
    (horizon : ℕ) (belief : Belief) :
    reducedPolicyValue model stateCost policy horizon belief =
      interpretedPolicyValue model stateCost policy horizon belief :=
  reachableBelief_policyValue_eq model stateCost policy horizon belief

/-- The concrete two-index Boolean model satisfies the exact positive-evidence
Bayesian update required by the reachable reduction. -/
theorem fep067_bool_reachable_update
    (belief action observation : Bool)
    (hEvidence : 0 < actionEvidence
      (boolReachablePOMDP.interpret belief) boolReachablePOMDP.transition
      boolReachablePOMDP.emission action observation) :
    boolReachablePOMDP.interpret
        (boolReachablePOMDP.update belief action observation) =
      actionBeliefUpdate (boolReachablePOMDP.interpret belief)
        boolReachablePOMDP.transition boolReachablePOMDP.emission action observation
        hEvidence :=
  boolReachablePOMDP_update_sound belief action observation hEvidence

end FEP067
```

13.67.2 Typeset statement signatures

$$\{\mathsf{Belief}\ \mathsf{State}\ \mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{model} : \mathsf{ReachableBeliefPOMDP}\ \mathsf{Belief}\ \mathsf{State}\ \mathsf{Action}\ \mathsf{Observation})$$
$$(\mathsf{stateCost} : \mathsf{State} \Rightarrow \mathsf{Action} \Rightarrow \mathbb{R})\ (\mathsf{policy} : \mathbb{N} \Rightarrow \mathsf{Belief} \Rightarrow \mathsf{Action})\ (\mathsf{horizon} : \mathbb{N})$$
$$(\mathsf{belief} : \mathsf{Belief})$$
$$\mathsf{reducedPolicyValue}\ \mathsf{model}\ \mathsf{stateCost}\ \mathsf{policy}\ \mathsf{horizon}\ \mathsf{belief}$$
$$= \mathsf{interpretedPolicyValue}\ \mathsf{model}\ \mathsf{stateCost}\ \mathsf{policy}\ \mathsf{horizon}\ \mathsf{belief}$$

$$\{\mathsf{Belief}\ \mathsf{State}\ \mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{belief}\ \mathsf{action}\ \mathsf{observation} : \mathsf{Bool})\ (\mathsf{hEvidence} : 0 < \mathsf{actionEvidence}$$
$$(\mathsf{boolReachablePOMDP.interpret}\ \mathsf{belief})\ \mathsf{boolReachablePOMDP.transition}$$
$$\mathsf{boolReachablePOMDP.emission}\ \mathsf{action}\ \mathsf{observation})$$
$$\mathsf{boolReachablePOMDP.interpret}$$
$$(\mathsf{boolReachablePOMDP.update}\ \mathsf{belief}\ \mathsf{action}\ \mathsf{observation}) = \mathsf{actionBeliefUpdate}$$
$$(\mathsf{boolReachablePOMDP.interpret}\ \mathsf{belief})\ \mathsf{boolReachablePOMDP.transition}$$
$$\mathsf{boolReachablePOMDP.emission}\ \mathsf{action}\ \mathsf{observation}\ \mathsf{hEvidence}$$

13.68 fep-068 — Soft Bellman Recursion

On a nonempty finite action carrier, a positive-temperature soft Bellman partition is positive, its successor value is the exact temperature-scaled log-sum-exp backup, and that soft value is no larger than any hard action energy. Assumption scope: State and action sets are finite, the action carrier is inhabited, and temperature is strictly positive before division, logarithmic aggregation, and the soft-minimum comparison. The statement is an exact finite-horizon recursion and pointwise hard/soft bound, not an infinite-horizon fixed-point or optimality-limit theorem. Semantic disposition: `formalized`.

13.68.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP068

open FEP FEP.ControlledMarkov Finset
open scoped BigOperators

variable {State Action : Type*}
  [Fintype State] [Fintype Action] [Nonempty Action]
```

```
/-- At positive temperature, every finite soft Bellman backup has a positive
partition, satisfies its exact recursion, and lies below each hard action
energy. -/
theorem fep068_softBellman_recursion (temperature : ℝ)
    (hTemperature : 0 < temperature) (stageCost : State → Action → ℝ)
    (transition : ControlledKernel State Action)
    (horizon : ℕ) (state : State) :
    (0 < ∑ action,
      Real.exp (-((softBellmanActionEnergy temperature stageCost transition
        horizon state action) / temperature))) ∧
    (softBellmanValue temperature stageCost transition (horizon + 1) state =
      -temperature * Real.log
        (∑ action,
          Real.exp (-((softBellmanActionEnergy temperature stageCost transition
            horizon state action) / temperature)))) ∧
    ∀ action,
      softBellmanValue temperature stageCost transition (horizon + 1) state ≤
        softBellmanActionEnergy temperature stageCost transition
          horizon state action := by
  exact
    ⟨softBellmanValue_partition_pos temperature stageCost transition horizon state,
      softBellmanValue_succ temperature stageCost transition horizon state,
      fun action ↦ softBellmanValue_le_actionEnergy temperature hTemperature
        stageCost transition horizon state action⟩

end FEP068
```

13.68.2 Typeset statement signatures

$$\{\mathsf{State}\ \mathsf{Action} : \mathsf{Type}^*\}$$
$$(\mathsf{temperature} : \mathbb{R})\ (\mathsf{hTemperature} : 0 < \mathsf{temperature})$$
$$(\mathsf{stageCost} : \mathsf{State} \Rightarrow \mathsf{Action} \Rightarrow \mathbb{R})\ (\mathsf{transition} : \mathsf{ControlledKernel}\ \mathsf{State}\ \mathsf{Action})$$
$$(\mathsf{horizon} : \mathbb{N})\ (\mathsf{state} : \mathsf{State})$$
$$(0 < \sum \mathsf{action}, \mathsf{Real.exp}\ (-((\mathsf{softBellmanActionEnergy}\ \mathsf{temperature}\ \mathsf{stageCost}$$
$$\mathsf{transition}\ \mathsf{horizon}\ \mathsf{state}\ \mathsf{action})/\mathsf{temperature}))) \wedge (\mathsf{softBellmanValue}$$
$$\mathsf{temperature}\ \mathsf{stageCost}\ \mathsf{transition}\ (\mathsf{horizon} + 1)\ \mathsf{state} = -\mathsf{temperature} \cdot \mathsf{Real.log}$$
$$(\sum \mathsf{action}, \mathsf{Real.exp}\ (-((\mathsf{softBellmanActionEnergy}\ \mathsf{temperature}\ \mathsf{stageCost}\ \mathsf{transition}$$
$$\mathsf{horizon}\ \mathsf{state}\ \mathsf{action})/\mathsf{temperature}))) \wedge \forall \mathsf{action}, \mathsf{softBellmanValue}\ \mathsf{temperature}$$
$$\mathsf{stageCost}\ \mathsf{transition}\ (\mathsf{horizon} + 1)\ \mathsf{state} \leq \mathsf{softBellmanActionEnergy}\ \mathsf{temperature}$$
$$\mathsf{stageCost}\ \mathsf{transition}\ \mathsf{horizon}\ \mathsf{state}\ \mathsf{action}$$

13.69 fep-069 — KL-Control Desirability Recursion

One finite desirability backup is an exponential state-cost tilt of a passive transition prediction and preserves nonnegativity. Assumption scope: The theorem gives the algebraic KL-control desirability operator on a finite Markov kernel. It does not derive this operator from a path-space KL objective, prove an optimal controlled kernel, or establish convergence of iteration. Semantic disposition: `structural_proxy`.

13.69.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP069

open FEP FEP.ControlledMarkov Finset
open scoped BigOperators

variable {State : Type*} [Fintype State]

/-- One KL-control desirability backup is an exponential state-cost tilt of
```

```
the passive-kernel prediction. -/
theorem fep069_desirability_recursion
    (passive : FiniteKernel State State) (stateCost desirability : State → ℝ)
    (state : State) :
    desirabilityStep passive stateCost desirability state =
      Real.exp (-stateCost state) *
        Σ nextState, passive state nextState * desirability nextState :=
  rfl

/-- The desirability recursion preserves nonnegativity. -/
theorem fep069_desirability_nonnegative
    (passive : FiniteKernel State State) (stateCost desirability : State → ℝ)
    (hDesirability : ∀ state, 0 ≤ desirability state) (state : State) :
    0 ≤ desirabilityStep passive stateCost desirability state :=
  desirabilityStep_nonneg passive stateCost desirability hDesirability state

/-- Zero state cost and unit desirability give an exact normalized fixed point. -/
theorem fep069_zeroCost_unitDesirability
    (passive : FiniteKernel State State) (state : State) :
    desirabilityStep passive (fun _ => 0) (fun _ => 1) state = 1 :=
  desirabilityStep_zero_cost_one passive state

end FEP069
```

13.69.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{passive} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\ (\mathsf{stateCost}\ \mathsf{desirability} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{state} : \mathsf{State})$$
$$\mathsf{desirabilityStep}\ \mathsf{passive}\ \mathsf{stateCost}\ \mathsf{desirability}\ \mathsf{state} = \mathsf{Real.exp}$$
$$(-\mathsf{stateCost}\ \mathsf{state}) \cdot \sum \mathsf{nextState},\ \mathsf{passive}\ \mathsf{state}\ \mathsf{nextState} \cdot \mathsf{desirability}$$
$$\mathsf{nextState}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{passive} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\ (\mathsf{stateCost}\ \mathsf{desirability} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hDesirability} : \forall \mathsf{state}, 0 \le \mathsf{desirability}\ \mathsf{state})\ (\mathsf{state} : \mathsf{State})$$
$$0 \le \mathsf{desirabilityStep}\ \mathsf{passive}\ \mathsf{stateCost}\ \mathsf{desirability}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{passive} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{desirabilityStep}\ \mathsf{passive}\ (\lambda\ \_ \mapsto 0)\ (\lambda\ \_ \mapsto 1)\ \mathsf{state} = 1$$

13.70 fep-070 — Control-as-Inference Policy Posterior

A prior-weighted exponential action score normalizes to a control posterior whose mass reconstructs the unnormalized score. Assumption scope: The action carrier is finite, precision and costs are real, and positivity of the exponential partition is supplied by the finite Boltzmann construction. The result is an exact algebraic posterior but does not derive its cost from a full generative observation model. Semantic disposition: `structural_proxy`.

13.70.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP070

open FEP FEP.ControlledMarkov Finset
open scoped BigOperators

variable {Action : Type*} [Fintype Action]
```

```
/-- The prior-weighted exponential control posterior is a normalized finite
action law. -/
theorem fep070_controlPosterior_normalized (prior : FiniteLaw Action)
    (precision : ℝ) (cost : Action → ℝ) :
    ∑ action, controlPosterior prior precision cost action = 1 :=
  controlPosterior_sum_one prior precision cost

/-- Posterior control mass reconstructs its prior-weighted exponential score. -/
theorem fep070_controlPosterior_reconstruction (prior : FiniteLaw Action)
    (precision : ℝ) (cost : Action → ℝ) (action : Action) :
    controlPosterior prior precision cost action *
        boltzmannPartition prior (fun candidate => precision * cost candidate) =
      boltzmannWeight prior (fun candidate => precision * cost candidate) action :=
  boltzmannPosterior_mul_partition prior
    (fun candidate => precision * cost candidate) action

/-- Zero control cost leaves the prior action law unchanged. -/
theorem fep070_zeroCost_posterior_eq_prior (prior : FiniteLaw Action)
    (precision : ℝ) :
    controlPosterior prior precision (fun _ => 0) = prior :=
  controlPosterior_zero_cost prior precision

end FEP070
```

13.70.2 Typeset statement signatures

$$\{\mathsf{Action} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Action}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{Action})\ (\mathsf{precision} : \mathbb{R})\ (\mathsf{cost} : \mathsf{Action} \Rightarrow \mathbb{R})$$
$$\sum \mathsf{action}, \mathsf{controlPosterior}\ \mathsf{prior}\ \mathsf{precision}\ \mathsf{cost}\ \mathsf{action} = 1$$

$$\{\mathsf{Action} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Action}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{Action})\ (\mathsf{precision} : \mathbb{R})\ (\mathsf{cost} : \mathsf{Action} \Rightarrow \mathbb{R})\ (\mathsf{action} : \mathsf{Action})$$
$$\mathsf{controlPosterior}\ \mathsf{prior}\ \mathsf{precision}\ \mathsf{cost}\ \mathsf{action} \cdot \mathsf{boltzmannPartition}\ \mathsf{prior}$$
$$(\lambda\, \mathsf{candidate} \mapsto \mathsf{precision} \cdot \mathsf{cost}\ \mathsf{candidate}) = \mathsf{boltzmannWeight}\ \mathsf{prior}$$
$$(\lambda\, \mathsf{candidate} \mapsto \mathsf{precision} \cdot \mathsf{cost}\ \mathsf{candidate})\ \mathsf{action}$$

$$\{\mathsf{Action} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Action}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{Action})\ (\mathsf{precision} : \mathbb{R})$$
$$\mathsf{controlPosterior}\ \mathsf{prior}\ \mathsf{precision}\ (\lambda\, \_ \mapsto 0) = \mathsf{prior}$$

13.71 fep-071 — Sophisticated Expected-Free-Energy Backward Induction

Finite sophisticated expected-free-energy recursion reoptimizes after each observation-dependent belief update, and a two-stage witness strictly separates feedback from every fixed open-loop second action. Assumption scope: Belief, action, and observation carriers are finite and Action is nonempty so each finite argmin exists. The theorem is a maintained finite-horizon recursion with an exact Boolean two-stage witness; it does not prove continuous-state or infinite-horizon sophisticated planning. Semantic disposition: `conditional_proxy`.

13.71.1 Lean sketch

```
import FepSketches.controlled_markov

namespace FEP071

open FEP FEP.ControlledMarkov Finset
open scoped BigOperators

variable {Belief Action Observation : Type*}
  [Fintype Belief] [Fintype Action] [Fintype Observation] [Nonempty Action]
```

```
/-- Sophisticated expected free energy applies a new finite minimum after the
observation-dependent continuation at every backward-induction node. -/
theorem fep071_sophisticatedEFE_backward_step
    (model : SophisticatedEFEModel Belief Action Observation)
    (horizon : ℕ) (belief : Belief) :
    sophisticatedEFEValue model (horizon + 1) belief =
      model.stageEFE horizon belief (sophisticatedEFEAction model horizon belief) +
        Σ observation,
          model.observationLaw belief (sophisticatedEFEAction model horizon belief)
              observation *
            sophisticatedEFEValue model horizon
              (model.update belief (sophisticatedEFEAction model horizon belief)
                observation) :=
  sophisticatedEFEValue_succ model horizon belief

/-- In the exact two-stage Boolean witness, different observations select
different second-stage actions. -/
theorem fep071_twoStage_feedback_changes_action :
    twoStageFeedback false ≠ twoStageFeedback true :=
  twoStageFeedback_changes_action

/-- The observation-dependent second action strictly improves on every fixed
open-loop Boolean action. -/
theorem fep071_twoStage_feedback_strictly_better (action : Bool) :
    twoStageFeedbackExpectedCost < twoStageOpenLoopExpectedCost action :=
  twoStageFeedback_beats_openLoop action

end FEP071
```

13.71.2 Typeset statement signatures

$$\{\text{Belief Action Observation} : \text{Type}^*\}$$
$$(\text{model} : \text{SophisticatedEFEModel Belief Action Observation})\ (\text{horizon} : \mathbb{N})$$
$$(\text{belief} : \text{Belief})$$
$$\text{sophisticatedEFEValue model}\ (\text{horizon} + 1)\ \text{belief} = \text{model.stageEFE horizon belief}$$
$$(\text{sophisticatedEFEAction model horizon belief}) + \sum \text{observation},$$
$$\text{model.observationLaw belief}\ (\text{sophisticatedEFEAction model horizon belief})$$
$$\text{observation} \cdot \text{sophisticatedEFEValue model horizon}\ (\text{model.update belief}$$
$$(\text{sophisticatedEFEAction model horizon belief})\ \text{observation})$$

$$\{\text{Belief Action Observation} : \text{Type}^*\}$$
$$\text{twoStageFeedback false} \neq \text{twoStageFeedback true}$$

$$\{\text{Belief Action Observation} : \text{Type}^*\}$$
$$(\text{action} : \text{Bool})$$
$$\text{twoStageFeedbackExpectedCost} < \text{twoStageOpenLoopExpectedCost action}$$

13.72 fep-072 — Hidden-Markov Forward Filtering Recursion

One positive-evidence finite hidden-Markov filtering step reconstructs its predicted state-observation joint, with an exact asymmetric Boolean posterior witness. Assumption scope: State and observation carriers are finite, transition and emission rows are normalized, and observed evidence is strictly positive. No filter is claimed at zero evidence and no continuous-state or Kalman–Bucy limit is inferred. Semantic disposition: `formalized`.

13.72.1 Lean sketch

```
import FepSketches.temporal_inference
```

```
namespace FEP072

open FEP FEP.TemporalInference Finset
open scoped BigOperators

variable {State Observation : Type*}
  [Fintype State] [Fintype Observation]

/-- One positive-evidence hidden-Markov forward update reconstructs the
predicted state-observation joint. -/
theorem fep072_forward_filtering_recursion (prior : FiniteLaw State)
    (transition : FiniteKernel State State)
    (emission : FiniteKernel State Observation) (observation : Observation)
    (hEvidence : 0 < forwardEvidence prior transition emission observation)
    (state : State) :
    forwardFilter prior transition emission observation hEvidence state *
        forwardEvidence prior transition emission observation =
      forwardPrediction prior transition state * emission state observation :=
  forwardFilter_reconstruction prior transition emission observation hEvidence state

/-- The exact asymmetric Boolean filter assigns mass `20/23` to `true`. -/
theorem fep072_bool_forward_mass : boolForwardFilter true = 20 / 23 :=
  boolForwardFilter_true_mass

/-- Zero evidence is outside the normalized forward-filter construction. -/
theorem fep072_zero_evidence_boundary (prior : FiniteLaw State)
    (transition : FiniteKernel State State)
    (emission : FiniteKernel State Observation) (observation : Observation)
    (hZero : forwardEvidence prior transition emission observation = 0) :
    ¬0 < forwardEvidence prior transition emission observation :=
  forwardEvidence_zero_boundary prior transition emission observation hZero

end FEP072
```

13.72.2 Typeset statement signatures

$$\{\text{State Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{FiniteKernel State State})$$
$$(\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})$$
$$(\text{hEvidence} : 0 < \text{forwardEvidence prior transition emission observation})$$
$$(\text{state} : \text{State})$$
$$\text{forwardFilter prior transition emission observation hEvidence state}$$
$$\cdot\ \text{forwardEvidence prior transition emission observation} = \text{forwardPrediction}$$
$$\text{prior transition state} \cdot \text{emission state observation}$$

$$\{\text{State Observation} : \text{Type}^*\}$$
$$\text{boolForwardFilter true} = 20/23$$

$$\{\text{State Observation} : \text{Type}^*\}$$
$$(\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{FiniteKernel State State})$$
$$(\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})\ (\text{hZero} :$$
$$\text{forwardEvidence prior transition emission observation} = 0)$$
$$\neg 0 < \text{forwardEvidence prior transition emission observation}$$

13.73 fep-073 — Backward Information-Message Recursion

Finite backward information messages obey the exact future-observation recursion and preserve nonnegativity from one message to the preceding one. Assumption scope: State and observation carriers are finite and transition and emission rows are normalized. Nonnegativity requires the

supplied next message to be pointwise nonnegative; the theorem does not normalize messages or assert a limiting smoother. Semantic disposition: `formalized`.

13.73.1 Lean sketch

```
import FepSketches.temporal_inference

namespace FEP073

open FEP FEP.TemporalInference

variable {State Observation : Type*}
  [Fintype State] [Fintype Observation]

/-- Backward information messages obey the exact finite future-observation
recursion. -/
theorem fep073_backward_message_recursion
    (transition : FiniteKernel State State)
    (emission : FiniteKernel State Observation) (observation : Observation)
    (future : List Observation) (state : State) :
    backwardMessage transition emission (observation :: future) state =
      backwardMessageStep transition emission observation
        (backwardMessage transition emission future) state :=
  backwardMessage_cons transition emission observation future state

/-- Nonnegative terminal information stays nonnegative under a backward step. -/
theorem fep073_backward_message_nonnegative
    (transition : FiniteKernel State State)
    (emission : FiniteKernel State Observation) (observation : Observation)
    (nextMessage : State → ℝ) (hMessage : ∀ state, 0 ≤ nextMessage state)
    (state : State) :
    0 ≤ backwardMessageStep transition emission observation nextMessage state :=
  backwardMessage_nonneg transition emission observation nextMessage hMessage state

end FEP073
```

13.73.2 Typeset statement signatures

$$
\begin{aligned}
&\{\mathsf{State}\ \mathsf{Observation} : \mathsf{Type}^{*}\}\\
&(\mathsf{transition} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\\
&(\mathsf{emission} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{Observation})\ (\mathsf{observation} : \mathsf{Observation})\\
&(\mathsf{future} : \mathsf{List}\ \mathsf{Observation})\ (\mathsf{state} : \mathsf{State})\\
&\mathsf{backwardMessage}\ \mathsf{transition}\ \mathsf{emission}\ (\mathsf{observation} :: \mathsf{future})\ \mathsf{state}\\
&\quad = \mathsf{backwardMessageStep}\ \mathsf{transition}\ \mathsf{emission}\ \mathsf{observation}\\
&\quad (\mathsf{backwardMessage}\ \mathsf{transition}\ \mathsf{emission}\ \mathsf{future})\ \mathsf{state}
\end{aligned}
$$

$$
\begin{aligned}
&\{\mathsf{State}\ \mathsf{Observation} : \mathsf{Type}^{*}\}\\
&(\mathsf{transition} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\\
&(\mathsf{emission} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{Observation})\ (\mathsf{observation} : \mathsf{Observation})\\
&(\mathsf{nextMessage} : \mathsf{State} \Rightarrow \mathbb{R})\ (\mathsf{hMessage} : \forall \mathsf{state}, 0 \le \mathsf{nextMessage}\ \mathsf{state})\\
&(\mathsf{state} : \mathsf{State})\\
&0 \le \mathsf{backwardMessageStep}\ \mathsf{transition}\ \mathsf{emission}\ \mathsf{observation}\ \mathsf{nextMessage}\ \mathsf{state}
\end{aligned}
$$

13.74 fep-074 — Forward–Backward Smoothing Factorization

At positive normalizer, a finite smoothing marginal factors into forward law times backward information, and one-step forward and backward computations agree on observation evidence. Assumption scope: The forward law is normalized, the backward message is nonnegative, and its weighted normalizer is strictly positive. The statement covers a finite factorization and does not assert positivity when all backward weights vanish. Semantic disposition: `formalized`.

13.74.1 Lean sketch

```
import FepSketches.temporal_inference

namespace FEP074

open FEP FEP.TemporalInference

variable {State : Type*} [Fintype State]

/-- At positive normalizer, a smoothing marginal factors exactly into its
forward law and backward information message. -/
theorem fep074_forwardBackward_smoothing_factorization
    (filtered : FiniteLaw State) (backward : State → ℝ)
    (hBackward : ∀ state, 0 ≤ backward state)
    (hNormalizer : 0 < smoothingNormalizer filtered backward) (state : State) :
    forwardBackwardSmoothing filtered backward hBackward hNormalizer state *
        smoothingNormalizer filtered backward =
      filtered state * backward state :=
  forwardBackwardSmoothing_factorization filtered backward hBackward hNormalizer state

/-- Forward and backward evaluation agree on one-step observation evidence. -/
theorem fep074_forward_backward_evidence_agreement
    {Observation : Type*} [Fintype Observation]
    (prior : FiniteLaw State) (transition : FiniteKernel State State)
    (emission : FiniteKernel State Observation) (observation : Observation) :
    forwardEvidence prior transition emission observation =
      backwardEvidence prior transition emission observation :=
  forward_backward_evidence_agree prior transition emission observation

end FEP074
```

13.74.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{filtered} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{backward} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hBackward} : \forall \mathsf{state}, 0 \leq \mathsf{backward}\ \mathsf{state})\ (\mathsf{hNormalizer} : 0 < \mathsf{smoothingNormalizer}$$
$$\mathsf{filtered}\ \mathsf{backward})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{forwardBackwardSmoothing}\ \mathsf{filtered}\ \mathsf{backward}\ \mathsf{hBackward}\ \mathsf{hNormalizer}\ \mathsf{state}$$
$$\cdot\ \mathsf{smoothingNormalizer}\ \mathsf{filtered}\ \mathsf{backward} = \mathsf{filtered}\ \mathsf{state} \cdot \mathsf{backward}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$\{\mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Observation}]\ (\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$(\mathsf{transition} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})$$
$$(\mathsf{emission} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{Observation})\ (\mathsf{observation} : \mathsf{Observation})$$
$$\mathsf{forwardEvidence}\ \mathsf{prior}\ \mathsf{transition}\ \mathsf{emission}\ \mathsf{observation} = \mathsf{backwardEvidence}\ \mathsf{prior}$$
$$\mathsf{transition}\ \mathsf{emission}\ \mathsf{observation}$$

13.75 fep-075 — Smoothing-Marginal Normalization

A nonnegative forward–backward product with positive normalizer defines a normalized finite smoothing marginal. Assumption scope: The state carrier is finite, backward weights are pointwise nonnegative, and their forward expectation is strictly positive. The construction explicitly excludes a zero normalizer rather than selecting an arbitrary fallback law. Semantic disposition: `formalized`.

13.75.1 Lean sketch

```
import FepSketches.temporal_inference

namespace FEP075
```

```
open FEP FEP.TemporalInference Finset
open scoped BigOperators

variable {State : Type*} [Fintype State]

/-- A positive-normalizer forward--backward smoothing marginal is normalized. -/
theorem fep075_smoothing_marginal_normalization
    (filtered : FiniteLaw State) (backward : State → ℝ)
    (hBackward : ∀ state, 0 ≤ backward state)
    (hNormalizer : 0 < smoothingNormalizer filtered backward) :
    ∑ state, forwardBackwardSmoothing filtered backward hBackward hNormalizer state =
      1 :=
  forwardBackwardSmoothing_sum_one filtered backward hBackward hNormalizer

/-- The nontrivial Boolean smoother normalizes with masses `7/46` and `39/46`. -/
theorem fep075_bool_smoothing_normalization :
    boolSmoothing false + boolSmoothing true = 1 :=
  boolSmoothing_sum_one

/-- A zero forward--backward normalizer is explicitly outside the smoother's
construction boundary. -/
theorem fep075_zero_normalizer_boundary (filtered : FiniteLaw State)
    (backward : State → ℝ) (hZero : smoothingNormalizer filtered backward = 0) :
    ¬0 < smoothingNormalizer filtered backward :=
  smoothingNormalizer_zero_boundary filtered backward hZero

end FEP075
```

13.75.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{filtered} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{backward} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hBackward} : \forall \mathsf{state}, 0 \leq \mathsf{backward}\ \mathsf{state})\ (\mathsf{hNormalizer} : 0 < \mathsf{smoothingNormalizer}$$
$$\mathsf{filtered}\ \mathsf{backward})$$
$$\sum \mathsf{state}, \mathsf{forwardBackwardSmoothing}\ \mathsf{filtered}\ \mathsf{backward}\ \mathsf{hBackward}\ \mathsf{hNormalizer}\ \mathsf{state}$$
$$= 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$\mathsf{boolSmoothing}\ \mathsf{false} + \mathsf{boolSmoothing}\ \mathsf{true} = 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{filtered} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{backward} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hZero} : \mathsf{smoothingNormalizer}\ \mathsf{filtered}\ \mathsf{backward} = 0)$$
$$\neg 0 < \mathsf{smoothingNormalizer}\ \mathsf{filtered}\ \mathsf{backward}$$

13.76 fep-076 — One-Step Normalized Variational State Update

A finite prior-weighted exponential energy tilt is normalized and reconstructs its unnormalized weight, with zero energy acting as the identity update. Assumption scope: The state carrier is finite and the real exponential makes the partition positive without a separate support premise. This is one exact normalized update, not convergence of a variational optimization algorithm. Semantic disposition: `formalized`.

13.76.1 Lean sketch

```
import FepSketches.temporal_inference

namespace FEP076

open FEP FEP.TemporalInference FEP.ControlledMarkov Finset
open scoped BigOperators
```

```
variable {State : Type*} [Fintype State]

/-- A prior-weighted one-step variational state update is a normalized finite
law. -/
theorem fep076_variational_state_update_normalized
    (prior : FiniteLaw State) (energy : State → ℝ) :
    Σ state, variationalStateUpdate prior energy state = 1 :=
  variationalStateUpdate_sum_one prior energy

/-- Normalized state mass reconstructs the corresponding unnormalized
variational tilt. -/
theorem fep076_variational_state_update_reconstruction
    (prior : FiniteLaw State) (energy : State → ℝ) (state : State) :
    variationalStateUpdate prior energy state * boltzmannPartition prior energy =
      boltzmannWeight prior energy state :=
  variationalStateUpdate_reconstruction prior energy state

/-- Zero state energy is the identity variational update. -/
theorem fep076_zeroEnergy_update_eq_prior (prior : FiniteLaw State) :
    variationalStateUpdate prior (fun _ => 0) = prior :=
  variationalStateUpdate_zero_energy prior

end FEP076
```

13.76.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{energy} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\sum \mathsf{state},\ \mathsf{variationalStateUpdate}\ \mathsf{prior}\ \mathsf{energy}\ \mathsf{state} = 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{energy} : \mathsf{State} \Rightarrow \mathbb{R})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{variationalStateUpdate}\ \mathsf{prior}\ \mathsf{energy}\ \mathsf{state} \cdot \mathsf{boltzmannPartition}\ \mathsf{prior}\ \mathsf{energy}$$
$$= \mathsf{boltzmannWeight}\ \mathsf{prior}\ \mathsf{energy}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$\mathsf{variationalStateUpdate}\ \mathsf{prior}\ (\lambda\ \_ \mapsto 0) = \mathsf{prior}$$

13.77 fep-077 — Two-Level Hierarchical Predictive Factorization

Two-level finite hierarchical prediction is exactly prediction through the chronological composition of its two normalized kernels. Assumption scope: Upper, middle, and observation carriers are finite and both conditional maps are FiniteKernel values. The result establishes two-level factorization only, not identifiability of latent levels or an unbounded hierarchy. Semantic disposition: `formalized`.

13.77.1 Lean sketch

```
import FepSketches.temporal_inference

namespace FEP077

open FEP FEP.TemporalInference

variable {Upper Middle Observation : Type*}
  [Fintype Upper] [Fintype Middle] [Fintype Observation]

/-- Two-level hierarchical prediction equals prediction through the composed
normalized kernel. -/
theorem fep077_hierarchical_predictive_factorization
```

```
    (top : FiniteLaw Upper) (upperKernel : FiniteKernel Upper Middle)
    (lowerKernel : FiniteKernel Middle Observation) :
    hierarchicalPredictive top upperKernel lowerKernel =
      (FiniteKernel.comp lowerKernel upperKernel).predictive top :=
  hierarchicalPredictive_eq top upperKernel lowerKernel

end FEP077
```

13.77.2 Typeset statement signatures

$$\{\text{Upper Middle Observation} : \text{Type}^*\}$$
$$(\text{top} : \text{FiniteLaw Upper})\ (\text{upperKernel} : \text{FiniteKernel Upper Middle})$$
$$(\text{lowerKernel} : \text{FiniteKernel Middle Observation})$$
$$\text{hierarchicalPredictive top upperKernel lowerKernel}$$
$$= (\text{FiniteKernel.comp lowerKernel upperKernel}).\text{predictive top}$$

13.78 fep-078 — Bayesian Model-Averaging Predictive Law

Bayesian model averaging is the prior-weighted finite mixture of model predictive laws and remains normalized. Assumption scope: Model and observation sets are finite, the model prior is normalized, and each model predictive row is normalized. No posterior model update, asymptotic selection, or continuous model class is claimed. Semantic disposition: `formalized`.

13.78.1 Lean sketch

```
import FepSketches.temporal_inference

namespace FEP078

open FEP FEP.TemporalInference Finset
open scoped BigOperators

variable {Model Observation : Type*} [Fintype Model] [Fintype Observation]

/-- Bayesian model averaging is the finite prior-weighted predictive mixture. -/
theorem fep078_modelAverage_predictive_law (modelPrior : FiniteLaw Model)
    (modelPredictive : FiniteKernel Model Observation)
    (observation : Observation) :
    modelAverage modelPrior modelPredictive observation =
      ∑ model, modelPrior model * modelPredictive model observation :=
  modelAverage_mass modelPrior modelPredictive observation

/-- The Bayesian model-averaged predictive mixture remains normalized. -/
theorem fep078_modelAverage_normalized (modelPrior : FiniteLaw Model)
    (modelPredictive : FiniteKernel Model Observation) :
    ∑ observation, modelAverage modelPrior modelPredictive observation = 1 :=
  modelAverage_sum_one modelPrior modelPredictive

/-- A nonuniform Boolean model prior and informative predictive kernel yield
the exact nontrivial mixture mass `13/20` on `true`. -/
theorem fep078_bool_modelAverage_true_mass :
    modelAverage boolInitialLaw boolAccurateEmission true = 13 / 20 :=
  boolModelAverage_true_mass

end FEP078
```

13.78.2 Typeset statement signatures

$$\{\mathsf{Model}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Model}]\ [\mathsf{Fintype}\ \mathsf{Observation}]$$
$$(\mathsf{modelPrior} : \mathsf{FiniteLaw}\ \mathsf{Model})$$
$$(\mathsf{modelPredictive} : \mathsf{FiniteKernel}\ \mathsf{Model}\ \mathsf{Observation})\ (\mathsf{observation} : \mathsf{Observation})$$
$$\mathsf{modelAverage}\ \mathsf{modelPrior}\ \mathsf{modelPredictive}\ \mathsf{observation} = \sum \mathsf{model},\ \mathsf{modelPrior}\ \mathsf{model}$$
$$\cdot\ \mathsf{modelPredictive}\ \mathsf{model}\ \mathsf{observation}$$

$$\{\mathsf{Model}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Model}]\ [\mathsf{Fintype}\ \mathsf{Observation}]$$
$$(\mathsf{modelPrior} : \mathsf{FiniteLaw}\ \mathsf{Model})$$
$$(\mathsf{modelPredictive} : \mathsf{FiniteKernel}\ \mathsf{Model}\ \mathsf{Observation})$$
$$\sum \mathsf{observation},\ \mathsf{modelAverage}\ \mathsf{modelPrior}\ \mathsf{modelPredictive}\ \mathsf{observation} = 1$$

$$\{\mathsf{Model}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Model}]\ [\mathsf{Fintype}\ \mathsf{Observation}]$$
$$\mathsf{modelAverage}\ \mathsf{boolInitialLaw}\ \mathsf{boolAccurateEmission}\ \mathsf{true} = 13/20$$

13.79 fep-079 — Blanket Factorization and Conditional-Mutual-Information Equivalence

On a finite conditional-blanket carrier with full blanket support, vanishing conditional mutual information is equivalent to pointwise internal–external factorization. Assumption scope: Blanket, internal, and external carriers are finite and every blanket atom is strictly positive for the equivalence. The static-model specialization proves zero conditional mutual information even with null blanket atoms, but the theorem does not identify a general physical Markov blanket. Semantic disposition: `conditional_proxy`.

13.79.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP079

open FEP FEP.CausalDynamics FEP.FiniteInformation FEP.MarkovBlanket

variable {Blanket Internal External Sensory Active : Type*}
  [Fintype Blanket] [Fintype Internal] [Fintype External]
  [Fintype Sensory] [Fintype Active]

/-- Under full blanket support, vanishing finite conditional mutual information
is equivalent to pointwise internal-external conditional factorization. -/
theorem fep079_blanketFactorization_iff_conditionalMutualInformation_zero
    (model : ConditionalBlanketModel Blanket Internal External)
    (hBlanket : ∀ blanket, 0 < model.blanketLaw blanket) :
    conditionalMutualInformation model = 0 ↔ Factorizes model :=
  conditionalMutualInformation_eq_zero_iff_factorizes model hBlanket

/-- A constructed static Markov-blanket model has zero conditional mutual
information even if some blanket values have zero probability. -/
theorem fep079_staticBlanket_conditionalMutualInformation_zero
    (model : StaticModel Internal Sensory Active External) :
    conditionalMutualInformation (ofStaticModel model) = 0 :=
  ofStaticModel_conditionalMutualInformation_zero model

end FEP079
```

13.79.2 Typeset statement signatures

$$\{\mathsf{Blanket}\ \mathsf{Internal}\ \mathsf{External}\ \mathsf{Sensory}\ \mathsf{Active} : \mathsf{Type}^*\}$$
$$(\mathsf{model} : \mathsf{ConditionalBlanketModel}\ \mathsf{Blanket}\ \mathsf{Internal}\ \mathsf{External})$$
$$(\mathsf{hBlanket} : \forall \mathsf{blanket}, 0 < \mathsf{model.blanketLaw}\ \mathsf{blanket})$$
$$\mathsf{conditionalMutualInformation}\ \mathsf{model} = 0 \leftrightarrow \mathsf{Factorizes}\ \mathsf{model}$$

$$\{\mathsf{Blanket\ Internal\ External\ Sensory\ Active} : \mathsf{Type}^*\}$$
$$(\mathsf{model} : \mathsf{StaticModel\ Internal\ Sensory\ Active\ External})$$
$$\mathsf{conditionalMutualInformation}\ (\mathsf{ofStaticModel\ model}) = 0$$

13.80 fep-080 — Shared-Conditional Mixture Preservation

Convex mixing of two finite input laws commutes exactly with forming their joint laws when both components share the same conditional kernel. Assumption scope: Input and output carriers are finite, the mixing weight lies in the closed unit interval, and both components use one identical normalized kernel. The result does not cover mixtures whose conditional mechanisms differ. Semantic disposition: `formalized`.

13.80.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP080

open FEP FEP.CausalDynamics

variable {Input Output : Type*} [Fintype Input] [Fintype Output]

/-- Convex mixing commutes with formation of a joint law whenever both
components share the same conditional kernel. -/
theorem fep080_sharedConditional_mixture_preservation
    (weight : ℝ) (hWeightNonneg : 0 ≤ weight)
    (hWeightLeOne : weight ≤ 1)
    (left right : FiniteLaw Input) (kernel : FiniteKernel Input Output) :
    kernel.joint (mixLaw weight hWeightNonneg hWeightLeOne left right) =
      mixLaw weight hWeightNonneg hWeightLeOne
        (kernel.joint left) (kernel.joint right) :=
  sharedConditional_mixture_preserves_joint
    weight hWeightNonneg hWeightLeOne left right kernel

/-- The half-mixture witness puts positive, unequal-component mass on both
Boolean atoms. -/
theorem fep080_boolHalfMixture_nonvacuity :
    boolHalfMixture false = 1 / 2 ∧ boolHalfMixture true = 1 / 2 :=
  ⟨boolHalfMixture_false, boolHalfMixture_true⟩

end FEP080
```

13.80.2 Typeset statement signatures

$$\{\mathsf{Input\ Output} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Input}]\ [\mathsf{Fintype\ Output}]$$
$$(\mathsf{weight} : \mathbb{R})\ (\mathsf{hWeightNonneg} : 0 \le \mathsf{weight})\ (\mathsf{hWeightLeOne} : \mathsf{weight} \le 1)$$
$$(\mathsf{left\ right} : \mathsf{FiniteLaw\ Input})\ (\mathsf{kernel} : \mathsf{FiniteKernel\ Input\ Output})$$
$$\mathsf{kernel.joint}\ (\mathsf{mixLaw\ weight\ hWeightNonneg\ hWeightLeOne\ left\ right}) = \mathsf{mixLaw}$$
$$\mathsf{weight\ hWeightNonneg\ hWeightLeOne}\ (\mathsf{kernel.joint\ left})\ (\mathsf{kernel.joint\ right})$$

$$\{\mathsf{Input\ Output} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Input}]\ [\mathsf{Fintype\ Output}]$$
$$\mathsf{boolHalfMixture\ false} = 1/2 \wedge \mathsf{boolHalfMixture\ true} = 1/2$$

13.81 fep-081 — Coupled-Subsystem Blanket Composition

Two conditionally factorized subsystems retain zero paired-row mutual information under an arbitrarily coupled joint blanket law. Assumption scope: Every carrier is finite and subsystem internal and external kernels factor by their own blanket coordinates. The conclusion is rowwise conditional mutual information for paired variables; it does not assert that the joint blanket factorizes or that the complete subsystems are marginally independent. Semantic disposition: `structural_proxy`.

13.81.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP081

open FEP FEP.CausalDynamics FEP.FiniteInformation FEP.MarkovBlanket

variable {Blanket₁ Blanket₂ Internal₁ Internal₂ External₁ External₂ : Type*}
  [Fintype Blanket₁] [Fintype Blanket₂]
  [Fintype Internal₁] [Fintype Internal₂]
  [Fintype External₁] [Fintype External₂]

/-- Two subsystems may have an arbitrarily coupled joint blanket law while
their paired internal and external rows retain zero conditional mutual
information. -/
theorem fep081_coupledSubsystem_blanketComposition
    (blanketLaw : FiniteLaw (Blanket₁ × Blanket₂))
    (internal₁ : FiniteKernel Blanket₁ Internal₁)
    (internal₂ : FiniteKernel Blanket₂ Internal₂)
    (external₁ : FiniteKernel Blanket₁ External₁)
    (external₂ : FiniteKernel Blanket₂ External₂)
    (blanket : Blanket₁ × Blanket₂) :
    mutualInformation
        (conditionalJoint
          (coupledBlanketModel blanketLaw internal₁ internal₂ external₁ external₂)
          blanket) = 0 :=
  coupledBlanket_conditionalMutualInformation_zero
    blanketLaw internal₁ internal₂ external₁ external₂ blanket

/-- The coupled-blanket carrier admits a non-product-looking diagonal Boolean
witness with zero cross mass. -/
theorem fep081_correlatedBlanket_nonvacuity :
    correlatedBoolBlanket (false, false) = 1 / 2 ∧
      correlatedBoolBlanket (true, true) = 1 / 2 ∧
      0 < mutualInformation correlatedBoolBlanket :=
  ⟨correlatedBoolBlanket_diagonal.1,
    correlatedBoolBlanket_diagonal.2,
    correlatedBoolBlanket_mutualInformation_pos⟩

end FEP081
```

13.81.2 Typeset statement signatures

$$\{Blanket_1\ Blanket_2\ Internal_1\ Internal_2\ External_1\ External_2 : \mathsf{Type}^*\}$$
$$(\mathsf{blanketLaw} : \mathsf{FiniteLaw}\ (Blanket_1 \times Blanket_2))$$
$$(internal_1 : \mathsf{FiniteKernel}\ Blanket_1\ Internal_1)$$
$$(internal_2 : \mathsf{FiniteKernel}\ Blanket_2\ Internal_2)$$
$$(external_1 : \mathsf{FiniteKernel}\ Blanket_1\ External_1)$$
$$(external_2 : \mathsf{FiniteKernel}\ Blanket_2\ External_2)\ (\mathsf{blanket} : Blanket_1 \times Blanket_2)$$
$$\mathsf{mutualInformation}\ (\mathsf{conditionalJoint}\ (\mathsf{coupledBlanketModel}\ \mathsf{blanketLaw}\ internal_1$$
$$internal_2\ external_1\ external_2)\ \mathsf{blanket}) = 0$$

$$\{Blanket_1\ Blanket_2\ Internal_1\ Internal_2\ External_1\ External_2 : \mathsf{Type}^*\}$$
$$\mathsf{correlatedBoolBlanket}\ (\mathsf{false}, \mathsf{false}) = 1/2 \wedge \mathsf{correlatedBoolBlanket}$$
$$(\mathsf{true}, \mathsf{true}) = 1/2 \wedge 0 < \mathsf{mutualInformation}\ \mathsf{correlatedBoolBlanket}$$

13.82 fep-082 — Finite Intervention-Kernel Normalization

A hard finite intervention kernel places unit mass on the chosen value, zero mass on every distinct value, and remains normalized in every context. Assumption scope: Context and value carriers are finite and value equality is decidable. The kernel represents a deterministic do-assignment only; it does not by itself encode graph surgery, confounding assumptions, or interventional identifiability. Semantic disposition: `formalized`.

13.82.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP082

open FEP FEP.CausalDynamics Finset
open scoped BigOperators

variable {Context Value : Type*} [Fintype Context] [Fintype Value]
  [DecidableEq Value]

/-- Every hard finite intervention kernel is normalized in every context. -/
theorem fep082_interventionKernel_normalization
    (chosen : Value) (context : Context) :
    ∑ value, interventionKernel (Context := Context) chosen context value = 1 :=
  interventionKernel_sum_one chosen context

/-- The hard intervention gives its chosen value unit mass. -/
theorem fep082_interventionKernel_chosen_mass
    (chosen : Value) (context : Context) :
    interventionKernel (Context := Context) chosen context chosen = 1 :=
  interventionKernel_chosen chosen context

/-- Every distinct value receives zero intervention mass. -/
theorem fep082_interventionKernel_other_mass
    (chosen other : Value) (hOther : other ≠ chosen) (context : Context) :
    interventionKernel (Context := Context) chosen context other = 0 :=
  interventionKernel_other chosen other hOther context

end FEP082
```

13.82.2 Typeset statement signatures

$$\{\mathsf{Context}\ \mathsf{Value} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Context}]\ [\mathsf{Fintype}\ \mathsf{Value}]$$
$$(\mathsf{chosen} : \mathsf{Value})\ (\mathsf{context} : \mathsf{Context})$$
$$\sum \mathsf{value},\ \mathsf{interventionKernel}\ (\mathsf{Context} := \mathsf{Context})\ \mathsf{chosen}\ \mathsf{context}\ \mathsf{value} = 1$$

$$\{\mathsf{Context}\ \mathsf{Value} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Context}]\ [\mathsf{Fintype}\ \mathsf{Value}]$$
$$(\mathsf{chosen} : \mathsf{Value})\ (\mathsf{context} : \mathsf{Context})$$
$$\mathsf{interventionKernel}\ (\mathsf{Context} := \mathsf{Context})\ \mathsf{chosen}\ \mathsf{context}\ \mathsf{chosen} = 1$$

$$\{\mathsf{Context}\ \mathsf{Value} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Context}]\ [\mathsf{Fintype}\ \mathsf{Value}]$$
$$(\mathsf{chosen\ other} : \mathsf{Value})\ (\mathsf{hOther} : \mathsf{other} \neq \mathsf{chosen})\ (\mathsf{context} : \mathsf{Context})$$
$$\mathsf{interventionKernel}\ (\mathsf{Context} := \mathsf{Context})\ \mathsf{chosen}\ \mathsf{context}\ \mathsf{other} = 0$$

13.83 fep-083 — Non-Descendant Intervention Invariance

Intervening on the root of the maintained ordered four-node model preserves the named non-descendant marginal while permitting a descendant marginal to change. Assumption scope: The result is scoped to OrderedFourNodeModel with finite carriers and a decidable root; the non-descendant is structurally assigned an independent root law. It does not prove intervention invariance for arbitrary DAGs or derive the graph from observational conditional independences. Semantic disposition: `conditional_proxy`.

13.83.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP083

open FEP FEP.CausalDynamics

variable {Root NonDescendant Mediator Outcome : Type*}
  [Fintype Root] [Fintype NonDescendant]
  [Fintype Mediator] [Fintype Outcome] [DecidableEq Root]

/-- Intervening on the ordered root preserves the marginal law of the named
non-descendant. -/
theorem fep083_nonDescendant_intervention_invariance
    (model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
    (root : Root) :
    nonDescendantMarginal (interventionalJoint model root) =
      model.nonDescendantLaw :=
  nonDescendant_intervention_invariant model root

/-- In the concrete four-node model the intervention changes the mediator
descendant from zero to unit true-mass while preserving the non-descendant. -/
theorem fep083_fourNode_descendantChange_nonDescendantPreservation :
    mediatorMarginal (interventionalJoint boolOrderedModel false) true = 0 ∧
      mediatorMarginal (interventionalJoint boolOrderedModel true) true = 1 ∧
      nonDescendantMarginal (interventionalJoint boolOrderedModel false) =
        nonDescendantMarginal (interventionalJoint boolOrderedModel true) :=
  ⟨boolIntervention_false_mediator_true_zero,
    boolIntervention_true_mediator_true_one,
    boolIntervention_preserves_named_nonDescendant⟩

end FEP083
```

13.83.2 Typeset statement signatures

$$\{\text{Root NonDescendant Mediator Outcome} : \text{Type}^*\}$$
$$(\text{model} : \text{OrderedFourNodeModel Root NonDescendant Mediator Outcome})\ (\text{root} : \text{Root})$$
$$\text{nonDescendantMarginal}\ (\text{interventionalJoint model root}) = \text{model.nonDescendantLaw}$$

$$\{\text{Root NonDescendant Mediator Outcome} : \text{Type}^*\}$$
$$\text{mediatorMarginal}\ (\text{interventionalJoint boolOrderedModel false})\ \text{true} = 0$$
$$\wedge\ \text{mediatorMarginal}\ (\text{interventionalJoint boolOrderedModel true})\ \text{true} = 1$$
$$\wedge\ \text{nonDescendantMarginal}\ (\text{interventionalJoint boolOrderedModel false})$$
$$= \text{nonDescendantMarginal}\ (\text{interventionalJoint boolOrderedModel true})$$

13.84 fep-084 — Ordered Finite Causal Factorization

The maintained four-node observational law is exactly its ordered product of two root laws and two parent-conditioned kernels, and that product normalizes. Assumption scope: Root, non-descendant, mediator, and outcome carriers are finite and every component is a normalized finite law or kernel. The carrier records one fixed parent order rather than a general acyclic-graph factorization calculus. Semantic disposition: `formalized`.

13.84.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP084

open FEP FEP.CausalDynamics Finset
open scoped BigOperators
```

```
variable {Root NonDescendant Mediator Outcome : Type*}
  [Fintype Root] [Fintype NonDescendant]
  [Fintype Mediator] [Fintype Outcome]

/-- The four-node observational law is exactly the product of its two root
laws and its two ordered parent kernels. -/
theorem fep084_orderedFiniteCausal_factorization
    (model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
    (root : Root) (nonDescendant : NonDescendant)
    (mediator : Mediator) (outcome : Outcome) :
    orderedJoint model (((root, nonDescendant), mediator), outcome) =
      ((model.rootLaw root * model.nonDescendantLaw nonDescendant) *
          model.mediatorGivenRoot root mediator) *
        model.outcomeGivenParents (nonDescendant, mediator) outcome :=
  orderedJoint_factorization model root nonDescendant mediator outcome

/-- The ordered factorization is a normalized joint law. -/
theorem fep084_orderedFiniteCausal_normalization
    (model : OrderedFourNodeModel Root NonDescendant Mediator Outcome) :
    Σ state, orderedJoint model state = 1 :=
  orderedJoint_sum_one model

end FEP084
```

13.84.2 Typeset statement signatures

$$\{\mathsf{Root}\ \mathsf{NonDescendant}\ \mathsf{Mediator}\ \mathsf{Outcome} : \mathsf{Type}^*\}$$
$$(\mathsf{model} : \mathsf{OrderedFourNodeModel}\ \mathsf{Root}\ \mathsf{NonDescendant}\ \mathsf{Mediator}\ \mathsf{Outcome})\ (\mathsf{root} : \mathsf{Root})$$
$$(\mathsf{nonDescendant} : \mathsf{NonDescendant})\ (\mathsf{mediator} : \mathsf{Mediator})\ (\mathsf{outcome} : \mathsf{Outcome})$$
$$\mathsf{orderedJoint}\ \mathsf{model}\ (((\mathsf{root}, \mathsf{nonDescendant}), \mathsf{mediator}), \mathsf{outcome})$$
$$= ((\mathsf{model.rootLaw}\ \mathsf{root} \cdot \mathsf{model.nonDescendantLaw}\ \mathsf{nonDescendant})$$
$$\cdot\ \mathsf{model.mediatorGivenRoot}\ \mathsf{root}\ \mathsf{mediator}) \cdot \mathsf{model.outcomeGivenParents}$$
$$(\mathsf{nonDescendant}, \mathsf{mediator})\ \mathsf{outcome}$$

$$\{\mathsf{Root}\ \mathsf{NonDescendant}\ \mathsf{Mediator}\ \mathsf{Outcome} : \mathsf{Type}^*\}$$
$$(\mathsf{model} : \mathsf{OrderedFourNodeModel}\ \mathsf{Root}\ \mathsf{NonDescendant}\ \mathsf{Mediator}\ \mathsf{Outcome})$$
$$\sum \mathsf{state}, \mathsf{orderedJoint}\ \mathsf{model}\ \mathsf{state} = 1$$

13.85 fep-085 — Local Markov Property from Ordered Factorization

In the ordered four-node carrier, conditioning on the outcome's two parents factorizes the outcome from its non-parent predecessor root and forces their finite mutual information to zero. Assumption scope: All carriers are finite and mediator evidence must be strictly positive. The result is a local statement for one authored ordering and parent set; it is not a proof of general d-separation, faithfulness, or native measurable conditional independence. Semantic disposition: `conditional_proxy`.

13.85.1 Lean sketch

```
import FepSketches.causal_dynamics

namespace FEP085

open FEP FEP.CausalDynamics FEP.FiniteInformation

variable {Root NonDescendant Mediator Outcome : Type*}
  [Fintype Root] [Fintype NonDescendant]
  [Fintype Mediator] [Fintype Outcome]

/-- In the maintained ordered carrier, the outcome and non-parent predecessor
root factorize after fixing the outcome's non-descendant and mediator parents. -/
```

```
theorem fep085_localMarkov_factorization_from_ordered
    (model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
    (nonDescendant : NonDescendant) (mediator : Mediator)
    (hEvidence : 0 < mediatorEvidence model mediator) :
    localMarkovConditional model nonDescendant mediator hEvidence =
      (model.mediatorGivenRoot.posterior model.rootLaw mediator hEvidence).product
        (model.outcomeGivenParents.row (nonDescendant, mediator)) :=
  localMarkov_factorization_from_ordered
    model nonDescendant mediator hEvidence

/-- The exact local product law has zero finite mutual information.  This is a
local Markov statement for the four-node carrier, not general d-separation. -/
theorem fep085_localMarkov_mutualInformation_zero
    (model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
    (nonDescendant : NonDescendant) (mediator : Mediator)
    (hEvidence : 0 < mediatorEvidence model mediator) :
    mutualInformation
        (localMarkovConditional model nonDescendant mediator hEvidence) = 0 :=
  localMarkov_mutualInformation_zero model nonDescendant mediator hEvidence

/-- The Boolean local-Markov theorem conditions on a positive half-mass
mediator event, providing a concrete nonvacuity witness. -/
theorem fep085_bool_localMarkov_nonvacuity :
    mutualInformation
        (localMarkovConditional boolOrderedModel false true
          (by rw [boolMediatorEvidence_true]; norm_num)) = 0 :=
  localMarkov_mutualInformation_zero boolOrderedModel false true
    (by rw [boolMediatorEvidence_true]; norm_num)

/-- Zero mediator evidence is the explicit conditioning boundary. -/
theorem fep085_zeroEvidence_boundary
    (model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
    (mediator : Mediator) (hZero : mediatorEvidence model mediator = 0) :
    ¬0 < mediatorEvidence model mediator :=
  localMarkov_zeroEvidence_boundary model mediator hZero

end FEP085
```

13.85.2 Typeset statement signatures

$$\{\text{Root NonDescendant Mediator Outcome} : \text{Type}^*\}$$
$$(\text{model} : \text{OrderedFourNodeModel Root NonDescendant Mediator Outcome})$$
$$(\text{nonDescendant} : \text{NonDescendant})\ (\text{mediator} : \text{Mediator})$$
$$(\text{hEvidence} : 0 < \text{mediatorEvidence model mediator})$$
$$\text{localMarkovConditional model nonDescendant mediator hEvidence}$$
$$= (\text{model.mediatorGivenRoot.posterior model.rootLaw mediator hEvidence}).\text{product}$$
$$(\text{model.outcomeGivenParents.row}\ (\text{nonDescendant}, \text{mediator}))$$

$$\{\text{Root NonDescendant Mediator Outcome} : \text{Type}^*\}$$
$$(\text{model} : \text{OrderedFourNodeModel Root NonDescendant Mediator Outcome})$$
$$(\text{nonDescendant} : \text{NonDescendant})\ (\text{mediator} : \text{Mediator})$$
$$(\text{hEvidence} : 0 < \text{mediatorEvidence model mediator})$$
$$\text{mutualInformation}\ (\text{localMarkovConditional model nonDescendant mediator hEvidence}) = 0$$

$$\{\text{Root NonDescendant Mediator Outcome} : \text{Type}^*\}$$
$$\text{mutualInformation}\ (\text{localMarkovConditional boolOrderedModel false true}\ (\text{by rw}\ [\text{boolMediatorEvidence\_true}]; \text{norm\_num})) = 0$$

$$\{\text{Root NonDescendant Mediator Outcome} : \text{Type}^*\}$$
$$(\text{model} : \text{OrderedFourNodeModel Root NonDescendant Mediator Outcome})$$
$$(\text{mediator} : \text{Mediator})\ (\text{hZero} : \text{mediatorEvidence model mediator} = 0)$$
$$\neg 0 < \text{mediatorEvidence model mediator}$$

13.86 fep-086 — Precision-Weighted Prediction-Error Energy

Nonnegative precision yields nonnegative scalar prediction-error energy, and under positive precision zero energy occurs exactly at exact prediction. Assumption scope: The carrier is scalar real arithmetic. Nonnegativity needs precision at least zero and separation needs precision strictly positive; zero precision cannot distinguish any estimate from the observation. Semantic disposition: `formalized`.

13.86.1 Lean sketch

```
import FepSketches.predictive_coding

namespace FEP086

open FEP.PredictiveCoding

/-- Nonnegative precision produces nonnegative squared prediction-error
energy. -/
theorem fep086_precisionWeighted_predictionError_nonnegative
    {precision : ℝ} (hPrecision : 0 ≤ precision)
    (observation estimate : ℝ) :
    0 ≤ precisionEnergy precision observation estimate :=
  precisionEnergy_nonneg hPrecision observation estimate

/-- At positive precision, zero energy occurs exactly at an exact prediction. -/
theorem fep086_precisionWeighted_zero_iff_exactPrediction
    {precision : ℝ} (hPrecision : 0 < precision)
    (observation estimate : ℝ) :
    precisionEnergy precision observation estimate = 0 ↔
      estimate = observation :=
  precisionEnergy_eq_zero_iff hPrecision observation estimate

/-- Unit prediction error at precision two has exactly unit energy. -/
theorem fep086_unitError_nonvacuity : precisionEnergy 2 1 0 = 1 := by
  norm_num [precisionEnergy, predictionError]

end FEP086
```

13.86.2 Typeset statement signatures

$$\{\text{precision} : \mathbb{R}\}\ (\text{hPrecision} : 0 \le \text{precision})\ (\text{observation estimate} : \mathbb{R})$$
$$0 \le \text{precisionEnergy precision observation estimate}$$

$$\{\text{precision} : \mathbb{R}\}\ (\text{hPrecision} : 0 < \text{precision})\ (\text{observation estimate} : \mathbb{R})$$
$$\text{precisionEnergy precision observation estimate} = 0 \leftrightarrow \text{estimate} = \text{observation}$$

$$\text{precisionEnergy}\ 2\ 1\ 0 = 1$$

13.87 fep-087 — Hierarchical Predictive-Coding Energy Decomposition

Finite hierarchical predictive-coding energy decomposes into the zeroth precision-weighted error plus all successor-level errors and is nonnegative when every precision is nonnegative. Assumption scope: The hierarchy is finite and energies are explicit scalar quadratics. Nonnegativity requires pointwise nonnegative precision; the result does not model neural implementation, learned precision, or an infinite hierarchy. Semantic disposition: `formalized`.

13.87.1 Lean sketch

```
import FepSketches.predictive_coding

namespace FEP087

open FEP.PredictiveCoding Finset
open scoped BigOperators

/-- A finite successor hierarchy splits into the zeroth prediction-error
energy and the sum over all higher levels. -/
theorem fep087_hierarchicalPredictiveCoding_decomposition {depth : ℕ}
    (precision error : Fin (depth + 1) → ℝ) :
    hierarchicalEnergy precision error =
      precision 0 / 2 * error 0 ^ 2 +
        Σ level : Fin depth,
          precision level.succ / 2 * error level.succ ^ 2 :=
  hierarchicalEnergy_succ precision error

/-- Nonnegative precision at every hierarchy level makes the total energy
nonnegative. -/
theorem fep087_hierarchicalPredictiveCoding_nonnegative
    {Level : Type*} [Fintype Level]
    (precision error : Level → ℝ)
    (hPrecision : ∀ level, 0 ≤ precision level) :
    0 ≤ hierarchicalEnergy precision error :=
  hierarchicalEnergy_nonneg precision error hPrecision

/-- Both levels contribute in the concrete energy value eleven. -/
theorem fep087_twoLevel_decomposition_nonvacuity :
    hierarchicalEnergy
      (fun level : Fin 2 => if level = 0 then 2 else 4)
      (fun level : Fin 2 => if level = 0 then 3 else 1) = 11 :=
  twoLevelEnergy_witness

end FEP087
```

13.87.2 Typeset statement signatures

$$\{\mathsf{depth} : \mathbb{N}\}\,(\mathsf{precision\ error} : \mathsf{Fin}\,(\mathsf{depth} + 1) \Rightarrow \mathbb{R})$$

$$\mathsf{hierarchicalEnergy\ precision\ error} = \mathsf{precision}\,0/2 \cdot \mathsf{error}\,0^2 + \sum \mathsf{level} :$$

$$\mathsf{Fin\ depth}, \mathsf{precision\ level.succ}/2 \cdot \mathsf{error\ level.succ}^2$$

$$\{\mathsf{Level} : \mathsf{Type}^*\}\,[\mathsf{Fintype\ Level}]\,(\mathsf{precision\ error} : \mathsf{Level} \Rightarrow \mathbb{R})$$

$$(\mathsf{hPrecision} : \forall \mathsf{level}, 0 \le \mathsf{precision\ level})$$

$$0 \le \mathsf{hierarchicalEnergy\ precision\ error}$$

$$\mathsf{hierarchicalEnergy}\,(\lambda\,\mathsf{level} : \mathsf{Fin}\,2 \mapsto \mathsf{if\ level} = 0\,\mathsf{then}\,2\,\mathsf{else}\,4)$$

$$(\lambda\,\mathsf{level} : \mathsf{Fin}\,2 \mapsto \mathsf{if\ level} = 0\,\mathsf{then}\,3\,\mathsf{else}\,1) = 11$$

13.88 fep-088 — Prediction-Error Gradient Identity

Differentiating scalar precision-weighted prediction-error energy with respect to the estimate gives exactly minus precision times signed prediction error. Assumption scope: The theorem is an exact HasDerivAt statement for a scalar quadratic and requires no positivity assumption for differentiation. It does not assert a gradient for a nonlinear decoder, a vector network, or a stochastic process. Semantic disposition: `formalized`.

13.88.1 Lean sketch

```
import FepSketches.predictive_coding
```

```
namespace FEP088

open FEP.PredictiveCoding

/-- The estimate derivative of precision-weighted error energy is exactly
minus precision times signed prediction error. -/
theorem fep088_predictionError_gradient_identity
    (precision observation estimate : ℝ) :
    HasDerivAt (fun candidate => precisionEnergy precision observation candidate)
      (-precision * predictionError observation estimate) estimate :=
  precisionEnergy_hasDerivAt precision observation estimate

/-- The named prediction-error gradient is definitionally the derivative
certified above. -/
theorem fep088_namedGradient_hasDerivAt
    (precision observation estimate : ℝ) :
    HasDerivAt (fun candidate => precisionEnergy precision observation candidate)
      (precisionEnergyGradient precision observation estimate) estimate :=
  precisionEnergy_derivative_eq_gradient precision observation estimate

/-- The gradient is nonzero for a concrete imperfect prediction. -/
theorem fep088_gradient_nonvacuity :
    precisionEnergyGradient 2 1 0 = -2 := by
  norm_num [precisionEnergyGradient, predictionError]

end FEP088
```

13.88.2 Typeset statement signatures

$$(\mathsf{precision\ observation\ estimate} : \mathbb{R})$$
$$\mathsf{HasDerivAt}\ (\lambda\ \mathsf{candidate} \mapsto \mathsf{precisionEnergy\ precision\ observation\ candidate})$$
$$(-\mathsf{precision} \cdot \mathsf{predictionError\ observation\ estimate})\ \mathsf{estimate}$$

$$(\mathsf{precision\ observation\ estimate} : \mathbb{R})$$
$$\mathsf{HasDerivAt}\ (\lambda\ \mathsf{candidate} \mapsto \mathsf{precisionEnergy\ precision\ observation\ candidate})$$
$$(\mathsf{precisionEnergyGradient\ precision\ observation\ estimate})\ \mathsf{estimate}$$

$$\mathsf{precisionEnergyGradient}\ 2\ 1\ 0 = -2$$

13.89 fep-089 — Finite Generalized-Coordinate Shift Semigroup

Natural-number shifts on a finite generalized-coordinate jet compose associatively, with the highest retained coordinate shifting into an explicit zero truncation boundary. Assumption scope: Generalized coordinates are finite jets represented by finitely supported coefficients. The semigroup is discrete and exact; no analytic infinite jet, time derivative, or continuous-flow convergence is claimed. Semantic disposition: `formalized`.

13.89.1 Lean sketch

```
import FepSketches.predictive_coding

namespace FEP089

open FEP.PredictiveCoding

/-- Generalized-coordinate shifts form a natural-number semigroup. -/
theorem fep089_finiteJet_shift_semigroup {order : ℕ}
    (first second : ℕ) (jet : FiniteJet order) :
    shift (first + second) jet = shift first (shift second jet) :=
  shift_add first second jet
```

```
/-- The highest retained coordinate shifts into the explicitly truncated zero
coefficient. -/
theorem fep089_finiteJet_terminal_boundary {order : ℕ}
    (jet : FiniteJet order) :
    (shift 1 jet).coefficient order = 0 :=
  shift_top_zero jet

/-- A first-order jet shifts velocity into position and exposes zero at its
terminal coordinate. -/
theorem fep089_firstOrder_shift_nonvacuity (value velocity : ℝ) :
    (shift 1 (firstOrderJet value velocity)).coefficient 0 = velocity ∧
      (shift 1 (firstOrderJet value velocity)).coefficient 1 = 0 :=
  firstOrderJet_shift_boundary value velocity

end FEP089
```

13.89.2 Typeset statement signatures

$$\{\mathsf{order} : \mathbb{N}\}\ (\mathsf{first}\ \mathsf{second} : \mathbb{N})\ (\mathsf{jet} : \mathsf{FiniteJet}\ \mathsf{order})$$
$$\mathsf{shift}\ (\mathsf{first} + \mathsf{second})\ \mathsf{jet} = \mathsf{shift}\ \mathsf{first}\ (\mathsf{shift}\ \mathsf{second}\ \mathsf{jet})$$

$$\{\mathsf{order} : \mathbb{N}\}\ (\mathsf{jet} : \mathsf{FiniteJet}\ \mathsf{order})$$
$$(\mathsf{shift}\ 1\ \mathsf{jet}).\mathsf{coefficient}\ \mathsf{order} = 0$$

$$(\mathsf{value}\ \mathsf{velocity} : \mathbb{R})$$
$$(\mathsf{shift}\ 1\ (\mathsf{firstOrderJet}\ \mathsf{value}\ \mathsf{velocity})).\mathsf{coefficient}\ 0 = \mathsf{velocity}$$
$$\wedge\ (\mathsf{shift}\ 1\ (\mathsf{firstOrderJet}\ \mathsf{value}\ \mathsf{velocity})).\mathsf{coefficient}\ 1 = 0$$

13.90 fep-090 — Finite-Jet Generalized-Filtering Correction Equation

One finite-jet generalized-filtering Euler step equals the current coordinate plus a truncated shift minus the certified quadratic energy gradient. Assumption scope: Step size and coordinate precisions are real and the jet order is finite. The pointwise correction has an exact scalar derivative witness, but the theorem does not identify an infinite generalized flow or prove stability of repeated coupled-jet updates. Semantic disposition: `formalized`.

13.90.1 Lean sketch

```
import FepSketches.predictive_coding

namespace FEP090

open FEP.PredictiveCoding

/-- One generalized-filtering Euler correction is the current jet plus step
size times truncated shift minus the certified quadratic energy gradient. -/
theorem fep090_finiteJet_generalizedFiltering_correctionEquation
    {order : ℕ} (stepSize : ℝ) (precision : ℕ → ℝ)
    (target estimate : FiniteJet order) (degree : ℕ) :
    (generalizedFilteringStep stepSize precision target estimate).coefficient degree =
      estimate.coefficient degree + stepSize *
        ((shift 1 estimate).coefficient degree -
          precisionEnergyGradient (precision degree)
            (target.coefficient degree) (estimate.coefficient degree)) :=
  generalizedFilteringStep_equation stepSize precision target estimate degree

/-- The correction's pointwise gradient is backed by an exact derivative
statement rather than a symbolic placeholder. -/
theorem fep090_finiteJet_coordinateGradient_hasDerivAt
    {order : ℕ} (precision : ℕ → ℝ)
    (target estimate : FiniteJet order) (degree : ℕ) :
```

```
    HasDerivAt
      (fun candidate =>
        precisionEnergy (precision degree) (target.coefficient degree) candidate)
      (precisionEnergyGradient (precision degree)
        (target.coefficient degree) (estimate.coefficient degree))
      (estimate.coefficient degree) :=
  generalizedCoordinateEnergy_hasDerivAt precision target estimate degree

/-- At the top finite coordinate the shift term is exactly absent. -/
theorem fep090_finiteJet_truncation_visible
    {order : ℕ} (precision : ℕ → ℝ)
    (target estimate : FiniteJet order) :
    (generalizedFlow precision target estimate).coefficient order =
      -precisionEnergyGradient (precision order)
        (target.coefficient order) (estimate.coefficient order) :=
  generalizedFlow_top_boundary precision target estimate

/-- Both coordinates of a concrete first-order jet receive nonzero, exact
corrections. -/
theorem fep090_firstOrder_correction_nonvacuity :
    let corrected := generalizedFilteringStep (1 / 2 : ℝ) (fun _ => 1)
      (firstOrderJet 2 0) (firstOrderJet 0 1)
    corrected.coefficient 0 = 3 / 2 ∧ corrected.coefficient 1 = 1 / 2 :=
  firstOrderCorrection_witness

end FEP090
```

13.90.2 Typeset statement signatures

$$\{\text{order} : \mathbb{N}\}\ (\text{stepSize} : \mathbb{R})\ (\text{precision} : \mathbb{N} \Rightarrow \mathbb{R})$$
$$(\text{target estimate} : \text{FiniteJet order})\ (\text{degree} : \mathbb{N})$$
$$(\text{generalizedFilteringStep stepSize precision target estimate}).\text{coefficient degree}$$
$$= \text{estimate.coefficient degree} + \text{stepSize} \cdot ((\text{shift } 1\ \text{estimate}).\text{coefficient}$$
$$\text{degree} - \text{precisionEnergyGradient}\ (\text{precision degree})\ (\text{target.coefficient degree})$$
$$(\text{estimate.coefficient degree}))$$

$$\{\text{order} : \mathbb{N}\}\ (\text{precision} : \mathbb{N} \Rightarrow \mathbb{R})\ (\text{target estimate} : \text{FiniteJet order})\ (\text{degree} : \mathbb{N})$$
$$\text{HasDerivAt}\ (\lambda\ \text{candidate} \mapsto \text{precisionEnergy}\ (\text{precision degree})$$
$$(\text{target.coefficient degree})\ \text{candidate})\ (\text{precisionEnergyGradient}\ (\text{precision}$$
$$\text{degree})\ (\text{target.coefficient degree})\ (\text{estimate.coefficient degree}))$$
$$(\text{estimate.coefficient degree})$$

$$\{\text{order} : \mathbb{N}\}\ (\text{precision} : \mathbb{N} \Rightarrow \mathbb{R})\ (\text{target estimate} : \text{FiniteJet order})$$
$$(\text{generalizedFlow precision target estimate}).\text{coefficient order}$$
$$= -\text{precisionEnergyGradient}\ (\text{precision order})\ (\text{target.coefficient order})$$
$$(\text{estimate.coefficient order})$$

$$\text{let corrected}$$

13.91 fep-091 — Precision Modulation of Prediction Error

Increasing nonnegative precision cannot decrease a fixed scalar prediction-error energy, and scaling precision scales its exact gradient by the same factor. Assumption scope: Energy monotonicity assumes a nonnegative lower precision and an ordered upper precision. Gradient scaling is algebraic for arbitrary real scale, including negative values, but only nonnegative precision retains an energy weighting interpretation. Semantic disposition: `formalized`.

13.91.1 Lean sketch

```
import FepSketches.predictive_coding

namespace FEP091

open FEP.PredictiveCoding

/-- Greater precision cannot lower the energy of a fixed prediction error. -/
theorem fep091_precisionModulation_energy_mono
    {lower upper : ℝ} (hLowerNonneg : 0 ≤ lower)
    (hPrecision : lower ≤ upper)
    (observation estimate : ℝ) :
    0 ≤ precisionEnergy lower observation estimate ∧
      precisionEnergy lower observation estimate ≤
        precisionEnergy upper observation estimate :=
  precisionEnergy_mono hLowerNonneg hPrecision observation estimate

/-- Scaling precision scales the exact prediction-error gradient by the same
factor. -/
theorem fep091_precisionModulation_gradient_scale
    (scale precision observation estimate : ℝ) :
    precisionEnergyGradient (scale * precision) observation estimate =
      scale * precisionEnergyGradient precision observation estimate :=
  precisionGradient_scale scale precision observation estimate

/-- Doubling precision strictly increases the energy of one concrete nonzero
error. -/
theorem fep091_doubledPrecision_nonvacuity :
    precisionEnergy 2 1 0 = 2 * precisionEnergy 1 1 0 ∧
      precisionEnergy 1 1 0 < precisionEnergy 2 1 0 :=
  doubledPrecision_nontrivial_witness

end FEP091
```

13.91.2 Typeset statement signatures

$$\{\mathsf{lower}\ \mathsf{upper} : \mathbb{R}\}\ (\mathsf{hLowerNonneg} : 0 \le \mathsf{lower})\ (\mathsf{hPrecision} : \mathsf{lower} \le \mathsf{upper})$$
$$(\mathsf{observation}\ \mathsf{estimate} : \mathbb{R})$$
$$0 \le \mathsf{precisionEnergy}\ \mathsf{lower}\ \mathsf{observation}\ \mathsf{estimate} \wedge \mathsf{precisionEnergy}\ \mathsf{lower}$$
$$\mathsf{observation}\ \mathsf{estimate} \le \mathsf{precisionEnergy}\ \mathsf{upper}\ \mathsf{observation}\ \mathsf{estimate}$$

$$(\mathsf{scale}\ \mathsf{precision}\ \mathsf{observation}\ \mathsf{estimate} : \mathbb{R})$$
$$\mathsf{precisionEnergyGradient}\ (\mathsf{scale} \cdot \mathsf{precision})\ \mathsf{observation}\ \mathsf{estimate} = \mathsf{scale}$$
$$\cdot\ \mathsf{precisionEnergyGradient}\ \mathsf{precision}\ \mathsf{observation}\ \mathsf{estimate}$$

$$\mathsf{precisionEnergy}\ 2\ 1\ 0 = 2 \cdot \mathsf{precisionEnergy}\ 1\ 1\ 0 \wedge \mathsf{precisionEnergy}\ 1\ 1\ 0$$
$$< \mathsf{precisionEnergy}\ 2\ 1\ 0$$

13.92 fep-092 — Quadratic Predictive-Coding Convergence

Exact scalar quadratic prediction updates contract energy by the squared factor one minus step size, strictly decrease positive-precision nonzero error for step sizes in the open interval zero to two, and drive error to zero. Assumption scope: Convergence requires 0 < stepSize < 2; strict descent additionally requires positive precision and unequal target and estimate. The result is scalar and exact-quadratic, not a convergence theorem for nonlinear predictive-coding networks. Semantic disposition: `formalized`.

13.92.1 Lean sketch

```
import FepSketches.predictive_coding

namespace FEP092
```

```
open FEP.PredictiveCoding Filter Topology

/-- Each quadratic prediction update contracts energy by the exact squared
factor `(1-stepSize)^2`. -/
theorem fep092_quadraticPredictiveCoding_contraction
    (precision stepSize target estimate : ℝ) :
    precisionEnergy precision target
        (predictionUpdate stepSize target estimate) =
      (1 - stepSize) ^ 2 * precisionEnergy precision target estimate :=
  predictionEnergy_contraction precision stepSize target estimate

/-- Positive precision and a nonzero error give strict descent for every step
size strictly between zero and two. -/
theorem fep092_quadraticPredictiveCoding_strictDecrease
    {precision stepSize target estimate : ℝ}
    (hPrecision : 0 < precision) (hStepPositive : 0 < stepSize)
    (hStepBelowTwo : stepSize < 2) (hError : target ≠ estimate) :
    precisionEnergy precision target (predictionUpdate stepSize target estimate) <
      precisionEnergy precision target estimate :=
  predictionEnergy_strictDecrease hPrecision hStepPositive hStepBelowTwo hError

/-- Under the same step-size boundary, repeated exact prediction error
converges to zero. -/
theorem fep092_quadraticPredictiveCoding_error_tendsto_zero
    {stepSize : ℝ} (hStepPositive : 0 < stepSize)
    (hStepBelowTwo : stepSize < 2) (target estimate : ℝ) :
    Tendsto
      (fun iterations => predictionError target
        (iteratePredictionUpdate stepSize target iterations estimate))
      atTop (nhds 0) :=
  iteratePredictionError_tendsto_zero
    hStepPositive hStepBelowTwo target estimate

/-- A half-step reduces a concrete unit energy to one quarter. -/
theorem fep092_halfStep_decrease_nonvacuity :
    precisionEnergy 2 1 0 = 1 ∧
      predictionUpdate (1 / 2) 1 0 = 1 / 2 ∧
      precisionEnergy 2 1 (predictionUpdate (1 / 2) 1 0) = 1 / 4 :=
  halfStep_energy_witness

end FEP092
```

13.92.2 Typeset statement signatures

$$(\mathsf{precision}\ \mathsf{stepSize}\ \mathsf{target}\ \mathsf{estimate} : \mathbb{R})$$
$$\mathsf{precisionEnergy}\ \mathsf{precision}\ \mathsf{target}\ (\mathsf{predictionUpdate}\ \mathsf{stepSize}\ \mathsf{target}\ \mathsf{estimate})$$
$$= (1 - \mathsf{stepSize})^2 \cdot \mathsf{precisionEnergy}\ \mathsf{precision}\ \mathsf{target}\ \mathsf{estimate}$$

$$\{\mathsf{precision}\ \mathsf{stepSize}\ \mathsf{target}\ \mathsf{estimate} : \mathbb{R}\}\ (\mathsf{hPrecision} : 0 < \mathsf{precision})$$
$$(\mathsf{hStepPositive} : 0 < \mathsf{stepSize})\ (\mathsf{hStepBelowTwo} : \mathsf{stepSize} < 2)$$
$$(\mathsf{hError} : \mathsf{target} \neq \mathsf{estimate})$$
$$\mathsf{precisionEnergy}\ \mathsf{precision}\ \mathsf{target}\ (\mathsf{predictionUpdate}\ \mathsf{stepSize}\ \mathsf{target}\ \mathsf{estimate})$$
$$< \mathsf{precisionEnergy}\ \mathsf{precision}\ \mathsf{target}\ \mathsf{estimate}$$

$$\{\mathsf{stepSize} : \mathbb{R}\}\ (\mathsf{hStepPositive} : 0 < \mathsf{stepSize})\ (\mathsf{hStepBelowTwo} : \mathsf{stepSize} < 2)$$
$$(\mathsf{target}\ \mathsf{estimate} : \mathbb{R})$$
$$\mathsf{Tendsto}\ (\lambda\ \mathsf{iterations} \mapsto \mathsf{predictionError}\ \mathsf{target}\ (\mathsf{iteratePredictionUpdate}$$
$$\mathsf{stepSize}\ \mathsf{target}\ \mathsf{iterations}\ \mathsf{estimate}))\ \mathsf{atTop}\ (\mathsf{nhds}\ 0)$$

$$\mathsf{precisionEnergy}\,2\,1\,0 = 1 \wedge \mathsf{predictionUpdate}\,(1/2)\,1\,0 = 1/2$$
$$\wedge\ \mathsf{precisionEnergy}\,2\,1\,(\mathsf{predictionUpdate}\,(1/2)\,1\,0) = 1/4$$

13.93 fep-093 — Forward and Reverse Finite Path-Law Ratio

A finite forward path law is reconstructed pointwise from its supported reverse-aligned law and likelihood ratio, while both path laws normalize and path reversal is involutive. Assumption scope: The path carrier is finite and reverse-aligned mass must be strictly positive for the ratio reconstruction. FinitePathProtocol stores normalized laws and an involution; it does not derive reversal or reverse dynamics from a microscopic process. Semantic disposition: `formalized`.

13.93.1 Lean sketch

```
import FepSketches.path_thermodynamics

namespace FEP093

open FEP FEP.PathThermodynamics Finset
open scoped BigOperators

variable {Path : Type*} [Fintype Path]

/-- The supported forward/reverse likelihood ratio reconstructs forward path
mass, while both finite path laws remain normalized. -/
theorem fep093_forward_reverse_pathLaw_ratio
    (protocol : FinitePathProtocol Path)
    (hReverse : ∀ path, 0 < protocol.reverseAligned path) (path : Path) :
    pathRatio protocol path * protocol.reverseAligned path =
      protocol.forward path :=
  pathRatio_mul_reverse protocol hReverse path

/-- Forward and reverse path-law normalization is structural. -/
theorem fep093_pathLaws_normalized
    (protocol : FinitePathProtocol Path) :
    (Σ path, protocol.forward path = 1) ∧
      Σ path, protocol.reverseAligned path = 1 :=
  pathLaw_normalization protocol

/-- The recorded path reversal is involutive. -/
theorem fep093_pathReversal_involutive
    (protocol : FinitePathProtocol Path) (path : Path) :
    protocol.reversal (protocol.reversal path) = path :=
  reverse_reverse protocol path

end FEP093
```

13.93.2 Typeset statement signatures

$$\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})$$
$$(\mathsf{hReverse} : \forall \mathsf{path}, 0 < \mathsf{protocol.reverseAligned}\ \mathsf{path})\ (\mathsf{path} : \mathsf{Path})$$
$$\mathsf{pathRatio}\ \mathsf{protocol}\ \mathsf{path} \cdot \mathsf{protocol.reverseAligned}\ \mathsf{path} = \mathsf{protocol.forward}\ \mathsf{path}$$

$$\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})$$
$$(\sum \mathsf{path}, \mathsf{protocol.forward}\ \mathsf{path} = 1) \wedge \sum \mathsf{path}, \mathsf{protocol.reverseAligned}\ \mathsf{path} = 1$$

$$\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})\ (\mathsf{path} : \mathsf{Path})$$
$$\mathsf{protocol.reversal}\ (\mathsf{protocol.reversal}\ \mathsf{path}) = \mathsf{path}$$

13.94 fep-094 — Entropy Production as Path KL

Mean finite path entropy production is the KL divergence from the forward to reverse-aligned path law, is nonnegative, and under full support equals the forward expectation of the pointwise log ratio. Assumption scope: The primary equality is the maintained definition of entropyProduction. Expected-log-ratio expansion requires strict support of both path laws, and the project's totalized real KL is not an extended divergence at zero reverse mass. Semantic disposition: `structural_proxy`.

13.94.1 Lean sketch

```
import FepSketches.path_thermodynamics

namespace FEP094

open FEP FEP.PathThermodynamics FEP.FiniteInformation Finset
open scoped BigOperators

variable {Path : Type*} [Fintype Path]

/-- Mean stochastic entropy production is exactly the finite path-space KL
divergence. -/
theorem fep094_entropyProduction_as_pathKL
    (protocol : FinitePathProtocol Path) :
    entropyProduction protocol =
      finiteKL protocol.forward protocol.reverseAligned :=
  rfl

/-- Full path support converts the KL definition to the expected log ratio. -/
theorem fep094_entropyProduction_expected_logRatio
    (protocol : FinitePathProtocol Path)
    (hForward : ∀ path, 0 < protocol.forward path)
    (hReverse : ∀ path, 0 < protocol.reverseAligned path) :
    entropyProduction protocol =
      Σ path, protocol.forward path * pathwiseEntropyProduction protocol path :=
  entropyProduction_eq_expected_logRatio protocol hForward hReverse

/-- Finite path entropy production is nonnegative. -/
theorem fep094_entropyProduction_nonnegative
    (protocol : FinitePathProtocol Path) :
    0 ≤ entropyProduction protocol :=
  entropyProduction_nonneg protocol

end FEP094
```

13.94.2 Typeset statement signatures

$$\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})$$
$$\mathsf{entropyProduction}\ \mathsf{protocol} = \mathsf{finiteKL}\ \mathsf{protocol.forward}\ \mathsf{protocol.reverseAligned}$$

$$\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})$$
$$(\mathsf{hForward} : \forall \mathsf{path}, 0 < \mathsf{protocol.forward}\ \mathsf{path})$$
$$(\mathsf{hReverse} : \forall \mathsf{path}, 0 < \mathsf{protocol.reverseAligned}\ \mathsf{path})$$
$$\mathsf{entropyProduction}\ \mathsf{protocol} = \sum \mathsf{path}, \mathsf{protocol.forward}\ \mathsf{path}$$
$$\cdot\ \mathsf{pathwiseEntropyProduction}\ \mathsf{protocol}\ \mathsf{path}$$

$$\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})$$
$$0 \le \mathsf{entropyProduction}\ \mathsf{protocol}$$

13.95 fep-095 — Detailed Fluctuation Symmetry

Under full path support, reverse-aligned mass times exponential pointwise entropy production reconstructs forward mass, and an exchanging reversal makes entropy production antisymmetric. Assumption scope: Forward and reverse-aligned laws are strictly positive. Antisymmetry further assumes that the recorded involution exchanges the two aligned laws in both directions; these exchange laws are not derived from protocol dynamics. Semantic disposition: `conditional_proxy`.

13.95.1 Lean sketch

```
import FepSketches.path_thermodynamics

namespace FEP095

open FEP FEP.PathThermodynamics

variable {Path : Type*} [Fintype Path]

/-- The supported pointwise detailed fluctuation symmetry reconstructs the
forward path law from the reverse law and exponential entropy production. -/
theorem fep095_detailedFluctuation_symmetry
    (protocol : FinitePathProtocol Path)
    (hForward : ∀ path, 0 < protocol.forward path)
    (hReverse : ∀ path, 0 < protocol.reverseAligned path) (path : Path) :
    protocol.reverseAligned path *
        Real.exp (pathwiseEntropyProduction protocol path) =
      protocol.forward path :=
  detailedFluctuation_identity protocol hForward hReverse path

/-- When reversal exchanges the aligned laws, pathwise entropy production is
odd under the involution. -/
theorem fep095_entropyProduction_reversal_antisymmetry
    (protocol : FinitePathProtocol Path)
    (hForward : ∀ path, 0 < protocol.forward path)
    (hReverse : ∀ path, 0 < protocol.reverseAligned path)
    (hExchangeForward : ∀ path,
      protocol.forward (protocol.reversal path) = protocol.reverseAligned path)
    (hExchangeReverse : ∀ path,
      protocol.reverseAligned (protocol.reversal path) = protocol.forward path)
    (path : Path) :
    pathwiseEntropyProduction protocol (protocol.reversal path) =
      -pathwiseEntropyProduction protocol path :=
  pathwiseEntropyProduction_reverse protocol hForward hReverse
    hExchangeForward hExchangeReverse path

end FEP095
```

13.95.2 Typeset statement signatures

$$\{\mathsf{Path} : \mathsf{Type}^*\}\,[\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})$$
$$(\mathsf{hForward} : \forall \mathsf{path}, 0 < \mathsf{protocol.forward}\ \mathsf{path})$$
$$(\mathsf{hReverse} : \forall \mathsf{path}, 0 < \mathsf{protocol.reverseAligned}\ \mathsf{path})\,(\mathsf{path} : \mathsf{Path})$$
$$\mathsf{protocol.reverseAligned}\ \mathsf{path} \cdot \mathsf{Real.exp}$$
$$(\mathsf{pathwiseEntropyProduction}\ \mathsf{protocol}\ \mathsf{path}) = \mathsf{protocol.forward}\ \mathsf{path}$$

$$
\begin{aligned}
&\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]\\
&(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})\\
&(\mathsf{hForward} : \forall \mathsf{path}, 0 < \mathsf{protocol.forward}\ \mathsf{path})\\
&(\mathsf{hReverse} : \forall \mathsf{path}, 0 < \mathsf{protocol.reverseAligned}\ \mathsf{path})\ (\mathsf{hExchangeForward} : \forall\\
&\mathsf{path}, \mathsf{protocol.forward}\ (\mathsf{protocol.reversal}\ \mathsf{path}) = \mathsf{protocol.reverseAligned}\ \mathsf{path})\\
&(\mathsf{hExchangeReverse} : \forall \mathsf{path}, \mathsf{protocol.reverseAligned}\ (\mathsf{protocol.reversal}\ \mathsf{path})\\
&= \mathsf{protocol.forward}\ \mathsf{path})\ (\mathsf{path} : \mathsf{Path})\\
&\mathsf{pathwiseEntropyProduction}\ \mathsf{protocol}\ (\mathsf{protocol.reversal}\ \mathsf{path})\\
&= -\mathsf{pathwiseEntropyProduction}\ \mathsf{protocol}\ \mathsf{path}
\end{aligned}
$$

13.96 fep-096 — Integral Fluctuation Theorem

The forward expectation of exponential negative pointwise entropy production equals one for normalized finite full-support forward and reverse path laws. Assumption scope: The path carrier is finite and both path laws have strict support, so the likelihood ratio and logarithm are ordinary real values. The theorem does not treat singular path measures or continuous-time trajectories. Semantic disposition: `formalized`.

13.96.1 Lean sketch

```
import FepSketches.path_thermodynamics

namespace FEP096

open FEP FEP.PathThermodynamics Finset
open scoped BigOperators

variable {Path : Type*} [Fintype Path]

/-- Supported normalized finite path laws obey the integral fluctuation
identity. -/
theorem fep096_integralFluctuation_theorem
    (protocol : FinitePathProtocol Path)
    (hForward : ∀ path, 0 < protocol.forward path)
    (hReverse : ∀ path, 0 < protocol.reverseAligned path) :
    Σ path, protocol.forward path *
        Real.exp (-pathwiseEntropyProduction protocol path) = 1 :=
  integralFluctuation_eq_one protocol hForward hReverse

end FEP096
```

13.96.2 Typeset statement signatures

$$
\begin{aligned}
&\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}]\\
&(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path})\\
&(\mathsf{hForward} : \forall \mathsf{path}, 0 < \mathsf{protocol.forward}\ \mathsf{path})\\
&(\mathsf{hReverse} : \forall \mathsf{path}, 0 < \mathsf{protocol.reverseAligned}\ \mathsf{path})\\
&\sum \mathsf{path}, \mathsf{protocol.forward}\ \mathsf{path} \cdot \mathsf{Real.exp}\\
&(-\mathsf{pathwiseEntropyProduction}\ \mathsf{protocol}\ \mathsf{path}) = 1
\end{aligned}
$$

13.97 fep-097 — Finite Jarzynski Equality

A normalized finite exponential-work protocol has work average equal to the exponential of negative inverse temperature times free-energy difference. Assumption scope: HasJarzynskiNormalization explicitly packages positive beta and the identity that the expectation of exp(-beta times (work minus free-energy difference)) equals one. The theorem algebraically extracts Jarzynski's equality from this normalization; it does not derive that premise from microscopic reversibility. Semantic disposition: `conditional_proxy`.

13.97.1 Lean sketch

```
import FepSketches.path_thermodynamics
```

```
namespace FEP097

open FEP FEP.PathThermodynamics

variable {Path : Type*} [Fintype Path]

/-- A positive inverse temperature and explicit exponential-work
normalization imply the finite Jarzynski equality. -/
theorem fep097_finiteJarzynski_equality
    (law : FiniteLaw Path) (beta deltaFreeEnergy : ℝ)
    (work : Path → ℝ)
    (hNormalization :
      HasJarzynskiNormalization law beta deltaFreeEnergy work) :
    exponentialWorkAverage law beta work =
      Real.exp (-beta * deltaFreeEnergy) :=
  finiteJarzynski_eq law beta deltaFreeEnergy work hNormalization

end FEP097
```

13.97.2 Typeset statement signatures

$$\{\mathsf{Path} : \mathsf{Type}^{*}\}\ [\mathsf{Fintype}\ \mathsf{Path}]$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{Path})\ (\mathsf{beta}\ \mathsf{deltaFreeEnergy} : \mathbb{R})\ (\mathsf{work} : \mathsf{Path} \Rightarrow \mathbb{R})$$
$$(\mathsf{hNormalization} : \mathsf{HasJarzynskiNormalization}\ \mathsf{law}\ \mathsf{beta}\ \mathsf{deltaFreeEnergy}\ \mathsf{work})$$
$$\mathsf{exponentialWorkAverage}\ \mathsf{law}\ \mathsf{beta}\ \mathsf{work} = \mathsf{Real.exp}\ (-\mathsf{beta} \cdot \mathsf{deltaFreeEnergy})$$

13.98 fep-098 — Local Detailed Balance and Current Cancellation

Detailed balance cancels every oriented finite probability current, while a zero reverse rate is exposed as a totalized-log affinity boundary. Assumption scope: The law and transition kernel are normalized finite carriers and current cancellation assumes IsReversible. The zero-rate theorem reports the project's real-log value zero and must not be read as a finite extended-real affinity. Semantic disposition: `formalized`.

13.98.1 Lean sketch

```
import FepSketches.path_thermodynamics

namespace FEP098

open FEP FEP.FiniteMarkovDynamics FEP.PathThermodynamics

variable {State : Type*} [Fintype State]

/-- Finite detailed balance cancels every oriented local probability current. -/
theorem fep098_localDetailedBalance_currentCancellation
    (law : FiniteLaw State) (kernel : FiniteKernel State State)
    (hReversible : IsReversible law kernel) (source target : State) :
    probabilityCurrent law kernel source target = 0 :=
  localDetailedBalance_current_zero law kernel hReversible source target

/-- A zero reverse rate is kept as an explicit totalized-log boundary rather
than reported as an extended-real affinity. -/
theorem fep098_zeroReverseRate_boundary
    (law : FiniteLaw State) (kernel : FiniteKernel State State)
    (source target : State) (hZero : kernel target source = 0) :
    localAffinity law kernel source target = 0 :=
  localAffinity_zero_reverseRate_boundary law kernel source target hZero

end FEP098
```

13.98.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})$$
$$(\mathsf{hReversible} : \mathsf{IsReversible}\ \mathsf{law}\ \mathsf{kernel})\ (\mathsf{source}\ \mathsf{target} : \mathsf{State})$$
$$\mathsf{probabilityCurrent}\ \mathsf{law}\ \mathsf{kernel}\ \mathsf{source}\ \mathsf{target} = 0$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})$$
$$(\mathsf{source}\ \mathsf{target} : \mathsf{State})\ (\mathsf{hZero} : \mathsf{kernel}\ \mathsf{target}\ \mathsf{source} = 0)$$
$$\mathsf{localAffinity}\ \mathsf{law}\ \mathsf{kernel}\ \mathsf{source}\ \mathsf{target} = 0$$

13.99 fep-099 — Reversible-Chain One-Step KL Dissipation

One strictly positive reversible finite Markov step cannot increase KL to its stationary law; the identity transition saturates equality and an explicit irreversible path protocol has positive entropy production. Assumption scope: State is finite and nonempty, actual and stationary laws have full support, every transition atom is positive, and detailed balance supplies stationarity. The result is one-step dissipation, not strict contraction for every nonidentity kernel or a continuous-time entropy-production theorem. Semantic disposition: `formalized`.

13.99.1 Lean sketch

```
import FepSketches.path_thermodynamics

namespace FEP099

open FEP FEP.FiniteInformation FEP.FiniteMarkovDynamics
  FEP.PathThermodynamics

variable {State : Type*} [Fintype State]

/-- One full-support reversible Markov step dissipates KL to its stationary
law; detailed balance supplies stationarity and finite channel data processing
supplies the inequality. -/
theorem fep099_reversibleChain_oneStep_KL_dissipation
    [Nonempty State]
    (actual stationary : FiniteLaw State)
    (kernel : FiniteKernel State State)
    (hActual : ∀ state, 0 < actual state)
    (hStationary : ∀ state, 0 < stationary state)
    (hKernel : ∀ source target, 0 < kernel source target)
    (hReversible : IsReversible stationary kernel) :
    finiteKL (kernel.predictive actual) stationary ≤
      finiteKL actual stationary :=
  reversibleKL_oneStep_dissipation actual stationary kernel hActual
    hStationary hKernel hReversible

/-- The identity transition is the exact reversible equality boundary. -/
theorem fep099_reversibleEquality_boundary [DecidableEq State]
    (actual stationary : FiniteLaw State) :
    finiteKL
        ((FiniteKernel.identity : FiniteKernel State State).predictive actual)
        stationary = finiteKL actual stationary :=
  identityKernel_KL_equality actual stationary

/-- A concrete unequal full-support Boolean forward/reverse pair has strictly
positive path entropy production. -/
theorem fep099_irreversiblePositiveProduction_witness :
    0 < entropyProduction irreversibleBoolProtocol :=
  irreversibleBool_entropyProduction_pos

end FEP099
```

13.99.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$[\mathsf{Nonempty}\ \mathsf{State}]\ (\mathsf{actual}\ \mathsf{stationary} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$(\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\ (\mathsf{hActual} : \forall \mathsf{state}, 0 < \mathsf{actual}\ \mathsf{state})$$
$$(\mathsf{hStationary} : \forall \mathsf{state}, 0 < \mathsf{stationary}\ \mathsf{state})$$
$$(\mathsf{hKernel} : \forall \mathsf{source}\ \mathsf{target}, 0 < \mathsf{kernel}\ \mathsf{source}\ \mathsf{target})$$
$$(\mathsf{hReversible} : \mathsf{IsReversible}\ \mathsf{stationary}\ \mathsf{kernel})$$
$$\mathsf{finiteKL}\ (\mathsf{kernel.predictive}\ \mathsf{actual})\ \mathsf{stationary} \le \mathsf{finiteKL}\ \mathsf{actual}\ \mathsf{stationary}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$[\mathsf{DecidableEq}\ \mathsf{State}]\ (\mathsf{actual}\ \mathsf{stationary} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$\mathsf{finiteKL}\ ((\mathsf{FiniteKernel.identity} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State}).\mathsf{predictive}\ \mathsf{actual})$$
$$\mathsf{stationary} = \mathsf{finiteKL}\ \mathsf{actual}\ \mathsf{stationary}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$0 < \mathsf{entropyProduction}\ \mathsf{irreversibleBoolProtocol}$$

13.100 fep-100 — Categorical Fisher Positivity on Simplex Tangents

The inherited Fisher metric is strictly positive on every certified nonzero categorical simplex tangent under full support, while a duplicated-score model exhibits an explicit nonzero null direction. Assumption scope: CategoricalFisherCarrier packages a full-support law, an existing ScoreModel, and exact metric and lowering-map representations on simplex tangents. The general positivity theorem depends on this certificate. A separate genuine Fin 2 categorical construction proves that its one-dimensional simplex tangent space is spanned by (1,-1) and is positive definite there; no claim is made that every arbitrary score parameterization is full rank. Semantic disposition: `conditional_proxy`.

13.100.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP100

open FEP FEP.GeometricOptimization FEP.InformationGeometry

variable {d : ℕ}

/-- The inherited Fisher metric is strictly positive on every certified
nonzero categorical simplex tangent under full support. -/
theorem fep100_categoricalFisher_simplexTangent_positivity
    (carrier : CategoricalFisherCarrier d)
    (tangent : Fin d → ℝ) (hTangent : IsSimplexTangent tangent)
    (hNonzero : tangent ≠ 0) :
    0 < fisherMetric carrier.model tangent tangent :=
  categoricalFisher_pos carrier tangent hTangent hNonzero

/-- The genuine two-outcome categorical carrier is positive definite on its
entire nonzero simplex tangent space. -/
theorem fep100_twoCategorical_simplexMetric_fullRank
    (tangent : Fin 2 → ℝ) (hTangent : IsSimplexTangent tangent)
    (hNonzero : tangent ≠ 0) :
    0 < fisherMetric twoCategoricalFisherCarrier.model tangent tangent :=
  twoCategorical_simplexMetric_fullRank tangent hTangent hNonzero

/-- Every two-outcome simplex tangent is generated by the explicit nonzero
`(1, -1)` direction. -/
theorem fep100_twoCategorical_tangent_spanning
    (tangent : Fin 2 → ℝ) (hTangent : IsSimplexTangent tangent) :
    tangent = fun coordinate ↦
      tangent 0 * twoCategoricalTangent coordinate :=
```

```
  twoCategoricalTangent_spans tangent hTangent

/-- A concrete nonzero simplex tangent has Fisher energy exactly four. -/
theorem fep100_fullRank_categorical_example :
    IsSimplexTangent twoCategoricalTangent ∧
      twoCategoricalTangent ≠ 0 ∧
      fisherMetric twoCategoricalFisherCarrier.model
        twoCategoricalTangent twoCategoricalTangent = 4 :=
  twoCategorical_nonzeroTangent_metric

/-- Duplicated score coordinates expose a nonzero null direction. -/
theorem fep100_rankDeficient_nullDirection_example :
    duplicatedScoreNullTangent ≠ 0 ∧
      fisherMetric duplicatedFairBernoulliScoreModel
        duplicatedScoreNullTangent duplicatedScoreNullTangent = 0 :=
  duplicatedScore_nullDirection_example

end FEP100
```

13.100.2 Typeset statement signatures

$$\{d : \mathbb{N}\}$$
$$(\mathsf{carrier} : \mathsf{CategoricalFisherCarrier}\ d)\ (\mathsf{tangent} : \mathsf{Fin}\ d \Rightarrow \mathbb{R})$$
$$(\mathsf{hTangent} : \mathsf{IsSimplexTangent}\ \mathsf{tangent})\ (\mathsf{hNonzero} : \mathsf{tangent} \neq 0)$$
$$0 < \mathsf{fisherMetric}\ \mathsf{carrier.model}\ \mathsf{tangent}\ \mathsf{tangent}$$

$$\{d : \mathbb{N}\}$$
$$(\mathsf{tangent} : \mathsf{Fin}\ 2 \Rightarrow \mathbb{R})\ (\mathsf{hTangent} : \mathsf{IsSimplexTangent}\ \mathsf{tangent})$$
$$(\mathsf{hNonzero} : \mathsf{tangent} \neq 0)$$
$$0 < \mathsf{fisherMetric}\ \mathsf{twoCategoricalFisherCarrier.model}\ \mathsf{tangent}\ \mathsf{tangent}$$

$$\{d : \mathbb{N}\}$$
$$(\mathsf{tangent} : \mathsf{Fin}\ 2 \Rightarrow \mathbb{R})\ (\mathsf{hTangent} : \mathsf{IsSimplexTangent}\ \mathsf{tangent})$$
$$\mathsf{tangent} = \lambda\, \mathsf{coordinate} \mapsto \mathsf{tangent}\ 0 \cdot \mathsf{twoCategoricalTangent}\ \mathsf{coordinate}$$

$$\{d : \mathbb{N}\}$$
$$\mathsf{IsSimplexTangent}\ \mathsf{twoCategoricalTangent} \wedge \mathsf{twoCategoricalTangent} \neq 0$$
$$\wedge\ \mathsf{fisherMetric}\ \mathsf{twoCategoricalFisherCarrier.model}\ \mathsf{twoCategoricalTangent}$$
$$\mathsf{twoCategoricalTangent} = 4$$

$$\{d : \mathbb{N}\}$$
$$\mathsf{duplicatedScoreNullTangent} \neq 0 \wedge \mathsf{fisherMetric}\ \mathsf{duplicatedFairBernoulliScoreModel}$$
$$\mathsf{duplicatedScoreNullTangent}\ \mathsf{duplicatedScoreNullTangent} = 0$$

13.101 fep-101 — Fisher Pullback under Reparameterization

Pulling the existing finite Fisher metric through a composite Jacobian equals successive pullback, and every such pullback is positive semidefinite. Assumption scope: Outcome and coordinate carriers are finite and Jacobians are finite real matrices. No invertibility is needed for pullback or semidefiniteness, so a rank-deficient chart may retain null directions; positive definiteness is not claimed here. Semantic disposition: `formalized`.

13.101.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP101

open FEP FEP.GeometricOptimization FEP.InformationGeometry
```

```
open scoped Matrix

variable {Outcome : Type*} [Fintype Outcome]
variable {d k : ℕ}

/-- Fisher pullback along a composite finite Jacobian equals successive
pullback through the two coordinate maps. -/
theorem fep101_fisherPullback_reparameterization
    (model : ScoreModel Outcome d)
    (outer : Matrix (Fin d) (Fin k) ℝ)
    (inner : Matrix (Fin k) (Fin d) ℝ)
    (left right : Fin d → ℝ) :
    pullbackMetric model (outer * inner) left right =
      pullbackMetric model outer (inner.mulVec left) (inner.mulVec right) :=
  fisherPullback_comp model outer inner left right

/-- Every Fisher pullback remains positive semidefinite. -/
theorem fep101_fisherPullback_nonnegative
    (model : ScoreModel Outcome d)
    (jacobian : Matrix (Fin d) (Fin k) ℝ) (tangent : Fin k → ℝ) :
    0 ≤ pullbackMetric model jacobian tangent tangent :=
  pullbackMetric_nonneg model jacobian tangent

end FEP101
```

13.101.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ \{d\ k : \mathbb{N}\}$$
$$(\mathsf{model} : \mathsf{ScoreModel}\ \mathsf{Outcome}\ d)\ (\mathsf{outer} : \mathsf{Matrix}\ (\mathsf{Fin}\ d)\ (\mathsf{Fin}\ k)\ \mathbb{R})$$
$$(\mathsf{inner} : \mathsf{Matrix}\ (\mathsf{Fin}\ k)\ (\mathsf{Fin}\ d)\ \mathbb{R})\ (\mathsf{left\ right} : \mathsf{Fin}\ d \Rightarrow \mathbb{R})$$
$$\mathsf{pullbackMetric\ model}\ (\mathsf{outer} \cdot \mathsf{inner})\ \mathsf{left\ right} = \mathsf{pullbackMetric\ model\ outer}$$
$$(\mathsf{inner.mulVec\ left})\ (\mathsf{inner.mulVec\ right})$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ \{d\ k : \mathbb{N}\}$$
$$(\mathsf{model} : \mathsf{ScoreModel}\ \mathsf{Outcome}\ d)\ (\mathsf{jacobian} : \mathsf{Matrix}\ (\mathsf{Fin}\ d)\ (\mathsf{Fin}\ k)\ \mathbb{R})$$
$$(\mathsf{tangent} : \mathsf{Fin}\ k \Rightarrow \mathbb{R})$$
$$0 \le \mathsf{pullbackMetric\ model\ jacobian\ tangent\ tangent}$$

13.102 fep-102 — Unbiased Scalar Cramér–Rao Bound under Score Regularity

Finite unbiased scalar estimation under score regularity and positive Fisher information satisfies variance times Fisher information at least one. Assumption scope: ScalarCramerRaoCertificate explicitly records centered unbiasedness, unit estimator–score covariance, and strictly positive scalar Fisher information for an existing one-parameter ScoreModel. The theorem does not derive score regularity by differentiating a parameterized family or cover biased estimators. Semantic disposition: `formalized`.

13.102.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP102

open FEP FEP.GeometricOptimization FEP.InformationGeometry

variable {Outcome : Type*} [Fintype Outcome]

/-- Unbiasedness, the finite score-regularity identity, and positive Fisher
information imply the scalar Cramér--Rao lower bound. -/
theorem fep102_unbiasedScalar_cramerRao
    (model : ScoreModel Outcome 1) (estimator : Outcome → ℝ)
    (target : ℝ)
```

```
    (certificate : ScalarCramerRaoCertificate model estimator target) :
    1 ≤ estimatorVariance model.law estimator target * scalarFisher model :=
  scalarCramerRao model estimator target certificate

end FEP102
```

13.102.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]$$
$$(\mathsf{model} : \mathsf{ScoreModel}\ \mathsf{Outcome}\ 1)\ (\mathsf{estimator} : \mathsf{Outcome} \Rightarrow \mathbb{R})\ (\mathsf{target} : \mathbb{R})$$
$$(\mathsf{certificate} : \mathsf{ScalarCramerRaoCertificate}\ \mathsf{model}\ \mathsf{estimator}\ \mathsf{target})$$
$$1 \leq \mathsf{estimatorVariance}\ \mathsf{model.law}\ \mathsf{estimator}\ \mathsf{target} \cdot \mathsf{scalarFisher}\ \mathsf{model}$$

13.103 fep-103 — Natural-Gradient Equivariance under an Invertible Full-Rank Chart

Under invertible Fisher and chart matrices, the natural gradient transports equivariantly so lowering the chart-coordinate vector yields the pulled-back covector. Assumption scope: The finite ScoreModel Fisher matrix and square chart Jacobian each carry an Invertible instance. Rank-deficient metrics or charts are excluded because the natural-gradient and chart-coordinate inverses would not be uniquely defined. The supporting weighted Cauchy–Schwarz theorem requires positive right energy. Semantic disposition: `formalized`.

13.103.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP103

open FEP FEP.GeometricOptimization FEP.InformationGeometry
open scoped Matrix

variable {Outcome : Type*} [Fintype Outcome]
variable {d : ℕ}

/-- An invertible full-rank Fisher metric and invertible chart transport the
natural gradient equivariantly to the pulled-back covector. -/
theorem fep103_naturalGradient_equivariance
    (model : ScoreModel Outcome d) [Invertible (fisherMatrix model)]
    (jacobian : Matrix (Fin d) (Fin d) ℝ) [Invertible jacobian]
    (covector : Fin d → ℝ) :
    chartPullbackLower model jacobian
        (chartCoordinates jacobian (naturalGradient model covector)) =
      chartCovector jacobian covector :=
  naturalGradient_equivariant model jacobian covector

/-- The slice's finite Cauchy--Schwarz proof spike remains directly available
at this gate topic, with its strict positive-energy boundary explicit. -/
theorem fep103_weightedCauchySchwarz_proofSpike
    (law : FiniteLaw Outcome) (left right : Outcome → ℝ)
    (hRight : 0 < ∑ outcome, law outcome * right outcome ^ 2) :
    (∑ outcome, law outcome * left outcome * right outcome) ^ 2 ≤
      (∑ outcome, law outcome * left outcome ^ 2) *
        ∑ outcome, law outcome * right outcome ^ 2 :=
  weightedCauchySchwarz_of_right_pos law left right hRight

end FEP103
```

13.103.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ \{d : \mathbb{N}\}$$
$$(\mathsf{model} : \mathsf{ScoreModel}\ \mathsf{Outcome}\ d)\ [\mathsf{Invertible}\ (\mathsf{fisherMatrix}\ \mathsf{model})]$$
$$(\mathsf{jacobian} : \mathsf{Matrix}\ (\mathsf{Fin}\ d)\ (\mathsf{Fin}\ d)\ \mathbb{R})\ [\mathsf{Invertible}\ \mathsf{jacobian}]$$
$$(\mathsf{covector} : \mathsf{Fin}\ d \Rightarrow \mathbb{R})$$
$$\mathsf{chartPullbackLower}\ \mathsf{model}\ \mathsf{jacobian}\ (\mathsf{chartCoordinates}\ \mathsf{jacobian}\ (\mathsf{naturalGradient}$$
$$\mathsf{model}\ \mathsf{covector})) = \mathsf{chartCovector}\ \mathsf{jacobian}\ \mathsf{covector}$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ \{d : \mathbb{N}\}$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{Outcome})\ (\mathsf{left}\ \mathsf{right} : \mathsf{Outcome} \Rightarrow \mathbb{R})\ (\mathsf{hRight} : 0 < \sum \mathsf{outcome},$$
$$\mathsf{law}\ \mathsf{outcome} \cdot \mathsf{right}\ \mathsf{outcome}^2)$$
$$(\sum \mathsf{outcome}, \mathsf{law}\ \mathsf{outcome} \cdot \mathsf{left}\ \mathsf{outcome} \cdot \mathsf{right}\ \mathsf{outcome})^2$$
$$\leq (\sum \mathsf{outcome}, \mathsf{law}\ \mathsf{outcome} \cdot \mathsf{left}\ \mathsf{outcome}^2) \cdot \sum \mathsf{outcome}, \mathsf{law}\ \mathsf{outcome} \cdot \mathsf{right}$$
$$\mathsf{outcome}^2$$

13.104 fep-104 — Mirror-Descent Three-Point Identity

The algebraic Bregman expression associated with supplied potential and gradient data obeys the exact three-point identity used by mirror descent. Assumption scope: The coordinate carrier is finite and potential and gradient are arbitrary supplied functions. The identity is algebraic and assumes neither convexity nor that the gradient is the derivative of the potential; those extra properties are required before interpreting every term as a nonnegative divergence or an algorithmic descent guarantee. Semantic disposition: `formalized`.

13.104.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP104

open FEP.GeometricOptimization

variable {d : ℕ}

/-- The Bregman divergence generated by supplied potential and gradient data
obeys the exact mirror-descent three-point identity. -/
theorem fep104_mirrorDescent_threePoint_identity
    (potential : (Fin d → ℝ) → ℝ)
    (gradient : (Fin d → ℝ) → Fin d → ℝ)
    (left middle right : Fin d → ℝ) :
    bregmanDivergence potential gradient left right -
        bregmanDivergence potential gradient left middle -
        bregmanDivergence potential gradient middle right =
      coordinatePairing
        (fun coordinate ↦
          gradient middle coordinate - gradient right coordinate)
        (fun coordinate ↦ left coordinate - middle coordinate) :=
  mirrorDescent_threePoint_identity potential gradient left middle right

end FEP104
```

13.104.2 Typeset statement signatures

$$
\begin{aligned}
&\{d : \mathbb{N}\}\\
&(\mathsf{potential} : (\mathsf{Fin}\, d \Rightarrow \mathbb{R}) \Rightarrow \mathbb{R})\,(\mathsf{gradient} : (\mathsf{Fin}\, d \Rightarrow \mathbb{R}) \Rightarrow \mathsf{Fin}\, d \Rightarrow \mathbb{R})\\
&(\mathsf{left\ middle\ right} : \mathsf{Fin}\, d \Rightarrow \mathbb{R})\\
&\mathsf{bregmanDivergence\ potential\ gradient\ left\ right} - \mathsf{bregmanDivergence\ potential}\\
&\mathsf{gradient\ left\ middle} - \mathsf{bregmanDivergence\ potential\ gradient\ middle\ right}\\
&= \mathsf{coordinatePairing}\,(\lambda\, \mathsf{coordinate} \mapsto \mathsf{gradient\ middle\ coordinate} - \mathsf{gradient}\\
&\mathsf{right\ coordinate})\,(\lambda\, \mathsf{coordinate} \mapsto \mathsf{left\ coordinate} - \mathsf{middle\ coordinate})
\end{aligned}
$$

13.105 fep-105 — Bregman Pythagorean Law for an Affine Information Projection

An affine projection certificate yields exact Bregman Pythagorean splitting, and a nonnegative residual divergence makes the certified projection a minimizer over the maintained affine set. Assumption scope: AffineBregmanProjection explicitly supplies projection membership and gradient-difference orthogonality for every candidate. Minimization separately assumes residual Bregman nonnegativity. Existence, uniqueness, convexity, and differentiability of the projection are not derived. Semantic disposition: `conditional_proxy`.

13.105.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP105

open FEP.GeometricOptimization

variable {d : ℕ}

/-- Orthogonality on the maintained affine family yields the exact Bregman
Pythagorean law. -/
theorem fep105_affineProjection_bregmanPythagorean
    (potential : (Fin d → ℝ) → ℝ)
    (gradient : (Fin d → ℝ) → Fin d → ℝ)
    (affineSet : Set (Fin d → ℝ))
    (base projection candidate : Fin d → ℝ)
    (hProjection : AffineBregmanProjection potential gradient affineSet
      base projection)
    (hCandidate : candidate ∈ affineSet) :
    bregmanDivergence potential gradient candidate base =
      bregmanDivergence potential gradient candidate projection +
        bregmanDivergence potential gradient projection base :=
  affineProjection_bregmanPythagorean potential gradient affineSet base
    projection candidate hProjection hCandidate

/-- Nonnegative residual divergence makes the affine projection a minimizer. -/
theorem fep105_affineProjection_minimizes
    (potential : (Fin d → ℝ) → ℝ)
    (gradient : (Fin d → ℝ) → Fin d → ℝ)
    (affineSet : Set (Fin d → ℝ))
    (base projection candidate : Fin d → ℝ)
    (hProjection : AffineBregmanProjection potential gradient affineSet
      base projection)
    (hCandidate : candidate ∈ affineSet)
    (hNonnegative :
      0 ≤ bregmanDivergence potential gradient candidate projection) :
    bregmanDivergence potential gradient projection base ≤
      bregmanDivergence potential gradient candidate base :=
  affineProjection_minimizes potential gradient affineSet base projection
    candidate hProjection hCandidate hNonnegative

end FEP105
```

13.105.2 Typeset statement signatures

$$
\begin{aligned}
&\{d : \mathbb{N}\}\\
&(\mathsf{potential} : (\mathsf{Fin}\, d \Rightarrow \mathbb{R}) \Rightarrow \mathbb{R})\,(\mathsf{gradient} : (\mathsf{Fin}\, d \Rightarrow \mathbb{R}) \Rightarrow \mathsf{Fin}\, d \Rightarrow \mathbb{R})\\
&(\mathsf{affineSet} : \mathsf{Set}\,(\mathsf{Fin}\, d \Rightarrow \mathbb{R}))\,(\mathsf{base\ projection\ candidate} : \mathsf{Fin}\, d \Rightarrow \mathbb{R})\\
&(\mathsf{hProjection} : \mathsf{AffineBregmanProjection\ potential\ gradient\ affineSet\ base}\\
&\mathsf{projection})\,(\mathsf{hCandidate} : \mathsf{candidate} \in \mathsf{affineSet})\\
&\mathsf{bregmanDivergence\ potential\ gradient\ candidate\ base} = \mathsf{bregmanDivergence}\\
&\mathsf{potential\ gradient\ candidate\ projection} + \mathsf{bregmanDivergence\ potential\ gradient}\\
&\mathsf{projection\ base}
\end{aligned}
$$

$$
\begin{aligned}
&\{d : \mathbb{N}\}\\
&(\mathsf{potential} : (\mathsf{Fin}\, d \Rightarrow \mathbb{R}) \Rightarrow \mathbb{R})\,(\mathsf{gradient} : (\mathsf{Fin}\, d \Rightarrow \mathbb{R}) \Rightarrow \mathsf{Fin}\, d \Rightarrow \mathbb{R})\\
&(\mathsf{affineSet} : \mathsf{Set}\,(\mathsf{Fin}\, d \Rightarrow \mathbb{R}))\,(\mathsf{base\ projection\ candidate} : \mathsf{Fin}\, d \Rightarrow \mathbb{R})\\
&(\mathsf{hProjection} : \mathsf{AffineBregmanProjection\ potential\ gradient\ affineSet\ base}\\
&\mathsf{projection})\,(\mathsf{hCandidate} : \mathsf{candidate} \in \mathsf{affineSet})\,(\mathsf{hNonnegative} : 0\\
&\leq \mathsf{bregmanDivergence\ potential\ gradient\ candidate\ projection})\\
&\mathsf{bregmanDivergence\ potential\ gradient\ projection\ base} \leq \mathsf{bregmanDivergence}\\
&\mathsf{potential\ gradient\ candidate\ base}
\end{aligned}
$$

13.106 fep-106 — Replicator–Natural-Gradient Equivalence

On a certified categorical Fisher carrier, the replicator vector is the natural gradient of centered fitness and has zero total mass. Assumption scope: CategoricalFisherCarrier supplies full support and the exact bridge from the existing ScoreModel lowering map to categorical coordinates. The result is an instantaneous finite-simplex vector-field identity, not positivity preservation or convergence of a time-integrated replicator flow. Semantic disposition: `conditional_proxy`.

13.106.1 Lean sketch

```
import FepSketches.geometric_optimization

namespace FEP106

open FEP FEP.GeometricOptimization FEP.InformationGeometry Finset
open scoped BigOperators

variable {d : ℕ}

/-- On a certified categorical Fisher carrier, the replicator field is the
natural gradient of centered fitness. -/
theorem fep106_replicator_naturalGradient_equivalence
    (carrier : CategoricalFisherCarrier d) (fitness : Fin d → ℝ) :
    IsNaturalGradient carrier.model
      (fun coordinate ↦ fitness coordinate - meanFitness carrier.law fitness)
      (replicatorVector carrier.law fitness) :=
  replicator_naturalGradient_equivalence carrier fitness

/-- Replicator dynamics preserves the affine simplex constraint. -/
theorem fep106_replicator_massConservation
    (law : FiniteLaw (Fin d)) (fitness : Fin d → ℝ) :
    ∑ coordinate, replicatorVector law fitness coordinate = 0 :=
  replicatorVector_isSimplexTangent law fitness

/-- A concrete nonconstant two-outcome fitness produces a nonzero replicator
field and realizes the categorical natural-gradient equivalence. -/
theorem fep106_twoCategorical_nonzero_replicator_witness :
    twoCategoricalFitness 0 ≠ twoCategoricalFitness 1 ∧
      replicatorVector twoCategoricalLaw twoCategoricalFitness ≠ 0 ∧
```

```
      IsNaturalGradient twoCategoricalFisherCarrier.model
        (fun coordinate ↦
          twoCategoricalFitness coordinate -
            meanFitness twoCategoricalLaw twoCategoricalFitness)
        (replicatorVector twoCategoricalLaw twoCategoricalFitness) :=
  twoCategorical_replicator_nonzero_witness

end FEP106
```

13.106.2 Typeset statement signatures

$$\{d : \mathbb{N}\}$$
$$(\mathsf{carrier} : \mathsf{CategoricalFisherCarrier}\ d)\ (\mathsf{fitness} : \mathsf{Fin}\ d \Rightarrow \mathbb{R})$$
$$\mathsf{IsNaturalGradient\ carrier.model}\ (\lambda\ \mathsf{coordinate} \mapsto \mathsf{fitness\ coordinate}$$
$$- \mathsf{meanFitness\ carrier.law\ fitness})\ (\mathsf{replicatorVector\ carrier.law\ fitness})$$

$$\{d : \mathbb{N}\}$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\ (\mathsf{Fin}\ d))\ (\mathsf{fitness} : \mathsf{Fin}\ d \Rightarrow \mathbb{R})$$
$$\sum \mathsf{coordinate}, \mathsf{replicatorVector\ law\ fitness\ coordinate} = 0$$

$$\{d : \mathbb{N}\}$$
$$\mathsf{twoCategoricalFitness}\ 0 \neq \mathsf{twoCategoricalFitness}\ 1 \wedge \mathsf{replicatorVector}$$
$$\mathsf{twoCategoricalLaw\ twoCategoricalFitness} \neq 0 \wedge \mathsf{IsNaturalGradient}$$
$$\mathsf{twoCategoricalFisherCarrier.model}\ (\lambda\ \mathsf{coordinate} \mapsto \mathsf{twoCategoricalFitness}$$
$$\mathsf{coordinate} - \mathsf{meanFitness\ twoCategoricalLaw\ twoCategoricalFitness})$$
$$(\mathsf{replicatorVector\ twoCategoricalLaw\ twoCategoricalFitness})$$

13.107 fep-107 — Product-Agent Generative Law

Independent product priors and product observation kernels yield a normalized joint kernel whose generative mass factors exactly into the two agent-level generative masses. Assumption scope: Both state and observation carriers are finite and each component law and kernel is normalized. Independence is built into FiniteLaw.product and the productKernel coordinate definition; the theorem does not infer independence from an arbitrary coupled multiagent law. Semantic disposition: `formalized`.

13.107.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Product-Agent Generative Law -/
namespace FEP107

open FEP Finset
open scoped BigOperators

variable {State₁ State₂ Observation₁ Observation₂ : Type*}
  [Fintype State₁] [Fintype State₂]
  [Fintype Observation₁] [Fintype Observation₂]

/-- Independent product kernels retain a normalized observation law at every
joint state. -/
theorem fep107_productKernel_normalized
    (left : FiniteKernel State₁ Observation₁)
    (right : FiniteKernel State₂ Observation₂)
    (state : State₁ × State₂) :
    Σ observation,
      FEP.CollectiveInference.productKernel left right state observation = 1 :=
  FEP.CollectiveInference.productKernel_sum_one left right state

/-- With product priors and product kernels, the joint generative mass is the
```

```
product of the two agent-level generative masses. -/
theorem fep107_productAgent_generative_mass
    (priorLeft : FiniteLaw State₁) (priorRight : FiniteLaw State₂)
    (kernelLeft : FiniteKernel State₁ Observation₁)
    (kernelRight : FiniteKernel State₂ Observation₂)
    (stateLeft : State₁) (stateRight : State₂)
    (observationLeft : Observation₁) (observationRight : Observation₂) :
    (FEP.CollectiveInference.productKernel kernelLeft kernelRight).joint
        (FiniteLaw.product priorLeft priorRight)
        ((stateLeft, stateRight), (observationLeft, observationRight)) =
      kernelLeft.joint priorLeft (stateLeft, observationLeft) *
        kernelRight.joint priorRight (stateRight, observationRight) :=
  FEP.CollectiveInference.productAgent_joint_mass
    priorLeft priorRight kernelLeft kernelRight
    stateLeft stateRight observationLeft observationRight

end FEP107
```

13.107.2 Typeset statement signatures

$$\{State_1\ State_2\ Observation_1\ Observation_2 : \mathsf{Type}^*\}$$
$$(\mathsf{left} : \mathsf{FiniteKernel}\ State_1\ Observation_1)$$
$$(\mathsf{right} : \mathsf{FiniteKernel}\ State_2\ Observation_2)\ (\mathsf{state} : State_1 \times State_2)$$
$$\sum \mathsf{observation},\ \mathsf{FEP.CollectiveInference.productKernel\ left\ right\ state}$$
$$\mathsf{observation} = 1$$

$$\{State_1\ State_2\ Observation_1\ Observation_2 : \mathsf{Type}^*\}$$
$$(\mathsf{priorLeft} : \mathsf{FiniteLaw}\ State_1)\ (\mathsf{priorRight} : \mathsf{FiniteLaw}\ State_2)$$
$$(\mathsf{kernelLeft} : \mathsf{FiniteKernel}\ State_1\ Observation_1)$$
$$(\mathsf{kernelRight} : \mathsf{FiniteKernel}\ State_2\ Observation_2)\ (\mathsf{stateLeft} : State_1)$$
$$(\mathsf{stateRight} : State_2)\ (\mathsf{observationLeft} : Observation_1)$$
$$(\mathsf{observationRight} : Observation_2)$$
$$(\mathsf{FEP.CollectiveInference.productKernel\ kernelLeft\ kernelRight}).\mathsf{joint}$$
$$(\mathsf{FiniteLaw.product\ priorLeft\ priorRight})\ ((\mathsf{stateLeft}, \mathsf{stateRight}),$$
$$(\mathsf{observationLeft}, \mathsf{observationRight})) = \mathsf{kernelLeft.joint\ priorLeft}$$
$$(\mathsf{stateLeft}, \mathsf{observationLeft}) \cdot \mathsf{kernelRight.joint\ priorRight}$$
$$(\mathsf{stateRight}, \mathsf{observationRight})$$

13.108 fep-108 — Additive Collective Variational Free Energy

Variational free energy of independent product laws with additive cost is exactly the sum of the two agent-level free energies. Assumption scope: State carriers are finite, both reference factors have full support, actual and reference joint laws are products, and joint cost is additive. The result does not discard a dependence or interaction term for a coupled law or nonadditive cost. Semantic disposition: `formalized`.

13.108.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Additive Collective Variational Free Energy -/
namespace FEP108

open FEP

variable {State₁ State₂ : Type*} [Fintype State₁] [Fintype State₂]

/-- Product-law VFE is additive under full-support reference factors and an
additive cost. -/
theorem fep108_collectiveVFE_additive
```

```
    (actualLeft referenceLeft : FiniteLaw State₁)
    (actualRight referenceRight : FiniteLaw State₂)
    (leftCost : State₁ → ℝ) (rightCost : State₂ → ℝ)
    (referenceLeftPositive : ∀ state, 0 < referenceLeft state)
    (referenceRightPositive : ∀ state, 0 < referenceRight state) :
    FEP.CollectiveInference.collectiveVFE
        actualLeft referenceLeft actualRight referenceRight leftCost rightCost =
      FEP.CollectiveInference.variationalFreeEnergy
          actualLeft referenceLeft leftCost +
        FEP.CollectiveInference.variationalFreeEnergy
          actualRight referenceRight rightCost :=
  FEP.CollectiveInference.collectiveVFE_additive
    actualLeft referenceLeft actualRight referenceRight leftCost rightCost
    referenceLeftPositive referenceRightPositive

/-- The expectation component has its own independent-product certificate. -/
theorem fep108_productCost_additive
    (left : FiniteLaw State₁) (right : FiniteLaw State₂)
    (leftCost : State₁ → ℝ) (rightCost : State₂ → ℝ) :
    FEP.CollectiveInference.expectedCost (FiniteLaw.product left right)
        (fun state => leftCost state.1 + rightCost state.2) =
      FEP.CollectiveInference.expectedCost left leftCost +
        FEP.CollectiveInference.expectedCost right rightCost :=
  FEP.CollectiveInference.productExpectedCost_additive
    left right leftCost rightCost

end FEP108
```

13.108.2 Typeset statement signatures

$$\{State_1\ State_2 : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ State_1]\ [\mathsf{Fintype}\ State_2]$$
$$(\mathsf{actualLeft}\ \mathsf{referenceLeft} : \mathsf{FiniteLaw}\ State_1)$$
$$(\mathsf{actualRight}\ \mathsf{referenceRight} : \mathsf{FiniteLaw}\ State_2)\ (\mathsf{leftCost} : State_1 \Rightarrow \mathbb{R})$$
$$(\mathsf{rightCost} : State_2 \Rightarrow \mathbb{R})\ (\mathsf{referenceLeftPositive} : \forall \mathsf{state}, 0 < \mathsf{referenceLeft}$$
$$\mathsf{state})\ (\mathsf{referenceRightPositive} : \forall \mathsf{state}, 0 < \mathsf{referenceRight}\ \mathsf{state})$$
$$\mathsf{FEP.CollectiveInference.collectiveVFE}\ \mathsf{actualLeft}\ \mathsf{referenceLeft}\ \mathsf{actualRight}$$
$$\mathsf{referenceRight}\ \mathsf{leftCost}\ \mathsf{rightCost}$$
$$= \mathsf{FEP.CollectiveInference.variationalFreeEnergy}\ \mathsf{actualLeft}\ \mathsf{referenceLeft}$$
$$\mathsf{leftCost} + \mathsf{FEP.CollectiveInference.variationalFreeEnergy}\ \mathsf{actualRight}$$
$$\mathsf{referenceRight}\ \mathsf{rightCost}$$

$$\{State_1\ State_2 : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ State_1]\ [\mathsf{Fintype}\ State_2]$$
$$(\mathsf{left} : \mathsf{FiniteLaw}\ State_1)\ (\mathsf{right} : \mathsf{FiniteLaw}\ State_2)\ (\mathsf{leftCost} : State_1 \Rightarrow \mathbb{R})$$
$$(\mathsf{rightCost} : State_2 \Rightarrow \mathbb{R})$$
$$\mathsf{FEP.CollectiveInference.expectedCost}\ (\mathsf{FiniteLaw.product}\ \mathsf{left}\ \mathsf{right})$$
$$(\lambda\ \mathsf{state} \mapsto \mathsf{leftCost}\ \mathsf{state}.1 + \mathsf{rightCost}\ \mathsf{state}.2)$$
$$= \mathsf{FEP.CollectiveInference.expectedCost}\ \mathsf{left}\ \mathsf{leftCost}$$
$$+ \mathsf{FEP.CollectiveInference.expectedCost}\ \mathsf{right}\ \mathsf{rightCost}$$

13.109 fep-109 — Independent-Agent Expected-Free-Energy Additivity

Expected free energy of product predictive and preference laws with additive ambiguity equals the sum of independent agent-level expected free energies. Assumption scope: Both finite preference factors have full support, predictive and preference joint laws are products, and ambiguity is coordinate-additive. No additivity is claimed for statistically coupled agents, shared outcomes, or a nonseparable ambiguity functional. Semantic disposition: `formalized`.

13.109.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Independent-Agent Expected-Free-Energy Additivity -/
namespace FEP109

open FEP

variable {State₁ State₂ : Type*} [Fintype State₁] [Fintype State₂]

/-- EFE additivity follows from product predictive and preference laws,
additive ambiguity, and full-support preferences. -/
theorem fep109_independentEFE_additive
    (predictiveLeft preferenceLeft : FiniteLaw State₁)
    (predictiveRight preferenceRight : FiniteLaw State₂)
    (ambiguityLeft : State₁ → ℝ) (ambiguityRight : State₂ → ℝ)
    (preferenceLeftPositive : ∀ state, 0 < preferenceLeft state)
    (preferenceRightPositive : ∀ state, 0 < preferenceRight state) :
    FEP.CollectiveInference.independentCollectiveEFE
        predictiveLeft preferenceLeft predictiveRight preferenceRight
        ambiguityLeft ambiguityRight =
      FEP.CollectiveInference.expectedFreeEnergy
          predictiveLeft preferenceLeft ambiguityLeft +
        FEP.CollectiveInference.expectedFreeEnergy
          predictiveRight preferenceRight ambiguityRight :=
  FEP.CollectiveInference.independentEFE_additive
    predictiveLeft preferenceLeft predictiveRight preferenceRight
    ambiguityLeft ambiguityRight preferenceLeftPositive preferenceRightPositive

/-- Without the product construction, only the component expectation identity
is available; no dependence term is silently discarded. -/
theorem fep109_additiveAmbiguity_expectation
    (left : FiniteLaw State₁) (right : FiniteLaw State₂)
    (ambiguityLeft : State₁ → ℝ) (ambiguityRight : State₂ → ℝ) :
    FEP.CollectiveInference.expectedCost (FiniteLaw.product left right)
        (fun state => ambiguityLeft state.1 + ambiguityRight state.2) =
      FEP.CollectiveInference.expectedCost left ambiguityLeft +
        FEP.CollectiveInference.expectedCost right ambiguityRight :=
  FEP.CollectiveInference.productExpectedCost_additive
    left right ambiguityLeft ambiguityRight

end FEP109
```

13.109.2 Typeset statement signatures

$$
\begin{aligned}
&\{State_1\ State_2 : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ State_1]\ [\mathsf{Fintype}\ State_2]\\
&(\mathsf{predictiveLeft}\ \mathsf{preferenceLeft} : \mathsf{FiniteLaw}\ State_1)\\
&(\mathsf{predictiveRight}\ \mathsf{preferenceRight} : \mathsf{FiniteLaw}\ State_2)\\
&(\mathsf{ambiguityLeft} : State_1 \Rightarrow \mathbb{R})\ (\mathsf{ambiguityRight} : State_2 \Rightarrow \mathbb{R})\\
&(\mathsf{preferenceLeftPositive} : \forall \mathsf{state}, 0 < \mathsf{preferenceLeft}\ \mathsf{state})\\
&(\mathsf{preferenceRightPositive} : \forall \mathsf{state}, 0 < \mathsf{preferenceRight}\ \mathsf{state})\\
&\mathsf{FEP.CollectiveInference.independentCollectiveEFE}\ \mathsf{predictiveLeft}\ \mathsf{preferenceLeft}\\
&\mathsf{predictiveRight}\ \mathsf{preferenceRight}\ \mathsf{ambiguityLeft}\ \mathsf{ambiguityRight}\\
&= \mathsf{FEP.CollectiveInference.expectedFreeEnergy}\ \mathsf{predictiveLeft}\ \mathsf{preferenceLeft}\\
&\mathsf{ambiguityLeft} + \mathsf{FEP.CollectiveInference.expectedFreeEnergy}\ \mathsf{predictiveRight}\\
&\mathsf{preferenceRight}\ \mathsf{ambiguityRight}
\end{aligned}
$$

$$
\begin{aligned}
&\{State_1\ State_2 : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ State_1]\ [\mathsf{Fintype}\ State_2] \\
&(\mathsf{left} : \mathsf{FiniteLaw}\ State_1)\ (\mathsf{right} : \mathsf{FiniteLaw}\ State_2) \\
&(\mathsf{ambiguityLeft} : State_1 \Rightarrow \mathbb{R})\ (\mathsf{ambiguityRight} : State_2 \Rightarrow \mathbb{R}) \\
&\mathsf{FEP.CollectiveInference.expectedCost}\ (\mathsf{FiniteLaw.product\ left\ right}) \\
&(\lambda\, \mathsf{state} \mapsto \mathsf{ambiguityLeft\ state}.1 + \mathsf{ambiguityRight\ state}.2) \\
&= \mathsf{FEP.CollectiveInference.expectedCost\ left\ ambiguityLeft} \\
&+ \mathsf{FEP.CollectiveInference.expectedCost\ right\ ambiguityRight}
\end{aligned}
$$

13.110 fep-110 — Unit-Weight Product-of-Experts Pool Normalization

The unit-weight finite product-of-experts pool is the normalized pointwise product of two laws and has total mass one whenever its normalizer is positive. Assumption scope: The state carrier is finite and the pointwise-product normalizer must be strictly positive. Disjoint supports are excluded. This unit-weight product uses exponents one, not the one-half exponents of an equal-weight logarithmic pool, and no externally Bayesian pooling property is claimed. Semantic disposition: `formalized`.

13.110.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Unit-Weight Product-of-Experts Pool Normalization -/
namespace FEP110

open FEP Finset
open scoped BigOperators

variable {State : Type*} [Fintype State]

/-- A positive pointwise-product normalizer yields a normalized unit-weight
product-of-experts pool. -/
theorem fep110_unitWeightProductOfExpertsPool_normalized
    (left right : FiniteLaw State)
    (normalizerPositive :
      0 < FEP.CollectiveInference.productOfExpertsNormalizer left right) :
    Σ state,
        FEP.CollectiveInference.unitWeightProductOfExpertsPool
          left right normalizerPositive state = 1 :=
  FEP.CollectiveInference.unitWeightProductOfExpertsPool_sum_one
    left right normalizerPositive

/-- The pool mass is exactly the normalized pointwise product. It is a
unit-weight product of experts, not a half-exponent logarithmic pool. -/
theorem fep110_unitWeightProductOfExpertsPool_mass
    (left right : FiniteLaw State)
    (normalizerPositive :
      0 < FEP.CollectiveInference.productOfExpertsNormalizer left right)
    (state : State) :
    FEP.CollectiveInference.unitWeightProductOfExpertsPool
        left right normalizerPositive state =
      left state * right state /
        FEP.CollectiveInference.productOfExpertsNormalizer left right :=
  rfl

end FEP110
```

13.110.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{normalizerPositive} : 0$$
$$< \mathsf{FEP.CollectiveInference.productOfExpertsNormalizer}\ \mathsf{left}\ \mathsf{right})$$
$$\sum \mathsf{state},\ \mathsf{FEP.CollectiveInference.unitWeightProductOfExpertsPool}\ \mathsf{left}\ \mathsf{right}$$
$$\mathsf{normalizerPositive}\ \mathsf{state} = 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{normalizerPositive} : 0$$
$$< \mathsf{FEP.CollectiveInference.productOfExpertsNormalizer}\ \mathsf{left}\ \mathsf{right})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{FEP.CollectiveInference.unitWeightProductOfExpertsPool}\ \mathsf{left}\ \mathsf{right}$$
$$\mathsf{normalizerPositive}\ \mathsf{state} = \mathsf{left}\ \mathsf{state} \cdot \mathsf{right}\ \mathsf{state}$$
$$/\mathsf{FEP.CollectiveInference.productOfExpertsNormalizer}\ \mathsf{left}\ \mathsf{right}$$

13.111 fep-111 — Consensus Mass Conservation

The specified doubly stochastic two-agent consensus update conserves combined mass at every state and therefore preserves total combined mass two. Assumption scope: The state carrier is finite and both inputs are normalized FiniteLaw values. Conservation is for the maintained two-agent mixing operator; it does not establish consensus, stationarity, or conservation for arbitrary interaction matrices. Semantic disposition: `formalized`.

13.111.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Consensus Mass Conservation -/
namespace FEP111

open FEP Finset
open scoped BigOperators

variable {State : Type*} [Fintype State]

/-- The doubly stochastic two-agent update conserves combined mass at every
state. -/
theorem fep111_consensus_pointwise_mass_conserved
    (left right : FiniteLaw State) (state : State) :
    FEP.CollectiveInference.consensusLeft left right state +
        FEP.CollectiveInference.consensusRight left right state =
      left state + right state :=
  FEP.CollectiveInference.consensusMass_conserved left right state

/-- Consequently, the two normalized post-update laws have total combined
mass two. -/
theorem fep111_consensus_total_mass_two
    (left right : FiniteLaw State) :
    (Σ state, FEP.CollectiveInference.consensusLeft left right state) +
        Σ state, FEP.CollectiveInference.consensusRight left right state = 2 := by
  rw [(FEP.CollectiveInference.consensusLeft left right).sum_one,
    (FEP.CollectiveInference.consensusRight left right).sum_one]
  norm_num

end FEP111
```

13.111.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ State}]$$
$$(\mathsf{left\ right} : \mathsf{FiniteLaw\ State})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{FEP.CollectiveInference.consensusLeft\ left\ right\ state}$$
$$+\ \mathsf{FEP.CollectiveInference.consensusRight\ left\ right\ state} = \mathsf{left\ state} + \mathsf{right\ state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ State}]$$
$$(\mathsf{left\ right} : \mathsf{FiniteLaw\ State})$$
$$(\sum \mathsf{state}, \mathsf{FEP.CollectiveInference.consensusLeft\ left\ right\ state}) + \sum \mathsf{state},$$
$$\mathsf{FEP.CollectiveInference.consensusRight\ left\ right\ state} = 2$$

13.112 fep-112 — Contractive Belief-Consensus Convergence

Iterating the maintained strictly stochastic two-agent update halves each atomwise belief gap and drives pointwise disagreement to zero. Assumption scope: State is finite and consensus uses the fixed one-quarter/three-quarter mixing matrix encoded by consensusLeft and consensusRight. Convergence is pointwise for two agents; no arbitrary network topology, time-varying weights, or rate in a global probability metric is claimed. Semantic disposition: `formalized`.

13.112.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Contractive Belief-Consensus Convergence -/
namespace FEP112

open FEP Filter

variable {State : Type*} [Fintype State]

/-- The strictly stochastic consensus matrix contracts every atomwise gap by
exactly one half. -/
theorem fep112_consensus_half_contraction
    (left right : FiniteLaw State) (state : State) :
    FEP.CollectiveInference.beliefGap
        (FEP.CollectiveInference.consensusLeft left right)
        (FEP.CollectiveInference.consensusRight left right) state =
      (1 / 2 : ℝ) *
        FEP.CollectiveInference.beliefGap left right state :=
  FEP.CollectiveInference.consensus_gap_contracts left right state

/-- Iterated consensus converges pointwise to zero disagreement. -/
theorem fep112_consensus_converges
    (left right : FiniteLaw State) (state : State) :
    Tendsto
      (fun iterations =>
        FEP.CollectiveInference.beliefGap
          (FEP.CollectiveInference.consensusIterate
            (left, right) iterations).1
          (FEP.CollectiveInference.consensusIterate
            (left, right) iterations).2 state)
      atTop (nhds 0) :=
  FEP.CollectiveInference.consensus_gap_tendsto_zero left right state

/-- The contraction is not a zero-gap tautology: opposite Boolean point masses
retain positive disagreement after one step and contract strictly. -/
theorem fep112_bool_nonzero_strict_witness :
    FEP.CollectiveInference.beliefGap
        (FiniteLaw.pointMass true) (FiniteLaw.pointMass false) true = 1 ∧
```

```
      FEP.CollectiveInference.beliefGap
          (FEP.CollectiveInference.consensusLeft
            (FiniteLaw.pointMass true) (FiniteLaw.pointMass false))
          (FEP.CollectiveInference.consensusRight
            (FiniteLaw.pointMass true) (FiniteLaw.pointMass false)) true = 1 / 2 ∧
      0 < FEP.CollectiveInference.beliefGap
          (FEP.CollectiveInference.consensusLeft
            (FiniteLaw.pointMass true) (FiniteLaw.pointMass false))
          (FEP.CollectiveInference.consensusRight
            (FiniteLaw.pointMass true) (FiniteLaw.pointMass false)) true ∧
      FEP.CollectiveInference.beliefGap
          (FEP.CollectiveInference.consensusLeft
            (FiniteLaw.pointMass true) (FiniteLaw.pointMass false))
          (FEP.CollectiveInference.consensusRight
            (FiniteLaw.pointMass true) (FiniteLaw.pointMass false)) true <
        FEP.CollectiveInference.beliefGap
          (FiniteLaw.pointMass true) (FiniteLaw.pointMass false) true :=
  FEP.CollectiveInference.boolConsensus_nonzero_strict_witness

end FEP112
```

13.112.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{FEP.CollectiveInference.beliefGap}$$
$$(\mathsf{FEP.CollectiveInference.consensusLeft}\ \mathsf{left}\ \mathsf{right})$$
$$(\mathsf{FEP.CollectiveInference.consensusRight}\ \mathsf{left}\ \mathsf{right})\ \mathsf{state} = (1/2 : \mathbb{R})$$
$$\cdot\ \mathsf{FEP.CollectiveInference.beliefGap}\ \mathsf{left}\ \mathsf{right}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{Tendsto}\ (\lambda\, \mathsf{iterations} \mapsto \mathsf{FEP.CollectiveInference.beliefGap}$$
$$(\mathsf{FEP.CollectiveInference.consensusIterate}\ (\mathsf{left}, \mathsf{right})\ \mathsf{iterations}).1$$
$$(\mathsf{FEP.CollectiveInference.consensusIterate}\ (\mathsf{left}, \mathsf{right})\ \mathsf{iterations}).2\ \mathsf{state})$$
$$\mathsf{atTop}\ (\mathsf{nhds}\ 0)$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$\mathsf{FEP.CollectiveInference.beliefGap}\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{true})$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{false})\ \mathsf{true} = 1 \wedge \mathsf{FEP.CollectiveInference.beliefGap}$$
$$(\mathsf{FEP.CollectiveInference.consensusLeft}\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{true})$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{false}))\ (\mathsf{FEP.CollectiveInference.consensusRight}$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{true})\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{false}))\ \mathsf{true} = 1/2 \wedge 0$$
$$< \mathsf{FEP.CollectiveInference.beliefGap}\ (\mathsf{FEP.CollectiveInference.consensusLeft}$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{true})\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{false}))$$
$$(\mathsf{FEP.CollectiveInference.consensusRight}\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{true})$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{false}))\ \mathsf{true} \wedge \mathsf{FEP.CollectiveInference.beliefGap}$$
$$(\mathsf{FEP.CollectiveInference.consensusLeft}\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{true})$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{false}))\ (\mathsf{FEP.CollectiveInference.consensusRight}$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{true})\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{false}))\ \mathsf{true}$$
$$< \mathsf{FEP.CollectiveInference.beliefGap}\ (\mathsf{FiniteLaw.pointMass}\ \mathsf{true})$$
$$(\mathsf{FiniteLaw.pointMass}\ \mathsf{false})\ \mathsf{true}$$

13.113 fep-113 — Coupled-Agent Potential Descent

The maintained consensus coupling quarters atomwise quadratic disagreement potential and decreases it strictly whenever the two input masses differ at that atom. Assumption scope: State is finite and the potential is the exact scalar square of the two-agent atomwise gap under the fixed consensus operator. Strictness requires explicit separation; the theorem does not cover a general coupled free-energy functional or nonlinear multiagent dynamics. Semantic disposition: `formalized`.

13.113.1 Lean sketch

```
import FepSketches.collective_inference

/-! # Coupled-Agent Potential Descent -/
namespace FEP113

open FEP

variable {State : Type*} [Fintype State]

/-- The specified consensus coupling quarters the atomwise quadratic
disagreement potential. -/
theorem fep113_coupledPotential_contracts
    (left right : FiniteLaw State) (state : State) :
    FEP.CollectiveInference.coupledPotential
        (FEP.CollectiveInference.consensusLeft left right)
        (FEP.CollectiveInference.consensusRight left right) state =
      (1 / 4 : ℝ) *
        FEP.CollectiveInference.coupledPotential left right state :=
  FEP.CollectiveInference.coupledPotential_contracts left right state

/-- Nonzero disagreement makes the potential descent strict. -/
theorem fep113_coupledPotential_strict_descent
    (left right : FiniteLaw State) (state : State)
    (separated : left state ≠ right state) :
    FEP.CollectiveInference.coupledPotential
        (FEP.CollectiveInference.consensusLeft left right)
        (FEP.CollectiveInference.consensusRight left right) state <
      FEP.CollectiveInference.coupledPotential left right state :=
  FEP.CollectiveInference.coupledPotential_strict_descent
    left right state separated

end FEP113
```

13.113.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{FEP.CollectiveInference.coupledPotential}$$
$$(\mathsf{FEP.CollectiveInference.consensusLeft}\ \mathsf{left}\ \mathsf{right})$$
$$(\mathsf{FEP.CollectiveInference.consensusRight}\ \mathsf{left}\ \mathsf{right})\ \mathsf{state} = (1/4 : \mathbb{R})$$
$$\cdot\ \mathsf{FEP.CollectiveInference.coupledPotential}\ \mathsf{left}\ \mathsf{right}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{state} : \mathsf{State})$$
$$(\mathsf{separated} : \mathsf{left}\ \mathsf{state} \neq \mathsf{right}\ \mathsf{state})$$
$$\mathsf{FEP.CollectiveInference.coupledPotential}$$
$$(\mathsf{FEP.CollectiveInference.consensusLeft}\ \mathsf{left}\ \mathsf{right})$$
$$(\mathsf{FEP.CollectiveInference.consensusRight}\ \mathsf{left}\ \mathsf{right})\ \mathsf{state}$$
$$<\ \mathsf{FEP.CollectiveInference.coupledPotential}\ \mathsf{left}\ \mathsf{right}\ \mathsf{state}$$

13.114 fep-114 — Sub-Gaussian Empirical-Mean Tail Bound

A finite independent family with certified sub-Gaussian moment-generating functions satisfies the explicit exponential upper-tail bound for its summed empirical-mean event. Assumption scope: Observables are indexed by Fin sampleCount, are independent under the supplied measure, and each has a nonnegative sub-Gaussian proxy variance. Deviation is nonnegative. The theorem neither assumes IID data nor asserts an asymptotic law; a positive sample count is needed for the usual empirical-mean interpretation even though real division is totalized at zero. Semantic disposition: `formalized`.

13.114.1 Lean sketch

```
import FepSketches.learning_theory

/-! # Sub-Gaussian Empirical-Mean Tail Bound -/
namespace FEP114

open MeasureTheory ProbabilityTheory
open scoped BigOperators ENNReal MeasureTheory NNReal ProbabilityTheory

/-- Independent sub-Gaussian observations obey the finite-sample upper-tail
bound for the empirical-mean event. -/
theorem fep114_subGaussian_empiricalMean_tail
    {Ω : Type*} [MeasurableSpace Ω]
    (μ : Measure Ω) {sampleCount : ℕ}
    (observables : Fin sampleCount → Ω → ℝ)
    (independent : iIndepFun observables μ)
    (proxyVariance : Fin sampleCount → ℝ≥0)
    (subGaussian : ∀ index,
      HasSubgaussianMGF (observables index) (proxyVariance index) μ)
    {deviation : ℝ} (deviationNonnegative : 0 ≤ deviation) :
    μ.real {outcome |
        (sampleCount : ℝ) * deviation ≤
          ∑ index, observables index outcome} ≤
      Real.exp
        (-((sampleCount : ℝ) * deviation) ^ 2 /
          (2 * ∑ index, proxyVariance index)) :=
  FEP.LearningTheory.subGaussian_empiricalMean_tail
    μ observables independent proxyVariance subGaussian deviationNonnegative

/-- The theorem fixes the finite sample through `Fin sampleCount`; no
asymptotic limit or IID premise is inferred beyond the stated independence. -/
theorem fep114_empiricalMean_definition
    {Ω : Type*} {sampleCount : ℕ}
    (observables : Fin sampleCount → Ω → ℝ) (outcome : Ω) :
    FEP.LearningTheory.empiricalMean observables outcome =
      (∑ index, observables index outcome) / sampleCount :=
  rfl

end FEP114
```

13.114.2 Typeset statement signatures

$$\{\Omega : \mathsf{Type}^*\}\,[\mathsf{MeasurableSpace}\Omega]\,(\mu : \mathsf{Measure}\Omega)\,\{\mathsf{sampleCount} : \mathbb{N}\}$$
$$(\mathsf{observables} : \mathsf{Fin}\,\mathsf{sampleCount} \Rightarrow \Omega \Rightarrow \mathbb{R})\,(\mathsf{independent} : \mathsf{iIndepFun}\,\mathsf{observables}\,\mu)$$
$$(\mathsf{proxyVariance} : \mathsf{Fin}\,\mathsf{sampleCount} \Rightarrow \mathbb{R} \geq 0)\,(\mathsf{subGaussian} : \forall \mathsf{index},$$
$$\mathsf{HasSubgaussianMGF}\,(\mathsf{observables}\,\mathsf{index})\,(\mathsf{proxyVariance}\,\mathsf{index})\,\mu)\,\{\mathsf{deviation} : \mathbb{R}\}$$
$$(\mathsf{deviationNonnegative} : 0 \leq \mathsf{deviation})$$
$$\mu.\mathsf{real}\,\{\mathsf{outcome} | (\mathsf{sampleCount} : \mathbb{R}) \cdot \mathsf{deviation} \leq \sum \mathsf{index}, \mathsf{observables}\,\mathsf{index}$$
$$\mathsf{outcome}\} \leq \mathsf{Real.exp}\,(-((\mathsf{sampleCount} : \mathbb{R}) \cdot \mathsf{deviation})^2/(2 \cdot \sum \mathsf{index},$$
$$\mathsf{proxyVariance}\,\mathsf{index}))$$

$$\{\Omega : \mathsf{Type}^*\}\,\{\mathsf{sampleCount} : \mathbb{N}\}\,(\mathsf{observables} : \mathsf{Fin}\,\mathsf{sampleCount} \Rightarrow \Omega \Rightarrow \mathbb{R})$$
$$(\mathsf{outcome} : \Omega)$$
$$\mathsf{FEP.LearningTheory.empiricalMean}\,\mathsf{observables}\,\mathsf{outcome}$$
$$= (\sum \mathsf{index}, \mathsf{observables}\,\mathsf{index}\,\mathsf{outcome})/\mathsf{sampleCount}$$

13.115 fep-115 — Simultaneous Finite-Alphabet Frequency Bound

Per-symbol empirical-frequency failure bounds combine by a finite union bound into an alphabet-wide failure probability bounded by alphabet size times the per-symbol rate. Assumption scope: The sampling measure is a probability measure, the alphabet is finite with decidable equality, sample count is nonzero, and every symbol has the supplied probability bound. No independence or concentration rate is inferred beyond those per-symbol premises. Semantic disposition: `formalized`.

13.115.1 Lean sketch

```
import FepSketches.learning_theory

/-! # Simultaneous Finite-Alphabet Frequency Bound -/
namespace FEP115

open MeasureTheory ProbabilityTheory
open scoped ENNReal MeasureTheory ProbabilityTheory

/-- A finite union bound lifts per-symbol empirical-frequency guarantees to a
simultaneous alphabet-wide guarantee. -/
theorem fep115_simultaneous_frequency_bound
    {Ω Alphabet : Type*} [MeasurableSpace Ω]
    [Fintype Alphabet] [DecidableEq Alphabet]
    (μ : Measure Ω) [IsProbabilityMeasure μ]
    {sampleCount : ℕ} [NeZero sampleCount]
    (sample : Ω → Fin sampleCount → Alphabet)
    (target : Alphabet → ℝ) (deviation perSymbolFailure : ℝ)
    (perSymbolBound : ∀ symbol,
      μ.real
          (FEP.LearningTheory.frequencyDeviationEvent
            sample target deviation symbol) ≤ perSymbolFailure) :
    μ.real (⋃ symbol,
        FEP.LearningTheory.frequencyDeviationEvent
          sample target deviation symbol) ≤
      Fintype.card Alphabet * perSymbolFailure :=
  FEP.LearningTheory.simultaneous_frequency_bound
    μ sample target deviation perSymbolFailure perSymbolBound

/-- The event is the actual two-sided deviation of a normalized finite count,
not an abstract placeholder event. -/
theorem fep115_frequencyDeviationEvent_membership
    {Ω Alphabet : Type*} [Fintype Alphabet] [DecidableEq Alphabet]
    {sampleCount : ℕ} [NeZero sampleCount]
    (sample : Ω → Fin sampleCount → Alphabet)
    (target : Alphabet → ℝ) (deviation : ℝ)
    (symbol : Alphabet) (outcome : Ω) :
    outcome ∈ FEP.LearningTheory.frequencyDeviationEvent
        sample target deviation symbol ↔
      deviation ≤
        |FEP.LearningTheory.empiricalFrequency
            (sample outcome) symbol - target symbol| :=
  Iff.rfl

end FEP115
```

13.115.2 Typeset statement signatures

$$\{\Omega\ \mathsf{Alphabet} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{Fintype\ Alphabet}]$$
$$[\mathsf{DecidableEq\ Alphabet}]\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{IsProbabilityMeasure}\ \mu]$$
$$\{\mathsf{sampleCount} : \mathbb{N}\}\ [\mathsf{NeZero\ sampleCount}]\ (\mathsf{sample} : \Omega \Rightarrow \mathsf{Fin\ sampleCount} \Rightarrow \mathsf{Alphabet})$$
$$(\mathsf{target} : \mathsf{Alphabet} \Rightarrow \mathbb{R})\ (\mathsf{deviation\ perSymbolFailure} : \mathbb{R})\ (\mathsf{perSymbolBound} : \forall$$
$$\mathsf{symbol},\ \mu.\mathsf{real}\ (\mathsf{FEP.LearningTheory.frequencyDeviationEvent\ sample\ target}$$
$$\mathsf{deviation\ symbol}) \leq \mathsf{perSymbolFailure})$$
$$\mu.\mathsf{real}\ (\bigcup \mathsf{symbol}, \mathsf{FEP.LearningTheory.frequencyDeviationEvent\ sample\ target}$$
$$\mathsf{deviation\ symbol}) \leq \mathsf{Fintype.card\ Alphabet} \cdot \mathsf{perSymbolFailure}$$

$$\{\Omega\ \mathsf{Alphabet} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Alphabet}]\ [\mathsf{DecidableEq\ Alphabet}]\ \{\mathsf{sampleCount} : \mathbb{N}\}$$
$$[\mathsf{NeZero\ sampleCount}]\ (\mathsf{sample} : \Omega \Rightarrow \mathsf{Fin\ sampleCount} \Rightarrow \mathsf{Alphabet})$$
$$(\mathsf{target} : \mathsf{Alphabet} \Rightarrow \mathbb{R})\ (\mathsf{deviation} : \mathbb{R})\ (\mathsf{symbol} : \mathsf{Alphabet})\ (\mathsf{outcome} : \Omega)$$
$$\mathsf{outcome} \in \mathsf{FEP.LearningTheory.frequencyDeviationEvent\ sample\ target\ deviation}$$
$$\mathsf{symbol} \leftrightarrow \mathsf{deviation} \leq |\mathsf{FEP.LearningTheory.empiricalFrequency}\ (\mathsf{sample\ outcome})$$
$$\mathsf{symbol} - \mathsf{target\ symbol}|$$

13.116 fep-116 — Finite-Hypothesis PAC-Bayes Loss-Gap Bound

For any finite posterior, a Gibbs certificate whose potential is inverse temperature times the population-minus-empirical loss gap converts a certified log-MGF budget into an empirical-loss plus KL-complexity population-loss bound. Assumption scope: Hypotheses are finite; inverse temperature is positive; the Gibbs reference equals the prior and carries full support; confidence lies in (0,1); and the certificate log partition is bounded by log(confidence^-1). The theorem is deterministic conditional on this log-MGF premise and does not itself prove that the premise holds with high probability over data. Semantic disposition: `formalized`.

13.116.1 Lean sketch

```
import FepSketches.learning_theory

/-! # Finite-Hypothesis PAC-Bayes Loss-Gap Bound -/
namespace FEP116

open FEP MeasureTheory ProbabilityTheory
open scoped ENNReal MeasureTheory ProbabilityTheory

/-- A finite posterior's population loss is controlled by its empirical loss,
finite KL complexity, and a certified log-MGF budget. The Gibbs potential is
explicitly the inverse-temperature-scaled loss gap. -/
theorem fep116_finitePACBayes_with_confidence
    {Hypothesis : Type*} [Fintype Hypothesis]
    (prior posterior : FiniteLaw Hypothesis)
    (certificate : FEP.VariationalDuality.GibbsCertificate Hypothesis)
    (referenceIsPrior : certificate.reference = prior)
    (empiricalLoss populationLoss : Hypothesis → ℝ)
    (inverseTemperature : ℝ)
    (inverseTemperaturePositive : 0 < inverseTemperature)
    (potentialIsLossGap : ∀ hypothesis,
      certificate.potential hypothesis = inverseTemperature *
        (populationLoss hypothesis - empiricalLoss hypothesis))
    (confidence : ℝ)
    (confidencePositive : 0 < confidence)
    (confidenceBelowOne : confidence < 1)
    (logMGFBound : certificate.logPartition ≤ Real.log confidence⁻¹) :
    FEP.VariationalDuality.expectation posterior populationLoss ≤
      FEP.VariationalDuality.expectation posterior empiricalLoss +
        (FEP.FiniteInformation.finiteKL posterior prior +
          Real.log confidence⁻¹) / inverseTemperature :=
  (FEP.LearningTheory.finitePACBayes_changeOfMeasure_with_confidence
```

```
    prior posterior certificate referenceIsPrior empiricalLoss populationLoss
    inverseTemperature inverseTemperaturePositive potentialIsLossGap confidence
    confidencePositive confidenceBelowOne logMGFBound).1

/-- A full-support finite prior has no zero-mass atom, matching the support
boundary required by finite KL and Gibbs log-density terms. -/
theorem fep116_prior_support_nonzero
    {Hypothesis : Type*} [Fintype Hypothesis]
    (prior : FiniteLaw Hypothesis)
    (priorPositive : ∀ hypothesis, 0 < prior hypothesis)
    (hypothesis : Hypothesis) :
    prior hypothesis ≠ 0 :=
  ne_of_gt (priorPositive hypothesis)

end FEP116
```

13.116.2 Typeset statement signatures

$$\{\mathsf{Hypothesis} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Hypothesis}]$$
$$(\mathsf{prior\ posterior} : \mathsf{FiniteLaw\ Hypothesis})\ (\mathsf{certificate} :$$
$$\mathsf{FEP.VariationalDuality.GibbsCertificate\ Hypothesis})$$
$$(\mathsf{referenceIsPrior} : \mathsf{certificate.reference} = \mathsf{prior})$$
$$(\mathsf{empiricalLoss\ populationLoss} : \mathsf{Hypothesis} \Rightarrow \mathbb{R})\ (\mathsf{inverseTemperature} : \mathbb{R})$$
$$(\mathsf{inverseTemperaturePositive} : 0 < \mathsf{inverseTemperature})\ (\mathsf{potentialIsLossGap} : \forall$$
$$\mathsf{hypothesis}, \mathsf{certificate.potential\ hypothesis} = \mathsf{inverseTemperature}$$
$$\cdot\ (\mathsf{populationLoss\ hypothesis} - \mathsf{empiricalLoss\ hypothesis}))\ (\mathsf{confidence} : \mathbb{R})$$
$$(\mathsf{confidencePositive} : 0 < \mathsf{confidence})\ (\mathsf{confidenceBelowOne} : \mathsf{confidence} < 1)$$
$$(\mathsf{logMGFBound} : \mathsf{certificate.logPartition} \le \mathsf{Real.log\ confidence}^{-1})$$
$$\mathsf{FEP.VariationalDuality.expectation\ posterior\ populationLoss}$$
$$\le \mathsf{FEP.VariationalDuality.expectation\ posterior\ empiricalLoss}$$
$$+\ (\mathsf{FEP.FiniteInformation.finiteKL\ posterior\ prior} + \mathsf{Real.log\ confidence}^{-1})$$
$$/\mathsf{inverseTemperature}$$

$$\{\mathsf{Hypothesis} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Hypothesis}]\ (\mathsf{prior} : \mathsf{FiniteLaw\ Hypothesis})$$
$$(\mathsf{priorPositive} : \forall \mathsf{hypothesis}, 0 < \mathsf{prior\ hypothesis})\ (\mathsf{hypothesis} : \mathsf{Hypothesis})$$
$$\mathsf{prior\ hypothesis} \neq 0$$

13.117 fep-117 — Posterior-Odds Multiplicative Recursion

At positive evidence, posterior odds equal prior odds times the likelihood ratio, and a zero-prior hypothesis retains zero posterior mass. Assumption scope: Hypothesis and evidence carriers are finite. Total evidence, the reference prior mass, and the reference likelihood must be strictly positive before forming ordinary real odds. The zero-prior theorem marks lack of posterior support rather than permitting recovery by conditioning. Semantic disposition: `formalized`.

13.117.1 Lean sketch

```
import FepSketches.learning_theory

/-! # Posterior-Odds Multiplicative Recursion -/
namespace FEP117

open FEP

/-- At positive evidence, posterior odds equal prior odds times the likelihood
ratio, provided the reference hypothesis has positive prior and likelihood. -/
theorem fep117_posteriorOdds_recursion
    {Hypothesis Evidence : Type*} [Fintype Hypothesis] [Fintype Evidence]
```

```
    (prior : FiniteLaw Hypothesis)
    (likelihood : FiniteKernel Hypothesis Evidence)
    (evidence : Evidence)
    (evidencePositive : 0 < likelihood.predictive prior evidence)
    (favored reference : Hypothesis)
    (referencePriorPositive : 0 < prior reference)
    (referenceLikelihoodPositive : 0 < likelihood reference evidence) :
    FEP.LearningTheory.posteriorOdds
        prior likelihood evidence evidencePositive favored reference =
      (prior favored / prior reference) *
        (likelihood favored evidence / likelihood reference evidence) :=
  FEP.LearningTheory.posteriorOdds_recursion
    prior likelihood evidence evidencePositive favored reference
    referencePriorPositive referenceLikelihoodPositive

/-- A zero-prior hypothesis cannot be recovered by conditioning on positive
evidence. -/
theorem fep117_zero_prior_boundary
    {Hypothesis Evidence : Type*} [Fintype Hypothesis] [Fintype Evidence]
    (prior : FiniteLaw Hypothesis)
    (likelihood : FiniteKernel Hypothesis Evidence)
    (evidence : Evidence)
    (evidencePositive : 0 < likelihood.predictive prior evidence)
    (hypothesis : Hypothesis) (priorZero : prior hypothesis = 0) :
    likelihood.posterior prior evidence evidencePositive hypothesis = 0 :=
  FEP.LearningTheory.posterior_zero_of_prior_zero
    prior likelihood evidence evidencePositive hypothesis priorZero

end FEP117
```

13.117.2 Typeset statement signatures

$$\{\text{Hypothesis Evidence} : \text{Type}^*\}\ [\text{Fintype Hypothesis}]\ [\text{Fintype Evidence}]\ (\text{prior} : \text{FiniteLaw Hypothesis})\ (\text{likelihood} : \text{FiniteKernel Hypothesis Evidence})\ (\text{evidence} : \text{Evidence})\ (\text{evidencePositive} : 0 < \text{likelihood.predictive prior evidence})\ (\text{favored reference} : \text{Hypothesis})\ (\text{referencePriorPositive} : 0 < \text{prior reference})\ (\text{referenceLikelihoodPositive} : 0 < \text{likelihood reference evidence})\ \text{FEP.LearningTheory.posteriorOdds prior likelihood evidence evidencePositive favored reference} = (\text{prior favored}/\text{prior reference}) \cdot (\text{likelihood favored evidence}/\text{likelihood reference evidence})$$

$$\{\text{Hypothesis Evidence} : \text{Type}^*\}\ [\text{Fintype Hypothesis}]\ [\text{Fintype Evidence}]\ (\text{prior} : \text{FiniteLaw Hypothesis})\ (\text{likelihood} : \text{FiniteKernel Hypothesis Evidence})\ (\text{evidence} : \text{Evidence})\ (\text{evidencePositive} : 0 < \text{likelihood.predictive prior evidence})\ (\text{hypothesis} : \text{Hypothesis})\ (\text{priorZero} : \text{prior hypothesis} = 0)\ \text{likelihood.posterior prior evidence evidencePositive hypothesis} = 0$$

13.118 fep-118 — Exponential Posterior Concentration from a Likelihood Gap

A nonnegative per-observation likelihood gap yields an explicit exponential finite-sample upper bound on inferior two-hypothesis posterior weight. Assumption scope: Good prior and likelihood weights are strictly positive, bad weights and the gap are nonnegative, and bad likelihood is bounded by exp(-gap) times good likelihood. These are scalar two-hypothesis weights, not a theorem of consistency for an arbitrary normalized model class. Semantic disposition: `formalized`.

13.118.1 Lean sketch

```
import FepSketches.learning_theory
```

```
/-! # Exponential Posterior Concentration from a Likelihood Gap -/
namespace FEP118

/-- A nonnegative per-observation likelihood gap gives an explicit exponential
finite-sample bound on inferior posterior mass. -/
theorem fep118_posteriorGap_concentration
    (priorGood priorBad likelihoodGood likelihoodBad likelihoodGap : ℝ)
    (sampleCount : ℕ)
    (priorGoodPositive : 0 < priorGood)
    (priorBadNonnegative : 0 ≤ priorBad)
    (likelihoodGoodPositive : 0 < likelihoodGood)
    (likelihoodBadNonnegative : 0 ≤ likelihoodBad)
    (likelihoodGapNonnegative : 0 ≤ likelihoodGap)
    (gapBound :
      likelihoodBad ≤ Real.exp (-likelihoodGap) * likelihoodGood) :
    FEP.LearningTheory.twoHypothesisPosteriorBad
        priorGood priorBad likelihoodGood likelihoodBad sampleCount ≤
      (priorBad / priorGood) *
        Real.exp (-((sampleCount : ℝ) * likelihoodGap)) :=
  FEP.LearningTheory.posteriorGap_concentration
    priorGood priorBad likelihoodGood likelihoodBad likelihoodGap sampleCount
    priorGoodPositive priorBadNonnegative likelihoodGoodPositive
    likelihoodBadNonnegative likelihoodGapNonnegative gapBound

/-- Equal priors and a `3:1` likelihood ratio give exact nonzero posterior bad
mass `1/10` after two observations. -/
theorem fep118_twoHypothesis_witness :
    FEP.LearningTheory.twoHypothesisPosteriorBad
      (1 / 2) (1 / 2) (3 / 4) (1 / 4) 2 = 1 / 10 :=
  FEP.LearningTheory.twoHypothesis_posterior_witness

end FEP118
```

13.118.2 Typeset statement signatures

$$(\mathsf{priorGood}\ \mathsf{priorBad}\ \mathsf{likelihoodGood}\ \mathsf{likelihoodBad}\ \mathsf{likelihoodGap} : \mathbb{R})$$
$$(\mathsf{sampleCount} : \mathbb{N})\ (\mathsf{priorGoodPositive} : 0 < \mathsf{priorGood})$$
$$(\mathsf{priorBadNonnegative} : 0 \le \mathsf{priorBad})$$
$$(\mathsf{likelihoodGoodPositive} : 0 < \mathsf{likelihoodGood})$$
$$(\mathsf{likelihoodBadNonnegative} : 0 \le \mathsf{likelihoodBad})$$
$$(\mathsf{likelihoodGapNonnegative} : 0 \le \mathsf{likelihoodGap})\ (\mathsf{gapBound} : \mathsf{likelihoodBad}$$
$$\le \mathsf{Real.exp}\ (-\mathsf{likelihoodGap}) \cdot \mathsf{likelihoodGood})$$
$$\mathsf{FEP.LearningTheory.twoHypothesisPosteriorBad}\ \mathsf{priorGood}\ \mathsf{priorBad}\ \mathsf{likelihoodGood}$$
$$\mathsf{likelihoodBad}\ \mathsf{sampleCount} \le (\mathsf{priorBad}/\mathsf{priorGood}) \cdot \mathsf{Real.exp}$$
$$(-((\mathsf{sampleCount} : \mathbb{R}) \cdot \mathsf{likelihoodGap}))$$

$$\mathsf{FEP.LearningTheory.twoHypothesisPosteriorBad}\ (1/2)\ (1/2)\ (3/4)\ (1/4)\ 2$$
$$= 1/10$$

13.119 fep-119 — Bayesian-Mixture Log-Loss Regret Bound

Finite Bayesian-mixture evidence dominates every nonnegative component contribution, so mixture log loss is at most selected-component log loss plus its exact negative-log-prior penalty. Assumption scope: Hypotheses are finite, all component likelihood weights are nonnegative, and the selected prior and likelihood weights are strictly positive before logarithms are compared. No sequential regret sum, stochastic calibration, or asymptotic optimality is claimed. Semantic disposition: `formalized`.

13.119.1 Lean sketch

```
import FepSketches.learning_theory
```

```
/-! # Bayesian-Mixture Log-Loss Regret Bound -/
namespace FEP119

open FEP

variable {Hypothesis : Type*} [Fintype Hypothesis]

/-- Mixture evidence dominates every nonnegative component contribution. -/
theorem fep119_mixtureEvidence_lower_bound
    (prior : FiniteLaw Hypothesis) (likelihood : Hypothesis → ℝ)
    (likelihoodNonnegative : ∀ hypothesis, 0 ≤ likelihood hypothesis)
    (selected : Hypothesis) :
    prior selected * likelihood selected ≤
      FEP.LearningTheory.mixtureEvidence prior likelihood :=
  FEP.LearningTheory.mixtureEvidence_lower_bound
    prior likelihood likelihoodNonnegative selected

/-- The selected model pays at most its own log loss plus the exact finite
prior penalty. -/
theorem fep119_mixtureLogLoss_regret
    (prior : FiniteLaw Hypothesis) (likelihood : Hypothesis → ℝ)
    (likelihoodNonnegative : ∀ hypothesis, 0 ≤ likelihood hypothesis)
    (selected : Hypothesis)
    (priorPositive : 0 < prior selected)
    (likelihoodPositive : 0 < likelihood selected) :
    -Real.log (FEP.LearningTheory.mixtureEvidence prior likelihood) ≤
      -Real.log (likelihood selected) - Real.log (prior selected) :=
  FEP.LearningTheory.mixtureLogLoss_regret
    prior likelihood likelihoodNonnegative selected
    priorPositive likelihoodPositive

end FEP119
```

13.119.2 Typeset statement signatures

$$\{\mathsf{Hypothesis} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Hypothesis}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw\ Hypothesis})\ (\mathsf{likelihood} : \mathsf{Hypothesis} \Rightarrow \mathbb{R})$$
$$(\mathsf{likelihoodNonnegative} : \forall \mathsf{hypothesis}, 0 \le \mathsf{likelihood\ hypothesis})$$
$$(\mathsf{selected} : \mathsf{Hypothesis})$$
$$\mathsf{prior\ selected} \cdot \mathsf{likelihood\ selected} \le \mathsf{FEP.LearningTheory.mixtureEvidence\ prior}$$
$$\mathsf{likelihood}$$

$$\{\mathsf{Hypothesis} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Hypothesis}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw\ Hypothesis})\ (\mathsf{likelihood} : \mathsf{Hypothesis} \Rightarrow \mathbb{R})$$
$$(\mathsf{likelihoodNonnegative} : \forall \mathsf{hypothesis}, 0 \le \mathsf{likelihood\ hypothesis})$$
$$(\mathsf{selected} : \mathsf{Hypothesis})\ (\mathsf{priorPositive} : 0 < \mathsf{prior\ selected})$$
$$(\mathsf{likelihoodPositive} : 0 < \mathsf{likelihood\ selected})$$
$$-\mathsf{Real.log}\ (\mathsf{FEP.LearningTheory.mixtureEvidence\ prior\ likelihood}) \le -\mathsf{Real.log}$$
$$(\mathsf{likelihood\ selected}) - \mathsf{Real.log}\ (\mathsf{prior\ selected})$$

13.120 fep-120 — Bayes-Factor Multiplicativity and Model-Evidence Update

Explicitly factorized evidence ratios multiply, and sequential model-odds updates equal one update using product evidence. Assumption scope: Both reference evidence factors must be nonzero for multiplicativity in the real field. Evidence factorization is supplied algebraically and does not establish probabilistic independence. At a zero denominator, totalized real division returns zero rather than an infinite Bayes factor. Semantic disposition: `formalized`.

13.120.1 Lean sketch

```
import FepSketches.learning_theory

/-! # Bayes-Factor Multiplicativity and Model-Evidence Update -/
namespace FEP120

/-- Explicitly factorized evidence terms multiply, and sequential
posterior-odds updates agree with one update using product evidence. -/
theorem fep120_bayesFactor_multiplicative_update
    (priorOdds firstFavored firstReference secondFavored secondReference : ℝ)
    (firstReferenceNonzero : firstReference ≠ 0)
    (secondReferenceNonzero : secondReference ≠ 0) :
    FEP.LearningTheory.bayesFactor
        (firstFavored * secondFavored) (firstReference * secondReference) =
        FEP.LearningTheory.bayesFactor firstFavored firstReference *
          FEP.LearningTheory.bayesFactor secondFavored secondReference ∧
      FEP.LearningTheory.updatedModelOdds priorOdds
          (firstFavored * secondFavored) (firstReference * secondReference) =
        FEP.LearningTheory.updatedModelOdds
          (FEP.LearningTheory.updatedModelOdds
            priorOdds firstFavored firstReference)
          secondFavored secondReference :=
  FEP.LearningTheory.bayesFactor_multiplicative
    priorOdds firstFavored firstReference secondFavored secondReference
    firstReferenceNonzero secondReferenceNonzero

/-- A concrete two-model evidence ratio updates unit prior odds to three. -/
theorem fep120_twoHypothesis_evidence_witness :
    FEP.LearningTheory.bayesFactor (3 / 4) (1 / 4) = 3 ∧
      FEP.LearningTheory.updatedModelOdds 1 (3 / 4) (1 / 4) = 3 :=
  FEP.LearningTheory.bayesFactor_twoHypothesis_witness

/-- At zero reference evidence, real division returns zero rather than an
extended infinite Bayes factor; supported comparisons require nonzero
reference evidence. -/
theorem fep120_zero_evidence_boundary (favored : ℝ) :
    FEP.LearningTheory.bayesFactor favored 0 = 0 :=
  FEP.LearningTheory.bayesFactor_zero_denominator_boundary favored

end FEP120
```

13.120.2 Typeset statement signatures

$$(\text{priorOdds firstFavored firstReference secondFavored secondReference} : \mathbb{R})$$
$$(\text{firstReferenceNonzero} : \text{firstReference} \neq 0)$$
$$(\text{secondReferenceNonzero} : \text{secondReference} \neq 0)$$
$$\text{FEP.LearningTheory.bayesFactor}\,(\text{firstFavored} \cdot \text{secondFavored})$$
$$(\text{firstReference} \cdot \text{secondReference}) = \text{FEP.LearningTheory.bayesFactor firstFavored}$$
$$\text{firstReference} \cdot \text{FEP.LearningTheory.bayesFactor secondFavored secondReference}$$
$$\wedge\ \text{FEP.LearningTheory.updatedModelOdds priorOdds}\,(\text{firstFavored} \cdot \text{secondFavored})$$
$$(\text{firstReference} \cdot \text{secondReference}) = \text{FEP.LearningTheory.updatedModelOdds}$$
$$(\text{FEP.LearningTheory.updatedModelOdds priorOdds firstFavored firstReference})$$
$$\text{secondFavored secondReference}$$

$$\text{FEP.LearningTheory.bayesFactor}\,(3/4)\,(1/4) = 3$$
$$\wedge\ \text{FEP.LearningTheory.updatedModelOdds}\,1\,(3/4)\,(1/4) = 3$$

$$(\text{favored} : \mathbb{R})$$
$$\text{FEP.LearningTheory.bayesFactor favored}\,0 = 0$$

13.121 fep-121 — Laplace-Smoothing Error Identity

For a positive finite sample, add-one smoothing decomposes forecast error into contracted empirical error plus an exact target-dependent offset. Assumption scope: The success count and sample count are natural numbers, sample count is positive, successes do not exceed it, and the target lies in the unit interval. The theorem is an algebraic identity, not a stochastic consistency or posterior-contraction result. Semantic disposition: `formalized`.

13.121.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Laplace-Smoothing Error Identity -/
namespace FEP121

open FEP.EmpiricalRisk

/-- For a positive finite sample, add-one error is contracted empirical error
plus the exact target-dependent offset. -/
theorem fep121_laplaceError_identity
    (successes sampleCount : ℕ) (target : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : successes ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1) :
    laplaceEstimate successes sampleCount - target =
      shrinkage sampleCount * (empiricalRate successes sampleCount - target) +
        laplaceBias sampleCount target := by
  have _ := successesAtMost
  have _ := targetBounds
  exact laplaceError_identity successes sampleCount target sampleCountPositive

/-- The contraction coefficient remains in the closed unit interval. -/
theorem fep121_shrinkage_mem_unitInterval (sampleCount : ℕ) :
    shrinkage sampleCount ∈ Set.Icc (0 : ℝ) 1 :=
  ⟨shrinkage_nonneg sampleCount, shrinkage_le_one sampleCount⟩

end FEP121
```

13.121.2 Typeset statement signatures

$$(\mathsf{successes}\ \mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target} : \mathbb{R})\ (\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})$$
$$(\mathsf{successesAtMost} : \mathsf{successes} \le \mathsf{sampleCount})$$
$$(\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\mathsf{laplaceEstimate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target} = \mathsf{shrinkage}\ \mathsf{sampleCount}$$
$$\cdot\ (\mathsf{empiricalRate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target}) + \mathsf{laplaceBias}\ \mathsf{sampleCount}$$
$$\mathsf{target}$$

$$(\mathsf{sampleCount} : \mathbb{N})$$
$$\mathsf{shrinkage}\ \mathsf{sampleCount} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1$$

13.122 fep-122 — Laplace-Smoothing Bias Bound

The absolute target-dependent Laplace offset is at most one pseudo-count divided by the smoothed sample size. Assumption scope: The target is confined to the closed unit interval. The bound controls the deterministic smoothing offset and does not identify statistical bias under an unspecified sampling distribution. Semantic disposition: `formalized`.

13.122.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Laplace-Smoothing Bias Bound -/
namespace FEP122
```

```
open FEP.EmpiricalRisk

/-- The target-dependent Laplace offset is bounded by one pseudo-count over
the smoothed sample size. -/
theorem fep122_laplaceBias_abs_le (sampleCount : ℕ) (target : ℝ)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1) :
    |laplaceBias sampleCount target| ≤
      1 / ((sampleCount : ℝ) + 2) :=
  laplaceBias_abs_le sampleCount target targetBounds

/-- The bias is genuinely nonzero at a boundary target. -/
theorem fep122_nonzero_boundary_witness :
    laplaceBias 2 0 = 1 / 4 ∧ laplaceBias 2 0 ≠ 0 :=
  laplaceBias_nonzero_witness

end FEP122
```

13.122.2 Typeset statement signatures

$$(\mathsf{sampleCount} : \mathbb{N})\,(\mathsf{target} : \mathbb{R})\,(\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$$
$$|\mathsf{laplaceBias}\,\mathsf{sampleCount}\,\mathsf{target}| \le 1/((\mathsf{sampleCount} : \mathbb{R}) + 2)$$

$$\mathsf{laplaceBias}\,2\,0 = 1/4 \wedge \mathsf{laplaceBias}\,2\,0 \neq 0$$

13.123 fep-123 — Absolute-Error Transfer through Laplace Smoothing

An absolute empirical-error certificate transfers through add-one smoothing with the exact shrinkage coefficient and one-pseudo-count offset. Assumption scope: Sample count is positive, successes are bounded by it, the target is in the unit interval, and the raw empirical error already has a supplied bound. No probability tail rate is inferred in this row. Semantic disposition: `formalized`.

13.123.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Absolute-Error Transfer -/
namespace FEP123

open FEP.EmpiricalRisk

/-- An empirical absolute-error certificate transfers through add-one
smoothing with exact shrinkage and offset terms. -/
theorem fep123_laplaceAbsoluteError_le
    (successes sampleCount : ℕ) (target error : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : successes ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1)
    (empiricalErrorBound :
      |empiricalRate successes sampleCount - target| ≤ error) :
    |laplaceEstimate successes sampleCount - target| ≤
      shrinkage sampleCount * error + 1 / ((sampleCount : ℝ) + 2) := by
  have _ := successesAtMost
  exact laplaceAbsoluteError_le successes sampleCount target error
    sampleCountPositive targetBounds empiricalErrorBound

/-- The transferred threshold cannot hide a negative empirical tolerance. -/
theorem fep123_error_nonnegative
    (successes sampleCount : ℕ) (target error : ℝ)
    (empiricalErrorBound :
      |empiricalRate successes sampleCount - target| ≤ error) :
    0 ≤ error :=
```

```
  (abs_nonneg _).trans empiricalErrorBound

end FEP123
```

13.123.2 Typeset statement signatures

$$(\mathsf{successes}\ \mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target}\ \mathsf{error} : \mathbb{R})$$
$$(\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})$$
$$(\mathsf{successesAtMost} : \mathsf{successes} \le \mathsf{sampleCount})$$
$$(\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)\ (\mathsf{empiricalErrorBound} :$$
$$|\mathsf{empiricalRate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target}| \le \mathsf{error})$$
$$|\mathsf{laplaceEstimate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target}| \le \mathsf{shrinkage}\ \mathsf{sampleCount} \cdot \mathsf{error}$$
$$+ 1/((\mathsf{sampleCount} : \mathbb{R}) + 2)$$

$$(\mathsf{successes}\ \mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target}\ \mathsf{error} : \mathbb{R})\ (\mathsf{empiricalErrorBound} :$$
$$|\mathsf{empiricalRate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target}| \le \mathsf{error})$$
$$0 \le \mathsf{error}$$

13.124 fep-124 — Squared-Risk Transfer through Laplace Smoothing

Squared add-one error is bounded pointwise by twice the contracted empirical squared error plus twice the squared pseudo-count offset, and the inequality integrates under every normalized finite sampling law. Assumption scope: Sample count is positive, each success count is bounded by it, and the target is in the unit interval. The factor-two inequality is conservative and finite-law expectation is not an asymptotic risk claim. Semantic disposition: `formalized`.

13.124.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Squared-Risk Transfer -/
namespace FEP124

open FEP FEP.EmpiricalRisk

/-- Pointwise squared Laplace error is controlled by contracted empirical
squared error and the squared pseudo-count offset. -/
theorem fep124_laplaceSquaredError_le
    (successes sampleCount : ℕ) (target : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : successes ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1) :
    (laplaceEstimate successes sampleCount - target) ^ 2 ≤
      2 * shrinkage sampleCount ^ 2 *
          (empiricalRate successes sampleCount - target) ^ 2 +
        2 * (1 / ((sampleCount : ℝ) + 2)) ^ 2 := by
  have _ := successesAtMost
  exact laplaceSquaredError_le successes sampleCount target
    sampleCountPositive targetBounds

/-- The same pointwise inequality integrates under every normalized finite
sampling law. -/
theorem fep124_laplaceSquaredRisk_le
    {Ω : Type*} [Fintype Ω] (sampling : FiniteLaw Ω)
    (successes : Ω → ℕ) (sampleCount : ℕ) (target : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : ∀ outcome, successes outcome ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1) :
    FEP.VariationalDuality.expectation sampling (fun outcome =>
      (laplaceEstimate (successes outcome) sampleCount - target) ^ 2) ≤
      2 * shrinkage sampleCount ^ 2 *
```

```
          FEP.VariationalDuality.expectation sampling (fun outcome =>
            (empiricalRate (successes outcome) sampleCount - target) ^ 2) +
        2 * (1 / ((sampleCount : ℝ) + 2)) ^ 2 :=
  laplaceSquaredRisk_le sampling successes sampleCount target
    sampleCountPositive successesAtMost targetBounds

end FEP124
```

13.124.2 Typeset statement signatures

$$(\mathsf{successes}\ \mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target} : \mathbb{R})\ (\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})$$
$$(\mathsf{successesAtMost} : \mathsf{successes} \le \mathsf{sampleCount})$$
$$(\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$(\mathsf{laplaceEstimate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target})^2 \le 2 \cdot \mathsf{shrinkage}$$
$$\mathsf{sampleCount}^2 \cdot (\mathsf{empiricalRate}\ \mathsf{successes}\ \mathsf{sampleCount} - \mathsf{target})^2 + 2$$
$$\cdot\ (1/((\mathsf{sampleCount} : \mathbb{R}) + 2))^2$$

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \Omega]\ (\mathsf{sampling} : \mathsf{FiniteLaw}\ \Omega)\ (\mathsf{successes} : \Omega \Rightarrow \mathbb{N})$$
$$(\mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target} : \mathbb{R})\ (\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})$$
$$(\mathsf{successesAtMost} : \forall \mathsf{outcome}, \mathsf{successes}\ \mathsf{outcome} \le \mathsf{sampleCount})$$
$$(\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\mathsf{FEP.VariationalDuality.expectation}\ \mathsf{sampling}\ (\lambda\ \mathsf{outcome} \mapsto (\mathsf{laplaceEstimate}$$
$$(\mathsf{successes}\ \mathsf{outcome})\ \mathsf{sampleCount} - \mathsf{target})^2) \le 2 \cdot \mathsf{shrinkage}\ \mathsf{sampleCount}^2$$
$$\cdot\ \mathsf{FEP.VariationalDuality.expectation}\ \mathsf{sampling}\ (\lambda\ \mathsf{outcome} \mapsto (\mathsf{empiricalRate}$$
$$(\mathsf{successes}\ \mathsf{outcome})\ \mathsf{sampleCount} - \mathsf{target})^2) + 2$$
$$\cdot\ (1/((\mathsf{sampleCount} : \mathbb{R}) + 2))^2$$

13.125 fep-125 — Bernoulli Brier Excess-Risk Identity

Bernoulli Brier excess risk relative to the true probability is exactly the squared forecast error. Assumption scope: The identity is polynomial and therefore holds over real target and forecast inputs; the probability interpretation requires both to lie in the unit interval, which is supplied by downstream statistical uses. Semantic disposition: `formalized`.

13.125.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Bernoulli Brier Excess-Risk Identity -/
namespace FEP125

open FEP.EmpiricalRisk

/-- Bernoulli Brier excess risk is exactly squared probability error. -/
theorem fep125_brierExcess_eq_sqError (target forecast : ℝ) :
    bernoulliBrierScore target forecast - bernoulliBrierScore target target =
      (forecast - target) ^ 2 :=
  brierExcess_eq_sqError target forecast

/-- Forecasting the target itself has zero excess Brier risk. -/
theorem fep125_brierExcess_self (target : ℝ) :
    bernoulliBrierScore target target - bernoulliBrierScore target target = 0 := by
  ring

end FEP125
```

13.125.2 Typeset statement signatures

$$(\mathsf{target\ forecast} : \mathbb{R})$$
$$\mathsf{bernoulliBrierScore\ target\ forecast} - \mathsf{bernoulliBrierScore\ target\ target} = (\mathsf{forecast} - \mathsf{target})^2$$

$$(\mathsf{target} : \mathbb{R})$$
$$\mathsf{bernoulliBrierScore\ target\ target} - \mathsf{bernoulliBrierScore\ target\ target} = 0$$

13.126 fep-126 — Finite-Law Laplace Brier-Risk Bound

Finite-law expected Brier excess for the add-one forecast inherits the squared-error transfer bound. Assumption scope: Sampling uses the normalized FiniteLaw carrier, sample count is positive, success counts are bounded, and the target is in the unit interval. This is an expected finite-model inequality, not an empirical performance report. Semantic disposition: `formalized`.

13.126.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Laplace Brier-Risk Bound -/
namespace FEP126

open FEP FEP.EmpiricalRisk

/-- The Brier excess risk of the add-one forecast inherits the finite-law
squared-error transfer bound. -/
theorem fep126_laplaceBrierRisk_le
    {Ω : Type*} [Fintype Ω] (sampling : FiniteLaw Ω)
    (successes : Ω → ℕ) (sampleCount : ℕ) (target : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : ∀ outcome, successes outcome ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1) :
    brierExcessRisk sampling
        (fun outcome => laplaceEstimate (successes outcome) sampleCount) target ≤
      2 * shrinkage sampleCount ^ 2 *
          FEP.VariationalDuality.expectation sampling (fun outcome =>
            (empiricalRate (successes outcome) sampleCount - target) ^ 2) +
        2 * (1 / ((sampleCount : ℝ) + 2)) ^ 2 :=
  laplaceBrierRisk_le sampling successes sampleCount target
    sampleCountPositive successesAtMost targetBounds

/-- The risk theorem is finite-law weighted; it does not require a continuous
sampling carrier. -/
theorem fep126_sampling_mass_one
    {Ω : Type*} [Fintype Ω] (sampling : FiniteLaw Ω) :
    ∑ outcome, sampling outcome = 1 :=
  sampling.sum_one

end FEP126
```

13.126.2 Typeset statement signatures

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \Omega]\ (\mathsf{sampling} : \mathsf{FiniteLaw}\ \Omega)\ (\mathsf{successes} : \Omega \Rightarrow \mathbb{N})$$
$$(\mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target} : \mathbb{R})\ (\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})$$
$$(\mathsf{successesAtMost} : \forall \mathsf{outcome}, \mathsf{successes}\ \mathsf{outcome} \leq \mathsf{sampleCount})$$
$$(\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\mathsf{brierExcessRisk}\ \mathsf{sampling}\ (\lambda\ \mathsf{outcome} \mapsto \mathsf{laplaceEstimate}\ (\mathsf{successes}\ \mathsf{outcome})$$
$$\mathsf{sampleCount})\ \mathsf{target} \leq 2 \cdot \mathsf{shrinkage}\ \mathsf{sampleCount}^2$$
$$\cdot\ \mathsf{FEP.VariationalDuality.expectation}\ \mathsf{sampling}\ (\lambda\ \mathsf{outcome} \mapsto (\mathsf{empiricalRate}$$
$$(\mathsf{successes}\ \mathsf{outcome})\ \mathsf{sampleCount} - \mathsf{target})^2) + 2$$
$$\cdot\ (1/((\mathsf{sampleCount} : \mathbb{R}) + 2))^2$$

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \Omega]\ (\mathsf{sampling} : \mathsf{FiniteLaw}\ \Omega)$$
$$\sum \mathsf{outcome}, \mathsf{sampling}\ \mathsf{outcome} = 1$$

13.127 fep-127 — Concentration-Event Transfer through Smoothing

A smoothed deviation beyond the transferred threshold is contained in the corresponding raw empirical-deviation event, so any supplied finite-law probability bound transfers monotonically. Assumption scope: Sample count is positive, success counts are bounded, the target is in the unit interval, and a raw event-probability bound is supplied. The theorem composes with concentration results but does not derive their assumptions or rates itself. Semantic disposition: `formalized`.

13.127.1 Lean sketch

```
import FepSketches.empirical_risk

/-! # Concentration-Event Transfer Through Smoothing -/
namespace FEP127

open FEP FEP.EmpiricalRisk

/-- A smoothed deviation beyond the transferred threshold is contained in the
corresponding raw empirical-deviation event. -/
theorem fep127_laplaceBadEvent_subset
    {Ω : Type*} [Fintype Ω] (successes : Ω → ℕ)
    (sampleCount : ℕ) (target error : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : ∀ outcome, successes outcome ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1) :
    laplaceBadEvent successes sampleCount target error ⊆
      empiricalBadEvent successes sampleCount target error :=
  laplaceBadEvent_subset successes sampleCount target error
    sampleCountPositive successesAtMost targetBounds

/-- Any finite-law bound on the raw event transfers to the contained smoothed
event without asserting a general posterior-contraction theorem. -/
theorem fep127_laplaceBadEvent_probability_le
    {Ω : Type*} [Fintype Ω] (sampling : FiniteLaw Ω)
    (successes : Ω → ℕ) (sampleCount : ℕ)
    (target error failure : ℝ)
    (sampleCountPositive : 0 < sampleCount)
    (successesAtMost : ∀ outcome, successes outcome ≤ sampleCount)
    (targetBounds : target ∈ Set.Icc (0 : ℝ) 1)
    (rawProbabilityBound :
      finiteEventProbability sampling
        (empiricalBadEvent successes sampleCount target error) ≤ failure) :
    finiteEventProbability sampling
        (laplaceBadEvent successes sampleCount target error) ≤ failure :=
  laplaceBadEvent_probability_le sampling successes sampleCount target error
```

```
    failure sampleCountPositive successesAtMost targetBounds rawProbabilityBound

end FEP127
```

13.127.2 Typeset statement signatures

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \Omega]\ (\mathsf{successes} : \Omega \Rightarrow \mathbb{N})\ (\mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target\ error} : \mathbb{R})$$
$$(\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})\ (\mathsf{successesAtMost} : \forall \mathsf{outcome}, \mathsf{successes}$$
$$\mathsf{outcome} \leq \mathsf{sampleCount})\ (\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\mathsf{laplaceBadEvent\ successes\ sampleCount\ target\ error} \subseteq \mathsf{empiricalBadEvent\ successes}$$
$$\mathsf{sampleCount\ target\ error}$$

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \Omega]\ (\mathsf{sampling} : \mathsf{FiniteLaw}\ \Omega)\ (\mathsf{successes} : \Omega \Rightarrow \mathbb{N})$$
$$(\mathsf{sampleCount} : \mathbb{N})\ (\mathsf{target\ error\ failure} : \mathbb{R})$$
$$(\mathsf{sampleCountPositive} : 0 < \mathsf{sampleCount})\ (\mathsf{successesAtMost} : \forall \mathsf{outcome}, \mathsf{successes}$$
$$\mathsf{outcome} \leq \mathsf{sampleCount})\ (\mathsf{targetBounds} : \mathsf{target} \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$(\mathsf{rawProbabilityBound} : \mathsf{finiteEventProbability\ sampling}\ (\mathsf{empiricalBadEvent}$$
$$\mathsf{successes\ sampleCount\ target\ error}) \leq \mathsf{failure})$$
$$\mathsf{finiteEventProbability\ sampling}$$
$$(\mathsf{laplaceBadEvent\ successes\ sampleCount\ target\ error}) \leq \mathsf{failure}$$

13.128 fep-128 — Observation-Contingent Policy-Tree Recursion

A finite policy-tree node equals its stage cost plus the observation-law weighted values of observation-contingent continuation trees. Assumption scope: Belief, action, and observation carriers are finite and the horizon is a natural-number depth. Observation laws and deterministic belief updates are supplied by the model; no infinite-horizon or continuous-belief claim is made. Semantic disposition: `formalized`.

13.128.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Observation-Contingent Policy-Tree Recursion -/
namespace FEP128

open FEP.PolicyTrees

/-- A node chooses one action and then follows the continuation indexed by the
realized observation. -/
theorem fep128_policyTreeValue_node
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation]
    (model : PolicyTreeModel Belief Action Observation)
    {depth : ℕ} (action : Action)
    (continuation : Observation → PolicyTree Action Observation depth)
    (belief : Belief) :
    policyTreeValue (depth := depth + 1) model (action, continuation) belief =
      model.stageCost depth belief action +
        ∑ observation,
          model.observationLaw belief action observation *
            policyTreeValue model (continuation observation)
              (model.update belief action observation) :=
  policyTreeValue_node model action continuation belief

/-- The depth-zero carrier has no action node and therefore zero value. -/
theorem fep128_policyTreeValue_leaf
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation]
    (model : PolicyTreeModel Belief Action Observation) (belief : Belief) :
    policyTreeValue (depth := 0) model
```

```
        (PUnit.unit : PolicyTree Action Observation 0) belief = 0 :=
  rfl

end FEP128
```

13.128.2 Typeset statement signatures

$$\{\text{Belief Action Observation} : \text{Type}^*\}\ [\text{Fintype Belief}]\ [\text{Fintype Action}]$$
$$[\text{Fintype Observation}]\ (\text{model} : \text{PolicyTreeModel Belief Action Observation})$$
$$\{\text{depth} : \mathbb{N}\}\ (\text{action} : \text{Action})\ (\text{continuation} : \text{Observation} \Rightarrow \text{PolicyTree Action}$$
$$\text{Observation depth})\ (\text{belief} : \text{Belief})$$
$$\text{policyTreeValue}\ (\text{depth} := \text{depth} + 1)\ \text{model}\ (\text{action}, \text{continuation})\ \text{belief}$$
$$= \text{model.stageCost depth belief action} + \sum \text{observation, model.observationLaw}$$
$$\text{belief action observation} \cdot \text{policyTreeValue model}\ (\text{continuation observation})$$
$$(\text{model.update belief action observation})$$

$$\{\text{Belief Action Observation} : \text{Type}^*\}\ [\text{Fintype Belief}]\ [\text{Fintype Action}]$$
$$[\text{Fintype Observation}]\ (\text{model} : \text{PolicyTreeModel Belief Action Observation})$$
$$(\text{belief} : \text{Belief})$$
$$\text{policyTreeValue}\ (\text{depth} := 0)\ \text{model}$$
$$(\text{PUnit.unit} : \text{PolicyTree Action Observation}\ 0)\ \text{belief} = 0$$

13.129 fep-129 — Finite Policy-Tree Bellman Minimum

Backward induction exposes an exact finite-action Bellman minimum of stage cost plus observation-weighted optimal continuation. Assumption scope: The action carrier is finite and nonempty, and belief and observation carriers are finite. The result is one finite-horizon recursion, not a contraction or fixed-point theorem. Semantic disposition: `formalized`.

13.129.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Finite Policy-Tree Bellman Minimum -/
namespace FEP129

open FEP.PolicyTrees

/-- The finite Bellman value is the selected action minimum of stage cost plus
the observation-weighted optimal continuation. -/
theorem fep129_optimalTreeValue_eq_min
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation] [Nonempty Action]
    (model : PolicyTreeModel Belief Action Observation)
    (depth : ℕ) (belief : Belief) :
    optimalTreeValue model (depth + 1) belief =
      model.stageCost depth belief (optimalTreeAction model depth belief) +
        Σ observation,
          model.observationLaw belief (optimalTreeAction model depth belief)
              observation *
            optimalTreeValue model depth
              (model.update belief (optimalTreeAction model depth belief)
                observation) :=
  optimalTreeValue_eq_min model depth belief

/-- The selected node action is no worse than any finite alternative. -/
theorem fep129_optimalTreeAction_le
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation] [Nonempty Action]
    (model : PolicyTreeModel Belief Action Observation)
```

```
    (depth : ℕ) (belief : Belief) (alternative : Action) :
    model.stageCost depth belief (optimalTreeAction model depth belief) +
        Σ observation,
          model.observationLaw belief (optimalTreeAction model depth belief)
              observation *
            optimalTreeValue model depth
              (model.update belief (optimalTreeAction model depth belief)
                observation) ≤
      model.stageCost depth belief alternative +
        Σ observation,
          model.observationLaw belief alternative observation *
            optimalTreeValue model depth
              (model.update belief alternative observation) :=
  optimalTreeAction_le model depth belief alternative

end FEP129
```

13.129.2 Typeset statement signatures

$$\{\text{Belief Action Observation} : \text{Type}^*\}\ [\text{Fintype Belief}]\ [\text{Fintype Action}]\ [\text{Fintype Observation}]\ [\text{Nonempty Action}]\ (\text{model} : \text{PolicyTreeModel Belief Action Observation})\ (\text{depth} : \mathbb{N})\ (\text{belief} : \text{Belief})\ \text{optimalTreeValue model}\ (\text{depth} + 1)\ \text{belief} = \text{model.stageCost depth belief}\ (\text{optimalTreeAction model depth belief}) + \sum \text{observation, model.observationLaw belief}\ (\text{optimalTreeAction model depth belief})\ \text{observation} \cdot \text{optimalTreeValue model depth}\ (\text{model.update belief}\ (\text{optimalTreeAction model depth belief})\ \text{observation})$$

$$\{\text{Belief Action Observation} : \text{Type}^*\}\ [\text{Fintype Belief}]\ [\text{Fintype Action}]\ [\text{Fintype Observation}]\ [\text{Nonempty Action}]\ (\text{model} : \text{PolicyTreeModel Belief Action Observation})\ (\text{depth} : \mathbb{N})\ (\text{belief} : \text{Belief})\ (\text{alternative} : \text{Action})\ \text{model.stageCost depth belief}\ (\text{optimalTreeAction model depth belief}) + \sum \text{observation, model.observationLaw belief}\ (\text{optimalTreeAction model depth belief})\ \text{observation} \cdot \text{optimalTreeValue model depth}\ (\text{model.update belief}\ (\text{optimalTreeAction model depth belief})\ \text{observation}) \le \text{model.stageCost depth belief alternative} + \sum \text{observation, model.observationLaw belief alternative}\ \text{observation} \cdot \text{optimalTreeValue model depth}\ (\text{model.update belief alternative observation})$$

13.130 fep-130 — Optimal Finite Policy-Tree Existence

Every finite-depth, finite-carrier policy-tree problem with a nonempty action set has a concrete backward-inducted tree attaining the recursive optimum. Assumption scope: All carriers and the horizon are finite and the action carrier is nonempty. Existence does not extend to unbounded horizon, arbitrary measurable policy spaces, or continuous beliefs. Semantic disposition: `formalized`.

13.130.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Optimal Finite Policy-Tree Existence -/
namespace FEP130

open FEP.PolicyTrees
```

```
/-- Backward induction supplies an attaining optimum on the exact finite
depth-indexed tree carrier. -/
theorem fep130_exists_optimalPolicyTree
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation] [Nonempty Action]
    (model : PolicyTreeModel Belief Action Observation)
    (depth : ℕ) (belief : Belief) :
    ∃ tree : PolicyTree Action Observation depth,
      policyTreeValue model tree belief = optimalTreeValue model depth belief ∧
        ∀ alternative : PolicyTree Action Observation depth,
          policyTreeValue model tree belief ≤
            policyTreeValue model alternative belief :=
  exists_optimalPolicyTree model depth belief

/-- The authored carrier is finite at every finite depth. -/
theorem fep130_policyTree_carrier_finite
    {Action Observation : Type*}
    [Fintype Action] [Fintype Observation] (depth : ℕ) :
    Finite (PolicyTree Action Observation depth) := by
  infer_instance

end FEP130
```

13.130.2 Typeset statement signatures

$$\{\text{Belief Action Observation} : \text{Type}^*\}\ [\text{Fintype Belief}]\ [\text{Fintype Action}]$$
$$[\text{Fintype Observation}]\ [\text{Nonempty Action}]$$
$$(\text{model} : \text{PolicyTreeModel Belief Action Observation})\ (\text{depth} : \mathbb{N})$$
$$(\text{belief} : \text{Belief})$$
$$\exists \text{tree} : \text{PolicyTree Action Observation depth},\ \text{policyTreeValue model tree belief}$$
$$= \text{optimalTreeValue model depth belief} \wedge \forall \text{alternative} : \text{PolicyTree Action}$$
$$\text{Observation depth},\ \text{policyTreeValue model tree belief} \leq \text{policyTreeValue model}$$
$$\text{alternative belief}$$

$$\{\text{Action Observation} : \text{Type}^*\}\ [\text{Fintype Action}]\ [\text{Fintype Observation}]\ (\text{depth} : \mathbb{N})$$
$$\text{Finite}\ (\text{PolicyTree Action Observation depth})$$

13.131 fep-131 — Open-Loop Plan Embedding into Policy Trees

Reusing the same continuation after every observation embeds an open-loop action plan into the policy-tree carrier without changing its recursive value. Assumption scope: The comparison uses the same model, belief, and finite depth. Open-loop plans ignore observations by construction; the theorem does not identify arbitrary state-feedback policies with open-loop sequences. Semantic disposition: `formalized`.

13.131.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Open-Loop Plan Embedding -/
namespace FEP131

open FEP.PolicyTrees

/-- Reusing the same continuation after every observation embeds an open-loop
plan without changing its recursive value. -/
theorem fep131_openLoopEmbedding_value
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation]
    (model : PolicyTreeModel Belief Action Observation)
    {depth : ℕ} (plan : OpenLoopPlan Action depth) (belief : Belief) :
    policyTreeValue model
```

```
        (openLoopEmbedding (Observation := Observation) plan) belief =
      openLoopValue model plan belief :=
  openLoopEmbedding_value model plan belief

/-- At depth zero, the embedding maps the unique empty plan to the unique
policy-tree leaf. -/
theorem fep131_openLoopEmbedding_leaf
    {Action Observation : Type*} :
    openLoopEmbedding (depth := 0) (Action := Action) (Observation := Observation)
        (PUnit.unit : OpenLoopPlan Action 0) =
      (PUnit.unit : PolicyTree Action Observation 0) :=
  rfl

end FEP131
```

13.131.2 Typeset statement signatures

$$
\begin{aligned}
&\{\mathsf{Belief}\ \mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Belief}]\ [\mathsf{Fintype}\ \mathsf{Action}]\\
&[\mathsf{Fintype}\ \mathsf{Observation}]\ (\mathsf{model} : \mathsf{PolicyTreeModel}\ \mathsf{Belief}\ \mathsf{Action}\ \mathsf{Observation})\\
&\{\mathsf{depth} : \mathbb{N}\}\ (\mathsf{plan} : \mathsf{OpenLoopPlan}\ \mathsf{Action}\ \mathsf{depth})\ (\mathsf{belief} : \mathsf{Belief})\\
&\mathsf{policyTreeValue}\ \mathsf{model}\ (\mathsf{openLoopEmbedding}\ (\mathsf{Observation} := \mathsf{Observation})\ \mathsf{plan})\\
&\mathsf{belief} = \mathsf{openLoopValue}\ \mathsf{model}\ \mathsf{plan}\ \mathsf{belief}
\end{aligned}
$$

$$
\begin{aligned}
&\{\mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}\\
&\mathsf{openLoopEmbedding}\ (\mathsf{depth} := 0)\ (\mathsf{Action} := \mathsf{Action})\ (\mathsf{Observation} := \mathsf{Observation})\\
&(\mathsf{PUnit.unit} : \mathsf{OpenLoopPlan}\ \mathsf{Action}\ 0)\\
&= (\mathsf{PUnit.unit} : \mathsf{PolicyTree}\ \mathsf{Action}\ \mathsf{Observation}\ 0)
\end{aligned}
$$

13.132 fep-132 — Closed-Loop Policy-Tree Dominance over Open Loop

The optimal observation-contingent finite policy tree is no worse than every open-loop plan embedded at the same depth. Assumption scope: Actions are finite and nonempty, all other carriers are finite, and both sides share one finite horizon and cost model. No infinite-horizon POMDP dominance is claimed. Semantic disposition: `formalized`.

13.132.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Closed-Loop Dominance over Open Loop -/
namespace FEP132

open FEP.PolicyTrees

/-- The finite closed-loop optimum is no worse than any observation-ignoring
open-loop plan embedded at the same depth. -/
theorem fep132_optimalTree_le_openLoop
    {Belief Action Observation : Type*}
    [Fintype Belief] [Fintype Action] [Fintype Observation] [Nonempty Action]
    (model : PolicyTreeModel Belief Action Observation)
    {depth : ℕ} (plan : OpenLoopPlan Action depth) (belief : Belief) :
    optimalTreeValue model depth belief ≤ openLoopValue model plan belief :=
  optimalTree_le_openLoop model plan belief

/-- Dominance compares equal finite horizons; it does not assert an
infinite-horizon POMDP optimum. -/
theorem fep132_finite_horizon_depth (depth : ℕ) : depth < depth + 1 :=
  Nat.lt_succ_self depth

end FEP132
```

13.132.2 Typeset statement signatures

$$\{\mathsf{Belief}\ \mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Belief}]\ [\mathsf{Fintype}\ \mathsf{Action}]$$
$$[\mathsf{Fintype}\ \mathsf{Observation}]\ [\mathsf{Nonempty}\ \mathsf{Action}]$$
$$(\mathsf{model} : \mathsf{PolicyTreeModel}\ \mathsf{Belief}\ \mathsf{Action}\ \mathsf{Observation})\ \{\mathsf{depth} : \mathbb{N}\}$$
$$(\mathsf{plan} : \mathsf{OpenLoopPlan}\ \mathsf{Action}\ \mathsf{depth})\ (\mathsf{belief} : \mathsf{Belief})$$
$$\mathsf{optimalTreeValue}\ \mathsf{model}\ \mathsf{depth}\ \mathsf{belief} \le \mathsf{openLoopValue}\ \mathsf{model}\ \mathsf{plan}\ \mathsf{belief}$$

$$(\mathsf{depth} : \mathbb{N})$$
$$\mathsf{depth} < \mathsf{depth} + 1$$

13.133 fep-133 — Treewise Expected-Free-Energy Decomposition

The one-step full-support expected-free-energy identity lifts through every branch of a finite policy tree, and the corresponding optimal values agree. Assumption scope: Every belief-indexed finite generative model satisfies the maintained FullSupport contract. Belief updates and horizons are finite; the theorem transports the existing risk-plus-ambiguity identity and does not establish its empirical adequacy. Semantic disposition: `formalized`.

13.133.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Treewise EFE Decomposition -/
namespace FEP133

open FEP.PolicyTrees

/-- The one-step full-support EFE identity lifts through every branch of a
finite observation-contingent tree. -/
theorem fep133_policyTree_efe_eq_risk_add_ambiguity
    {Belief State Action Observation : Type*}
    [Fintype Belief] [Fintype State] [Fintype Action] [Fintype Observation]
    (model : EFEPolicyTreeModel Belief State Action Observation)
    {depth : ℕ} (tree : PolicyTree Action Observation depth)
    (belief : Belief) :
    policyTreeValue (efePolicyTreeModel model) tree belief =
      policyTreeValue (riskAmbiguityPolicyTreeModel model) tree belief :=
  policyTree_efe_eq_risk_add_ambiguity model tree belief

/-- Consequently, the finite optimal EFE and optimal risk-plus-ambiguity
values agree. -/
theorem fep133_optimalValues_agree
    {Belief State Action Observation : Type*}
    [Fintype Belief] [Fintype State] [Fintype Action] [Fintype Observation]
    [Nonempty Action]
    (model : EFEPolicyTreeModel Belief State Action Observation)
    (depth : ℕ) (belief : Belief) :
    optimalTreeValue (efePolicyTreeModel model) depth belief =
      optimalTreeValue (riskAmbiguityPolicyTreeModel model) depth belief :=
  optimalEFEValue_eq_riskAmbiguity model depth belief

end FEP133
```

13.133.2 Typeset statement signatures

$$\{\mathsf{Belief}\ \mathsf{State}\ \mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Belief}]\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$[\mathsf{Fintype}\ \mathsf{Action}]\ [\mathsf{Fintype}\ \mathsf{Observation}]\ (\mathsf{model} : \mathsf{EFEPolicyTreeModel}\ \mathsf{Belief}\ \mathsf{State}$$
$$\mathsf{Action}\ \mathsf{Observation})\ \{\mathsf{depth} : \mathbb{N}\}\ (\mathsf{tree} : \mathsf{PolicyTree}\ \mathsf{Action}\ \mathsf{Observation}\ \mathsf{depth})$$
$$(\mathsf{belief} : \mathsf{Belief})$$
$$\mathsf{policyTreeValue}\ (\mathsf{efePolicyTreeModel}\ \mathsf{model})\ \mathsf{tree}\ \mathsf{belief} = \mathsf{policyTreeValue}$$
$$(\mathsf{riskAmbiguityPolicyTreeModel}\ \mathsf{model})\ \mathsf{tree}\ \mathsf{belief}$$

$$\{\mathsf{Belief}\ \mathsf{State}\ \mathsf{Action}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Belief}]\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$[\mathsf{Fintype}\ \mathsf{Action}]\ [\mathsf{Fintype}\ \mathsf{Observation}]\ [\mathsf{Nonempty}\ \mathsf{Action}]\ (\mathsf{model} :$$
$$\mathsf{EFEPolicyTreeModel}\ \mathsf{Belief}\ \mathsf{State}\ \mathsf{Action}\ \mathsf{Observation})\ (\mathsf{depth} : \mathbb{N})$$
$$(\mathsf{belief} : \mathsf{Belief})$$
$$\mathsf{optimalTreeValue}\ (\mathsf{efePolicyTreeModel}\ \mathsf{model})\ \mathsf{depth}\ \mathsf{belief} = \mathsf{optimalTreeValue}$$
$$(\mathsf{riskAmbiguityPolicyTreeModel}\ \mathsf{model})\ \mathsf{depth}\ \mathsf{belief}$$

13.134 fep-134 — Strict Boolean Feedback Advantage

In the explicit two-step Boolean model, observation-contingent feedback has value zero while every fixed open-loop second action has value one half. Assumption scope: This is a finite exact counterexample on fair Boolean observations and a mismatch terminal cost. It demonstrates possible feedback advantage, not a universal quantitative gap for active-inference agents. Semantic disposition: `formalized`.

13.134.1 Lean sketch

```
import FepSketches.policy_tree

/-! # Strict Boolean Feedback Advantage -/
namespace FEP134

open FEP.PolicyTrees

/-- Observation-contingent Boolean feedback has value zero, every fixed
open-loop continuation has value one half, and the comparison is strict. -/
theorem fep134_boolFeedback_strictlyBetter (action : Bool) :
    policyTreeValue boolFeedbackModel boolFeedbackTree false = 0 ∧
      openLoopValue boolFeedbackModel (boolOpenLoopPlan action) false = 1 / 2 ∧
      policyTreeValue boolFeedbackModel boolFeedbackTree false <
        openLoopValue boolFeedbackModel (boolOpenLoopPlan action) false :=
  ⟨boolFeedbackTree_value_zero, boolOpenLoop_value_half action,
    boolFeedbackTree_strictlyBetter action⟩

/-- The witness is genuinely closed loop: its second action changes with the
first observation. -/
theorem fep134_feedback_continuation_changes :
    (boolFeedbackTree.2 false).1 ≠ (boolFeedbackTree.2 true).1 := by
  decide

end FEP134
```

13.134.2 Typeset statement signatures

$$(\mathsf{action} : \mathsf{Bool})$$
$$\mathsf{policyTreeValue}\ \mathsf{boolFeedbackModel}\ \mathsf{boolFeedbackTree}\ \mathsf{false} = 0 \wedge \mathsf{openLoopValue}$$
$$\mathsf{boolFeedbackModel}\ (\mathsf{boolOpenLoopPlan}\ \mathsf{action})\ \mathsf{false} = 1/2 \wedge \mathsf{policyTreeValue}$$
$$\mathsf{boolFeedbackModel}\ \mathsf{boolFeedbackTree}\ \mathsf{false} < \mathsf{openLoopValue}\ \mathsf{boolFeedbackModel}$$
$$(\mathsf{boolOpenLoopPlan}\ \mathsf{action})\ \mathsf{false}$$

$$(\mathsf{boolFeedbackTree}.2\ \mathsf{false}).1 \neq (\mathsf{boolFeedbackTree}.2\ \mathsf{true}).1$$

13.135 fep-135 — Finite-Law Measure Embedding

A normalized finite law embeds as a weighted sum of Dirac measures that preserves every singleton mass and has native measure mass one. Assumption scope: The carrier is finite with its discrete measurable space. Finite real masses enter the native nonnegative-extended-real measure through ENNReal.ofReal; normalization and nonnegativity are inherited from FiniteLaw. Semantic disposition: `formalized`.

13.135.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP135

open FEP MeasureTheory

variable {State : Type*} [Fintype State]
  [MeasurableSpace State] [DiscreteMeasurableSpace State]

/-- The weighted-Dirac embedding preserves every finite singleton mass. -/
theorem fep135_embeddedLaw_apply_singleton
    (law : FiniteLaw State) (state : State) :
    FEP.NativeBlanket.embeddedLaw law {state} = ENNReal.ofReal (law state) :=
  FEP.NativeBlanket.embeddedLaw_apply_singleton law state

/-- The embedded finite law is normalized as a native measure. -/
theorem fep135_embeddedLaw_normalized (law : FiniteLaw State) :
    FEP.NativeBlanket.embeddedLaw law Set.univ = 1 :=
  FEP.NativeBlanket.embeddedLaw_apply_univ law

end FEP135
```

13.135.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\mathsf{State}]$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\,\mathsf{State})\,(\mathsf{state} : \mathsf{State})$$
$$\mathsf{FEP.NativeBlanket.embeddedLaw}\,\mathsf{law}\,\{\mathsf{state}\} = \mathsf{ENNReal.ofReal}\,(\mathsf{law}\,\mathsf{state})$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\,[\mathsf{Fintype}\,\mathsf{State}]$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\,\mathsf{State})$$
$$\mathsf{FEP.NativeBlanket.embeddedLaw}\,\mathsf{law}\,\Omega = 1$$

13.136 fep-136 — Finite-Law Embedding Injectivity and Expectation Transfer

Equality of native weighted-Dirac measures reflects equality of the source finite laws, and native integration agrees exactly with the finite weighted expectation sum. Assumption scope: The carrier is finite and discretely measurable and observables are real-valued. Finiteness supplies integrability; no general integration or measure-equivalence theorem is inferred. Semantic disposition: `formalized`.

13.136.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP136

open FEP MeasureTheory
open scoped BigOperators MeasureTheory

variable {State : Type*} [Fintype State]
  [MeasurableSpace State] [DiscreteMeasurableSpace State]

/-- Equality of native embeddings reflects equality of finite laws. -/
theorem fep136_embeddedLaw_injective :
    Function.Injective
```

```
      (FEP.NativeBlanket.embeddedLaw : FiniteLaw State → Measure State) :=
  FEP.NativeBlanket.embeddedLaw_injective

/-- Native expectation is exactly the authored finite weighted sum. -/
theorem fep136_embeddedLaw_expectation
    (law : FiniteLaw State) (observable : State → ℝ) :
    ∫ state, observable state ∂FEP.NativeBlanket.embeddedLaw law =
      Σ state, law state * observable state :=
  FEP.NativeBlanket.embeddedLaw_integral_eq_sum law observable

end FEP136
```

13.136.2 Typeset statement signatures

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$\mathsf{Function.Injective}\ (\mathsf{FEP.NativeBlanket.embeddedLaw} : \mathsf{FiniteLaw}\ \mathsf{State} \Rightarrow \mathsf{Measure}\ \mathsf{State})$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{observable} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\int \mathsf{state}, \mathsf{observable}\ \mathsf{state}\ \mathrm{d}\ \mathsf{FEP.NativeBlanket.embeddedLaw}\ \mathsf{law} = \sum \mathsf{state}, \mathsf{law}\ \mathsf{state} \cdot \mathsf{observable}\ \mathsf{state}$$

13.137 fep-137 — Embedded Finite-Kernel Composition

Embedding finite prediction equals native composition of the embedded finite kernel with the embedded prior measure. Assumption scope: Input and output carriers are finite and discretely measurable and each kernel row is normalized. The theorem is a carrier-transport identity, not a general disintegration or arbitrary-kernel approximation result. Semantic disposition: `formalized`.

13.137.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP137

open FEP MeasureTheory ProbabilityTheory
open scoped ENNReal MeasureTheory ProbabilityTheory

variable {Input Output : Type*} [Fintype Input] [Fintype Output]
  [MeasurableSpace Input] [MeasurableSpace Output]
  [DiscreteMeasurableSpace Input] [DiscreteMeasurableSpace Output]

/-- Finite prediction agrees with native measure-kernel composition. -/
theorem fep137_embeddedPredictive_eq_comp
    (prior : FiniteLaw Input) (kernel : FiniteKernel Input Output) :
    FEP.NativeBlanket.embeddedLaw (kernel.predictive prior) =
      FEP.NativeBlanket.embeddedKernel kernel ∘ₘ
        FEP.NativeBlanket.embeddedLaw prior :=
  FEP.NativeBlanket.embeddedPredictive_eq_comp prior kernel

end FEP137
```

13.137.2 Typeset statement signatures

$$\{\mathsf{Input}\ \mathsf{Output} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Input}]\ [\mathsf{Fintype}\ \mathsf{Output}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{Input})\ (\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{Input}\ \mathsf{Output})$$
$$\mathsf{FEP.NativeBlanket.embeddedLaw}\ (\mathsf{kernel.predictive}\ \mathsf{prior})$$
$$= \mathsf{FEP.NativeBlanket.embeddedKernel}\ \mathsf{kernel} \circ_m \mathsf{FEP.NativeBlanket.embeddedLaw}\ \mathsf{prior}$$

13.138 fep-138 — Markov-Blanket Rectangle Factorization

Every singleton rectangle of blanket, internal, and external coordinates in the embedded static joint has the exact blanket-marginal times two conditional-mass factorization. Assumption scope: All four carriers are finite and discretely measurable and the static model supplies normalized blanket and conditional laws. The theorem covers the discrete singleton generator and does not by itself state Mathlib conditional independence. Semantic disposition: `formalized`.

13.138.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP138

open FEP FEP.MarkovBlanket MeasureTheory

variable {Internal Sensory Active External : Type*}
  [Fintype Internal] [Fintype Sensory] [Fintype Active] [Fintype External]
  [MeasurableSpace Internal] [MeasurableSpace Sensory]
  [MeasurableSpace Active] [MeasurableSpace External]
  [DiscreteMeasurableSpace Internal] [DiscreteMeasurableSpace Sensory]
  [DiscreteMeasurableSpace Active] [DiscreteMeasurableSpace External]

/-- Every singleton coordinate rectangle has the exact native blanket
factorization inherited from the finite joint. -/
theorem fep138_staticJoint_rectangle_factorization
    (model : StaticModel Internal Sensory Active External)
    (blanket : Blanket Sensory Active) (internal : Internal)
    (external : External) :
    FEP.NativeBlanket.embeddedLaw (staticJoint model)
        (FEP.NativeBlanket.blanketCoordinate ⁻¹' {blanket} ∩
          FEP.NativeBlanket.internalCoordinate ⁻¹' {internal} ∩
          FEP.NativeBlanket.externalCoordinate ⁻¹' {external}) =
      ENNReal.ofReal (model.blanketLaw blanket) *
        ENNReal.ofReal (model.internalGiven blanket internal) *
          ENNReal.ofReal (model.externalGiven blanket external) :=
  FEP.NativeBlanket.staticJoint_rectangle_factorization
    model blanket internal external

end FEP138
```

13.138.2 Typeset statement signatures

$$\{\mathsf{Internal}\ \mathsf{Sensory}\ \mathsf{Active}\ \mathsf{External} : \mathsf{Type}^{*}\}$$
$$(\mathsf{model} : \mathsf{StaticModel}\ \mathsf{Internal}\ \mathsf{Sensory}\ \mathsf{Active}\ \mathsf{External})$$
$$(\mathsf{blanket} : \mathsf{Blanket}\ \mathsf{Sensory}\ \mathsf{Active})\ (\mathsf{internal} : \mathsf{Internal})\ (\mathsf{external} : \mathsf{External})$$
$$\mathsf{FEP.NativeBlanket.embeddedLaw}\ (\mathsf{staticJoint}\ \mathsf{model})$$
$$(\mathsf{FEP.NativeBlanket.blanketCoordinate}^{-1}\{\mathsf{blanket}\} \cap$$
$$\mathsf{FEP.NativeBlanket.internalCoordinate}^{-1}\{\mathsf{internal}\} \cap$$
$$\mathsf{FEP.NativeBlanket.externalCoordinate}^{-1}\{\mathsf{external}\}) = \mathsf{ENNReal.ofReal}$$
$$(\mathsf{model.blanketLaw}\ \mathsf{blanket}) \cdot \mathsf{ENNReal.ofReal}$$
$$(\mathsf{model.internalGiven}\ \mathsf{blanket}\ \mathsf{internal}) \cdot \mathsf{ENNReal.ofReal}$$
$$(\mathsf{model.externalGiven}\ \mathsf{blanket}\ \mathsf{external})$$

13.139 fep-139 — Native Markov-Blanket Conditional Independence

The embedded finite blanket factorization satisfies Mathlib's native CondIndepFun predicate for internal and external coordinates conditioned on the sensory-active blanket coordinate. Assumption scope: Internal and external carriers are nonempty; all carriers are finite, standard Borel through their discrete measurable structures, and the static model supplies exact normalized factorized rows. The proof identifies authored

kernels with native conditional distributions and does not infer the result from zero finite mutual information. Semantic disposition: `formalized`.

13.139.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP139

open FEP FEP.MarkovBlanket MeasureTheory ProbabilityTheory

variable {Internal Sensory Active External : Type*}
  [Fintype Internal] [Fintype Sensory] [Fintype Active] [Fintype External]
  [Nonempty Internal] [Nonempty External]
  [MeasurableSpace Internal] [MeasurableSpace Sensory]
  [MeasurableSpace Active] [MeasurableSpace External]
  [DiscreteMeasurableSpace Internal] [DiscreteMeasurableSpace Sensory]
  [DiscreteMeasurableSpace Active] [DiscreteMeasurableSpace External]

/-- The embedded static finite joint satisfies Mathlib's native conditional
independence predicate at the actual internal, external, and blanket maps. -/
theorem fep139_staticJoint_condIndepFun
    (model : StaticModel Internal Sensory Active External) :
    CondIndepFun
      (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate inferInstance)
      FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
      FEP.NativeBlanket.internalCoordinate
      FEP.NativeBlanket.externalCoordinate
      (FEP.NativeBlanket.embeddedLaw (staticJoint model)) :=
  FEP.NativeBlanket.staticJoint_condIndepFun model

/-- A concrete two-regime Boolean witness makes the native stop/go theorem
nonvacuous at the topic boundary. -/
theorem fep139_correlatedBlanket_nonvacuous :
    letI : MeasurableSpace Bool := ⊤
    CondIndepFun
        (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate inferInstance)
        FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
        FEP.NativeBlanket.internalCoordinate
        FEP.NativeBlanket.externalCoordinate
        (FEP.NativeBlanket.embeddedLaw
          (staticJoint FEP.NativeBlanket.correlatedBlanketModel)) ∧
      staticJoint FEP.NativeBlanket.correlatedBlanketModel
          (((false, false), false), false) = 1 / 2 ∧
      staticJoint FEP.NativeBlanket.correlatedBlanketModel
          (((true, true), true), true) = 1 / 2 ∧
      staticJoint FEP.NativeBlanket.correlatedBlanketModel
          (((false, true), false), false) ≠ 1 / 2 :=
  FEP.NativeBlanket.correlatedBlanket_nonvacuous

end FEP139
```

13.139.2 Typeset statement signatures

$$
\begin{aligned}
&\{\mathsf{Internal\ Sensory\ Active\ External} : \mathsf{Type}^{*}\}\\
&(\mathsf{model} : \mathsf{StaticModel\ Internal\ Sensory\ Active\ External})\\
&\mathsf{CondIndepFun}\ (\mathsf{MeasurableSpace.comap\ FEP.NativeBlanket.blanketCoordinate}\\
&\mathsf{inferInstance})\ \mathsf{FEP.NativeBlanket.blanketCoordinate\_measurable.comap\_le}\\
&\mathsf{FEP.NativeBlanket.internalCoordinate\ FEP.NativeBlanket.externalCoordinate}\\
&(\mathsf{FEP.NativeBlanket.embeddedLaw}\ (\mathsf{staticJoint\ model}))
\end{aligned}
$$

{Internal Sensory Active External : Type*}
letI : MeasurableSpaceBool

13.140 fep-140 — Conditional Independence under Measurable Coarsening

Native blanket conditional independence is preserved under measurable images of both internal and external coordinates. Assumption scope: The source blanket theorem retains its finite, discrete, nonempty endpoint hypotheses and both image maps are explicitly measurable. The result is Mathlib's composition law and does not cover transformations of the conditioning coordinate. Semantic disposition: `formalized`.

13.140.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP140

open FEP FEP.MarkovBlanket MeasureTheory ProbabilityTheory

variable {Internal Sensory Active External : Type*}
  [Fintype Internal] [Fintype Sensory] [Fintype Active] [Fintype External]
  [Nonempty Internal] [Nonempty External]
  [MeasurableSpace Internal] [MeasurableSpace Sensory]
  [MeasurableSpace Active] [MeasurableSpace External]
  [DiscreteMeasurableSpace Internal] [DiscreteMeasurableSpace Sensory]
  [DiscreteMeasurableSpace Active] [DiscreteMeasurableSpace External]

/-- Mathlib's composition law preserves the native blanket statement under
measurable coarsenings of both endpoint coordinates. -/
theorem fep140_condIndepFun_measurableImages
    {InternalImage ExternalImage : Type*}
    [MeasurableSpace InternalImage] [MeasurableSpace ExternalImage]
    (model : StaticModel Internal Sensory Active External)
    (internalImage : Internal → InternalImage)
    (externalImage : External → ExternalImage)
    (internalMeasurable : Measurable internalImage)
    (externalMeasurable : Measurable externalImage) :
    CondIndepFun
      (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate inferInstance)
      FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
      (internalImage ∘ FEP.NativeBlanket.internalCoordinate)
      (externalImage ∘ FEP.NativeBlanket.externalCoordinate)
      (FEP.NativeBlanket.embeddedLaw (staticJoint model)) :=
  (FEP.NativeBlanket.staticJoint_condIndepFun model).comp
    internalMeasurable externalMeasurable

end FEP140
```

13.140.2 Typeset statement signatures

$$
\begin{aligned}
&\{\mathsf{Internal}\ \mathsf{Sensory}\ \mathsf{Active}\ \mathsf{External} : \mathsf{Type}^{*}\}\\
&\{\mathsf{InternalImage}\ \mathsf{ExternalImage} : \mathsf{Type}^{*}\}\ [\mathsf{MeasurableSpaceInternalImage}]\\
&[\mathsf{MeasurableSpaceExternalImage}]\\
&(\mathsf{model} : \mathsf{StaticModel}\ \mathsf{Internal}\ \mathsf{Sensory}\ \mathsf{Active}\ \mathsf{External})\\
&(\mathsf{internalImage} : \mathsf{Internal} \Rightarrow \mathsf{InternalImage})\\
&(\mathsf{externalImage} : \mathsf{External} \Rightarrow \mathsf{ExternalImage})\\
&(\mathsf{internalMeasurable} : \mathsf{Measurable}\ \mathsf{internalImage})\\
&(\mathsf{externalMeasurable} : \mathsf{Measurable}\ \mathsf{externalImage})\\
&\mathsf{CondIndepFun}\ (\mathsf{MeasurableSpace.comap}\ \mathsf{FEP.NativeBlanket.blanketCoordinate}\\
&\mathsf{inferInstance})\ \mathsf{FEP.NativeBlanket.blanketCoordinate\_measurable.comap\_le}\\
&(\mathsf{internalImage} \circ \mathsf{FEP.NativeBlanket.internalCoordinate})\\
&(\mathsf{externalImage} \circ \mathsf{FEP.NativeBlanket.externalCoordinate})\\
&(\mathsf{FEP.NativeBlanket.embeddedLaw}\ (\mathsf{staticJoint}\ \mathsf{model}))
\end{aligned}
$$

13.141 fep-141 — Native Blanket-Transition Closure

Every authored factorized transition row induces a next-state embedded law whose internal and external coordinates are natively conditionally independent given its next blanket. Assumption scope: All carriers are finite and discretely measurable and internal and external carriers are nonempty. The theorem is rowwise for one supplied current state; arbitrary mixtures over current-state uncertainty require additional hypotheses and are not claimed. Semantic disposition: `formalized`.

13.141.1 Lean sketch

```
import FepSketches.native_blanket

namespace FEP141

open FEP FEP.MarkovBlanket MeasureTheory ProbabilityTheory

variable {Internal Sensory Active External : Type*}
  [Fintype Internal] [Fintype Sensory] [Fintype Active] [Fintype External]
  [Nonempty Internal] [Nonempty External]
  [MeasurableSpace Internal] [MeasurableSpace Sensory]
  [MeasurableSpace Active] [MeasurableSpace External]
  [DiscreteMeasurableSpace Internal] [DiscreteMeasurableSpace Sensory]
  [DiscreteMeasurableSpace Active] [DiscreteMeasurableSpace External]

/-- A factorized transition row induces a predicted native blanket law. -/
theorem fep141_prediction_preserves_nativeBlanket
    (dynamics : Dynamics Internal Sensory Active External)
    (current : DynamicState Internal Sensory Active External) :
    CondIndepFun
      (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate inferInstance)
      FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
      FEP.NativeBlanket.internalCoordinate
      FEP.NativeBlanket.externalCoordinate
      (FEP.NativeBlanket.embeddedLaw
        (staticJoint (nextStaticModel dynamics current))) :=
  FEP.NativeBlanket.prediction_preserves_nativeBlanket dynamics current

end FEP141
```

13.141.2 Typeset statement signatures

$$\{\text{Internal Sensory Active External} : \text{Type}^*\}$$
$$(\text{dynamics} : \text{Dynamics Internal Sensory Active External})\ (\text{current} : \text{DynamicState}$$
$$\text{Internal Sensory Active External})$$
$$\text{CondIndepFun}\ (\text{MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate}$$
$$\text{inferInstance})\ \text{FEP.NativeBlanket.blanketCoordinate\_measurable.comap\_le}$$
$$\text{FEP.NativeBlanket.internalCoordinate FEP.NativeBlanket.externalCoordinate}$$
$$(\text{FEP.NativeBlanket.embeddedLaw}\ (\text{staticJoint}\ (\text{nextStaticModel dynamics current})))$$

13.142 fep-142 — Finite Exponential-Family Normalization and Support

Strictly positive base weights normalize the finite scalar exponential family to a full-support probability law at every natural parameter. Assumption scope: The outcome carrier is finite and nonempty, base weights are strictly positive, and the sufficient statistic is real valued. No arbitrary manifold or multidimensional family is claimed. Semantic disposition: `formalized`.

13.142.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP142

open FEP FEP.ExponentialFamily Finset
open scoped BigOperators

variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

/-- Positive finite base weights normalize to a probability law at every
natural parameter. -/
theorem fep142_exponentialFamily_sum_one
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ) :
    ∑ outcome, family.law parameter outcome = 1 :=
  (family.law parameter).sum_one

/-- Strictly positive base weights give pointwise full support. -/
theorem fep142_exponentialFamily_pointwise_pos
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ)
    (outcome : Outcome) :
    0 < family.law parameter outcome :=
  family.law_pos parameter outcome

end FEP142
```

13.142.2 Typeset statement signatures

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\sum \text{outcome}, \text{family.law parameter outcome} = 1$$

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})\ (\text{outcome} : \text{Outcome})$$
$$0 < \text{family.law parameter outcome}$$

13.143 fep-143 — Affine Exponential-Family Log-Density Ratio

The supported log-density ratio between two parameters is affine in the sufficient statistic with the exact log-partition difference. Assumption scope: Finiteness, nonemptiness, and strict base-weight support make every logarithm ordinary and finite. The result is scalar and does not assert an extended-real identity at zero support. Semantic disposition: `formalized`.

13.143.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP143

open FEP FEP.ExponentialFamily

variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

/-- The supported log-density ratio is affine in the sufficient statistic. -/
theorem fep143_logDensityRatio_eq
    (family : ScalarExponentialFamily Outcome)
    (left right : ℝ) (outcome : Outcome) :
    Real.log
        (family.law left outcome / family.law right outcome) =
      (left - right) * family.statistic outcome -
        (family.logPartition left - family.logPartition right) :=
  family.logDensityRatio_eq left right outcome

end FEP143
```

13.143.2 Typeset statement signatures

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{left right} : \mathbb{R})\ (\text{outcome} : \text{Outcome})$$
$$\text{Real.log}\ (\text{family.law left outcome} / \text{family.law right outcome}) = (\text{left} - \text{right})$$
$$\cdot\ \text{family.statistic outcome}$$
$$-\ (\text{family.logPartition left} - \text{family.logPartition right})$$

13.144 fep-144 — Finite Log-Partition Gradient

The derivative of the finite scalar log-partition function equals the mean sufficient statistic. Assumption scope: The proof differentiates a finite sum of positive exponential weights and uses the positive partition denominator. It makes no infinite-sum exchange or general smooth-manifold claim. Semantic disposition: `formalized`.

13.144.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP144

open FEP FEP.ExponentialFamily

variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

/-- The derivative of the finite log-partition function is the expected
sufficient statistic. -/
theorem fep144_logPartition_hasDerivAt
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ) :
    HasDerivAt family.logPartition (family.mean parameter) parameter :=
  family.logPartition_hasDerivAt parameter

end FEP144
```

13.144.2 Typeset statement signatures

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\text{HasDerivAt family.logPartition}\ (\text{family.mean parameter})\ \text{parameter}$$

13.145 fep-145 — Centered Exponential-Family Score Identity

The scalar score is the sufficient statistic centered by its family mean and therefore has exactly zero expectation. Assumption scope: The score is defined on the finite full-support exponential family. This is a natural-parameter derivative identity, not a general score-regularity theorem for arbitrary likelihoods. Semantic disposition: `formalized`.

13.145.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP145

open FEP FEP.ExponentialFamily Finset
open scoped BigOperators

variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

omit [Nonempty Outcome] in
/-- Differentiating log density centers the sufficient statistic. -/
theorem fep145_score_eq_statistic_sub_mean
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ)
    (outcome : Outcome) :
    family.score parameter outcome =
      family.statistic outcome - family.mean parameter :=
  family.score_eq_statistic_sub_mean parameter outcome

/-- The normalized finite score has zero expectation. -/
theorem fep145_score_mean_zero
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ) :
    ∑ outcome,
      family.law parameter outcome * family.score parameter outcome = 0 :=
  family.mean_score_zero parameter

end FEP145
```

13.145.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ [\mathsf{Nonempty}\ \mathsf{Outcome}]$$
$$(\mathsf{family} : \mathsf{ScalarExponentialFamily}\ \mathsf{Outcome})\ (\mathsf{parameter} : \mathbb{R})\ (\mathsf{outcome} : \mathsf{Outcome})$$
$$\mathsf{family.score}\ \mathsf{parameter}\ \mathsf{outcome} = \mathsf{family.statistic}\ \mathsf{outcome} - \mathsf{family.mean}$$
$$\mathsf{parameter}$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ [\mathsf{Nonempty}\ \mathsf{Outcome}]$$
$$(\mathsf{family} : \mathsf{ScalarExponentialFamily}\ \mathsf{Outcome})\ (\mathsf{parameter} : \mathbb{R})$$
$$\sum \mathsf{outcome}, \mathsf{family.law}\ \mathsf{parameter}\ \mathsf{outcome} \cdot \mathsf{family.score}\ \mathsf{parameter}\ \mathsf{outcome} = 0$$

13.146 fep-146 — Log-Partition Hessian–Variance–Fisher Identity

Differentiating the log-partition mean coordinate gives statistic variance, which equals scalar Fisher information; a concrete three-state family has variance two thirds at zero. Assumption scope: The result is the derivative of the already-proved first-derivative mean coordinate on a finite scalar family. Strict positivity is witnessed for a named nonconstant family and is not inferred for constant statistics. Semantic disposition: `formalized`.

13.146.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP146

open FEP FEP.ExponentialFamily
```

```
variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

/-- Since the first log-partition derivative is the mean coordinate, its
second derivative is the sufficient-statistic variance. -/
theorem fep146_logPartition_secondDeriv_eq_variance
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ) :
    HasDerivAt family.mean (family.variance parameter) parameter :=
  family.mean_hasDerivAt parameter

/-- Scalar Fisher information is exactly the same centered variance. -/
theorem fep146_fisher_eq_variance
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ) :
    family.fisher parameter = family.variance parameter :=
  family.fisher_eq_variance parameter

/-- The explicit nonconstant three-state statistic has strictly positive
variance at the origin. -/
theorem fep146_threeState_variance_positive :
    0 < ScalarExponentialFamily.threeStateFamily.variance 0 :=
  ScalarExponentialFamily.threeState_variance_zero_pos

/-- A constant statistic is the exact zero-Fisher boundary. -/
theorem fep146_constantStatistic_zero_boundary
    (base : Outcome → ℝ) (hBase : ∀ outcome, 0 < base outcome)
    (constant parameter : ℝ) :
    (ScalarExponentialFamily.constantStatisticFamily base hBase constant).fisher
        parameter = 0 := by
  rw [ScalarExponentialFamily.fisher_eq_variance,
    ScalarExponentialFamily.constantStatistic_variance_zero]

end FEP146
```

13.146.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ [\mathsf{Nonempty}\ \mathsf{Outcome}]$$
$$(\mathsf{family} : \mathsf{ScalarExponentialFamily}\ \mathsf{Outcome})\ (\mathsf{parameter} : \mathbb{R})$$
$$\mathsf{HasDerivAt}\ \mathsf{family.mean}\ (\mathsf{family.variance}\ \mathsf{parameter})\ \mathsf{parameter}$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ [\mathsf{Nonempty}\ \mathsf{Outcome}]$$
$$(\mathsf{family} : \mathsf{ScalarExponentialFamily}\ \mathsf{Outcome})\ (\mathsf{parameter} : \mathbb{R})$$
$$\mathsf{family.fisher}\ \mathsf{parameter} = \mathsf{family.variance}\ \mathsf{parameter}$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ [\mathsf{Nonempty}\ \mathsf{Outcome}]$$
$$0 < \mathsf{ScalarExponentialFamily.threeStateFamily.variance}\ 0$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Outcome}]\ [\mathsf{Nonempty}\ \mathsf{Outcome}]$$
$$(\mathsf{base} : \mathsf{Outcome} \Rightarrow \mathbb{R})\ (\mathsf{hBase} : \forall \mathsf{outcome}, 0 < \mathsf{base}\ \mathsf{outcome})$$
$$(\mathsf{constant}\ \mathsf{parameter} : \mathbb{R})$$
$$(\mathsf{ScalarExponentialFamily.constantStatisticFamily}\ \mathsf{base}\ \mathsf{hBase}\ \mathsf{constant}).\mathsf{fisher}$$
$$\mathsf{parameter} = 0$$

13.147 fep-147 — Exponential-Family KL–Bregman Identity

Finite KL between two members of the same supported scalar exponential family equals the oriented Bregman divergence of the log-partition potential. Assumption scope: The carrier is finite and nonempty and strict base support supplies the logarithmic KL identity. The project finiteKL convention is totalized, but this theorem stays inside its ordinary full-support regime. Semantic disposition: `formalized`.

13.147.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP147

open FEP FEP.ExponentialFamily FEP.FiniteInformation

variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

/-- Full-support finite KL equals the Bregman divergence of the
log-partition potential. -/
theorem fep147_exponentialFamily_KL_eq_bregman
    (family : ScalarExponentialFamily Outcome) (left right : ℝ) :
    finiteKL (family.law left) (family.law right) =
      family.logPartitionBregman left right :=
  family.finiteKL_eq_logPartitionBregman left right

/-- The support premise used by the logarithmic KL representation is
constructively available at every atom. -/
theorem fep147_exponentialFamily_fullSupport
    (family : ScalarExponentialFamily Outcome) (parameter : ℝ)
    (outcome : Outcome) :
    0 < family.law parameter outcome :=
  family.law_pos parameter outcome

end FEP147
```

13.147.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Outcome}]\ [\mathsf{Nonempty\ Outcome}]\ (\mathsf{family} : \mathsf{ScalarExponentialFamily\ Outcome})\ (\mathsf{left\ right} : \mathbb{R})\ \mathsf{finiteKL}\ (\mathsf{family.law\ left})\ (\mathsf{family.law\ right}) = \mathsf{family.logPartitionBregman\ left\ right}$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Outcome}]\ [\mathsf{Nonempty\ Outcome}]\ (\mathsf{family} : \mathsf{ScalarExponentialFamily\ Outcome})\ (\mathsf{parameter} : \mathbb{R})\ (\mathsf{outcome} : \mathsf{Outcome})\ 0 < \mathsf{family.law\ parameter\ outcome}$$

13.148 fep-148 — Natural-to-Mean Coordinate Injectivity

Positive statistic variance throughout a closed interval makes the natural-to-mean coordinate map strictly monotone and injective there. Assumption scope: Variance positivity is an explicit interval premise. The theorem is local to a stated scalar interval and does not assert global dual-flat coordinates or positivity for degenerate statistics. Semantic disposition: `formalized`.

13.148.1 Lean sketch

```
import FepSketches.exponential_family

namespace FEP148

open FEP FEP.ExponentialFamily

variable {Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]

/-- Positive variance throughout a closed interval makes the natural-to-mean
coordinate map strictly monotone there. -/
theorem fep148_meanParameter_strictMono
    (family : ScalarExponentialFamily Outcome) {lower upper : ℝ}
    (hVariance : ∀ parameter ∈ Set.Icc lower upper,
      0 < family.variance parameter) :
```

```
    StrictMonoOn family.mean (Set.Icc lower upper) :=
  family.meanParameter_strictMono hVariance

/-- Hence the mean coordinate is injective on the same stated interval. -/
theorem fep148_meanParameter_injective
    (family : ScalarExponentialFamily Outcome) {lower upper : ℝ}
    (hVariance : ∀ parameter ∈ Set.Icc lower upper,
      0 < family.variance parameter) :
    Set.InjOn family.mean (Set.Icc lower upper) :=
  family.meanParameter_injectiveOn hVariance

end FEP148
```

13.148.2 Typeset statement signatures

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Outcome}]\ [\mathsf{Nonempty\ Outcome}]$$
$$(\mathsf{family} : \mathsf{ScalarExponentialFamily\ Outcome})\ \{\mathsf{lower\ upper} : \mathbb{R}\}\ (\mathsf{hVariance} : \forall$$
$$\mathsf{parameter} \in \mathsf{Set.Icc\ lower\ upper}, 0 < \mathsf{family.variance\ parameter})$$
$$\mathsf{StrictMonoOn\ family.mean}\ (\mathsf{Set.Icc\ lower\ upper})$$

$$\{\mathsf{Outcome} : \mathsf{Type}^*\}\ [\mathsf{Fintype\ Outcome}]\ [\mathsf{Nonempty\ Outcome}]$$
$$(\mathsf{family} : \mathsf{ScalarExponentialFamily\ Outcome})\ \{\mathsf{lower\ upper} : \mathbb{R}\}\ (\mathsf{hVariance} : \forall$$
$$\mathsf{parameter} \in \mathsf{Set.Icc\ lower\ upper}, 0 < \mathsf{family.variance\ parameter})$$
$$\mathsf{Set.InjOn\ family.mean}\ (\mathsf{Set.Icc\ lower\ upper})$$

13.149 fep-149 — Two-State Continuous-Time Markov Kernel

Positive two-state rates define a nonnegative transition matrix with every row summing to one for nonnegative time. Assumption scope: The state carrier is Boolean, both rates are strictly positive, and time is nonnegative. This is an exact two-state kernel rather than a general continuous-time Markov-chain construction. Semantic disposition: `formalized`.

13.149.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP149

open FEP FEP.ContinuousTimeMarkov Finset
open scoped BigOperators

/-- Every row of the exact two-state transition function has unit mass. -/
theorem fep149_twoStateSemigroup_rowSum
    (rates : TwoStateRates) (time : ℝ) (source : Bool) :
    ∑ target, rates.transition time source target = 1 :=
  rates.transition_rowSum time source

/-- Positive rates and nonnegative time make every entry nonnegative. -/
theorem fep149_twoStateSemigroup_nonnegative
    (rates : TwoStateRates) {time : ℝ} (hTime : 0 ≤ time)
    (source target : Bool) :
    0 ≤ rates.transition time source target :=
  rates.transition_nonneg hTime source target

/-- The plot-ready case uses exactly the requested rates. -/
theorem fep149_benchmarkRates_exact :
    TwoStateRates.benchmarkRates.forward = 0.7 ∧
      TwoStateRates.benchmarkRates.backward = 0.3 :=
  TwoStateRates.benchmarkRates_exact

end FEP149
```

13.149.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{time} : \mathbb{R})\ (\mathsf{source} : \mathsf{Bool})$$
$$\sum \mathsf{target},\ \mathsf{rates.transition\ time\ source\ target} = 1$$

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ \{\mathsf{time} : \mathbb{R}\}\ (\mathsf{hTime} : 0 \le \mathsf{time})\ (\mathsf{source\ target} : \mathsf{Bool})$$
$$0 \le \mathsf{rates.transition\ time\ source\ target}$$

$$\mathsf{TwoStateRates.benchmarkRates.forward} = 0.7$$
$$\wedge\ \mathsf{TwoStateRates.benchmarkRates.backward} = 0.3$$

13.150 fep-150 — Continuous-Time Identity at Zero

The exact two-state continuous-time transition matrix is the identity at time zero. Assumption scope: Both rates remain strictly positive through the maintained rate structure. The equality is entrywise on the Boolean carrier. Semantic disposition: `formalized`.

13.150.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP150

open FEP.ContinuousTimeMarkov

/-- At zero time the exact transition is the identity matrix. -/
theorem fep150_twoStateSemigroup_zero
    (rates : TwoStateRates) (source target : Bool) :
    rates.transition 0 source target = if source = target then 1 else 0 :=
  rates.transition_zero source target

end FEP150
```

13.150.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{source\ target} : \mathsf{Bool})$$
$$\mathsf{rates.transition}\ 0\ \mathsf{source\ target} = \mathsf{if\ source} = \mathsf{target\ then}\ 1\ \mathsf{else}\ 0$$

13.151 fep-151 — Two-State Chapman–Kolmogorov Semigroup

The exact two-state transition matrices satisfy the Chapman–Kolmogorov additive semigroup law entrywise. Assumption scope: Both times are nonnegative and rates are strictly positive. Composition is the finite Boolean matrix sum and no path-space or general generator existence theorem is inferred. Semantic disposition: `formalized`.

13.151.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP151

open FEP.ContinuousTimeMarkov Finset
open scoped BigOperators

/-- The exact transition function obeys Chapman--Kolmogorov at all real
times; its probability-kernel interpretation uses nonnegative times. -/
theorem fep151_twoStateSemigroup_add
    (rates : TwoStateRates) (left right : ℝ) (source target : Bool) :
    rates.transition (left + right) source target =
      Σ middle,
        rates.transition left source middle *
          rates.transition right middle target :=
  rates.transition_add left right source target
```

```
end FEP151
```

13.151.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{left}\ \mathsf{right} : \mathbb{R})\ (\mathsf{source}\ \mathsf{target} : \mathsf{Bool})$$
$$\mathsf{rates.transition}\ (\mathsf{left} + \mathsf{right})\ \mathsf{source}\ \mathsf{target} = \sum \mathsf{middle},\ \mathsf{rates.transition}\ \mathsf{left}$$
$$\mathsf{source}\ \mathsf{middle} \cdot \mathsf{rates.transition}\ \mathsf{right}\ \mathsf{middle}\ \mathsf{target}$$

13.152 fep-152 — Two-State Continuous-Time Master Equation

Every transition entry is differentiable and its derivative equals both the generator-times-transition and transition-times-generator finite sums. Assumption scope: The generator and transition matrices are the maintained explicit Boolean formulas. This proves the forward and backward master equations for one two-state semigroup, not a general CTMC existence theorem. Semantic disposition: `formalized`.

13.152.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP152

open FEP.ContinuousTimeMarkov Finset
open scoped BigOperators

/-- The exact two-state transition derivative equals both generator products
entrywise. -/
theorem fep152_twoStateSemigroup_hasDerivAt
    (rates : TwoStateRates) (time : ℝ) (source target : Bool) :
    HasDerivAt (fun candidate ↦ rates.transition candidate source target)
        (∑ middle,
          rates.generator source middle * rates.transition time middle target)
        time ∧
      HasDerivAt (fun candidate ↦ rates.transition candidate source target)
        (∑ middle,
          rates.transition time source middle * rates.generator middle target)
        time :=
  rates.transition_masterEquation time source target

end FEP152
```

13.152.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{time} : \mathbb{R})\ (\mathsf{source}\ \mathsf{target} : \mathsf{Bool})$$
$$\mathsf{HasDerivAt}\ (\lambda\ \mathsf{candidate} \mapsto \mathsf{rates.transition}\ \mathsf{candidate}\ \mathsf{source}\ \mathsf{target})\ (\sum \mathsf{middle},$$
$$\mathsf{rates.generator}\ \mathsf{source}\ \mathsf{middle} \cdot \mathsf{rates.transition}\ \mathsf{time}\ \mathsf{middle}\ \mathsf{target})\ \mathsf{time}$$
$$\wedge\ \mathsf{HasDerivAt}\ (\lambda\ \mathsf{candidate} \mapsto \mathsf{rates.transition}\ \mathsf{candidate}\ \mathsf{source}\ \mathsf{target})\ (\sum$$
$$\mathsf{middle},\ \mathsf{rates.transition}\ \mathsf{time}\ \mathsf{source}\ \mathsf{middle} \cdot \mathsf{rates.generator}\ \mathsf{middle}\ \mathsf{target})$$
$$\mathsf{time}$$

13.153 fep-153 — Continuous-Time Stationarity and Detailed Balance

The explicit stationary law is invariant under every nonnegative-time transition and satisfies detailed balance entrywise. Assumption scope: Both rates are strictly positive and time is nonnegative. Reversibility is proved only for this exact two-state family and does not cover driven or nonequilibrium generators. Semantic disposition: `formalized`.

13.153.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP153
```

```
open FEP FEP.ContinuousTimeMarkov Finset
open scoped BigOperators

/-- The closed-form stationary law is invariant at every time. -/
theorem fep153_twoStateSemigroup_stationary
    (rates : TwoStateRates) (time : ℝ) (target : Bool) :
    (∑ source,
        rates.stationaryLaw source * rates.transition time source target) =
      rates.stationaryLaw target :=
  rates.transition_stationary time target

/-- The same law satisfies entrywise detailed balance. -/
theorem fep153_twoStateSemigroup_detailedBalance
    (rates : TwoStateRates) (time : ℝ) (source target : Bool) :
    rates.stationaryLaw source * rates.transition time source target =
      rates.stationaryLaw target * rates.transition time target source :=
  rates.transition_detailedBalance time source target

end FEP153
```

13.153.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{time} : \mathbb{R})\ (\mathsf{target} : \mathsf{Bool})$$
$$\left(\sum \mathsf{source},\ \mathsf{rates.stationaryLaw\ source} \cdot \mathsf{rates.transition\ time\ source\ target}\right)$$
$$= \mathsf{rates.stationaryLaw\ target}$$

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{time} : \mathbb{R})\ (\mathsf{source\ target} : \mathsf{Bool})$$
$$\mathsf{rates.stationaryLaw\ source} \cdot \mathsf{rates.transition\ time\ source\ target}$$
$$= \mathsf{rates.stationaryLaw\ target} \cdot \mathsf{rates.transition\ time\ target\ source}$$

13.154 fep-154 — Exact Two-State Exponential Relaxation

The true-state mass of any initial Boolean law relaxes toward stationarity by the exact factor exp(-(a+b)t). Assumption scope: Rates are positive, time is nonnegative, and prediction uses the explicit two-state transition. This is exact finite-state relaxation, not a mixing theorem for arbitrary chains. Semantic disposition: `formalized`.

13.154.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP154

open FEP FEP.ContinuousTimeMarkov

/-- Every initial two-state law relaxes exactly at rate `a+b`. -/
theorem fep154_twoStateRelaxation_exact
    (rates : TwoStateRates) (initial : FiniteLaw Bool) (time : ℝ) :
    rates.trueMass initial time - rates.stationaryTrue =
      rates.rho time * (initial true - rates.stationaryTrue) :=
  rates.relaxation_exact initial time

/-- The plot-ready initial point mass is genuinely nonstationary. -/
theorem fep154_benchmarkInitial_nonstationary :
    TwoStateRates.benchmarkInitial true ≠
      TwoStateRates.benchmarkRates.stationaryTrue :=
  TwoStateRates.benchmarkInitial_nonstationary

end FEP154
```

13.154.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{initial} : \mathsf{FiniteLaw}\ \mathsf{Bool})\ (\mathsf{time} : \mathbb{R})$$
$$\mathsf{rates.trueMass}\ \mathsf{initial}\ \mathsf{time} - \mathsf{rates.stationaryTrue} = \mathsf{rates.rho}\ \mathsf{time}$$
$$\cdot\ (\mathsf{initial}\ \mathsf{true} - \mathsf{rates.stationaryTrue})$$

$$\mathsf{TwoStateRates.benchmarkInitial}\ \mathsf{true}$$
$$\neq \mathsf{TwoStateRates.benchmarkRates.stationaryTrue}$$

13.155 fep-155 — Exact Two-State Quadratic Lyapunov Decay

Squared deviation from stationarity decays by exp(-2(a+b)t), and its exact derivative is minus twice the decay rate times the Lyapunov value. Assumption scope: The Lyapunov function is the scalar square of the true-state mass deviation for the explicit two-state semigroup. It is not a general entropy-production or physical free-energy theorem. Semantic disposition: `formalized`.

13.155.1 Lean sketch

```
import FepSketches.continuous_time_markov

namespace FEP155

open FEP FEP.ContinuousTimeMarkov

/-- Squared deviation from stationarity decays by the exact squared
semigroup factor. -/
theorem fep155_twoStateLyapunov_exact
    (rates : TwoStateRates) (initial : FiniteLaw Bool) (time : ℝ) :
    rates.lyapunov initial time =
      rates.rho time ^ 2 *
        (initial true - rates.stationaryTrue) ^ 2 :=
  rates.lyapunov_exact initial time

/-- Its derivative is exactly `-2(a+b)V(t)`. -/
theorem fep155_twoStateLyapunov_hasDerivAt
    (rates : TwoStateRates) (initial : FiniteLaw Bool) (time : ℝ) :
    HasDerivAt (rates.lyapunov initial)
      (-2 * rates.decayRate * rates.lyapunov initial time) time :=
  rates.lyapunov_hasDerivAt initial time

/-- For rates `(0.7,0.3)` and a nonstationary point mass, the Lyapunov
derivative is strictly negative at time zero. -/
theorem fep155_benchmarkLyapunov_strictlyDecreasing :
    -2 * TwoStateRates.benchmarkRates.decayRate *
        TwoStateRates.benchmarkRates.lyapunov
          TwoStateRates.benchmarkInitial 0 < 0 :=
  TwoStateRates.benchmarkLyapunov_deriv_zero_neg

end FEP155
```

13.155.2 Typeset statement signatures

$$(\mathsf{rates} : \mathsf{TwoStateRates})\,(\mathsf{initial} : \mathsf{FiniteLaw\,Bool})\,(\mathsf{time} : \mathbb{R})$$
$$\mathsf{rates.lyapunov\,initial\,time} = \mathsf{rates.rho\,time}^2$$
$$\cdot\,(\mathsf{initial\,true} - \mathsf{rates.stationaryTrue})^2$$

$$(\mathsf{rates} : \mathsf{TwoStateRates})\,(\mathsf{initial} : \mathsf{FiniteLaw\,Bool})\,(\mathsf{time} : \mathbb{R})$$
$$\mathsf{HasDerivAt}\,(\mathsf{rates.lyapunov\,initial})$$
$$(-2 \cdot \mathsf{rates.decayRate} \cdot \mathsf{rates.lyapunov\,initial\,time})\,\mathsf{time}$$

$$-2 \cdot \mathsf{TwoStateRates.benchmarkRates.decayRate}$$
$$\cdot\,\mathsf{TwoStateRates.benchmarkRates.lyapunov\,TwoStateRates.benchmarkInitial}\,0 < 0$$

14 Appendix C: Typeset Equations

14.1 fep-001 — Variational Free Energy Bound

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{approximation\ posterior} : \mathsf{Measure}\ \alpha)\ (\mathsf{surprisal} : \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{surprisal} \leq \mathsf{fep001\_variationalUpperBound\ approximation\ posterior\ surprisal}$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{approximation\ posterior} : \mathsf{Measure}\ \alpha)\ [\mathsf{IsFiniteMeasure\ approximation}]$$
$$[\mathsf{IsFiniteMeasure\ posterior}]$$
$$\mathsf{fep001\_variationalGap\ approximation\ posterior} = 0 \leftrightarrow \mathsf{approximation} = \mathsf{posterior}$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{approximation\ posterior} : \mathsf{Measure}\ \alpha)\ [\mathsf{IsFiniteMeasure\ approximation}]$$
$$[\mathsf{IsFiniteMeasure\ posterior}]\ \{\mathsf{surprisal} : \mathbb{R}^{\infty}_{\geq 0}\}\ (\mathsf{hsurprisal} : \mathsf{surprisal} \neq \infty)$$
$$\mathsf{fep001\_variationalUpperBound\ approximation\ posterior\ surprisal} = \mathsf{surprisal}$$
$$\leftrightarrow \mathsf{approximation} = \mathsf{posterior}$$

14.2 fep-002 — Variational Evidence Bound via KL Divergence

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ [\mathsf{IsProbabilityMeasure}\ \mu]$$
$$\mu\,\Omega = 1$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ [\mathsf{IsProbabilityMeasure}\ \mu]\ \{s : \mathsf{Set}\ \alpha\}\ (\mathsf{hs} : \mathsf{MeasurableSet} s)$$
$$(\mathsf{hfin} : \mu\, s \neq \top)$$
$$\mu\, s^{\mathsf{c}} = 1 - \mu\, s$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{logEvidence\ freeEnergy\ klDiv} : \mathbb{R})\ (\mathsf{h\_decomp} : \mathsf{logEvidence} = \mathsf{freeEnergy} + \mathsf{klDiv})$$
$$(\mathsf{h\_kl\_nonneg} : 0 \leq \mathsf{klDiv})$$
$$\mathsf{freeEnergy} \leq \mathsf{logEvidence}$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(q\ \mathsf{posterior} : \mathsf{Measure}\ \alpha)\ (\mathsf{surprisal} : \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{surprisal} \leq \mathsf{fep002\_variationalFreeEnergy}\ q\ \mathsf{posterior\ surprisal}$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mathsf{posterior} : \mathsf{Measure}\ \alpha)\ [\mathsf{SigmaFinite\ posterior}]\ (\mathsf{surprisal} : \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{fep002\_variationalFreeEnergy\ posterior\ posterior\ surprisal} = \mathsf{surprisal}$$

14.3 fep-003 — Discounted Pragmatic Cost

$$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{stageCost} : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0})$$
$$\mathsf{fep003\_pragmaticCost\ discount\ stageCost}\ 0 = 0$$

$$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{stageCost} : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{horizon} : \mathbb{N})$$
$$\mathsf{fep003\_pragmaticCost\ discount\ stageCost}\ (\mathsf{horizon} + 1) = \mathsf{fep003\_pragmaticCost}$$
$$\mathsf{discount\ stageCost\ horizon} + \mathsf{discount}^{\mathsf{horizon}} \cdot \mathsf{stageCost\ horizon}$$

$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\ \{cost_1\ cost_2 : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0}\}$
$(\mathsf{hcost} : \forall t, cost_1\ t \leq cost_2\ t)\ (\mathsf{horizon} : \mathbb{N})$
$\mathsf{fep003\_pragmaticCost}\ \mathsf{discount}\ cost_1\ \mathsf{horizon} \leq \mathsf{fep003\_pragmaticCost}\ \mathsf{discount}$
$cost_2\ \mathsf{horizon}$

$(\mathsf{discount} : \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{stageCost} : \mathbb{N} \Rightarrow \mathbb{R}^{\infty}_{\geq 0})\ (\mathsf{horizon} : \mathbb{N})$
$\mathsf{fep003\_pragmaticCost}\ \mathsf{discount}\ \mathsf{stageCost}\ \mathsf{horizon} \leq \mathsf{fep003\_pragmaticCost}\ \mathsf{discount}$
$\mathsf{stageCost}\ (\mathsf{horizon} + 1)$

14.4 fep-004 — Finite Diagonal Fisher Metric

$\{\iota : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{information}\ u\ v : \iota \Rightarrow \mathbb{R})$
$\mathsf{fep004\_fisherMetric}\ \mathsf{information}\ u\ v = \mathsf{fep004\_fisherMetric}\ \mathsf{information}\ v\ u$

$\{\iota : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{information}\ v : \iota \Rightarrow \mathbb{R})$
$(\mathsf{hinformation} : \forall i, 0 \leq \mathsf{information}\ i)$
$0 \leq \mathsf{fep004\_fisherMetric}\ \mathsf{information}\ v\ v$

$\{\iota : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{information}\ v : \iota \Rightarrow \mathbb{R})$
$(\mathsf{hinformation} : \forall i, 0 < \mathsf{information}\ i)$
$\mathsf{fep004\_fisherMetric}\ \mathsf{information}\ v\ v = 0 \leftrightarrow \forall i, v\ i = 0$

14.5 fep-005 — Four-Block State Partition

$(\mathsf{assign} : \mathsf{Fin}\ 20 \Rightarrow \mathsf{BlkPart})\ (i\ j : \mathsf{BlkPart})\ (\mathsf{hij} : i \neq j)$
$\mathsf{Disjoint}\ (\mathsf{fep005\_partitionCover}\ \mathsf{assign}\ i)\ (\mathsf{fep005\_partitionCover}\ \mathsf{assign}\ j)$

$(\mathsf{assign} : \mathsf{Fin}\ 20 \Rightarrow \mathsf{BlkPart})$
$\forall s : \mathsf{Fin}\ 20, \exists k : \mathsf{BlkPart}, s \in \mathsf{fep005\_partitionCover}\ \mathsf{assign}\ k$

$(\mathsf{assign} : \mathsf{Fin}\ 20 \Rightarrow \mathsf{BlkPart})\ (s : \mathsf{Fin}\ 20)\ (k : \mathsf{BlkPart})$
$s \in \mathsf{fep005\_partitionCover}\ \mathsf{assign}\ k \leftrightarrow \mathsf{assign}\ s = k$

$(\mathsf{assign} : \mathsf{Fin}\ 20 \Rightarrow \mathsf{BlkPart})\ (s : \mathsf{Fin}\ 20)$
$\exists! k : \mathsf{BlkPart}, s \in \mathsf{fep005\_partitionCover}\ \mathsf{assign}\ k$

14.6 fep-006 — Measurable Discrete-Time Flow

$\{\alpha : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ \{\mathsf{step} : \alpha \Rightarrow \alpha\}\ (\mathsf{hstep} : \mathsf{Measurable}\ \mathsf{step})\ (\mathsf{time} : \mathbb{N})$
$\mathsf{Measurable}\ (\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ \mathsf{time})$

$\{\alpha : \mathsf{Type}^*\}$
$(\mathsf{step} : \alpha \Rightarrow \alpha)$
$\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ 0 = \mathsf{id}$

$\{\alpha : \mathsf{Type}^*\}$
$(\mathsf{step} : \alpha \Rightarrow \alpha)\ (m\ n : \mathbb{N})\ (\mathsf{state} : \alpha)$
$\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ (m + n)\ \mathsf{state} = \mathsf{fep006\_iterateFlow}\ \mathsf{step}\ m$
$(\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ n\ \mathsf{state})$

$\{\alpha : \mathsf{Type}^*\}$
$(\mathsf{step} : \alpha \Rightarrow \alpha)\ \{\mathsf{state} : \alpha\}\ (\mathsf{hfixed} : \mathsf{Function.IsFixedPt}\ \mathsf{step}\ \mathsf{state})\ (\mathsf{time} : \mathbb{N})$
$\mathsf{fep006\_iterateFlow}\ \mathsf{step}\ \mathsf{time}\ \mathsf{state} = \mathsf{state}$

14.7 fep-007 — Normalized Finite Sum-Product Message

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (i\,j : \mathsf{Node})\ (h\psi : 0 \le \psi\,i\,j)\ (\mathsf{hj} : 0 \le \psi\,j\,i)$$
$$0 \le \psi\,i\,j \cdot \psi\,j\,i$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{msg} : \mathsf{Node} \Rightarrow \mathbb{R})\ (N : \mathsf{Finset}\ \mathsf{Node})\ (i : \mathsf{Node})$$
$$(h\psi : \forall j, 0 \le \psi\,i\,j)\ (\mathsf{hm} : \forall j, 0 \le \mathsf{msg}\,j)$$
$$0 \le \sum j \in N, \psi\,i\,j \cdot \mathsf{msg}\,j$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (m_1\,m_2 : \mathsf{Node} \Rightarrow \mathbb{R})\ (N : \mathsf{Finset}\ \mathsf{Node})\ (i : \mathsf{Node})$$
$$(h\psi : \forall j, 0 \le \psi\,i\,j)\ (h : \forall j \in N, m_1\,j \le m_2\,j)$$
$$\sum j \in N, \psi\,i\,j \cdot m_1\,j \le \sum j \in N, \psi\,i\,j \cdot m_2\,j$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (N : \mathsf{Finset}\ \mathsf{Node})\ (i : \mathsf{Node})$$
$$\sum j \in N, \psi\,i\,j \cdot (0 : \mathbb{R}) = 0$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{incoming} : \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{neighbors} : \mathsf{Finset}\ \mathsf{Node})\ (i : \mathsf{Node})$$
$$(\mathsf{hne} : \mathsf{neighbors.Nonempty})\ (h\psi : \forall j \in \mathsf{neighbors}, 0 < \psi\,i\,j)$$
$$(\mathsf{hincoming} : \forall j \in \mathsf{neighbors}, 0 < \mathsf{incoming}\,j)$$
$$0 < \mathsf{fep007\_messageNormalizer}\,\psi\,\mathsf{incoming}\,\mathsf{neighbors}\,i$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{incoming} : \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{neighbors} : \mathsf{Finset}\ \mathsf{Node})$$
$$(i\,j : \mathsf{Node})\ (\mathsf{hne} : \mathsf{neighbors.Nonempty})\ (h\psi : \forall k \in \mathsf{neighbors}, 0 < \psi\,i\,k)$$
$$(\mathsf{hincoming} : \forall k \in \mathsf{neighbors}, 0 < \mathsf{incoming}\,k)$$
$$0 \le \mathsf{fep007\_normalizedMessage}\,\psi\,\mathsf{incoming}\,\mathsf{neighbors}\,i\,j$$

$$(\psi : \mathsf{Node} \Rightarrow \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{incoming} : \mathsf{Node} \Rightarrow \mathbb{R})\ (\mathsf{neighbors} : \mathsf{Finset}\ \mathsf{Node})\ (i : \mathsf{Node})$$
$$(\mathsf{hne} : \mathsf{neighbors.Nonempty})\ (h\psi : \forall j \in \mathsf{neighbors}, 0 < \psi\,i\,j)$$
$$(\mathsf{hincoming} : \forall j \in \mathsf{neighbors}, 0 < \mathsf{incoming}\,j)$$
$$\sum j \in \mathsf{neighbors}, \mathsf{fep007\_normalizedMessage}\,\psi\,\mathsf{incoming}\,\mathsf{neighbors}\,i\,j = 1$$

14.8 fep-008 — Finite Policy Objective Minimizer

$$(\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (\mathsf{hne} : \mathsf{policies.Nonempty})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})$$
$$\exists p \in \mathsf{policies}, \forall p' \in \mathsf{policies}, G\,p \le G\,p'$$

$$(\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ ({}_h ne : \mathsf{policies.Nonempty})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})$$
$$(p\,p' : \mathsf{Policy})\ (\mathsf{hp} : p \in \mathsf{policies})\ (\mathsf{hp}' : p' \in \mathsf{policies})$$
$$(\mathsf{hmin} : \forall p'' \in \mathsf{policies}, G\,p \le G\,p'')\ (\mathsf{hmin}' : \forall p'' \in \mathsf{policies}, G\,p' \le G\,p'')$$
$$G\,p = G\,p'$$

$$(\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (\mathsf{hne} : \mathsf{policies.Nonempty})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})$$
$$\exists p \in \mathsf{policies}, \forall p' \in \mathsf{policies}, G\,p' \le G\,p$$

$$(\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (p : \mathsf{Policy})\ ({}_h p : p \in \mathsf{policies})$$
$$(\mathsf{hmin} : \forall p' \in \mathsf{policies}, G\,p \le G\,p')$$
$$\forall p' \in \mathsf{policies}, G\,p \le G\,p'$$

14.9 fep-009 — Conditional Independence Laws

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$(m'm_1\,m_2 : \mathsf{MeasurableSpace}\Omega)\ (\mathsf{hm}' : m' \le m\Omega)\ (\mu : @\mathsf{Measure}\Omega\,m\Omega)$
$[\mathsf{IsFiniteMeasure}\,\mu]\ (h : \mathsf{CondIndep}\,m'm_1\,m_2\,\mathsf{hm}'\mu)$
$\mathsf{CondIndep}\,m'm_2\,m_1\,\mathsf{hm}'\mu$

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$(m'm_1 : \mathsf{MeasurableSpace}\Omega)\ (\mathsf{hm}' : m' \le m\Omega)\ (\mu : @\mathsf{Measure}\Omega\,m\Omega)$
$[\mathsf{IsFiniteMeasure}\,\mu]$
$\mathsf{CondIndep}\,m'm_1\bot\mathsf{hm}'\mu$

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$(\mu : \mathsf{Measure}\ \alpha)\ (\nu : \mathsf{Measure}\beta)\ (s : \mathsf{Set}\ \alpha)\ (t : \mathsf{Set}\ \beta)$
$0 \le \mu\,s \cdot \nu\,t$

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$\{\mu : \mathsf{Measure}\ \alpha\}\ \{s\,t : \mathsf{Set}\ \alpha\}\ (h : s \subseteq t)$
$\mu\,s \le \mu\,t$

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$(\mu : \mathsf{Measure}\ \alpha)\ \{f : \alpha \Rightarrow \beta\}\ ({}_hf : \mathsf{Measurable}\ f)\ (s : \mathsf{Set}\ \beta)$
$0 \le \mu.\mathsf{map}\ f\,s$

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$(\mu : \mathsf{Measure}\ \alpha)$
$\mu\emptyset = 0$

$\{\Omega : \mathsf{Type}^*\}\ [m\Omega : \mathsf{MeasurableSpace}\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ \{\alpha\,\beta : \mathsf{Type}^*\}$
$[\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]$
$(\mu : \mathsf{Measure}\ \alpha)\ (s\,t : \mathsf{Set}\ \alpha)$
$\mu\,(s \cup t) \le \mu\,s + \mu\,t$

14.10 fep-010 — Detailed Balance Implies Invariance

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\kappa : \mathsf{Kernel}\,\alpha\,\alpha)\ (\pi : \mathsf{Measure}\ \alpha)\ [\mathsf{IsMarkovKernel}\,\kappa]$
$(\mathsf{hrev} : \mathsf{Kernel.IsReversible}\,\kappa\,\pi)$
$\mathsf{Kernel.Invariant}\,\kappa\,\pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\pi : \mathsf{Measure}\ \alpha)$
$\mathsf{Kernel.IsReversible}\ (\mathsf{Kernel.id} : \mathsf{Kernel}\,\alpha\,\alpha)\,\pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\pi : \mathsf{Measure}\ \alpha)$
$\mathsf{Kernel.Invariant}\ (\mathsf{Kernel.id} : \mathsf{Kernel}\,\alpha\,\alpha)\,\pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\kappa : \mathsf{Kernel}\ \alpha\ \alpha)\ (\pi : \mathsf{Measure}\ \alpha)\ (\mathsf{hinv} : \mathsf{Kernel.Invariant}\ \kappa\ \pi)$
$\pi.\mathsf{bind}\ \kappa = \pi$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$(\kappa\ \eta : \mathsf{Kernel}\ \alpha\ \alpha)\ (\pi : \mathsf{Measure}\ \alpha)\ (h\kappa : \mathsf{Kernel.Invariant}\ \kappa\ \pi)$
$(h\eta : \mathsf{Kernel.Invariant}\ \eta\ \pi)$
$\mathsf{Kernel.Invariant}\ (\kappa \circ_k\ \eta)\ \pi$

14.11 fep-011 — Surprise and Self-Information

$\{x : \mathbb{R}\}\ (\mathsf{hx} : 1 \le x)$
$0 \le \mathsf{Real.log}\ x$

$\{u : \mathbb{R}\}\ (\mathsf{hu} : 0 < u)\ (\mathsf{hu1} : u \le 1)$
$0 \le \mathsf{fep011\_surprise}\ u$

$\{a\ b : \mathbb{R}\}\ (\mathsf{ha} : 0 < a)\ (\mathsf{hb} : 0 < b)$
$\mathsf{fep011\_surprise}\ (a \cdot b) = \mathsf{fep011\_surprise}\ a + \mathsf{fep011\_surprise}\ b$

$\mathsf{fep011\_surprise}\ 1 = 0$

$\{p\ q : \mathbb{R}\}\ (\mathsf{hp} : 0 < p)\ (h : p \le q)$
$\mathsf{fep011\_surprise}\ q \le \mathsf{fep011\_surprise}\ p$

$\{p : \mathbb{R}\}\ (\mathsf{hp} : 0 < p)$
$\mathsf{fep011\_surprise}\ p = 0 \leftrightarrow p = 1$

$\{p\ q : \mathbb{R}\}\ (\mathsf{hp} : 0 < p)\ (\mathsf{hpq} : p < q)$
$\mathsf{fep011\_surprise}\ q < \mathsf{fep011\_surprise}\ p$

14.12 fep-012 — Finite Policy Shannon Entropy Regularization

$(q : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{hq} : \forall p, q\ p \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$
$0 \le \mathsf{fep012\_policyEntropy}\ q$

$(\mathsf{chosen} : \mathsf{Policy})$
$\mathsf{fep012\_policyEntropy}\ (\lambda\ p \mapsto \mathsf{if}\ p = \mathsf{chosen}\ \mathsf{then}\ 1\ \mathsf{else}\ 0) = 0$

$(q\ \mathsf{cost} : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\tau : \mathbb{R})\ (h\tau : 0 \le \tau)\ (\mathsf{hq} : \forall p, q\ p \in \mathsf{Set.Icc}\ (0 : \mathbb{R})\ 1)$
$\mathsf{fep012\_entropyRegularizedCost}\ q\ \mathsf{cost}\ \tau \le \mathsf{fep012\_expectedCost}\ q\ \mathsf{cost}$

$(q\ \mathsf{cost} : \mathsf{Policy} \Rightarrow \mathbb{R})$
$\mathsf{fep012\_entropyRegularizedCost}\ q\ \mathsf{cost}\ 0 = \mathsf{fep012\_expectedCost}\ q\ \mathsf{cost}$

$(G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (s : \mathsf{Finset}\ \mathsf{Policy})$
$0 \le \sum p \in s, \mathsf{Real.exp}\ (-G\ p)$

$(G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (s : \mathsf{Finset}\ \mathsf{Policy})\ (\mathsf{hne} : \mathsf{s.Nonempty})$
$0 < \sum p \in s, \mathsf{Real.exp}\ (-G\ p)$

$(G_1\ G_2 : \mathsf{Policy})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (h : G\ G_1 \le G\ G_2)$
$\mathsf{Real.exp}\ (-G\ G_2) \le \mathsf{Real.exp}\ (-G\ G_1)$

14.13 fep-013 — Equilibrium Helmholtz Differential Identities

$$(U\ S : \mathbb{R})$$
$$\mathsf{fep013\_helmholtz}\ U\ 0\ S = U$$

$$(U\ T : \mathbb{R})\ (S_1\ S_2 : \mathbb{R})\ (\mathsf{hT} : 0 < T)\ (h : S_1 \le S_2)$$
$$\mathsf{fep013\_helmholtz}\ U\ T\ S_2 \le \mathsf{fep013\_helmholtz}\ U\ T\ S_1$$

$$(U_1\ U_2\ T\ S : \mathbb{R})\ (h : U_1 \le U_2)$$
$$\mathsf{fep013\_helmholtz}\ U_1\ T\ S \le \mathsf{fep013\_helmholtz}\ U_2\ T\ S$$

$$(U_1\ U_2\ T\ S_1\ S_2 : \mathbb{R})$$
$$\mathsf{fep013\_helmholtz}\ U_2\ T\ S_2 - \mathsf{fep013\_helmholtz}\ U_1\ T\ S_1 = (U_2 - U_1) - T \cdot (S_2 - S_1)$$

$$\{U\ S : \mathbb{R} \Rightarrow \mathbb{R}\}\ \{U'\ S'\ T : \mathbb{R}\}\ (\mathsf{hU} : \mathsf{HasDerivAt}\ U\ U'\ T)\ (\mathsf{hS} : \mathsf{HasDerivAt}\ S\ S'\ T)$$
$$\mathsf{HasDerivAt}\ (\lambda\, t \mapsto \mathsf{fep013\_helmholtz}\ (U\ t)\ t\ (S\ t))\ (U' - (S\ T + T \cdot S'))\ T$$

$$\{U\ S : \mathbb{R} \Rightarrow \mathbb{R}\}\ \{U'\ S'\ T : \mathbb{R}\}\ (\mathsf{hU} : \mathsf{HasDerivAt}\ U\ U'\ T)\ (\mathsf{hS} : \mathsf{HasDerivAt}\ S\ S'\ T)$$
$$(\mathsf{hfirstLaw} : U' = T \cdot S')$$
$$\mathsf{HasDerivAt}\ (\lambda\, t \mapsto \mathsf{fep013\_helmholtz}\ (U\ t)\ t\ (S\ t))\ (-S\ T)\ T$$

14.14 fep-014 — KL Divergence: Non-Negativity, Identity, and Chain Rule

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu\ \nu : \mathsf{Measure}\ \alpha)$$
$$0 \le \mathsf{InformationTheory.klDiv}\ \mu\ \nu$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ [\mathsf{SigmaFinite}\ \mu]$$
$$\mathsf{InformationTheory.klDiv}\ \mu\ \mu = 0$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu\ \nu : \mathsf{Measure}\ \alpha)\ [\mathsf{IsFiniteMeasure}\ \mu]\ [\mathsf{IsFiniteMeasure}\ \nu]$$
$$\mathsf{InformationTheory.klDiv}\ \mu\ \nu = 0 \leftrightarrow \mu = \nu$$

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$(\mu\ \nu : \mathsf{Measure}\ \alpha)\ (\kappa\ \eta : \mathsf{Kernel}\ \alpha\ \beta)\ [\mathsf{IsFiniteMeasure}\ \mu]\ [\mathsf{IsFiniteMeasure}\ \nu]$$
$$[\mathsf{IsMarkovKernel}\ \kappa]\ [\mathsf{IsMarkovKernel}\ \eta]$$
$$\mathsf{InformationTheory.klDiv}\ (\mu \otimes_m\ \kappa)\ (\nu \otimes_m\ \eta) = \mathsf{InformationTheory.klDiv}\ \mu\ \nu$$
$$+\ \mathsf{InformationTheory.klDiv}\ (\mu \otimes_m\ \kappa)\ (\mu \otimes_m\ \eta)$$

14.15 fep-015 — Measurable Variational Integrand

$$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$$
$$\{\mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative} : \alpha \Rightarrow \mathbb{R}\}\ (\mathsf{henergy} : \mathsf{Measurable}\ \mathsf{energy})$$
$$(\mathsf{happroximation} : \mathsf{Measurable}\ \mathsf{logApproximation})$$
$$(\mathsf{hgenerative} : \mathsf{Measurable}\ \mathsf{logGenerative})$$
$$\mathsf{Measurable}\ (\mathsf{fep015\_variationalIntegrand}\ \mathsf{energy}\ \mathsf{logApproximation}\ \mathsf{logGenerative})$$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$\{\beta : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\beta]\ \{\mathsf{energy\ logApproximation\ logGenerative} : \alpha \Rightarrow \mathbb{R}\}$
$\{\mathsf{reparameterize} : \beta \Rightarrow \alpha\}\ (\mathsf{henergy} : \mathsf{Measurable\ energy})$
$(\mathsf{happroximation} : \mathsf{Measurable\ logApproximation})$
$(\mathsf{hgenerative} : \mathsf{Measurable\ logGenerative})$
$(\mathsf{hreparameterize} : \mathsf{Measurable\ reparameterize})$
$\mathsf{Measurable}\ (\mathsf{fep015\_variationalIntegrand\ energy\ logApproximation\ logGenerative} \circ$
$\mathsf{reparameterize})$

$\{\alpha : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]$
$\{\mathsf{energy\ logApproximation\ logGenerative} : \alpha \Rightarrow \mathbb{R}\}\ (\mathsf{henergy} : \mathsf{Measurable\ energy})$
$(\mathsf{happroximation} : \mathsf{Measurable\ logApproximation})$
$(\mathsf{hgenerative} : \mathsf{Measurable\ logGenerative})\ (\mathsf{baseline} : \mathbb{R})$
$\mathsf{Measurable}\ (\lambda\, \mathsf{state} \mapsto \mathsf{fep015\_variationalIntegrand\ energy\ logApproximation}$
$\mathsf{logGenerative\ state} + \mathsf{baseline})$

14.16 fep-016 — Exact Scalar Quadratic Laplace Kernel

$$(x : \mathbb{R})$$
$$0 \le x^2$$

$$(\mu : \mathbb{R})$$
$$(\mu - \mu)^2 = 0$$

$$(\mathsf{prec}\ x\ \mu : \mathbb{R})\ (\mathsf{hp} : 0 \le \mathsf{prec})$$
$$0 \le \mathsf{prec} \cdot (x - \mu)^2$$

$$(x\ \mu : \mathbb{R})$$
$$(x - \mu)^2 = (\mu - x)^2$$

$$(x\ \mu : \mathbb{R})$$
$$(x - \mu)^2 = x^2 - 2 \cdot \mu \cdot x + \mu^2$$

$\{\mathsf{precision} : \mathbb{R}\}\ (\mathsf{hprecision} : 0 < \mathsf{precision})$
$\mathsf{MeasureTheory.Integrable}\ (\mathsf{fep016\_laplaceKernel\ precision})$

$\{\mathsf{precision} : \mathbb{R}\}\ (\mathsf{hprecision} : 0 < \mathsf{precision})$
$(\int x : \mathbb{R}, \mathsf{fep016\_laplaceKernel\ precision}\ x) = \mathsf{Real.sqrt}$
$(\mathsf{Real.pi}/(\mathsf{precision}/2))$

14.17 fep-017 — Posterior Kernel Reconstruction

$\{\Omega\ \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\kappa : \mathsf{Kernel}\ \Omega\ \mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\ \Omega]\ [\mathsf{Nonempty}\ \Omega]$
$[\mathsf{IsFiniteMeasure}\ \mu]\ [\mathsf{IsFiniteKernel}\ \kappa]\ (x : \mathcal{X})$
$\mathsf{fep017\_posterior}\ \kappa\ \mu\ x\ \Omega = 1$

$\{\Omega\ \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\kappa : \mathsf{Kernel}\ \Omega\ \mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\ \Omega]\ [\mathsf{Nonempty}\ \Omega]$
$[\mathsf{IsFiniteMeasure}\ \mu]\ [\mathsf{IsFiniteKernel}\ \kappa]$
$(\kappa \circ_m\ \mu) \otimes_m\ \mathsf{fep017\_posterior}\ \kappa\ \mu = (\mu \otimes_m\ \kappa).\mathsf{map\ Prod.swap}$

$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\,\Omega]\ [\mathsf{Nonempty}\,\Omega]$
$[\mathsf{IsFiniteMeasure}\,\mu]\ [\mathsf{IsMarkovKernel}\,\kappa]$
$\mathsf{fep017\_posterior}\,\kappa\,\mu \circ_m\ \kappa \circ_m\ \mu = \mu$

$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$[\mathsf{Countable}\,\Omega]\ [\mathsf{StandardBorelSpace}\,\Omega]\ [\mathsf{Nonempty}\,\Omega]\ [\mathsf{StandardBorelSpace}\,\mathcal{X}]$
$[\mathsf{Nonempty}\,\mathcal{X}]\ (\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{IsFiniteMeasure}\,\mu]$
$[\mathsf{IsFiniteKernel}\,\kappa]$
$\forall^{\text{a.e.}}\, x\, \mathrm{d}\,(\kappa \circ_m\ \mu), \mathsf{fep017\_posterior}\,\kappa\,\mu\,x = \mu.\mathsf{withDensity}$
$(\lambda\,\omega \mapsto (\kappa\,\omega).\mathsf{rnDeriv}\,(\kappa \circ_m\ \mu)\,x)$

14.18 fep-018 — Bernoulli Fisher–Rao Coordinate Distance

$(p\,q : \mathbb{R})$
$0 \le \mathsf{fep018\_fisherRaoDistance}\,p\,q$

$(p\,q : \mathbb{R})$
$\mathsf{fep018\_fisherRaoDistance}\,p\,q = \mathsf{fep018\_fisherRaoDistance}\,q\,p$

$(p\,q\,r : \mathbb{R})$
$\mathsf{fep018\_fisherRaoDistance}\,p\,r \le \mathsf{fep018\_fisherRaoDistance}\,p\,q$
$+\ \mathsf{fep018\_fisherRaoDistance}\,q\,r$

$\{p\,q : \mathbb{R}\}\ (\mathsf{hp} : p \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)\ (\mathsf{hq} : q \in \mathsf{Set.Icc}\,(0 : \mathbb{R})\,1)$
$\mathsf{fep018\_fisherRaoDistance}\,p\,q = 0 \leftrightarrow p = q$

14.19 fep-019 — Prior-Predictive Kernel Composition

$\{\Omega\, \mathcal{X}\, \mathcal{Y} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]\ [\mathsf{MeasurableSpace}\mathcal{Y}]$
$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{IsMarkovKernel}\,\kappa]$
$\mathsf{fep019\_priorPredictive}\,\kappa\,\mu\,\Omega = \mu\,\Omega$

$\{\Omega\, \mathcal{X}\, \mathcal{Y} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]\ [\mathsf{MeasurableSpace}\mathcal{Y}]$
$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{IsProbabilityMeasure}\,\mu]\ [\mathsf{IsMarkovKernel}\,\kappa]$
$\mathsf{fep019\_priorPredictive}\,\kappa\,\mu\,\Omega = 1$

$\{\Omega\, \mathcal{X}\, \mathcal{Y} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]\ [\mathsf{MeasurableSpace}\mathcal{Y}]$
$(\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\eta : \mathsf{Kernel}\,\mathcal{X}\,\mathcal{Y})\ (\mu : \mathsf{Measure}\Omega)$
$\eta \circ_m\ \mathsf{fep019\_priorPredictive}\,\kappa\,\mu = \mathsf{fep019\_priorPredictive}\,(\eta \circ_k\ \kappa)\,\mu$

14.20 fep-020 — Two-State Markov Relaxation

$\{\alpha : \mathbb{R}\}\ (h\alpha 0 : 0 \le \alpha)\ (h\alpha 1 : \alpha \le 1)\ (\mathsf{source}\ \mathsf{target} : \mathsf{Bool})$
$0 \le \mathsf{fep020\_transition}\,\alpha\,\mathsf{source}\,\mathsf{target}$

$(\alpha : \mathbb{R})\ (\mathsf{source} : \mathsf{Bool})$
$\sum \mathsf{target} : \mathsf{Bool}, \mathsf{fep020\_transition}\,\alpha\,\mathsf{source}\,\mathsf{target} = 1$

$(\alpha\,p : \mathbb{R})$
$\mathsf{fep020\_evolve}\,\alpha\,p = \alpha + (1 - 2 \cdot \alpha) \cdot p$

$(\alpha : \mathbb{R})$
$\mathsf{fep020\_evolve}\,\alpha\,(1/2 : \mathbb{R}) = 1/2$

$(\alpha\, p : \mathbb{R})$
$\mathsf{fep020\_evolve}\ \alpha\, p - 1/2 = (1 - 2 \cdot \alpha) \cdot (p - 1/2)$

$(\alpha\, p : \mathbb{R})\ (n : \mathbb{N})$
$((\mathsf{fep020\_evolve}\ \alpha)^{[n]})\ p - 1/2 = (1 - 2 \cdot \alpha)^n \cdot (p - 1/2)$

$\{\alpha : \mathbb{R}\}\ (h\alpha 0 : 0 < \alpha)\ (h\alpha 1 : \alpha < 1)\ (p : \mathbb{R})$
$\mathsf{Filter.Tendsto}\ (\lambda\, n \mapsto ((\mathsf{fep020\_evolve}\ \alpha)^{[n]})\ p)\ \mathsf{Filter.atTop}$
$(\mathsf{nhds}\ (1/2 : \mathbb{R}))$

14.21 fep-021 — EFE Epistemic–Pragmatic Balance

$(\mathsf{pragmaticCost}\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0})$
$0 \leq \mathsf{fep021\_expectedFreeEnergy}\ \mathsf{pragmaticCost}\ \mathsf{epistemicValue}$

$\{\mathsf{pragmaticCost}\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0}\}$
$(\mathsf{hvalue} : \mathsf{epistemicValue} \leq \mathsf{pragmaticCost})$
$\mathsf{fep021\_expectedFreeEnergy}\ \mathsf{pragmaticCost}\ \mathsf{epistemicValue} + \mathsf{epistemicValue}$
$= \mathsf{pragmaticCost}$

$\{pragmatic_1\ pragmatic_2\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0}\}$
$(\mathsf{hpragmatic} : pragmatic_1 \leq pragmatic_2)$
$\mathsf{fep021\_expectedFreeEnergy}\ pragmatic_1\ \mathsf{epistemicValue} \leq \mathsf{fep021\_expectedFreeEnergy}$
$pragmatic_2\ \mathsf{epistemicValue}$

$\{\mathsf{pragmaticCost}\ epistemic_1\ epistemic_2 : \mathbb{R}^{\infty}_{\geq 0}\}$
$(\mathsf{hepistemic} : epistemic_1 \leq epistemic_2)$
$\mathsf{fep021\_expectedFreeEnergy}\ \mathsf{pragmaticCost}\ epistemic_2 \leq \mathsf{fep021\_expectedFreeEnergy}$
$\mathsf{pragmaticCost}\ epistemic_1$

$(\mathsf{pragmaticCost}\ \mathsf{epistemicValue} : \mathbb{R}^{\infty}_{\geq 0})$
$\mathsf{fep021\_expectedFreeEnergy}\ \mathsf{pragmaticCost}\ \mathsf{epistemicValue} = 0 \leftrightarrow \mathsf{pragmaticCost}$
$\leq \mathsf{epistemicValue}$

14.22 fep-022 — Posterior-Predictive Kernel and Brier Properness

$\{\mathcal{C}\ \Omega\ \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\mathcal{C}]\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\mathsf{posterior} : \mathsf{Kernel}\ \mathcal{C}\ \Omega)\ (\mathsf{likelihood} : \mathsf{Kernel}\ \Omega\ \mathcal{X})\ [\mathsf{IsMarkovKernel\ posterior}]$
$[\mathsf{IsMarkovKernel\ likelihood}]\ (c : \mathcal{C})$
$\mathsf{fep022\_posteriorPredictive\ posterior\ likelihood}\ c\ \Omega = 1$

$\{\mathcal{C}\ \Omega\ \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\mathcal{C}]\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(p\ q : \mathbb{R})$
$\mathsf{fep022\_brierScore}\ p\ q = p \cdot (1 - p) + (q - p)^2$

$\{\mathcal{C}\ \Omega\ \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\mathcal{C}]\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(p\ q : \mathbb{R})$
$\mathsf{fep022\_brierScore}\ p\ q = p \cdot (1 - p) \leftrightarrow q = p$

$\{\mathcal{C}\ \Omega\ \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\mathcal{C}]\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\mathsf{posterior} : \mathsf{Kernel}\ \mathcal{C}\ \Omega)\ (\mathsf{likelihood} : \mathsf{Kernel}\ \Omega\ \mathsf{Bool})\ (c : \mathcal{C})\ (p : \mathbb{R})$
$\mathsf{fep022\_brierScore}\ p\ (\mathsf{fep022\_predictiveTrueMass\ posterior\ likelihood}\ c) = p$
$\cdot (1 - p) + (\mathsf{fep022\_predictiveTrueMass\ posterior\ likelihood}\ c - p)^2$

14.23 fep-023 — Policy-Indexed Reachable Laws

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$(\text{policies} : \text{Set Policy})\ (\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})\ \{\text{policy} : \text{Policy}\}$$
$$(\text{hpolicy} : \text{policy} \in \text{policies})$$
$$\text{law policy} \in \text{fep023\_reachableLaws policies law}$$

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$\{policies_1\ policies_2 : \text{Set Policy}\}\ (\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})$$
$$(\text{hpolicies} : policies_1 \subseteq policies_2)$$
$$\text{fep023\_reachableLaws}\ policies_1\ \text{law} \subseteq \text{fep023\_reachableLaws}\ policies_2\ \text{law}$$

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$(\text{policies} : \text{Set Policy})\ (\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})$$
$$(\text{hlaw} : \forall \text{policy} \in \text{policies}, \text{law policy}\ \Omega = 1)\ \{\mu : \text{MeasureOutcome}\}$$
$$(h\mu : \mu \in \text{fep023\_reachableLaws policies law})$$
$$\mu\ \Omega = 1$$

$$\{\text{Policy Outcome} : \text{Type}^*\}\ [\text{MeasurableSpaceOutcome}]$$
$$(\text{law} : \text{Policy} \Rightarrow \text{MeasureOutcome})$$
$$\text{fep023\_reachableLaws}\ (\emptyset : \text{Set Policy})\ \text{law} = \emptyset$$

14.24 fep-024 — KL Regularization in Objectives

$$\{\alpha : \text{Type}^*\}\ [\text{MeasurableSpace}\ \alpha]$$
$$(\text{base weight} : \mathbb{R}^{\infty}_{\geq 0})\ (\text{approximation prior} : \text{Measure}\ \alpha)$$
$$\text{base} \leq \text{fep024\_klRegularizedObjective base weight approximation prior}$$

$$\{\alpha : \text{Type}^*\}\ [\text{MeasurableSpace}\ \alpha]$$
$$(\text{base} : \mathbb{R}^{\infty}_{\geq 0})\ (\text{approximation prior} : \text{Measure}\ \alpha)$$
$$\text{fep024\_klRegularizedObjective base}\ 0\ \text{approximation prior} = \text{base}$$

$$\{\alpha : \text{Type}^*\}\ [\text{MeasurableSpace}\ \alpha]$$
$$(\text{base} : \mathbb{R}^{\infty}_{\geq 0})\ \{weight_1\ weight_2 : \mathbb{R}^{\infty}_{\geq 0}\}\ (\text{hweight} : weight_1 \leq weight_2)$$
$$(\text{approximation prior} : \text{Measure}\ \alpha)$$
$$\text{fep024\_klRegularizedObjective base}\ weight_1\ \text{approximation prior}$$
$$\leq \text{fep024\_klRegularizedObjective base}\ weight_2\ \text{approximation prior}$$

$$\{\alpha : \text{Type}^*\}\ [\text{MeasurableSpace}\ \alpha]$$
$$(\text{base weight} : \mathbb{R}^{\infty}_{\geq 0})\ (\text{prior} : \text{Measure}\ \alpha)\ [\text{SigmaFinite prior}]$$
$$\text{fep024\_klRegularizedObjective base weight prior prior} = \text{base}$$

14.25 fep-025 — Conserved Nonequilibrium Probability Currents

$$\{n : \mathbb{N}\}\ (\text{flow} : \text{Matrix}\ (\text{Fin}\ n)\ (\text{Fin}\ n)\ \mathbb{R})\ (i\ j : \text{Fin}\ n)$$
$$\text{fep025\_probabilityCurrent flow}\ i\ j = -\text{fep025\_probabilityCurrent flow}\ j\ i$$

$$\{n : \mathbb{N}\}\ (\text{flow} : \text{Matrix}\ (\text{Fin}\ n)\ (\text{Fin}\ n)\ \mathbb{R})\ (i : \text{Fin}\ n)$$
$$\text{fep025\_probabilityCurrent flow}\ i\ i = 0$$

$$\{n : \mathbb{N}\}\ (\text{current} : \text{Matrix}\ (\text{Fin}\ n)\ (\text{Fin}\ n)\ \mathbb{R})$$
$$(\text{hcurrent} : \forall i\ j, \text{current}\ i\ j = -\text{current}\ j\ i)$$
$$\sum i, \text{fep025\_divergence current}\ i = 0$$

$$\{n : \mathbb{N}\}\ (\mathsf{stationary} : \mathsf{Fin}\ n \Rightarrow \mathbb{R})\ (\mathsf{transition} : \mathsf{Matrix}\ (\mathsf{Fin}\ n)\ (\mathsf{Fin}\ n)\ \mathbb{R})$$
$$(\mathsf{hrow} : \forall i, \sum j, \mathsf{transition}\ i\ j = 1)\ (\mathsf{hstationary} : \forall j, \sum i, \mathsf{stationary}\ i$$
$$\cdot\ \mathsf{transition}\ i\ j = \mathsf{stationary}\ j)\ (i : \mathsf{Fin}\ n)$$
$$\mathsf{fep025\_divergence}\ (\mathsf{fep025\_transitionCurrent}\ \mathsf{stationary}\ \mathsf{transition})\ i = 0$$

$$(i : \mathsf{Fin}\ 3)$$
$$\sum j, \mathsf{fep025\_cycleTransition}\ i\ j = 1$$

$$(j : \mathsf{Fin}\ 3)$$
$$\sum i, \mathsf{fep025\_cycleStationary}\ i \cdot \mathsf{fep025\_cycleTransition}\ i\ j$$
$$= \mathsf{fep025\_cycleStationary}\ j$$

$$(i : \mathsf{Fin}\ 3)$$
$$\mathsf{fep025\_divergence}\ \mathsf{fep025\_cycleCurrent}\ i = 0$$

$$\mathsf{fep025\_cycleCurrent}\ 0\ 1 \neq 0$$

14.26 fep-026 — Negative-Log Prior Complexity

$$\{a\ b : \mathbb{R}\}\ (\mathsf{ha} : 0 < a)\ (\mathsf{hab} : a \leq b)$$
$$\mathsf{Real.log}\ a \leq \mathsf{Real.log}\ b$$

$$\{a\ b : \mathbb{R}\}\ (\mathsf{ha} : 0 < a)\ (\mathsf{hb} : 0 < b)$$
$$\mathsf{Real.log}\ (a/b) = \mathsf{Real.log}\ a - \mathsf{Real.log}\ b$$

$$\{p\ q : \mathbb{R}\}\ (\mathsf{hp} : 0 < p)\ (_h q : 0 < q)\ (h : p \leq q)$$
$$\mathsf{fep026\_priorComplexity}\ q \leq \mathsf{fep026\_priorComplexity}\ p$$

$$\mathsf{fep026\_priorComplexity}\ 1 = 0$$

$$\{a\ b : \mathbb{R}\}\ (\mathsf{ha} : 0 < a)\ (\mathsf{hb} : 0 < b)$$
$$\mathsf{fep026\_priorComplexity}\ (a \cdot b) = \mathsf{fep026\_priorComplexity}\ a$$
$$+\ \mathsf{fep026\_priorComplexity}\ b$$

$$\{p : \mathbb{R}\}\ (\mathsf{hp} : 0 < p)$$
$$\mathsf{fep026\_priorComplexity}\ p = 0 \leftrightarrow p = 1$$

$$\{p\ q : \mathbb{R}\}\ (\mathsf{hp} : 0 < p)\ (\mathsf{hpq} : p < q)$$
$$\mathsf{fep026\_priorComplexity}\ q < \mathsf{fep026\_priorComplexity}\ p$$

14.27 fep-027 — Hierarchical Kernel Factorization

$$\{\alpha\ \beta\ \gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\ \alpha\ \beta)\ [\mathsf{IsProbabilityMeasure}\ \mu]\ [\mathsf{IsMarkovKernel}\ \kappa]$$
$$\mathsf{fep027\_hierarchicalJoint}\ \mu\ \kappa\ \Omega = 1$$

$$\{\alpha\ \beta\ \gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$$
$$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\ \alpha\ \beta)\ [\mathsf{SFinite}\ \mu]\ [\mathsf{IsMarkovKernel}\ \kappa]$$
$$(\mathsf{fep027\_hierarchicalJoint}\ \mu\ \kappa).\mathsf{fst} = \mu$$

$\{\alpha\,\beta\,\gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$
$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\ \alpha\ \beta)\ [\mathsf{SFinite}\ \mu]\ [\mathsf{IsSFiniteKernel}\ \kappa]$
$(\mathsf{fep027\_hierarchicalJoint}\ \mu\ \kappa).\mathsf{snd} = \kappa \circ_m\ \mu$

$\{\alpha\,\beta\,\gamma : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\ \alpha]\ [\mathsf{MeasurableSpace}\beta]\ [\mathsf{MeasurableSpace}\gamma]$
$(\mu : \mathsf{Measure}\ \alpha)\ (\kappa : \mathsf{Kernel}\ \alpha\ \beta)\ (\eta : \mathsf{Kernel}\ (\alpha \times \beta)\ \gamma)$
$(\mathsf{fep027\_hierarchicalJoint}\ \mu\ \kappa \otimes_m\ \eta).\mathsf{map}\ \mathsf{MeasurableEquiv.prodAssoc}$
$= \mathsf{fep027\_hierarchicalJoint}\ \mu\ (\kappa \otimes_k\ \eta)$

14.28 fep-028 — Support-Aware Finite Softmax Policy

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (p : \mathsf{Policy})$
$(\mathsf{hne} : \mathsf{policies.Nonempty})$
$0 \le \mathsf{fep028\_softmax}\ \gamma\ G\ \mathsf{policies}\ p$

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (p : \mathsf{Policy})$
$(\mathsf{hne} : \mathsf{policies.Nonempty})$
$\mathsf{fep028\_softmax}\ \gamma\ G\ \mathsf{policies}\ p \le 1$

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (\mathsf{hne} : \mathsf{policies.Nonempty})$
$\sum p \in \mathsf{policies}, \mathsf{fep028\_softmax}\ \gamma\ G\ \mathsf{policies}\ p = 1$

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ \{p : \mathsf{Policy}\}$
$(\mathsf{hp} : p \notin \mathsf{policies})$
$\mathsf{fep028\_softmax}\ \gamma\ G\ \mathsf{policies}\ p = 0$

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (\mathsf{hne} : \mathsf{policies.Nonempty})$
$\sum p : \mathsf{Policy}, \mathsf{fep028\_softmax}\ \gamma\ G\ \mathsf{policies}\ p = 1$

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (p : \mathsf{Policy})$
$0 < \mathsf{Real.exp}\ (-\gamma \cdot G\ p)$

$(\gamma : \mathbb{R})\ (G : \mathsf{Policy} \Rightarrow \mathbb{R})\ (\mathsf{policies} : \mathsf{Finset}\ \mathsf{Policy})\ (\mathsf{hne} : \mathsf{policies.Nonempty})$
$0 < \sum p' \in \mathsf{policies}, \mathsf{Real.exp}\ (-\gamma \cdot G\ p')$

14.29 fep-029 — Quadratic Bregman Divergence

$(a\,b\,t : \mathbb{R})\ (\mathsf{ht0} : 0 \le t)\ (_h t1 : t \le 1)\ (\mathsf{hab} : a \le b)$
$(1 - t) \cdot a + t \cdot b \ge a$

$(a\,b\,t : \mathbb{R})\ (_h t0 : 0 \le t)\ (\mathsf{ht1} : t \le 1)\ (\mathsf{hab} : a \le b)$
$(1 - t) \cdot a + t \cdot b \le b$

$(a\,b : \mathbb{R})$
$(1 - 0) \cdot a + 0 \cdot b = a$

$(a\,b : \mathbb{R})$
$(1 - 1) \cdot a + 1 \cdot b = b$

$(x\,y : \mathbb{R})$
$\mathsf{fep029\_quadraticBregman}\ x\ y = (x - y)^2$

$$(x\ y : \mathbb{R})$$
$$0 \le \mathsf{fep029\_quadraticBregman}\ x\ y$$

$$(x\ y : \mathbb{R})$$
$$\mathsf{fep029\_quadraticBregman}\ x\ y = 0 \leftrightarrow x = y$$

14.30 fep-030 — Maximum Entropy Principle

$$(p : \mathbb{R})$$
$$\mathsf{Real.binEntropy}\ p \le \mathsf{Real.log}\ 2$$

$$(p : \mathbb{R})$$
$$\mathsf{Real.binEntropy}\ p = \mathsf{Real.log}\ 2 \leftrightarrow p = (2 : \mathbb{R})^{-1}$$

$$(n : \mathbb{N})\ ({}_h n : 0 < n)$$
$$0 \le (1 : \mathbb{R})/n$$

$$(n : \mathbb{N})\ (\mathsf{hn} : 0 < n)$$
$$\sum \_ \in \mathsf{Finset.range}\ n, (1 : \mathbb{R})/n = 1$$

$$(n : \mathbb{N})\ (\mathsf{hn} : 1 \le n)$$
$$0 \le \mathsf{Real.log}\ n$$

$$(n : \mathbb{N})\ (\mathsf{hn} : 0 < n)$$
$$-\sum \_ \in \mathsf{Finset.range}\ n, (1 : \mathbb{R})/n \cdot \mathsf{Real.log}\ ((1 : \mathbb{R})/n) = \mathsf{Real.log}\ n$$

14.31 fep-031 — Finite Boltzmann–Gibbs Weights

$$(\beta\ E : \mathbb{R})$$
$$0 < \mathsf{Real.exp}\ (-\beta \cdot E)$$

$$(\beta\ E_1\ E_2 : \mathbb{R})\ (h\beta : 0 < \beta)\ (\mathsf{hE} : E_1 \le E_2)$$
$$\mathsf{Real.exp}\ (-\beta \cdot E_2) \le \mathsf{Real.exp}\ (-\beta \cdot E_1)$$

$$(\beta : \mathbb{R})\ (n : \mathbb{N})\ (E : \mathsf{Fin}\ n \Rightarrow \mathbb{R})\ (S : \mathsf{Finset}\ (\mathsf{Fin}\ n))\ (\mathsf{hS} : \mathsf{S.Nonempty})$$
$$0 < \sum i \in S, \mathsf{Real.exp}\ (-\beta \cdot E\ i)$$

$$(\beta : \mathbb{R})\ (n : \mathbb{N})\ (E : \mathsf{Fin}\ n \Rightarrow \mathbb{R})\ (S : \mathsf{Finset}\ (\mathsf{Fin}\ n))\ (\mathsf{hS} : \mathsf{S.Nonempty})$$
$$(i : \mathsf{Fin}\ n)$$
$$0 \le \mathsf{fep031\_gibbsProbability}\ \beta\ n\ E\ S\ i$$

$$(\beta : \mathbb{R})\ (n : \mathbb{N})\ (E : \mathsf{Fin}\ n \Rightarrow \mathbb{R})\ (S : \mathsf{Finset}\ (\mathsf{Fin}\ n))\ (\mathsf{hS} : \mathsf{S.Nonempty})$$
$$\sum i \in S, \mathsf{fep031\_gibbsProbability}\ \beta\ n\ E\ S\ i = 1$$

14.32 fep-032 — Exact Quadratic Gradient-Descent Dynamics

$$(\text{step center } x : \mathbb{R})$$
$$\text{fep032\_quadraticUpdate step center } x - \text{center} = (1 - \text{step}) \cdot (x - \text{center})$$

$$(\text{step center} : \mathbb{R})$$
$$\text{fep032\_quadraticUpdate step center center} = \text{center}$$

$$(\text{step center } x : \mathbb{R})\ (n : \mathbb{N})$$
$$((\text{fep032\_quadraticUpdate step center})^{[n]})\ x - \text{center} = (1 - \text{step})^n \cdot (x - \text{center})$$

$$\{\text{step center } x : \mathbb{R}\}\ (\text{hstep0} : 0 \le \text{step})\ (\text{hstep2} : \text{step} \le 2)$$
$$(\text{fep032\_quadraticUpdate step center } x - \text{center})^2 \le (x - \text{center})^2$$

$$\{\text{step} : \mathbb{R}\}\ (\text{hstep0} : 0 < \text{step})\ (\text{hstep2} : \text{step} < 2)\ (\text{center } x : \mathbb{R})$$
$$\text{Filter.Tendsto}\ (\lambda\, n \mapsto ((\text{fep032\_quadraticUpdate step center})^{[n]})\ x)$$
$$\text{Filter.atTop}\ (\text{nhds center})$$

14.33 fep-033 — Finite-Horizon Bellman Recursion

$$\{\text{State} : \text{Type}^*\}\ (\text{discount} : \mathbb{R}_{\ge 0}^{\infty})\ (\text{stageCost} : \text{State} \Rightarrow \mathbb{R}_{\ge 0}^{\infty})$$
$$(\text{step} : \text{State} \Rightarrow \text{State})\ (\text{terminalCost} : \text{State} \Rightarrow \mathbb{R}_{\ge 0}^{\infty})\ (\text{horizon} : \mathbb{N})$$
$$(\text{state} : \text{State})$$
$$\text{fep033\_value discount stageCost step terminalCost}\ (\text{horizon} + 1)\ \text{state}$$
$$= \text{stageCost state} + \text{discount} \cdot \text{fep033\_value discount stageCost step terminalCost horizon}\ (\text{step state})$$

$$\{\text{State} : \text{Type}^*\}\ (\text{discount} : \mathbb{R}_{\ge 0}^{\infty})$$
$$\{stage_1\ stage_2\ terminal_1\ terminal_2 : \text{State} \Rightarrow \mathbb{R}_{\ge 0}^{\infty}\}\ (\text{step} : \text{State} \Rightarrow \text{State})$$
$$(\text{hstage} : \forall \text{state}, stage_1\ \text{state} \le stage_2\ \text{state})$$
$$(\text{hterminal} : \forall \text{state}, terminal_1\ \text{state} \le terminal_2\ \text{state})\ (\text{horizon} : \mathbb{N})$$
$$(\text{state} : \text{State})$$
$$\text{fep033\_value discount } stage_1 \text{ step } terminal_1 \text{ horizon state} \le \text{fep033\_value}$$
$$\text{discount } stage_2 \text{ step } terminal_2 \text{ horizon state}$$

$$\{\text{State} : \text{Type}^*\}\ (\text{discount} : \mathbb{R}_{\ge 0}^{\infty})\ (\text{step} : \text{State} \Rightarrow \text{State})\ (\text{horizon} : \mathbb{N})$$
$$(\text{state} : \text{State})$$
$$\text{fep033\_value discount}\ (\lambda \_ \mapsto 0)\ \text{step}\ (\lambda \_ \mapsto 0)\ \text{horizon state} = 0$$

$$\{\text{State} : \text{Type}^*\}\ (\text{stageCost} : \text{State} \Rightarrow \mathbb{R}_{\ge 0}^{\infty})\ (\text{step} : \text{State} \Rightarrow \text{State})$$
$$(\text{terminalCost} : \text{State} \Rightarrow \mathbb{R}_{\ge 0}^{\infty})\ (\text{horizon} : \mathbb{N})\ (\text{state} : \text{State})$$
$$\text{fep033\_value}\ 0\ \text{stageCost step terminalCost}\ (\text{horizon} + 1)\ \text{state} = \text{stageCost state}$$

14.34 fep-034 — Normalized Bayesian Filtering

$$\{\Omega\ \mathcal{X} : \text{Type}^*\}\ [\text{MeasurableSpace}\Omega]\ [\text{MeasurableSpace}\mathcal{X}]$$
$$(\tau : \text{Kernel}\ \Omega\ \Omega)\ (\mu : \text{Measure}\Omega)\ [\text{IsMarkovKernel}\ \tau]$$
$$\text{fep034\_predictivePrior}\ \tau\ \mu\ \Omega = \mu\ \Omega$$

$$\{\Omega\ \mathcal{X} : \text{Type}^*\}\ [\text{MeasurableSpace}\Omega]\ [\text{MeasurableSpace}\mathcal{X}]$$
$$(\tau : \text{Kernel}\ \Omega\ \Omega)\ (\kappa : \text{Kernel}\ \Omega\ \mathcal{X})\ (\mu : \text{Measure}\Omega)\ [\text{StandardBorelSpace}\ \Omega]$$
$$[\text{Nonempty}\ \Omega]\ [\text{IsFiniteMeasure}\ \mu]\ [\text{IsFiniteKernel}\ \tau]\ [\text{IsFiniteKernel}\ \kappa]\ (x : \mathcal{X})$$
$$\text{fep034\_filter}\ \tau\ \kappa\ \mu\ x\ \Omega = 1$$

$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\tau : \mathsf{Kernel}\,\Omega\,\Omega)\ (\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\,\Omega]$
$[\mathsf{Nonempty}\,\Omega]\ [\mathsf{IsFiniteMeasure}\,\mu]\ [\mathsf{IsFiniteKernel}\,\tau]\ [\mathsf{IsFiniteKernel}\,\kappa]$
$(\kappa \circ_m \mathsf{fep034\_predictivePrior}\,\tau\,\mu) \otimes_m \mathsf{fep034\_filter}\,\tau\,\kappa\,\mu$
$= (\mathsf{fep034\_predictivePrior}\,\tau\,\mu \otimes_m \kappa).\mathsf{map}\ \mathsf{Prod.swap}$

$\{\Omega\, \mathcal{X} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{MeasurableSpace}\mathcal{X}]$
$(\tau : \mathsf{Kernel}\,\Omega\,\Omega)\ (\kappa : \mathsf{Kernel}\,\Omega\,\mathcal{X})\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{StandardBorelSpace}\,\Omega]$
$[\mathsf{Nonempty}\,\Omega]\ [\mathsf{IsFiniteMeasure}\,\mu]\ [\mathsf{IsFiniteKernel}\,\tau]\ [\mathsf{IsMarkovKernel}\,\kappa]$
$\mathsf{fep034\_filter}\,\tau\,\kappa\,\mu \circ_m \kappa \circ_m \mathsf{fep034\_predictivePrior}\,\tau\,\mu = \mathsf{fep034\_predictivePrior}\,\tau\,\mu$

14.35 fep-035 — Jensen's Inequality for Log

$$\mathsf{StrictConcaveOn}\,\mathbb{R}\,(\mathsf{Set.Ioi}\,0)\,\mathsf{Real.log}$$

$\{x\, y\, a\, b : \mathbb{R}\}\ (\mathsf{hx} : 0 < x)\ (\mathsf{hy} : 0 < y)\ (\mathsf{hxy} : x \neq y)\ (\mathsf{ha} : 0 < a)\ (\mathsf{hb} : 0 < b)$
$(\mathsf{hab} : a + b = 1)$
$a \cdot \mathsf{Real.log}\,x + b \cdot \mathsf{Real.log}\,y < \mathsf{Real.log}\,(a \cdot x + b \cdot y)$

$\{a\, b : \mathbb{R}\}\ (\mathsf{ha} : 0 < a)\ (\mathsf{hb} : 0 < b)$
$\mathsf{Real.log}\,(a \cdot b) = \mathsf{Real.log}\,a + \mathsf{Real.log}\,b$

$\{a : \mathbb{R}\}\ ({}_h a : 0 < a)\ (n : \mathbb{N})$
$\mathsf{Real.log}\,(a^n) = n \cdot \mathsf{Real.log}\,a$

$(x : \mathbb{R})$
$\mathsf{Real.log}\,(\mathsf{Real.exp}\,x) = x$

$\{a\, b : \mathbb{R}\}\ (\mathsf{ha} : 0 < a)\ (h : a \leq b)$
$\mathsf{Real.log}\,a \leq \mathsf{Real.log}\,b$

$$\mathsf{Real.log}\,(1 : \mathbb{R}) = 0$$

14.36 fep-036 — Laplace-Smoothed Empirical Bernoulli Prior

$(p : \mathbb{R}_{\geq 0})\ (\mathsf{hp} : p \leq 1)\ (\mathsf{trials} : \mathbb{N})\ (\mathsf{sample} : \mathbb{N})$
$(\mathsf{fep036\_binomialModel}\,p\,\mathsf{hp}\,\mathsf{trials}).\mathsf{real}\,\{\mathsf{sample}\} = (\mathsf{trials.choose}\,\mathsf{sample} : \mathbb{R})$
$\cdot\,(p : \mathbb{R})^{\mathsf{sample}} \cdot (1 - (p : \mathbb{R}))^{(\mathsf{trials} - \mathsf{sample})}$

$(\mathsf{successes}\,\mathsf{trials} : \mathbb{N})$
$0 < \mathsf{fep036\_smoothedRate}\,\mathsf{successes}\,\mathsf{trials}$

$\{\mathsf{successes}\,\mathsf{trials} : \mathbb{N}\}\ (h : \mathsf{successes} \leq \mathsf{trials})$
$\mathsf{fep036\_smoothedRate}\,\mathsf{successes}\,\mathsf{trials} < 1$

$\{\mathsf{successes}\,\mathsf{trials} : \mathbb{N}\}\ (h : \mathsf{successes} \leq \mathsf{trials})$
$\mathsf{fep036\_smoothedRate}\,\mathsf{successes}\,\mathsf{trials} \in \mathsf{Set.Ioo}\,(0 : \mathbb{R})\,1$

$(\mathsf{trials} : \mathbb{N})\ (\mathsf{sample} : \mathsf{Fin}\,(\mathsf{trials} + 1))$
$\mathsf{fep036\_empiricalPrior}\,\mathsf{trials}\,\mathsf{sample} \in \mathsf{Set.Ioo}\,(0 : \mathbb{R})\,1$

$\{s_1\ s_2\ \mathsf{trials} : \mathbb{N}\}\ (h : s_1 \le s_2)$
$\mathsf{fep036\_smoothedRate}\ s_1\ \mathsf{trials} \le \mathsf{fep036\_smoothedRate}\ s_2\ \mathsf{trials}$

$\{\mathsf{successes\ trials} : \mathbb{N}\}\ (\mathsf{htrials} : 0 < \mathsf{trials})$
$\mathsf{fep036\_smoothedRate\ successes\ trials} = ((\mathsf{successes} : \mathbb{R})/\mathsf{trials})$
$\cdot\ ((\mathsf{trials} : \mathbb{R})/(\mathsf{trials} + 2)) + 1/((\mathsf{trials} : \mathbb{R}) + 2)$

$(\mathsf{successes} : \mathbb{N} \Rightarrow \mathbb{N})\ (p : \mathbb{R})\ (\mathsf{hraw} : \mathsf{Filter.Tendsto}\ (\lambda\, n : \mathbb{N} \mapsto (\mathsf{successes}\ n : \mathbb{R})/n)\ \mathsf{Filter.atTop}\ (\mathsf{nhds}\ p))$
$\mathsf{Filter.Tendsto}\ (\lambda\, n : \mathbb{N} \mapsto \mathsf{fep036\_smoothedRate}\ (\mathsf{successes}\ n)\ n)\ \mathsf{Filter.atTop}\ (\mathsf{nhds}\ p)$

14.37 fep-037 — Two-State Correlation–Response Law

$(\alpha\ \mathsf{variance} : \mathbb{R})\ (n : \mathbb{N})$
$\mathsf{fep037\_autocorrelation}\ \alpha\ \mathsf{variance}\ (n+1) = \mathsf{fep037\_relaxation}\ \alpha$
$\cdot\ \mathsf{fep037\_autocorrelation}\ \alpha\ \mathsf{variance}\ n$

$\{\alpha : \mathbb{R}\}\ (h\alpha 0 : 0 < \alpha)\ (h\alpha 1 : \alpha < 1)\ (\mathsf{variance} : \mathbb{R})$
$\mathsf{Filter.Tendsto}\ (\mathsf{fep037\_autocorrelation}\ \alpha\ \mathsf{variance})\ \mathsf{Filter.atTop}\ (\mathsf{nhds}\ 0)$

$(\beta\ \alpha\ \mathsf{variance} : \mathbb{R})\ (n : \mathbb{N})$
$\mathsf{fep037\_response}\ \beta\ \alpha\ \mathsf{variance}\ n = 2 \cdot \beta \cdot \alpha \cdot \mathsf{variance} \cdot \mathsf{fep037\_relaxation}\ \alpha^n$

$\{\beta\ \alpha\ \mathsf{variance} : \mathbb{R}\}\ (h\beta : 0 \le \beta)\ (h\alpha 0 : 0 \le \alpha)\ (h\alpha half : \alpha \le 1/2)$
$(\mathsf{hvariance} : 0 \le \mathsf{variance})\ (n : \mathbb{N})$
$0 \le \mathsf{fep037\_response}\ \beta\ \alpha\ \mathsf{variance}\ n$

$\{\alpha : \mathbb{R}\}\ (h\alpha 0 : 0 < \alpha)\ (h\alpha 1 : \alpha < 1)\ (\beta\ \mathsf{variance} : \mathbb{R})$
$\mathsf{Filter.Tendsto}\ (\lambda\, n \mapsto \mathsf{fep037\_response}\ \beta\ \alpha\ \mathsf{variance}\ n)\ \mathsf{Filter.atTop}\ (\mathsf{nhds}\ 0)$

14.38 fep-038 — Bernoulli Fisher Information and Natural Gradient

$(p : \mathbb{R})\ (b : \mathsf{Bool})$
$\mathsf{HasDerivAt}\ (\lambda\, q \mapsto \mathsf{fep038\_bernoulliMass}\ q\ b)\ (\mathsf{fep038\_bernoulliMassDeriv}\ b)\ p$

$(p : \mathbb{R})$
$\sum b : \mathsf{Bool}, \mathsf{fep038\_bernoulliMass}\ p\ b = 1$

$\{p : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$
$\sum b : \mathsf{Bool}, \mathsf{fep038\_bernoulliMass}\ p\ b \cdot \mathsf{fep038\_score}\ p\ b = 0$

$\{p : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$
$\mathsf{fep038\_fisherInformation}\ p = 1/(p \cdot (1-p))$

$\{p\ v : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)\ (\mathsf{hv} : v \ne 0)$
$0 < \mathsf{fep038\_fisherMetric}\ p\ v\ v$

$\{p\ \mathsf{gradient} : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$
$\mathsf{fep038\_fisherInformation}\ p \cdot \mathsf{fep038\_naturalGradient}\ p\ \mathsf{gradient} = \mathsf{gradient}$

$\{p : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$
$\mathsf{fep038\_coordinateJacobian}\ p^2 = \mathsf{fep038\_fisherInformation}\ p$

$$\{p\, v\, w : \mathbb{R}\}\ (\mathsf{hp0} : 0 < p)\ (\mathsf{hp1} : p < 1)$$
$$\mathsf{fep038\_fisherMetric}\ p\ v\ w = (\mathsf{fep038\_coordinateJacobian}\ p \cdot v) \cdot (\mathsf{fep038\_coordinateJacobian}\ p \cdot w)$$

14.39 fep-039 — Four-Block Additive Free Energy

$$(\mathsf{local\_fe} : \mathsf{Fin}\ 4 \Rightarrow \mathbb{R})\ (h : \forall i, 0 \le \mathsf{local\_fe}\ i)$$
$$0 \le \mathsf{fep039\_global\_fe}\ \mathsf{local\_fe}$$

$$(f\ g : \mathsf{Fin}\ 4 \Rightarrow \mathbb{R})\ (h : \forall i, f\ i \le g\ i)$$
$$\mathsf{fep039\_global\_fe}\ f \le \mathsf{fep039\_global\_fe}\ g$$

$$\mathsf{fep039\_global\_fe}\ (\lambda \_ \mapsto 0) = 0$$

$$(f\ g : \mathsf{Fin}\ 4 \Rightarrow \mathbb{R})$$
$$\mathsf{fep039\_global\_fe}\ (\lambda\, i \mapsto f\ i + g\ i) = \mathsf{fep039\_global\_fe}\ f + \mathsf{fep039\_global\_fe}\ g$$

$$(\mathsf{local\_fe} : \mathsf{Fin}\ 4 \Rightarrow \mathbb{R})\ (\mathsf{hnonneg} : \forall i, 0 \le \mathsf{local\_fe}\ i)$$
$$\mathsf{fep039\_global\_fe}\ \mathsf{local\_fe} = 0 \leftrightarrow \forall i, \mathsf{local\_fe}\ i = 0$$

14.40 fep-040 — Native Gaussian Law and Thermal Entropy Response

$$(\mathsf{mean} : \mathbb{R})\ (\mathsf{variance} : \mathbb{R}_{\ge 0})$$
$$\mathsf{fep040\_gaussianLaw}\ \mathsf{mean}\ \mathsf{variance}\ \Omega = 1$$

$$(\mathsf{mean} : \mathbb{R})\ (\mathsf{variance} : \mathbb{R}_{\ge 0})$$
$$\int x, x\ \mathrm{d}\ \mathsf{fep040\_gaussianLaw}\ \mathsf{mean}\ \mathsf{variance} = \mathsf{mean}$$

$$(\mathsf{mean} : \mathbb{R})\ (\mathsf{variance} : \mathbb{R}_{\ge 0})$$
$$\mathsf{ProbabilityTheory.variance}\ \mathsf{id}\ (\mathsf{fep040\_gaussianLaw}\ \mathsf{mean}\ \mathsf{variance}) = \mathsf{variance}$$

$$\{v_1\ v_2 : \mathbb{R}\}\ (hv_1 : 0 < v_1)\ (\mathsf{hv} : v_1 < v_2)$$
$$\mathsf{fep040\_gaussianEntropy}\ v_1 < \mathsf{fep040\_gaussianEntropy}\ v_2$$

$$\{v : \mathbb{R}\}\ (\mathsf{hv} : 0 < v)$$
$$\mathsf{HasDerivAt}\ \mathsf{fep040\_gaussianEntropy}\ (1/(2 \cdot v))\ v$$

$$\{\kappa\, T : \mathbb{R}\}\ (h\kappa : 0 < \kappa)\ (\mathsf{hT} : 0 < T)$$
$$\mathsf{HasDerivAt}\ (\mathsf{fep040\_thermalEntropy}\ \kappa)\ (1/(2 \cdot T))\ T$$

$$\{T : \mathbb{R}\}\ (\mathsf{hT} : 0 < T)$$
$$\mathsf{fep040\_heatCapacity}\ T = 1/2$$

14.41 fep-041 — Expected Information Gain via Measure KL

$$\{\text{Obs State} : \text{Type}^*\}\ [\text{MeasurableSpaceObs}]\ [\text{MeasurableSpaceState}]$$
$$(\text{posterior prior} : \text{MeasureState})$$
$$0 \le \text{fep041\_informationGain posterior prior}$$

$$\{\text{Obs State} : \text{Type}^*\}\ [\text{MeasurableSpaceObs}]\ [\text{MeasurableSpaceState}]$$
$$(\text{posterior prior} : \text{MeasureState})\ [\text{IsFiniteMeasure posterior}]$$
$$[\text{IsFiniteMeasure prior}]$$
$$\text{fep041\_informationGain posterior prior} = 0 \leftrightarrow \text{posterior} = \text{prior}$$

$$\{\text{Obs State} : \text{Type}^*\}\ [\text{MeasurableSpaceObs}]\ [\text{MeasurableSpaceState}]$$
$$(\text{predictive} : \text{MeasureObs})\ (\text{posterior} : \text{Obs} \Rightarrow \text{MeasureState})$$
$$(\text{prior} : \text{MeasureState})\ [\text{SigmaFinite prior}]\ (\text{hposterior} : \forall^{\text{a.e.}}\ \text{observation}$$
$$\text{d predictive, posterior observation} = \text{prior})$$
$$\text{fep041\_expectedInformationGain predictive posterior prior} = 0$$

14.42 fep-042 — Bernoulli Sufficient-Statistic Factorization

$$(p : \mathbb{R})\ (\text{data} : \text{List Bool})$$
$$\text{fep042\_bernoulliLikelihood}\ p\ \text{data} = p^{\text{fep042\_successCount data}} \cdot (1-p)^{\text{fep042\_failureCount data}}$$

$$(p : \mathbb{R})\ (\text{xs ys} : \text{List Bool})\ (h : \text{fep042\_sufficientStatistic xs} = \text{fep042\_sufficientStatistic ys})$$
$$\text{fep042\_bernoulliLikelihood}\ p\ \text{xs} = \text{fep042\_bernoulliLikelihood}\ p\ \text{ys}$$

14.43 fep-043 — Fermat Criticality and Quadratic Curvature

$$\{f : \mathbb{R} \Rightarrow \mathbb{R}\}\ \{x_0 : \mathbb{R}\}\ (\text{hmin} : \text{IsLocalMin}\ f\ x_0)$$
$$\text{deriv}\ f\ x_0 = 0$$

$$(a\ x_0\ c\ x : \mathbb{R})$$
$$\text{HasDerivAt}\ (\text{fep043\_quadraticFreeEnergy}\ a\ x_0\ c)\ (\text{fep043\_quadraticGradient}\ a\ x_0\ x)\ x$$

$$\{a\ x_0\ c\ x : \mathbb{R}\}\ (\text{ha} : 0 < a)$$
$$\text{fep043\_quadraticFreeEnergy}\ a\ x_0\ c\ x = \text{fep043\_quadraticFreeEnergy}\ a\ x_0\ c\ x_0 \leftrightarrow x = x_0$$

$$\{a\ x_0\ c\ x : \mathbb{R}\}\ (\text{ha} : 0 < a)$$
$$\text{deriv}\ (\text{fep043\_quadraticFreeEnergy}\ a\ x_0\ c)\ x = 0 \leftrightarrow x = x_0$$

$$(a\ x_0\ x : \mathbb{R})$$
$$\text{HasDerivAt}\ (\text{fep043\_quadraticGradient}\ a\ x_0)\ (2 \cdot a)\ x$$

$$\{a : \mathbb{R}\}\ (\text{ha} : 0 < a)$$
$$0 < 2 \cdot a$$

14.44 fep-044 — Bernoulli Squared Hellinger Divergence

$$(p\,q : \mathbb{R})$$
$$0 \le \mathsf{fep044\_hellingerSq}\,p\,q$$

$$(p\,q : \mathbb{R})$$
$$\mathsf{fep044\_hellingerSq}\,p\,q = \mathsf{fep044\_hellingerSq}\,q\,p$$

$$\{p\,q : \mathbb{R}\}\,(\mathsf{hp} : p \in \mathsf{Set.Icc}\,(0:\mathbb{R})\,1)\,(\mathsf{hq} : q \in \mathsf{Set.Icc}\,(0:\mathbb{R})\,1)$$
$$\mathsf{fep044\_hellingerSq}\,p\,q = 0 \leftrightarrow p = q$$

$$(p\,q : \mathbb{R})$$
$$\mathsf{fep044\_hellingerSq}\,(1-p)\,(1-q) = \mathsf{fep044\_hellingerSq}\,p\,q$$

14.45 fep-045 — Finite Bernoulli Conjugate Closure

$$\{p\,l_0\,l_1 : \mathbb{R}\}\,(hp_0 : 0 < p)\,(hp_1 : p < 1)\,(hl_0 : 0 < l_0)\,(hl_1 : 0 < l_1)$$
$$0 < \mathsf{fep045\_evidence}\,p\,l_0\,l_1$$

$$\{p\,l_0\,l_1 : \mathbb{R}\}\,(hp_0 : 0 < p)\,(hp_1 : p < 1)\,(hl_0 : 0 < l_0)\,(hl_1 : 0 < l_1)$$
$$0 \le \mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1 \wedge \mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1 \le 1$$

$$\{p\,l_0\,l_1 : \mathbb{R}\}\,(\mathsf{hZ} : \mathsf{fep045\_evidence}\,p\,l_0\,l_1 \neq 0)\,(b : \mathsf{Bool})$$
$$\mathsf{fep045\_bernoulliMass}\,(\mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1)\,b$$
$$= \mathsf{fep045\_bernoulliMass}\,p\,b \cdot \mathsf{fep045\_binaryLikelihood}\,l_0\,l_1\,b / \mathsf{fep045\_evidence}\,p\,l_0\,l_1$$

$$(p\,l_0\,l_1 : \mathbb{R})$$
$$\sum b : \mathsf{Bool}, \mathsf{fep045\_bernoulliMass}\,(\mathsf{fep045\_posteriorParameter}\,p\,l_0\,l_1)\,b = 1$$

14.46 fep-046 — Finite Stick-Breaking Mass Conservation

$$(\mathsf{breaks} : \mathsf{List}\,\mathbb{R})$$
$$(\mathsf{fep046\_stickWeights\,breaks}).\mathsf{sum} + \mathsf{fep046\_remainder\,breaks} = 1$$

$$(\mathsf{breaks} : \mathsf{List}\,\mathbb{R})\,(\mathsf{hbreaks} : \forall v \in \mathsf{breaks}, v \in \mathsf{Set.Icc}\,(0:\mathbb{R})\,1)$$
$$0 \le \mathsf{fep046\_remainder\,breaks}$$

$$(\mathsf{breaks} : \mathsf{List}\,\mathbb{R})\,(\mathsf{hbreaks} : \forall v \in \mathsf{breaks}, v \in \mathsf{Set.Icc}\,(0:\mathbb{R})\,1)$$
$$\mathsf{fep046\_remainder\,breaks} \le 1$$

14.47 fep-047 — Compositional Finite Sum-Product Propagation

$$(\mathsf{factor} : \mathsf{Factor})\,(\mathsf{incoming} : \mathsf{State} \Rightarrow \mathbb{R})\,(\mathsf{hfactor} : \forall x\,y, 0 \le \mathsf{factor}\,x\,y)$$
$$(\mathsf{hincoming} : \forall y, 0 \le \mathsf{incoming}\,y)\,(x : \mathsf{State})$$
$$0 \le \mathsf{fep047\_forward\,factor\,incoming}\,x$$

$$(\mathsf{factor} : \mathsf{Factor})\,(incoming_1\,incoming_2 : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hfactor} : \forall x\,y, 0 \le \mathsf{factor}\,x\,y)\,(\mathsf{hincoming} : \forall y, incoming_1\,y \le incoming_2\,y)$$
$$(x : \mathsf{State})$$
$$\mathsf{fep047\_forward\,factor}\,incoming_1\,x \le \mathsf{fep047\_forward\,factor}\,incoming_2\,x$$

$$(\mathsf{factor} : \mathsf{Factor})$$
$$\mathsf{fep047\_forward\,factor}\,(\lambda \_ \mapsto 0) = 0$$

$$(\mathsf{incoming} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\mathsf{fep047\_forward}\ (1 : \mathsf{Factor})\ \mathsf{incoming} = \mathsf{incoming}$$

$$(\mathsf{outer}\ \mathsf{inner} : \mathsf{Factor})\ (\mathsf{incoming} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\mathsf{fep047\_forward}\ \mathsf{outer}\ (\mathsf{fep047\_forward}\ \mathsf{inner}\ \mathsf{incoming}) = \mathsf{fep047\_forward}$$
$$(\mathsf{outer} \cdot \mathsf{inner})\ \mathsf{incoming}$$

14.48 fep-048 — Contractive Policy Update Uniqueness

$$(a\ b : \mathbb{R})\ (\mathsf{ha} : 0 \le a)\ (\mathsf{hb} : 0 \le b)$$
$$0 \le a + b$$

$$(f\ g : \mathbb{R} \Rightarrow \mathbb{R})\ (\mathsf{hf} : \mathsf{Monotone}\ f)\ (\mathsf{hg} : \mathsf{Monotone}\ g)$$
$$\mathsf{Monotone}\ (g \circ f)$$

$$(f : \mathbb{R} \Rightarrow \mathbb{R})\ (c : \mathbb{R}_{\ge 0})\ (\mathsf{hcontract} : \mathsf{ContractingWith}\ c\ f)\ (x\ y : \mathbb{R})$$
$$(\mathsf{hfx} : \mathsf{Function.IsFixedPt}\ f\ x)\ (\mathsf{hfy} : \mathsf{Function.IsFixedPt}\ f\ y)$$
$$x = y$$

$$\mathsf{ContractingWith}\ (1/2 : \mathbb{R}_{\ge 0})\ \mathsf{fep048\_halfUpdate}$$

$$(x : \mathbb{R})\ (\mathsf{hx} : \mathsf{Function.IsFixedPt}\ \mathsf{fep048\_halfUpdate}\ x)$$
$$x = 0$$

$$(x : \mathbb{R})$$
$$\mathsf{Filter.Tendsto}\ (\lambda\, n \mapsto \mathsf{fep048\_halfUpdate}^{[n]}\ x)\ \mathsf{Filter.atTop}\ (\mathcal{N}\ 0)$$

$$\mathsf{Monotone}\ (\mathsf{id} : \mathbb{R} \Rightarrow \mathbb{R})$$

$$(f\ g : \mathbb{R} \Rightarrow \mathbb{R})\ (\mathsf{hf} : \mathsf{Monotone}\ f)\ (\mathsf{hg} : \mathsf{Monotone}\ g)$$
$$\mathsf{Monotone}\ (\lambda\, x \mapsto f\ x + g\ x)$$

14.49 fep-049 — Linear Constitutive Entropy Production

$$\{\iota : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{conductance}\ \mathsf{force} : \iota \Rightarrow \mathbb{R})$$
$$(\sum i, \mathsf{force}\ i \cdot \mathsf{fep049\_linearFlux}\ \mathsf{conductance}\ \mathsf{force}\ i)$$
$$= \mathsf{fep049\_entropyProduction}\ \mathsf{conductance}\ \mathsf{force}$$

$$\{\iota : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{conductance}\ \mathsf{force} : \iota \Rightarrow \mathbb{R})$$
$$(\mathsf{hconductance} : \forall i, 0 \le \mathsf{conductance}\ i)$$
$$0 \le \mathsf{fep049\_entropyProduction}\ \mathsf{conductance}\ \mathsf{force}$$

$$\{\iota : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \iota]\ (\mathsf{conductance}\ \mathsf{force} : \iota \Rightarrow \mathbb{R})$$
$$(\mathsf{hconductance} : \forall i, 0 < \mathsf{conductance}\ i)$$
$$\mathsf{fep049\_entropyProduction}\ \mathsf{conductance}\ \mathsf{force} = 0 \leftrightarrow \forall i, \mathsf{force}\ i = 0$$

$$\{\mathsf{forward}\ \mathsf{reverse} : \mathbb{R}\}\ (\mathsf{hforward} : 0 < \mathsf{forward})\ (\mathsf{hreverse} : 0 < \mathsf{reverse})$$
$$0 \le \mathsf{fep049\_edgeProduction}\ \mathsf{forward}\ \mathsf{reverse}$$

$$\{\mathsf{forward}\ \mathsf{reverse} : \mathbb{R}\}\ (\mathsf{hforward} : 0 < \mathsf{forward})\ (\mathsf{hreverse} : 0 < \mathsf{reverse})$$
$$\mathsf{fep049\_edgeProduction}\ \mathsf{forward}\ \mathsf{reverse} = 0 \leftrightarrow \mathsf{forward} = \mathsf{reverse}$$

14.50 fep-050 — Logical Erasure and the Landauer Bound

$$\neg \mathsf{Function.Injective}\ \mathsf{fep050\_erasure}$$

$$(\mathsf{kB} : \mathbb{R})$$
$$\mathsf{fep050\_erasureEntropyLoss}\ \mathsf{kB} = \mathsf{kB} \cdot \mathsf{Real.log}\ 2$$

$$\{Q\ T\ \mathsf{kB} : \mathbb{R}\}\ (\mathsf{hT} : 0 < T)\ (\mathsf{hsecondLaw} : 0 \le \mathsf{fep050\_totalEntropyChange}\ Q\ T\ \mathsf{kB})$$
$$\mathsf{fep050\_landauerBound}\ \mathsf{kB}\ T \le Q$$

$$\{W\ Q\ T\ \mathsf{kB} : \mathbb{R}\}\ (\mathsf{hT} : 0 < T)\ (\mathsf{hsecondLaw} : 0 \le \mathsf{fep050\_totalEntropyChange}\ Q\ T\ \mathsf{kB})$$
$$(\mathsf{hworkHeat} : Q \le W)$$
$$\mathsf{fep050\_landauerBound}\ \mathsf{kB}\ T \le W$$

$$\{\mathsf{kB}\ T : \mathbb{R}\}\ (\mathsf{hkB} : 0 < \mathsf{kB})\ (\mathsf{hT} : 0 < T)$$
$$0 < \mathsf{fep050\_landauerBound}\ \mathsf{kB}\ T$$

14.51 fep-051 — Likelihood-Ratio Reconstruction under Absolute Continuity

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]$$
$$(\mathsf{target\ reference} : \mathsf{Measure}\Omega)\ [\mathsf{target.HaveLebesgueDecomposition\ reference}]$$
$$(\mathsf{h\_ac} : \mathsf{target} \ll \mathsf{reference})$$
$$\mathsf{reference.withDensity}\ (\mathsf{target.rnDeriv\ reference}) = \mathsf{target}$$

$$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]$$
$$(\mathsf{target\ reference} : \mathsf{Measure}\Omega)\ [\mathsf{target.HaveLebesgueDecomposition\ reference}]$$
$$\mathsf{reference.withDensity}\ (\mathsf{target.rnDeriv\ reference}) = \mathsf{target} \leftrightarrow \mathsf{target} \ll \mathsf{reference}$$

14.52 fep-052 — Posterior Density as a Likelihood Tilt

$$\{\mathsf{Parameter\ Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{prior} : \mathsf{MeasureParameter})\ [\mathsf{IsFiniteMeasure\ prior}]$$
$$(\mathsf{likelihood} : \mathsf{Kernel\ Parameter\ Observation})\ [\mathsf{IsFiniteKernel\ likelihood}]\ (\mathsf{h\_ac} :$$
$$\forall^{\text{a.e.}}\ \mathsf{parameter\ d\ prior},\ \mathsf{likelihood\ parameter} \ll \mathsf{likelihood} \circ_m \mathsf{prior})$$
$$\forall^{\text{a.e.}}\ \mathsf{observation\ d\ likelihood} \circ_m \mathsf{prior},\ (\mathsf{likelihood}^\dagger \mathsf{prior})\ \mathsf{observation}$$
$$= \mathsf{prior.withDensity}\ (\lambda\ \mathsf{parameter} \mapsto \mathsf{likelihood.rnDeriv}\ (\mathsf{Kernel.const\ Parameter}$$
$$(\mathsf{likelihood} \circ_m \mathsf{prior}))\ \mathsf{parameter\ observation})$$

$$\{\mathsf{Parameter\ Observation} : \mathsf{Type}^*\}$$
$$\{\mathsf{Parameter} : \mathsf{Type}^*\}\ [\mathsf{Countable\ Parameter}]\ [\mathsf{MeasurableSpaceParameter}]$$
$$[\mathsf{Nonempty\ Parameter}]\ [\mathsf{StandardBorelSpace\ Parameter}]\ (\mathsf{prior} : \mathsf{MeasureParameter})$$
$$[\mathsf{IsFiniteMeasure\ prior}]\ (\mathsf{likelihood} : \mathsf{Kernel\ Parameter\ Observation})$$
$$[\mathsf{IsFiniteKernel\ likelihood}]$$
$$\forall^{\text{a.e.}}\ \mathsf{observation\ d\ likelihood} \circ_m \mathsf{prior},\ (\mathsf{likelihood}^\dagger \mathsf{prior})\ \mathsf{observation}$$
$$= \mathsf{prior.withDensity}\ (\lambda\ \mathsf{parameter} \mapsto (\mathsf{likelihood\ parameter}).\mathsf{rnDeriv}\ (\mathsf{likelihood}$$
$$\circ_m \mathsf{prior})\ \mathsf{observation})$$

14.53 fep-053 — Kernel Bayesian-Inversion Reconstruction

{Parameter Observation : Type*}
(prior : MeasureParameter) [IsFiniteMeasure prior]
(likelihood : Kernel Parameter Observation) [IsFiniteKernel likelihood]
(likelihood $\circ_m$ prior) $\otimes_m$ likelihood†prior = (prior $\otimes_m$ likelihood).map Prod.swap

{Parameter Observation : Type*}
{FiniteParameter FiniteEvidence : Type*} [Fintype FiniteParameter]
[Fintype FiniteEvidence] (prior : FiniteLaw FiniteParameter) (likelihood :
FiniteKernel FiniteParameter FiniteEvidence) (evidence : FiniteEvidence)
(hEvidence : $0 <$ likelihood.predictive prior evidence)
(parameter : FiniteParameter)
likelihood.posterior prior evidence hEvidence parameter · likelihood.predictive
prior evidence = prior parameter · likelihood parameter evidence

14.54 fep-054 — Bayes Involution for Finite Joint Laws

{Parameter Observation : Type*}
(prior : MeasureParameter) [IsFiniteMeasure prior]
(likelihood : Kernel Parameter Observation) [StandardBorelSpace Observation]
[Nonempty Observation] [IsFiniteKernel likelihood] [IsMarkovKernel likelihood]
(likelihood†prior)†(likelihood $\circ_m$ prior) $\overset{\text{a.e.}}{=}$ [prior] likelihood

{Parameter Observation : Type*}
(prior : MeasureParameter) [IsFiniteMeasure prior]
(likelihood : Kernel Parameter Observation) [IsFiniteKernel likelihood]
[IsMarkovKernel likelihood]
likelihood†prior $\circ_m$ likelihood $\circ_m$ prior = prior

14.55 fep-055 — Composite-Kernel Bayesian Inversion

{Parameter Intermediate Observation : Type*}
(prior : MeasureParameter) [IsFiniteMeasure prior]
[StandardBorelSpace Parameter] [Nonempty Parameter]
[StandardBorelSpace Intermediate] [Nonempty Intermediate]
(earlier : Kernel Parameter Intermediate) [IsFiniteKernel earlier]
(later : Kernel Intermediate Observation) [IsFiniteKernel later]
(later $\circ_k$ earlier)†prior $\overset{\text{a.e.}}{=}$ [later $\circ_m$ earlier $\circ_m$ prior] earlier†prior$\circ_k$
later†(earlier $\circ_m$ prior)

{Parameter Intermediate Observation : Type*}
(prior : MeasureParameter) [IsFiniteMeasure prior]
(earlier : Kernel Parameter Intermediate) [IsFiniteKernel earlier]
(later : Kernel Intermediate Observation) [IsFiniteKernel later]
(later $\circ_k$ earlier) $\circ_m$ prior = later $\circ_m$ earlier $\circ_m$ prior

14.56 fep-056 — Standard-Borel Conditional-Kernel Reconstruction

{Conditioning Conditioned : Type*}
(joint : Measure(Conditioning × Conditioned)) [IsFiniteMeasure joint]
joint.fst $\otimes_m$ joint.condKernel = joint

{Conditioning Conditioned : Type*}
(joint : Measure(Conditioning × Conditioned)) [IsFiniteMeasure joint]
(conditioning : Conditioning)
joint.condKernel conditioning Ω = 1

14.57 fep-057 — Conditional-Expectation Tower through a Kernel

{Conditioning Conditioned : Type*}
(joint : Measure(Conditioning × Conditioned)) [IsFiniteMeasure joint]
{observable : Conditioning × Conditioned ⇒ ℝ}
(hintegrable : Integrable observable joint)
$$\left(\int \text{conditioning}, \int \text{conditioned}, \text{observable (conditioning, conditioned)}\right.$$
$$\left.\mathrm{d}\,\text{joint.condKernel conditioning}\,\mathrm{d}\,\text{joint.fst}\right) = \int \text{pair}, \text{observable pair}\,\mathrm{d}\,\text{joint}$$

{Conditioning Conditioned : Type*}
(joint : Measure(Conditioning × Conditioned)) [IsFiniteMeasure joint]
{observable : Conditioning × Conditioned ⇒ $\mathbb{R}_{\geq 0}^{\infty}$}
(hmeasurable : Measurable observable)
$$\left(\int^{-} \text{conditioning}, \int^{-} \text{conditioned}, \text{observable (conditioning, conditioned)}\right.$$
$$\left.\mathrm{d}\,\text{joint.condKernel conditioning}\,\mathrm{d}\,\text{joint.fst}\right) = \int^{-} \text{pair}, \text{observable pair}\,\mathrm{d}\,\text{joint}$$

14.58 fep-058 — Finite Gibbs Variational Principle

{State : Type*} [Fintype State]
(certificate : GibbsCertificate State) (candidate : FiniteLaw State)
− certificate.logPartition ≤ gibbsFreeEnergy certificate candidate

{State : Type*} [Fintype State]
(certificate : GibbsCertificate State)
gibbsFreeEnergy certificate certificate.optimizer = −certificate.logPartition

{State : Type*} [Fintype State]
[Nonempty State]
dvObjective (uniformZeroPotentialGibbs (α := State))
(uniformZeroPotentialGibbs (α := State)).optimizer = 0

14.59 fep-059 — Finite Donsker–Varadhan Equality with a Full-Support Gibbs Optimizer

{State : Type*} [Fintype State]
(certificate : GibbsCertificate State) (candidate : FiniteLaw State)
dvObjective certificate candidate ≤ certificate.logPartition

{State : Type*} [Fintype State]
(certificate : GibbsCertificate State)
dvObjective certificate certificate.optimizer = certificate.logPartition

{State : Type*} [Fintype State]
(certificate : GibbsCertificate State) (candidate : FiniteLaw State)
dvObjective certificate candidate = certificate.logPartition ↔ candidate
= certificate.optimizer

14.60 fep-060 — Coordinate ELBO Decomposition

$\{\text{Latent Observation} : \text{Type}^*\}$
$(\text{actualPrior referencePrior} : \text{FiniteLaw Latent})$ $(\text{actualKernel referenceKernel} :$
$\text{FiniteKernel Latent Observation})$ $(\text{hprior} : \forall \text{latent}, 0 < \text{referencePrior latent})$
$(\text{hkernel} : \forall \text{latent observation}, 0 < \text{referenceKernel latent observation})$
$\text{jointELBO actualPrior referencePrior actualKernel referenceKernel} = -\text{finiteKL}$
$\text{actualPrior referencePrior} - \text{conditionalKL actualPrior actualKernel}$
referenceKernel

$\{\text{Latent Observation} : \text{Type}^*\}$
$(\text{prior} : \text{FiniteLaw Latent})$ $(\text{actual reference} : \text{FiniteKernel Latent Observation})$
$0 \leq \text{conditionalKL prior actual reference}$

14.61 fep-061 — Mean-Field Coordinate Free-Energy Optimum

$\{\text{Fixed Coordinate} : \text{Type}^*\}$
$(\text{fixed} : \text{FiniteLaw Fixed})$ $(\text{candidate target} : \text{FiniteLaw Coordinate})$
$(\text{hfixed} : \forall x, 0 < \text{fixed}\, x)$ $(\text{htarget} : \forall y, 0 < \text{target}\, y)$
$\text{meanFieldCoordinateFreeEnergy fixed candidate target}$
$= \text{FEP.FiniteInformation.finiteKL candidate target}$

$\{\text{Fixed Coordinate} : \text{Type}^*\}$
$(\text{fixed} : \text{FiniteLaw Fixed})$ $(\text{candidate target} : \text{FiniteLaw Coordinate})$
$(\text{hfixed} : \forall x, 0 < \text{fixed}\, x)$ $(\text{htarget} : \forall y, 0 < \text{target}\, y)$
$\text{meanFieldCoordinateFreeEnergy fixed candidate target} = 0 \leftrightarrow \text{candidate} = \text{target}$

14.62 fep-062 — Fixed-Sample Importance-Weighted Jensen Bound

$\{\text{Sample} : \text{Type}^*\}$ $[\text{Fintype Sample}]$
$(\text{base} : \text{FiniteLaw Sample})$ $(\text{sampleCount} : \mathbb{N})$ $[\text{NeZero sampleCount}]$
$(\text{weight} : \text{Sample} \Rightarrow \mathbb{R})$ $(\text{hweight} : \forall \text{sample}, 0 < \text{weight sample})$
$\sum \text{sampleTuple}, \text{iidProductLaw base sampleCount sampleTuple} \cdot \text{Real.log}$
$(\text{sampleMeanWeight sampleCount weight sampleTuple}) \leq \text{Real.log}\,(\sum \text{sampleTuple},$
$\text{iidProductLaw base sampleCount sampleTuple} \cdot \text{sampleMeanWeight sampleCount weight}$
$\text{sampleTuple})$

$\{\text{Sample} : \text{Type}^*\}$ $[\text{Fintype Sample}]$
$(\text{sampling} : \text{FiniteLaw Sample})$ $(\text{weight} : \text{Sample} \Rightarrow \mathbb{R})$
$(\text{hweight} : \forall \text{sample}, 0 < \text{weight sample})$
$\sum \text{sample}, \text{sampling sample} \cdot \text{Real.log}\,(\text{weight sample}) \leq \text{Real.log}$
$(\sum \text{sample}, \text{sampling sample} \cdot \text{weight sample})$

$\{\text{Sample} : \text{Type}^*\}$ $[\text{Fintype Sample}]$
$[\text{Nonempty Sample}]$ $(\text{sampling} : \text{FiniteLaw Sample})$ $(\text{weight} : \text{Sample} \Rightarrow \mathbb{R})$
$(\text{hsampling} : \forall \text{sample}, 0 < \text{sampling sample})$
$(\text{hweight} : \forall \text{sample}, 0 < \text{weight sample})$
$0 < \sum \text{sample}, \text{sampling sample} \cdot \text{weight sample}$

14.63 fep-063 — Finite-Channel KL Data-Processing Inequality

$\{\text{Input Output} : \text{Type}^*\}$
$(\text{actual reference} : \text{FiniteLaw Input})\ (\text{channel} : \text{FiniteKernel Input Output})$
$(\text{hactual} : \forall \text{input}, 0 < \text{actual input})$
$(\text{hreference} : \forall \text{input}, 0 < \text{reference input})$
$(\text{hchannel} : \forall \text{input output}, 0 < \text{channel input output})$
$\text{finiteKL}\ (\text{channel.predictive actual})\ (\text{channel.predictive reference}) \leq \text{finiteKL}$
actual reference

$\{\text{Input Output} : \text{Type}^*\}$
$(\text{reference} : \text{FiniteLaw Input})\ (\text{channel} : \text{FiniteKernel Input Output})$
$(\text{hreference} : \forall \text{input}, 0 < \text{reference input})$
$(\text{hchannel} : \forall \text{input output}, 0 < \text{channel input output})\ (\text{output} : \text{Output})$
$0 < \text{channel.predictive reference output}$

14.64 fep-064 — Rate–Distortion Lagrangian Weak Duality

$\{\text{Source Code} : \text{Type}^*\}\ [\text{Fintype Source}]\ [\text{Fintype Code}]$
$(\text{joint} : \text{FiniteLaw}\ (\text{Source} \times \text{Code}))\ (\text{distortion} : \text{Source} \Rightarrow \text{Code} \Rightarrow \mathbb{R})$
$(\text{multiplier rateLower distortionLower} : \mathbb{R})\ (\text{hmultiplier} : 0 \leq \text{multiplier})$
$(\text{hrate} : \text{rateLower} \leq \text{mutualInformation joint})\ (\text{hdistortion} : \text{distortionLower}$
$\leq \text{expectedDistortion joint distortion})$
$\text{rateLower} + \text{multiplier} \cdot \text{distortionLower} \leq \text{rateDistortionLagrangian joint}$
$\text{distortion multiplier}$

$\{\text{Source Code} : \text{Type}^*\}\ [\text{Fintype Source}]\ [\text{Fintype Code}]$
$(\text{joint} : \text{FiniteLaw}\ (\text{Source} \times \text{Code}))\ (\text{distortion} : \text{Source} \Rightarrow \text{Code} \Rightarrow \mathbb{R})$
$\text{rateDistortionLagrangian joint distortion}\ 0 = \text{mutualInformation joint}$

14.65 fep-065 — Controlled-Kernel Normalization

$\{\text{State Action} : \text{Type}^*\}\ [\text{Fintype State}]\ [\text{Fintype Action}]$
$(\text{transition} : \text{ControlledKernel State Action})\ (\text{action} : \text{Action})\ (\text{state} : \text{State})$
$\sum \text{nextState}, \text{transition action state nextState} = 1$

14.66 fep-066 — Action-Conditioned Bayesian Belief Update

$\{\text{State Action Observation} : \text{Type}^*\}$
$(\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{ControlledKernel State Action})$
$(\text{emission} : \text{FiniteKernel State Observation})\ (\text{action} : \text{Action})$
$(\text{observation} : \text{Observation})\ (\text{hEvidence} : 0 < \text{actionEvidence prior transition}$
$\text{emission action observation})\ (\text{state} : \text{State})$
$\text{actionBeliefUpdate prior transition emission action observation hEvidence state}$
$\cdot\ \text{actionEvidence prior transition emission action observation} = \text{actionPrediction}$
$\text{prior transition action state} \cdot \text{emission state observation}$

{State Action Observation : Type*}
(prior : FiniteLaw State) (transition : ControlledKernel State Action)
(emission : FiniteKernel State Observation) (action : Action)
(observation : Observation) (hEvidence : $0 <$ actionEvidence prior transition
emission action observation)
$\sum$ state, actionBeliefUpdate prior transition emission action observation
hEvidence state $= 1$

{State Action Observation : Type*}
(prior : FiniteLaw State) (transition : ControlledKernel State Action)
(emission : FiniteKernel State Observation) (action : Action)
(observation : Observation) (hZero : actionEvidence prior transition emission
action observation $= 0$)
$\neg 0 <$ actionEvidence prior transition emission action observation

14.67 fep-067 — Reachable Finite-Belief POMDP Reduction

{Belief State Action Observation : Type*}
(model : ReachableBeliefPOMDP Belief State Action Observation)
(stateCost : State ⇒ Action ⇒ ℝ) (policy : ℕ ⇒ Belief ⇒ Action) (horizon : ℕ)
(belief : Belief)
reducedPolicyValue model stateCost policy horizon belief
$=$ interpretedPolicyValue model stateCost policy horizon belief

{Belief State Action Observation : Type*}
(belief action observation : Bool) (hEvidence : $0 <$ actionEvidence
(boolReachablePOMDP.interpret belief) boolReachablePOMDP.transition
boolReachablePOMDP.emission action observation)
boolReachablePOMDP.interpret
(boolReachablePOMDP.update belief action observation) $=$ actionBeliefUpdate
(boolReachablePOMDP.interpret belief) boolReachablePOMDP.transition
boolReachablePOMDP.emission action observation hEvidence

14.68 fep-068 — Soft Bellman Recursion

{State Action : Type*}
(temperature : ℝ) (hTemperature : $0 <$ temperature)
(stageCost : State ⇒ Action ⇒ ℝ) (transition : ControlledKernel State Action)
(horizon : ℕ) (state : State)
$(0 < \sum$ action, Real.exp $(-((\text{softBellmanActionEnergy temperature stageCost}$
transition horizon state action$)/\text{temperature})))$ ∧ (softBellmanValue
temperature stageCost transition $(\text{horizon} + 1)$ state $= -\text{temperature} \cdot$ Real.log
$(\sum$ action, Real.exp $(-((\text{softBellmanActionEnergy temperature stageCost transition}$
horizon state action$)/\text{temperature}))))$ ∧ ∀action, softBellmanValue temperature
stageCost transition $(\text{horizon} + 1)$ state ≤ softBellmanActionEnergy temperature
stageCost transition horizon state action

14.69 fep-069 — KL-Control Desirability Recursion

$$\{\text{State} : \text{Type}^*\}\ [\text{Fintype State}]$$
$$(\text{passive} : \text{FiniteKernel State State})\ (\text{stateCost desirability} : \text{State} \Rightarrow \mathbb{R})$$
$$(\text{state} : \text{State})$$
$$\text{desirabilityStep passive stateCost desirability state} = \text{Real.exp}$$
$$(-\text{stateCost state}) \cdot \sum \text{nextState, passive state nextState} \cdot \text{desirability}$$
$$\text{nextState}$$

$$\{\text{State} : \text{Type}^*\}\ [\text{Fintype State}]$$
$$(\text{passive} : \text{FiniteKernel State State})\ (\text{stateCost desirability} : \text{State} \Rightarrow \mathbb{R})$$
$$(\text{hDesirability} : \forall \text{state}, 0 \le \text{desirability state})\ (\text{state} : \text{State})$$
$$0 \le \text{desirabilityStep passive stateCost desirability state}$$

$$\{\text{State} : \text{Type}^*\}\ [\text{Fintype State}]$$
$$(\text{passive} : \text{FiniteKernel State State})\ (\text{state} : \text{State})$$
$$\text{desirabilityStep passive}\ (\lambda \_ \mapsto 0)\ (\lambda \_ \mapsto 1)\ \text{state} = 1$$

14.70 fep-070 — Control-as-Inference Policy Posterior

$$\{\text{Action} : \text{Type}^*\}\ [\text{Fintype Action}]$$
$$(\text{prior} : \text{FiniteLaw Action})\ (\text{precision} : \mathbb{R})\ (\text{cost} : \text{Action} \Rightarrow \mathbb{R})$$
$$\sum \text{action, controlPosterior prior precision cost action} = 1$$

$$\{\text{Action} : \text{Type}^*\}\ [\text{Fintype Action}]$$
$$(\text{prior} : \text{FiniteLaw Action})\ (\text{precision} : \mathbb{R})\ (\text{cost} : \text{Action} \Rightarrow \mathbb{R})\ (\text{action} : \text{Action})$$
$$\text{controlPosterior prior precision cost action} \cdot \text{boltzmannPartition prior}$$
$$(\lambda\, \text{candidate} \mapsto \text{precision} \cdot \text{cost candidate}) = \text{boltzmannWeight prior}$$
$$(\lambda\, \text{candidate} \mapsto \text{precision} \cdot \text{cost candidate})\ \text{action}$$

$$\{\text{Action} : \text{Type}^*\}\ [\text{Fintype Action}]$$
$$(\text{prior} : \text{FiniteLaw Action})\ (\text{precision} : \mathbb{R})$$
$$\text{controlPosterior prior precision}\ (\lambda \_ \mapsto 0) = \text{prior}$$

14.71 fep-071 — Sophisticated Expected-Free-Energy Backward Induction

$$\{\text{Belief Action Observation} : \text{Type}^*\}$$
$$(\text{model} : \text{SophisticatedEFEModel Belief Action Observation})\ (\text{horizon} : \mathbb{N})$$
$$(\text{belief} : \text{Belief})$$
$$\text{sophisticatedEFEValue model}\ (\text{horizon} + 1)\ \text{belief} = \text{model.stageEFE horizon belief}$$
$$(\text{sophisticatedEFEAction model horizon belief}) + \sum \text{observation,}$$
$$\text{model.observationLaw belief}\ (\text{sophisticatedEFEAction model horizon belief})$$
$$\text{observation} \cdot \text{sophisticatedEFEValue model horizon}\ (\text{model.update belief}$$
$$(\text{sophisticatedEFEAction model horizon belief})\ \text{observation})$$

$$\{\text{Belief Action Observation} : \text{Type}^*\}$$
$$\text{twoStageFeedback false} \neq \text{twoStageFeedback true}$$

$$\{\text{Belief Action Observation} : \text{Type}^*\}$$
$$(\text{action} : \text{Bool})$$
$$\text{twoStageFeedbackExpectedCost} < \text{twoStageOpenLoopExpectedCost action}$$

14.72 fep-072 — Hidden-Markov Forward Filtering Recursion

$$\{\text{State Observation} : \text{Type}^*\}\ (\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{FiniteKernel State State})\ (\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})\ (\text{hEvidence} : 0 < \text{forwardEvidence prior transition emission observation})\ (\text{state} : \text{State})\ \text{forwardFilter prior transition emission observation hEvidence state} \cdot \text{forwardEvidence prior transition emission observation} = \text{forwardPrediction prior transition state} \cdot \text{emission state observation}$$

$$\{\text{State Observation} : \text{Type}^*\}\ \text{boolForwardFilter true} = 20/23$$

$$\{\text{State Observation} : \text{Type}^*\}\ (\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{FiniteKernel State State})\ (\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})\ (\text{hZero} : \text{forwardEvidence prior transition emission observation} = 0)\ \neg 0 < \text{forwardEvidence prior transition emission observation}$$

14.73 fep-073 — Backward Information-Message Recursion

$$\{\text{State Observation} : \text{Type}^*\}\ (\text{transition} : \text{FiniteKernel State State})\ (\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})\ (\text{future} : \text{List Observation})\ (\text{state} : \text{State})\ \text{backwardMessage transition emission (observation :: future) state} = \text{backwardMessageStep transition emission observation (backwardMessage transition emission future) state}$$

$$\{\text{State Observation} : \text{Type}^*\}\ (\text{transition} : \text{FiniteKernel State State})\ (\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})\ (\text{nextMessage} : \text{State} \Rightarrow \mathbb{R})\ (\text{hMessage} : \forall \text{state}, 0 \le \text{nextMessage state})\ (\text{state} : \text{State})\ 0 \le \text{backwardMessageStep transition emission observation nextMessage state}$$

14.74 fep-074 — Forward–Backward Smoothing Factorization

$$\{\text{State} : \text{Type}^*\}\ [\text{Fintype State}]\ (\text{filtered} : \text{FiniteLaw State})\ (\text{backward} : \text{State} \Rightarrow \mathbb{R})\ (\text{hBackward} : \forall \text{state}, 0 \le \text{backward state})\ (\text{hNormalizer} : 0 < \text{smoothingNormalizer filtered backward})\ (\text{state} : \text{State})\ \text{forwardBackwardSmoothing filtered backward hBackward hNormalizer state} \cdot \text{smoothingNormalizer filtered backward} = \text{filtered state} \cdot \text{backward state}$$

$$\{\text{State} : \text{Type}^*\}\ [\text{Fintype State}]\ \{\text{Observation} : \text{Type}^*\}\ [\text{Fintype Observation}]\ (\text{prior} : \text{FiniteLaw State})\ (\text{transition} : \text{FiniteKernel State State})\ (\text{emission} : \text{FiniteKernel State Observation})\ (\text{observation} : \text{Observation})\ \text{forwardEvidence prior transition emission observation} = \text{backwardEvidence prior transition emission observation}$$

14.75 fep-075 — Smoothing-Marginal Normalization

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{filtered} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{backward} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hBackward} : \forall \mathsf{state}, 0 \le \mathsf{backward}\ \mathsf{state})\ (\mathsf{hNormalizer} : 0 < \mathsf{smoothingNormalizer}$$
$$\mathsf{filtered}\ \mathsf{backward})$$
$$\sum \mathsf{state}, \mathsf{forwardBackwardSmoothing}\ \mathsf{filtered}\ \mathsf{backward}\ \mathsf{hBackward}\ \mathsf{hNormalizer}\ \mathsf{state}$$
$$= 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$\mathsf{boolSmoothing}\ \mathsf{false} + \mathsf{boolSmoothing}\ \mathsf{true} = 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{filtered} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{backward} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$(\mathsf{hZero} : \mathsf{smoothingNormalizer}\ \mathsf{filtered}\ \mathsf{backward} = 0)$$
$$\neg 0 < \mathsf{smoothingNormalizer}\ \mathsf{filtered}\ \mathsf{backward}$$

14.76 fep-076 — One-Step Normalized Variational State Update

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{energy} : \mathsf{State} \Rightarrow \mathbb{R})$$
$$\sum \mathsf{state}, \mathsf{variationalStateUpdate}\ \mathsf{prior}\ \mathsf{energy}\ \mathsf{state} = 1$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{energy} : \mathsf{State} \Rightarrow \mathbb{R})\ (\mathsf{state} : \mathsf{State})$$
$$\mathsf{variationalStateUpdate}\ \mathsf{prior}\ \mathsf{energy}\ \mathsf{state} \cdot \mathsf{boltzmannPartition}\ \mathsf{prior}\ \mathsf{energy}$$
$$= \mathsf{boltzmannWeight}\ \mathsf{prior}\ \mathsf{energy}\ \mathsf{state}$$

$$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$$
$$(\mathsf{prior} : \mathsf{FiniteLaw}\ \mathsf{State})$$
$$\mathsf{variationalStateUpdate}\ \mathsf{prior}\ (\lambda \_ \mapsto 0) = \mathsf{prior}$$

14.77 fep-077 — Two-Level Hierarchical Predictive Factorization

$$\{\mathsf{Upper}\ \mathsf{Middle}\ \mathsf{Observation} : \mathsf{Type}^*\}$$
$$(\mathsf{top} : \mathsf{FiniteLaw}\ \mathsf{Upper})\ (\mathsf{upperKernel} : \mathsf{FiniteKernel}\ \mathsf{Upper}\ \mathsf{Middle})$$
$$(\mathsf{lowerKernel} : \mathsf{FiniteKernel}\ \mathsf{Middle}\ \mathsf{Observation})$$
$$\mathsf{hierarchicalPredictive}\ \mathsf{top}\ \mathsf{upperKernel}\ \mathsf{lowerKernel}$$
$$= (\mathsf{FiniteKernel.comp}\ \mathsf{lowerKernel}\ \mathsf{upperKernel}).\mathsf{predictive}\ \mathsf{top}$$

14.78 fep-078 — Bayesian Model-Averaging Predictive Law

$$\{\mathsf{Model}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Model}]\ [\mathsf{Fintype}\ \mathsf{Observation}]$$
$$(\mathsf{modelPrior} : \mathsf{FiniteLaw}\ \mathsf{Model})$$
$$(\mathsf{modelPredictive} : \mathsf{FiniteKernel}\ \mathsf{Model}\ \mathsf{Observation})\ (\mathsf{observation} : \mathsf{Observation})$$
$$\mathsf{modelAverage}\ \mathsf{modelPrior}\ \mathsf{modelPredictive}\ \mathsf{observation} = \sum \mathsf{model}, \mathsf{modelPrior}\ \mathsf{model}$$
$$\cdot\ \mathsf{modelPredictive}\ \mathsf{model}\ \mathsf{observation}$$

$$\{\mathsf{Model}\ \mathsf{Observation} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Model}]\ [\mathsf{Fintype}\ \mathsf{Observation}]$$
$$(\mathsf{modelPrior} : \mathsf{FiniteLaw}\ \mathsf{Model})$$
$$(\mathsf{modelPredictive} : \mathsf{FiniteKernel}\ \mathsf{Model}\ \mathsf{Observation})$$
$$\sum \mathsf{observation}, \mathsf{modelAverage}\ \mathsf{modelPrior}\ \mathsf{modelPredictive}\ \mathsf{observation} = 1$$

{Model Observation : Type*} [Fintype Model] [Fintype Observation]
modelAverage boolInitialLaw boolAccurateEmission true = 13/20

14.79 fep-079 — Blanket Factorization and Conditional-Mutual-Information Equivalence

{Blanket Internal External Sensory Active : Type*}
(model : ConditionalBlanketModel Blanket Internal External)
(hBlanket : ∀blanket, 0 < model.blanketLaw blanket)
conditionalMutualInformation model = 0 ↔ Factorizes model

{Blanket Internal External Sensory Active : Type*}
(model : StaticModel Internal Sensory Active External)
conditionalMutualInformation (ofStaticModel model) = 0

14.80 fep-080 — Shared-Conditional Mixture Preservation

{Input Output : Type*} [Fintype Input] [Fintype Output]
(weight : ℝ) (hWeightNonneg : 0 ≤ weight) (hWeightLeOne : weight ≤ 1)
(left right : FiniteLaw Input) (kernel : FiniteKernel Input Output)
kernel.joint (mixLaw weight hWeightNonneg hWeightLeOne left right) = mixLaw
weight hWeightNonneg hWeightLeOne (kernel.joint left) (kernel.joint right)

{Input Output : Type*} [Fintype Input] [Fintype Output]
boolHalfMixture false = 1/2 ∧ boolHalfMixture true = 1/2

14.81 fep-081 — Coupled-Subsystem Blanket Composition

{$Blanket_1$ $Blanket_2$ $Internal_1$ $Internal_2$ $External_1$ $External_2$: Type*}
(blanketLaw : FiniteLaw ($Blanket_1$ × $Blanket_2$))
($internal_1$: FiniteKernel $Blanket_1$ $Internal_1$)
($internal_2$: FiniteKernel $Blanket_2$ $Internal_2$)
($external_1$: FiniteKernel $Blanket_1$ $External_1$)
($external_2$: FiniteKernel $Blanket_2$ $External_2$) (blanket : $Blanket_1$ × $Blanket_2$)
mutualInformation (conditionalJoint (coupledBlanketModel blanketLaw $internal_1$
$internal_2$ $external_1$ $external_2$) blanket) = 0

{$Blanket_1$ $Blanket_2$ $Internal_1$ $Internal_2$ $External_1$ $External_2$: Type*}
correlatedBoolBlanket (false, false) = 1/2 ∧ correlatedBoolBlanket
(true, true) = 1/2 ∧ 0 < mutualInformation correlatedBoolBlanket

14.82 fep-082 — Finite Intervention-Kernel Normalization

{Context Value : Type*} [Fintype Context] [Fintype Value]
(chosen : Value) (context : Context)
∑ value, interventionKernel (Context := Context) chosen context value = 1

{Context Value : Type*} [Fintype Context] [Fintype Value]
(chosen : Value) (context : Context)
interventionKernel (Context := Context) chosen context chosen = 1

{Context Value : Type*} [Fintype Context] [Fintype Value]
(chosen other : Value) (hOther : other ≠ chosen) (context : Context)
interventionKernel (Context := Context) chosen context other = 0

14.83 fep-083 — Non-Descendant Intervention Invariance

{Root NonDescendant Mediator Outcome : Type*}
(model : OrderedFourNodeModel Root NonDescendant Mediator Outcome) (root : Root)
nonDescendantMarginal (interventionalJoint model root) = model.nonDescendantLaw

{Root NonDescendant Mediator Outcome : Type*}
mediatorMarginal (interventionalJoint boolOrderedModel false) true = 0
∧ mediatorMarginal (interventionalJoint boolOrderedModel true) true = 1
∧ nonDescendantMarginal (interventionalJoint boolOrderedModel false)
= nonDescendantMarginal (interventionalJoint boolOrderedModel true)

14.84 fep-084 — Ordered Finite Causal Factorization

{Root NonDescendant Mediator Outcome : Type*}
(model : OrderedFourNodeModel Root NonDescendant Mediator Outcome) (root : Root)
(nonDescendant : NonDescendant) (mediator : Mediator) (outcome : Outcome)
orderedJoint model (((root, nonDescendant), mediator), outcome)
= ((model.rootLaw root · model.nonDescendantLaw nonDescendant)
· model.mediatorGivenRoot root mediator) · model.outcomeGivenParents
(nonDescendant, mediator) outcome

{Root NonDescendant Mediator Outcome : Type*}
(model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
$\sum$ state, orderedJoint model state = 1

14.85 fep-085 — Local Markov Property from Ordered Factorization

{Root NonDescendant Mediator Outcome : Type*}
(model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
(nonDescendant : NonDescendant) (mediator : Mediator)
(hEvidence : 0 < mediatorEvidence model mediator)
localMarkovConditional model nonDescendant mediator hEvidence
= (model.mediatorGivenRoot.posterior model.rootLaw mediator hEvidence).product
(model.outcomeGivenParents.row (nonDescendant, mediator))

{Root NonDescendant Mediator Outcome : Type*}
(model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
(nonDescendant : NonDescendant) (mediator : Mediator)
(hEvidence : 0 < mediatorEvidence model mediator)
mutualInformation (localMarkovConditional model nonDescendant mediator
hEvidence) = 0

{Root NonDescendant Mediator Outcome : Type*}
mutualInformation (localMarkovConditional boolOrderedModel false true (by rw
[boolMediatorEvidence_true]; norm_num)) = 0

{Root NonDescendant Mediator Outcome : Type*}
(model : OrderedFourNodeModel Root NonDescendant Mediator Outcome)
(mediator : Mediator) (hZero : mediatorEvidence model mediator = 0)
¬0 < mediatorEvidence model mediator

14.86 fep-086 — Precision-Weighted Prediction-Error Energy

$$\{\text{precision} : \mathbb{R}\}\ (\text{hPrecision} : 0 \le \text{precision})\ (\text{observation estimate} : \mathbb{R})$$
$$0 \le \text{precisionEnergy precision observation estimate}$$

$$\{\text{precision} : \mathbb{R}\}\ (\text{hPrecision} : 0 < \text{precision})\ (\text{observation estimate} : \mathbb{R})$$
$$\text{precisionEnergy precision observation estimate} = 0 \leftrightarrow \text{estimate} = \text{observation}$$

$$\text{precisionEnergy}\ 2\ 1\ 0 = 1$$

14.87 fep-087 — Hierarchical Predictive-Coding Energy Decomposition

$$\{\text{depth} : \mathbb{N}\}\ (\text{precision error} : \text{Fin}\ (\text{depth} + 1) \Rightarrow \mathbb{R})$$
$$\text{hierarchicalEnergy precision error} = \text{precision}\ 0/2 \cdot \text{error}\ 0^2 + \sum \text{level} :$$
$$\text{Fin depth}, \text{precision level.succ}/2 \cdot \text{error level.succ}^2$$

$$\{\text{Level} : \text{Type}^*\}\ [\text{Fintype Level}]\ (\text{precision error} : \text{Level} \Rightarrow \mathbb{R})$$
$$(\text{hPrecision} : \forall \text{level}, 0 \le \text{precision level})$$
$$0 \le \text{hierarchicalEnergy precision error}$$

$$\text{hierarchicalEnergy}\ (\lambda\ \text{level} : \text{Fin}\ 2 \mapsto \text{if level} = 0\ \text{then}\ 2\ \text{else}\ 4)$$
$$(\lambda\ \text{level} : \text{Fin}\ 2 \mapsto \text{if level} = 0\ \text{then}\ 3\ \text{else}\ 1) = 11$$

14.88 fep-088 — Prediction-Error Gradient Identity

$$(\text{precision observation estimate} : \mathbb{R})$$
$$\text{HasDerivAt}\ (\lambda\ \text{candidate} \mapsto \text{precisionEnergy precision observation candidate})$$
$$(-\text{precision} \cdot \text{predictionError observation estimate})\ \text{estimate}$$

$$(\text{precision observation estimate} : \mathbb{R})$$
$$\text{HasDerivAt}\ (\lambda\ \text{candidate} \mapsto \text{precisionEnergy precision observation candidate})$$
$$(\text{precisionEnergyGradient precision observation estimate})\ \text{estimate}$$

$$\text{precisionEnergyGradient}\ 2\ 1\ 0 = -2$$

14.89 fep-089 — Finite Generalized-Coordinate Shift Semigroup

$$\{\text{order} : \mathbb{N}\}\ (\text{first second} : \mathbb{N})\ (\text{jet} : \text{FiniteJet order})$$
$$\text{shift}\ (\text{first} + \text{second})\ \text{jet} = \text{shift first}\ (\text{shift second jet})$$

$$\{\text{order} : \mathbb{N}\}\ (\text{jet} : \text{FiniteJet order})$$
$$(\text{shift}\ 1\ \text{jet}).\text{coefficient order} = 0$$

$$(\text{value velocity} : \mathbb{R})$$
$$(\text{shift}\ 1\ (\text{firstOrderJet value velocity})).\text{coefficient}\ 0 = \text{velocity}$$
$$\wedge\ (\text{shift}\ 1\ (\text{firstOrderJet value velocity})).\text{coefficient}\ 1 = 0$$

14.90 fep-090 — Finite-Jet Generalized-Filtering Correction Equation

$$\{\text{order} : \mathbb{N}\}\ (\text{stepSize} : \mathbb{R})\ (\text{precision} : \mathbb{N} \Rightarrow \mathbb{R})$$
$$(\text{target estimate} : \text{FiniteJet order})\ (\text{degree} : \mathbb{N})$$
$$(\text{generalizedFilteringStep stepSize precision target estimate}).\text{coefficient degree}$$
$$= \text{estimate.coefficient degree} + \text{stepSize} \cdot ((\text{shift}\ 1\ \text{estimate}).\text{coefficient}$$
$$\text{degree} - \text{precisionEnergyGradient}\ (\text{precision degree})\ (\text{target.coefficient degree})$$
$$(\text{estimate.coefficient degree}))$$

$$\{\text{order} : \mathbb{N}\}\ (\text{precision} : \mathbb{N} \Rightarrow \mathbb{R})\ (\text{target estimate} : \text{FiniteJet order})\ (\text{degree} : \mathbb{N})$$
$$\text{HasDerivAt}\ (\lambda\ \text{candidate} \mapsto \text{precisionEnergy}\ (\text{precision degree})$$
$$(\text{target.coefficient degree})\ \text{candidate})\ (\text{precisionEnergyGradient}\ (\text{precision}$$
$$\text{degree})\ (\text{target.coefficient degree})\ (\text{estimate.coefficient degree}))$$
$$(\text{estimate.coefficient degree})$$

$$\{\text{order} : \mathbb{N}\}\ (\text{precision} : \mathbb{N} \Rightarrow \mathbb{R})\ (\text{target estimate} : \text{FiniteJet order})$$
$$(\text{generalizedFlow precision target estimate}).\text{coefficient order}$$
$$= -\text{precisionEnergyGradient}\ (\text{precision order})\ (\text{target.coefficient order})$$
$$(\text{estimate.coefficient order})$$

$$\text{let corrected}$$

14.91 fep-091 — Precision Modulation of Prediction Error

$$\{\text{lower upper} : \mathbb{R}\}\ (\text{hLowerNonneg} : 0 \le \text{lower})\ (\text{hPrecision} : \text{lower} \le \text{upper})$$
$$(\text{observation estimate} : \mathbb{R})$$
$$0 \le \text{precisionEnergy lower observation estimate} \wedge \text{precisionEnergy lower}$$
$$\text{observation estimate} \le \text{precisionEnergy upper observation estimate}$$

$$(\text{scale precision observation estimate} : \mathbb{R})$$
$$\text{precisionEnergyGradient}\ (\text{scale} \cdot \text{precision})\ \text{observation estimate} = \text{scale}$$
$$\cdot\ \text{precisionEnergyGradient precision observation estimate}$$

$$\text{precisionEnergy}\ 2\ 1\ 0 = 2 \cdot \text{precisionEnergy}\ 1\ 1\ 0 \wedge \text{precisionEnergy}\ 1\ 1\ 0$$
$$< \text{precisionEnergy}\ 2\ 1\ 0$$

14.92 fep-092 — Quadratic Predictive-Coding Convergence

$$(\text{precision stepSize target estimate} : \mathbb{R})$$
$$\text{precisionEnergy precision target}\ (\text{predictionUpdate stepSize target estimate})$$
$$= (1 - \text{stepSize})^2 \cdot \text{precisionEnergy precision target estimate}$$

$$\{\text{precision stepSize target estimate} : \mathbb{R}\}\ (\text{hPrecision} : 0 < \text{precision})$$
$$(\text{hStepPositive} : 0 < \text{stepSize})\ (\text{hStepBelowTwo} : \text{stepSize} < 2)$$
$$(\text{hError} : \text{target} \neq \text{estimate})$$
$$\text{precisionEnergy precision target}\ (\text{predictionUpdate stepSize target estimate})$$
$$< \text{precisionEnergy precision target estimate}$$

$$\{\text{stepSize} : \mathbb{R}\}\ (\text{hStepPositive} : 0 < \text{stepSize})\ (\text{hStepBelowTwo} : \text{stepSize} < 2)$$
$$(\text{target estimate} : \mathbb{R})$$
$$\text{Tendsto}\ (\lambda\ \text{iterations} \mapsto \text{predictionError target}\ (\text{iteratePredictionUpdate}$$
$$\text{stepSize target iterations estimate}))\ \text{atTop}\ (\text{nhds}\ 0)$$

$$\text{precisionEnergy}\ 2\ 1\ 0 = 1 \wedge \text{predictionUpdate}\ (1/2)\ 1\ 0 = 1/2$$
$$\wedge\ \text{precisionEnergy}\ 2\ 1\ (\text{predictionUpdate}\ (1/2)\ 1\ 0) = 1/4$$

14.93 fep-093 — Forward and Reverse Finite Path-Law Ratio

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$(\text{hReverse} : \forall \text{path}, 0 < \text{protocol.reverseAligned path})\ (\text{path} : \text{Path})$$
$$\text{pathRatio protocol path} \cdot \text{protocol.reverseAligned path} = \text{protocol.forward path}$$

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$(\sum \text{path}, \text{protocol.forward path} = 1) \wedge \sum \text{path}, \text{protocol.reverseAligned path} = 1$$

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})\ (\text{path} : \text{Path})$$
$$\text{protocol.reversal}\ (\text{protocol.reversal path}) = \text{path}$$

14.94 fep-094 — Entropy Production as Path KL

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$\text{entropyProduction protocol} = \text{finiteKL protocol.forward protocol.reverseAligned}$$

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$(\text{hForward} : \forall \text{path}, 0 < \text{protocol.forward path})$$
$$(\text{hReverse} : \forall \text{path}, 0 < \text{protocol.reverseAligned path})$$
$$\text{entropyProduction protocol} = \sum \text{path}, \text{protocol.forward path}$$
$$\cdot\ \text{pathwiseEntropyProduction protocol path}$$

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$0 \le \text{entropyProduction protocol}$$

14.95 fep-095 — Detailed Fluctuation Symmetry

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$(\text{hForward} : \forall \text{path}, 0 < \text{protocol.forward path})$$
$$(\text{hReverse} : \forall \text{path}, 0 < \text{protocol.reverseAligned path})\ (\text{path} : \text{Path})$$
$$\text{protocol.reverseAligned path} \cdot \text{Real.exp}$$
$$(\text{pathwiseEntropyProduction protocol path}) = \text{protocol.forward path}$$

$$\{\text{Path} : \text{Type}^*\}\ [\text{Fintype Path}]$$
$$(\text{protocol} : \text{FinitePathProtocol Path})$$
$$(\text{hForward} : \forall \text{path}, 0 < \text{protocol.forward path})$$
$$(\text{hReverse} : \forall \text{path}, 0 < \text{protocol.reverseAligned path})\ (\text{hExchangeForward} : \forall$$
$$\text{path}, \text{protocol.forward}\ (\text{protocol.reversal path}) = \text{protocol.reverseAligned path})$$
$$(\text{hExchangeReverse} : \forall \text{path}, \text{protocol.reverseAligned}\ (\text{protocol.reversal path})$$
$$= \text{protocol.forward path})\ (\text{path} : \text{Path})$$
$$\text{pathwiseEntropyProduction protocol}\ (\text{protocol.reversal path})$$
$$= -\text{pathwiseEntropyProduction protocol path}$$

14.96 fep-096 — Integral Fluctuation Theorem

$$
\begin{aligned}
&\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}] \\
&(\mathsf{protocol} : \mathsf{FinitePathProtocol}\ \mathsf{Path}) \\
&(\mathsf{hForward} : \forall \mathsf{path}, 0 < \mathsf{protocol.forward}\ \mathsf{path}) \\
&(\mathsf{hReverse} : \forall \mathsf{path}, 0 < \mathsf{protocol.reverseAligned}\ \mathsf{path}) \\
&\sum \mathsf{path}, \mathsf{protocol.forward}\ \mathsf{path} \cdot \mathsf{Real.exp} \\
&(-\mathsf{pathwiseEntropyProduction}\ \mathsf{protocol}\ \mathsf{path}) = 1
\end{aligned}
$$

14.97 fep-097 — Finite Jarzynski Equality

$$
\begin{aligned}
&\{\mathsf{Path} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Path}] \\
&(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{Path})\ (\mathsf{beta}\ \mathsf{deltaFreeEnergy} : \mathbb{R})\ (\mathsf{work} : \mathsf{Path} \Rightarrow \mathbb{R}) \\
&(\mathsf{hNormalization} : \mathsf{HasJarzynskiNormalization}\ \mathsf{law}\ \mathsf{beta}\ \mathsf{deltaFreeEnergy}\ \mathsf{work}) \\
&\mathsf{exponentialWorkAverage}\ \mathsf{law}\ \mathsf{beta}\ \mathsf{work} = \mathsf{Real.exp}\ (-\mathsf{beta} \cdot \mathsf{deltaFreeEnergy})
\end{aligned}
$$

14.98 fep-098 — Local Detailed Balance and Current Cancellation

$$
\begin{aligned}
&\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}] \\
&(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State}) \\
&(\mathsf{hReversible} : \mathsf{IsReversible}\ \mathsf{law}\ \mathsf{kernel})\ (\mathsf{source}\ \mathsf{target} : \mathsf{State}) \\
&\mathsf{probabilityCurrent}\ \mathsf{law}\ \mathsf{kernel}\ \mathsf{source}\ \mathsf{target} = 0
\end{aligned}
$$

$$
\begin{aligned}
&\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}] \\
&(\mathsf{law} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State}) \\
&(\mathsf{source}\ \mathsf{target} : \mathsf{State})\ (\mathsf{hZero} : \mathsf{kernel}\ \mathsf{target}\ \mathsf{source} = 0) \\
&\mathsf{localAffinity}\ \mathsf{law}\ \mathsf{kernel}\ \mathsf{source}\ \mathsf{target} = 0
\end{aligned}
$$

14.99 fep-099 — Reversible-Chain One-Step KL Dissipation

$$
\begin{aligned}
&\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}] \\
&[\mathsf{Nonempty}\ \mathsf{State}]\ (\mathsf{actual}\ \mathsf{stationary} : \mathsf{FiniteLaw}\ \mathsf{State}) \\
&(\mathsf{kernel} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State})\ (\mathsf{hActual} : \forall \mathsf{state}, 0 < \mathsf{actual}\ \mathsf{state}) \\
&(\mathsf{hStationary} : \forall \mathsf{state}, 0 < \mathsf{stationary}\ \mathsf{state}) \\
&(\mathsf{hKernel} : \forall \mathsf{source}\ \mathsf{target}, 0 < \mathsf{kernel}\ \mathsf{source}\ \mathsf{target}) \\
&(\mathsf{hReversible} : \mathsf{IsReversible}\ \mathsf{stationary}\ \mathsf{kernel}) \\
&\mathsf{finiteKL}\ (\mathsf{kernel.predictive}\ \mathsf{actual})\ \mathsf{stationary} \le \mathsf{finiteKL}\ \mathsf{actual}\ \mathsf{stationary}
\end{aligned}
$$

$$
\begin{aligned}
&\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}] \\
&[\mathsf{DecidableEq}\ \mathsf{State}]\ (\mathsf{actual}\ \mathsf{stationary} : \mathsf{FiniteLaw}\ \mathsf{State}) \\
&\mathsf{finiteKL}\ ((\mathsf{FiniteKernel.identity} : \mathsf{FiniteKernel}\ \mathsf{State}\ \mathsf{State}).\mathsf{predictive}\ \mathsf{actual}) \\
&\mathsf{stationary} = \mathsf{finiteKL}\ \mathsf{actual}\ \mathsf{stationary}
\end{aligned}
$$

$$
\begin{aligned}
&\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}] \\
&0 < \mathsf{entropyProduction}\ \mathsf{irreversibleBoolProtocol}
\end{aligned}
$$

14.100 fep-100 — Categorical Fisher Positivity on Simplex Tangents

$$
\begin{aligned}
&\{d : \mathbb{N}\} \\
&(\mathsf{carrier} : \mathsf{CategoricalFisherCarrier}\ d)\ (\mathsf{tangent} : \mathsf{Fin}\ d \Rightarrow \mathbb{R}) \\
&(\mathsf{hTangent} : \mathsf{IsSimplexTangent}\ \mathsf{tangent})\ (\mathsf{hNonzero} : \mathsf{tangent} \neq 0) \\
&0 < \mathsf{fisherMetric}\ \mathsf{carrier.model}\ \mathsf{tangent}\ \mathsf{tangent}
\end{aligned}
$$

$\{d : \mathbb{N}\}$
(tangent : Fin $2 \Rightarrow \mathbb{R}$) (hTangent : IsSimplexTangent tangent)
(hNonzero : tangent $\neq 0$)
$0 <$ fisherMetric twoCategoricalFisherCarrier.model tangent tangent

$\{d : \mathbb{N}\}$
(tangent : Fin $2 \Rightarrow \mathbb{R}$) (hTangent : IsSimplexTangent tangent)
tangent $= \lambda$ coordinate $\mapsto$ tangent $0 \cdot$ twoCategoricalTangent coordinate

$\{d : \mathbb{N}\}$
IsSimplexTangent twoCategoricalTangent $\wedge$ twoCategoricalTangent $\neq 0$
$\wedge$ fisherMetric twoCategoricalFisherCarrier.model twoCategoricalTangent
twoCategoricalTangent $= 4$

$\{d : \mathbb{N}\}$
duplicatedScoreNullTangent $\neq 0 \wedge$ fisherMetric duplicatedFairBernoulliScoreModel
duplicatedScoreNullTangent duplicatedScoreNullTangent $= 0$

14.101 fep-101 — Fisher Pullback under Reparameterization

{Outcome : Type*} [Fintype Outcome] $\{d\, k : \mathbb{N}\}$
(model : ScoreModel Outcome d) (outer : Matrix (Fin d) (Fin k) $\mathbb{R}$)
(inner : Matrix (Fin k) (Fin d) $\mathbb{R}$) (left right : Fin $d \Rightarrow \mathbb{R}$)
pullbackMetric model (outer $\cdot$ inner) left right $=$ pullbackMetric model outer
(inner.mulVec left) (inner.mulVec right)

{Outcome : Type*} [Fintype Outcome] $\{d\, k : \mathbb{N}\}$
(model : ScoreModel Outcome d) (jacobian : Matrix (Fin d) (Fin k) $\mathbb{R}$)
(tangent : Fin $k \Rightarrow \mathbb{R}$)
$0 \leq$ pullbackMetric model jacobian tangent tangent

14.102 fep-102 — Unbiased Scalar Cramér–Rao Bound under Score Regularity

{Outcome : Type*} [Fintype Outcome]
(model : ScoreModel Outcome 1) (estimator : Outcome $\Rightarrow \mathbb{R}$) (target : $\mathbb{R}$)
(certificate : ScalarCramerRaoCertificate model estimator target)
$1 \leq$ estimatorVariance model.law estimator target $\cdot$ scalarFisher model

14.103 fep-103 — Natural-Gradient Equivariance under an Invertible Full-Rank Chart

{Outcome : Type*} [Fintype Outcome] $\{d : \mathbb{N}\}$
(model : ScoreModel Outcome d) [Invertible (fisherMatrix model)]
(jacobian : Matrix (Fin d) (Fin d) $\mathbb{R}$) [Invertible jacobian]
(covector : Fin $d \Rightarrow \mathbb{R}$)
chartPullbackLower model jacobian (chartCoordinates jacobian (naturalGradient
model covector)) $=$ chartCovector jacobian covector

$\{\mathsf{Outcome} : \mathsf{Type}^*\}$ [Fintype Outcome] $\{d : \mathbb{N}\}$
(law : FiniteLaw Outcome) (left right : Outcome $\Rightarrow \mathbb{R}$) (hRight : $0 < \sum$ outcome, law outcome $\cdot$ right outcome^2)
$(\sum$ outcome, law outcome $\cdot$ left outcome $\cdot$ right outcome$)^2$
$\leq (\sum$ outcome, law outcome $\cdot$ left outcome$^2) \cdot \sum$ outcome, law outcome $\cdot$ right outcome^2

14.104 fep-104 — Mirror-Descent Three-Point Identity

$\{d : \mathbb{N}\}$
(potential : (Fin $d \Rightarrow \mathbb{R}$) $\Rightarrow \mathbb{R}$) (gradient : (Fin $d \Rightarrow \mathbb{R}$) $\Rightarrow$ Fin $d \Rightarrow \mathbb{R}$)
(left middle right : Fin $d \Rightarrow \mathbb{R}$)
bregmanDivergence potential gradient left right $-$ bregmanDivergence potential gradient left middle $-$ bregmanDivergence potential gradient middle right
$=$ coordinatePairing (λ coordinate $\mapsto$ gradient middle coordinate $-$ gradient right coordinate) (λ coordinate $\mapsto$ left coordinate $-$ middle coordinate)

14.105 fep-105 — Bregman Pythagorean Law for an Affine Information Projection

$\{d : \mathbb{N}\}$
(potential : (Fin $d \Rightarrow \mathbb{R}$) $\Rightarrow \mathbb{R}$) (gradient : (Fin $d \Rightarrow \mathbb{R}$) $\Rightarrow$ Fin $d \Rightarrow \mathbb{R}$)
(affineSet : Set (Fin $d \Rightarrow \mathbb{R}$)) (base projection candidate : Fin $d \Rightarrow \mathbb{R}$)
(hProjection : AffineBregmanProjection potential gradient affineSet base projection) (hCandidate : candidate $\in$ affineSet)
bregmanDivergence potential gradient candidate base $=$ bregmanDivergence potential gradient candidate projection $+$ bregmanDivergence potential gradient projection base

$\{d : \mathbb{N}\}$
(potential : (Fin $d \Rightarrow \mathbb{R}$) $\Rightarrow \mathbb{R}$) (gradient : (Fin $d \Rightarrow \mathbb{R}$) $\Rightarrow$ Fin $d \Rightarrow \mathbb{R}$)
(affineSet : Set (Fin $d \Rightarrow \mathbb{R}$)) (base projection candidate : Fin $d \Rightarrow \mathbb{R}$)
(hProjection : AffineBregmanProjection potential gradient affineSet base projection) (hCandidate : candidate $\in$ affineSet) (hNonnegative : 0 $\leq$ bregmanDivergence potential gradient candidate projection)
bregmanDivergence potential gradient projection base $\leq$ bregmanDivergence potential gradient candidate base

14.106 fep-106 — Replicator–Natural-Gradient Equivalence

$\{d : \mathbb{N}\}$
(carrier : CategoricalFisherCarrier d) (fitness : Fin $d \Rightarrow \mathbb{R}$)
IsNaturalGradient carrier.model (λ coordinate $\mapsto$ fitness coordinate $-$ meanFitness carrier.law fitness) (replicatorVector carrier.law fitness)

$\{d : \mathbb{N}\}$
(law : FiniteLaw (Fin d)) (fitness : Fin $d \Rightarrow \mathbb{R}$)
$\sum$ coordinate, replicatorVector law fitness coordinate $= 0$

$\{d : \mathbb{N}\}$
twoCategoricalFitness $0 \neq$ twoCategoricalFitness $1 \wedge$ replicatorVector
twoCategoricalLaw twoCategoricalFitness $\neq 0 \wedge$ IsNaturalGradient
twoCategoricalFisherCarrier.model (λ coordinate $\mapsto$ twoCategoricalFitness
coordinate $-$ meanFitness twoCategoricalLaw twoCategoricalFitness)
(replicatorVector twoCategoricalLaw twoCategoricalFitness)

14.107 fep-107 — Product-Agent Generative Law

$\{State_1\ State_2\ Observation_1\ Observation_2 : \mathsf{Type}^*\}$
(left : FiniteKernel $State_1\ Observation_1$)
(right : FiniteKernel $State_2\ Observation_2$) (state : $State_1 \times State_2$)
$\sum$ observation, FEP.CollectiveInference.productKernel left right state
observation $= 1$

$\{State_1\ State_2\ Observation_1\ Observation_2 : \mathsf{Type}^*\}$
(priorLeft : FiniteLaw $State_1$) (priorRight : FiniteLaw $State_2$)
(kernelLeft : FiniteKernel $State_1\ Observation_1$)
(kernelRight : FiniteKernel $State_2\ Observation_2$) (stateLeft : $State_1$)
(stateRight : $State_2$) (observationLeft : $Observation_1$)
(observationRight : $Observation_2$)
(FEP.CollectiveInference.productKernel kernelLeft kernelRight).joint
(FiniteLaw.product priorLeft priorRight) ((stateLeft, stateRight),
(observationLeft, observationRight)) $=$ kernelLeft.joint priorLeft
(stateLeft, observationLeft) $\cdot$ kernelRight.joint priorRight
(stateRight, observationRight)

14.108 fep-108 — Additive Collective Variational Free Energy

$\{State_1\ State_2 : \mathsf{Type}^*\}$ [Fintype $State_1$] [Fintype $State_2$]
(actualLeft referenceLeft : FiniteLaw $State_1$)
(actualRight referenceRight : FiniteLaw $State_2$) (leftCost : $State_1 \Rightarrow \mathbb{R}$)
(rightCost : $State_2 \Rightarrow \mathbb{R}$) (referenceLeftPositive : $\forall$state, $0 <$ referenceLeft
state) (referenceRightPositive : $\forall$state, $0 <$ referenceRight state)
FEP.CollectiveInference.collectiveVFE actualLeft referenceLeft actualRight
referenceRight leftCost rightCost
$=$ FEP.CollectiveInference.variationalFreeEnergy actualLeft referenceLeft
leftCost $+$ FEP.CollectiveInference.variationalFreeEnergy actualRight
referenceRight rightCost

$\{State_1\ State_2 : \mathsf{Type}^*\}$ [Fintype $State_1$] [Fintype $State_2$]
(left : FiniteLaw $State_1$) (right : FiniteLaw $State_2$) (leftCost : $State_1 \Rightarrow \mathbb{R}$)
(rightCost : $State_2 \Rightarrow \mathbb{R}$)
FEP.CollectiveInference.expectedCost (FiniteLaw.product left right)
(λ state $\mapsto$ leftCost state.1 $+$ rightCost state.2)
$=$ FEP.CollectiveInference.expectedCost left leftCost
$+$ FEP.CollectiveInference.expectedCost right rightCost

14.109 fep-109 — Independent-Agent Expected-Free-Energy Additivity

$\{State_1\ State_2 : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ State_1]\ [\mathsf{Fintype}\ State_2]$
$(\mathsf{predictiveLeft}\ \mathsf{preferenceLeft} : \mathsf{FiniteLaw}\ State_1)$
$(\mathsf{predictiveRight}\ \mathsf{preferenceRight} : \mathsf{FiniteLaw}\ State_2)$
$(\mathsf{ambiguityLeft} : State_1 \Rightarrow \mathbb{R})\ (\mathsf{ambiguityRight} : State_2 \Rightarrow \mathbb{R})$
$(\mathsf{preferenceLeftPositive} : \forall \mathsf{state}, 0 < \mathsf{preferenceLeft}\ \mathsf{state})$
$(\mathsf{preferenceRightPositive} : \forall \mathsf{state}, 0 < \mathsf{preferenceRight}\ \mathsf{state})$
FEP.CollectiveInference.independentCollectiveEFE predictiveLeft preferenceLeft predictiveRight preferenceRight ambiguityLeft ambiguityRight
$=$ FEP.CollectiveInference.expectedFreeEnergy predictiveLeft preferenceLeft ambiguityLeft $+$ FEP.CollectiveInference.expectedFreeEnergy predictiveRight preferenceRight ambiguityRight

$\{State_1\ State_2 : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ State_1]\ [\mathsf{Fintype}\ State_2]$
$(\mathsf{left} : \mathsf{FiniteLaw}\ State_1)\ (\mathsf{right} : \mathsf{FiniteLaw}\ State_2)$
$(\mathsf{ambiguityLeft} : State_1 \Rightarrow \mathbb{R})\ (\mathsf{ambiguityRight} : State_2 \Rightarrow \mathbb{R})$
FEP.CollectiveInference.expectedCost $(\mathsf{FiniteLaw.product}\ \mathsf{left}\ \mathsf{right})$
$(\lambda\, \mathsf{state} \mapsto \mathsf{ambiguityLeft}\ \mathsf{state}.1 + \mathsf{ambiguityRight}\ \mathsf{state}.2)$
$=$ FEP.CollectiveInference.expectedCost left ambiguityLeft
$+$ FEP.CollectiveInference.expectedCost right ambiguityRight

14.110 fep-110 — Unit-Weight Product-of-Experts Pool Normalization

$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$
$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{normalizerPositive} : 0$
$<$ FEP.CollectiveInference.productOfExpertsNormalizer left right$)$
$\sum \mathsf{state}$, FEP.CollectiveInference.unitWeightProductOfExpertsPool left right normalizerPositive state $= 1$

$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$
$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{normalizerPositive} : 0$
$<$ FEP.CollectiveInference.productOfExpertsNormalizer left right$)\ (\mathsf{state} : \mathsf{State})$
FEP.CollectiveInference.unitWeightProductOfExpertsPool left right normalizerPositive state $=$ left state $\cdot$ right state
/FEP.CollectiveInference.productOfExpertsNormalizer left right

14.111 fep-111 — Consensus Mass Conservation

$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$
$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})\ (\mathsf{state} : \mathsf{State})$
FEP.CollectiveInference.consensusLeft left right state
$+$ FEP.CollectiveInference.consensusRight left right state $=$ left state $+$ right state

$\{\mathsf{State} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{State}]$
$(\mathsf{left}\ \mathsf{right} : \mathsf{FiniteLaw}\ \mathsf{State})$
$(\sum \mathsf{state}$, FEP.CollectiveInference.consensusLeft left right state$) + \sum \mathsf{state}$, FEP.CollectiveInference.consensusRight left right state $= 2$

14.112 fep-112 — Contractive Belief-Consensus Convergence

```
{State : Type*} [Fintype State]
(left right : FiniteLaw State) (state : State)
FEP.CollectiveInference.beliefGap
(FEP.CollectiveInference.consensusLeft left right)
(FEP.CollectiveInference.consensusRight left right) state = (1/2 : ℝ)
 · FEP.CollectiveInference.beliefGap left right state
```

```
{State : Type*} [Fintype State]
(left right : FiniteLaw State) (state : State)
Tendsto (λ iterations ↦ FEP.CollectiveInference.beliefGap
(FEP.CollectiveInference.consensusIterate (left, right) iterations).1
(FEP.CollectiveInference.consensusIterate (left, right) iterations).2 state)
atTop (nhds 0)
```

```
{State : Type*} [Fintype State]
FEP.CollectiveInference.beliefGap (FiniteLaw.pointMass true)
(FiniteLaw.pointMass false) true = 1 ∧ FEP.CollectiveInference.beliefGap
(FEP.CollectiveInference.consensusLeft (FiniteLaw.pointMass true)
(FiniteLaw.pointMass false)) (FEP.CollectiveInference.consensusRight
(FiniteLaw.pointMass true) (FiniteLaw.pointMass false)) true = 1/2 ∧ 0
 < FEP.CollectiveInference.beliefGap (FEP.CollectiveInference.consensusLeft
(FiniteLaw.pointMass true) (FiniteLaw.pointMass false))
(FEP.CollectiveInference.consensusRight (FiniteLaw.pointMass true)
(FiniteLaw.pointMass false)) true ∧ FEP.CollectiveInference.beliefGap
(FEP.CollectiveInference.consensusLeft (FiniteLaw.pointMass true)
(FiniteLaw.pointMass false)) (FEP.CollectiveInference.consensusRight
(FiniteLaw.pointMass true) (FiniteLaw.pointMass false)) true
 < FEP.CollectiveInference.beliefGap (FiniteLaw.pointMass true)
(FiniteLaw.pointMass false) true
```

14.113 fep-113 — Coupled-Agent Potential Descent

```
{State : Type*} [Fintype State]
(left right : FiniteLaw State) (state : State)
FEP.CollectiveInference.coupledPotential
(FEP.CollectiveInference.consensusLeft left right)
(FEP.CollectiveInference.consensusRight left right) state = (1/4 : ℝ)
 · FEP.CollectiveInference.coupledPotential left right state
```

```
{State : Type*} [Fintype State]
(left right : FiniteLaw State) (state : State)
(separated : left state ≠ right state)
FEP.CollectiveInference.coupledPotential
(FEP.CollectiveInference.consensusLeft left right)
(FEP.CollectiveInference.consensusRight left right) state
 < FEP.CollectiveInference.coupledPotential left right state
```

14.114 fep-114 — Sub-Gaussian Empirical-Mean Tail Bound

$\{\Omega : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ (\mu : \mathsf{Measure}\Omega)\ \{\mathsf{sampleCount} : \mathbb{N}\}$
$(\mathsf{observables} : \mathsf{Fin}\ \mathsf{sampleCount} \Rightarrow \Omega \Rightarrow \mathbb{R})\ (\mathsf{independent} : \mathsf{iIndepFun}\ \mathsf{observables}\ \mu)$
$(\mathsf{proxyVariance} : \mathsf{Fin}\ \mathsf{sampleCount} \Rightarrow \mathbb{R} \geq 0)\ (\mathsf{subGaussian} : \forall \mathsf{index},$
$\mathsf{HasSubgaussianMGF}\ (\mathsf{observables}\ \mathsf{index})\ (\mathsf{proxyVariance}\ \mathsf{index})\ \mu)\ \{\mathsf{deviation} : \mathbb{R}\}$
$(\mathsf{deviationNonnegative} : 0 \leq \mathsf{deviation})$
$\mu.\mathsf{real}\ \{\mathsf{outcome} | (\mathsf{sampleCount} : \mathbb{R}) \cdot \mathsf{deviation} \leq \sum \mathsf{index}, \mathsf{observables}\ \mathsf{index}$
$\mathsf{outcome}\} \leq \mathsf{Real.exp}\ (-((\mathsf{sampleCount} : \mathbb{R}) \cdot \mathsf{deviation})^2/(2 \cdot \sum \mathsf{index},$
$\mathsf{proxyVariance}\ \mathsf{index}))$

$\{\Omega : \mathsf{Type}^*\}\ \{\mathsf{sampleCount} : \mathbb{N}\}\ (\mathsf{observables} : \mathsf{Fin}\ \mathsf{sampleCount} \Rightarrow \Omega \Rightarrow \mathbb{R})$
$(\mathsf{outcome} : \Omega)$
$\mathsf{FEP.LearningTheory.empiricalMean}\ \mathsf{observables}\ \mathsf{outcome}$
$= (\sum \mathsf{index}, \mathsf{observables}\ \mathsf{index}\ \mathsf{outcome})/\mathsf{sampleCount}$

14.115 fep-115 — Simultaneous Finite-Alphabet Frequency Bound

$\{\Omega\ \mathsf{Alphabet} : \mathsf{Type}^*\}\ [\mathsf{MeasurableSpace}\Omega]\ [\mathsf{Fintype}\ \mathsf{Alphabet}]$
$[\mathsf{DecidableEq}\ \mathsf{Alphabet}]\ (\mu : \mathsf{Measure}\Omega)\ [\mathsf{IsProbabilityMeasure}\ \mu]$
$\{\mathsf{sampleCount} : \mathbb{N}\}\ [\mathsf{NeZero}\ \mathsf{sampleCount}]\ (\mathsf{sample} : \Omega \Rightarrow \mathsf{Fin}\ \mathsf{sampleCount} \Rightarrow \mathsf{Alphabet})$
$(\mathsf{target} : \mathsf{Alphabet} \Rightarrow \mathbb{R})\ (\mathsf{deviation}\ \mathsf{perSymbolFailure} : \mathbb{R})\ (\mathsf{perSymbolBound} : \forall$
$\mathsf{symbol}, \mu.\mathsf{real}\ (\mathsf{FEP.LearningTheory.frequencyDeviationEvent}\ \mathsf{sample}\ \mathsf{target}$
$\mathsf{deviation}\ \mathsf{symbol}) \leq \mathsf{perSymbolFailure})$
$\mu.\mathsf{real}\ (\bigcup \mathsf{symbol}, \mathsf{FEP.LearningTheory.frequencyDeviationEvent}\ \mathsf{sample}\ \mathsf{target}$
$\mathsf{deviation}\ \mathsf{symbol}) \leq \mathsf{Fintype.card}\ \mathsf{Alphabet} \cdot \mathsf{perSymbolFailure}$

$\{\Omega\ \mathsf{Alphabet} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Alphabet}]\ [\mathsf{DecidableEq}\ \mathsf{Alphabet}]\ \{\mathsf{sampleCount} : \mathbb{N}\}$
$[\mathsf{NeZero}\ \mathsf{sampleCount}]\ (\mathsf{sample} : \Omega \Rightarrow \mathsf{Fin}\ \mathsf{sampleCount} \Rightarrow \mathsf{Alphabet})$
$(\mathsf{target} : \mathsf{Alphabet} \Rightarrow \mathbb{R})\ (\mathsf{deviation} : \mathbb{R})\ (\mathsf{symbol} : \mathsf{Alphabet})\ (\mathsf{outcome} : \Omega)$
$\mathsf{outcome} \in \mathsf{FEP.LearningTheory.frequencyDeviationEvent}\ \mathsf{sample}\ \mathsf{target}\ \mathsf{deviation}$
$\mathsf{symbol} \leftrightarrow \mathsf{deviation} \leq |\mathsf{FEP.LearningTheory.empiricalFrequency}\ (\mathsf{sample}\ \mathsf{outcome})$
$\mathsf{symbol} - \mathsf{target}\ \mathsf{symbol}|$

14.116 fep-116 — Finite-Hypothesis PAC-Bayes Loss-Gap Bound

$\{\mathsf{Hypothesis} : \mathsf{Type}^*\}\ [\mathsf{Fintype}\ \mathsf{Hypothesis}]$
$(\mathsf{prior}\ \mathsf{posterior} : \mathsf{FiniteLaw}\ \mathsf{Hypothesis})\ (\mathsf{certificate} :$
$\mathsf{FEP.VariationalDuality.GibbsCertificate}\ \mathsf{Hypothesis})$
$(\mathsf{referenceIsPrior} : \mathsf{certificate.reference} = \mathsf{prior})$
$(\mathsf{empiricalLoss}\ \mathsf{populationLoss} : \mathsf{Hypothesis} \Rightarrow \mathbb{R})\ (\mathsf{inverseTemperature} : \mathbb{R})$
$(\mathsf{inverseTemperaturePositive} : 0 < \mathsf{inverseTemperature})\ (\mathsf{potentialIsLossGap} : \forall$
$\mathsf{hypothesis}, \mathsf{certificate.potential}\ \mathsf{hypothesis} = \mathsf{inverseTemperature}$
$\cdot\ (\mathsf{populationLoss}\ \mathsf{hypothesis} - \mathsf{empiricalLoss}\ \mathsf{hypothesis}))\ (\mathsf{confidence} : \mathbb{R})$
$(\mathsf{confidencePositive} : 0 < \mathsf{confidence})\ (\mathsf{confidenceBelowOne} : \mathsf{confidence} < 1)$
$(\mathsf{logMGFBound} : \mathsf{certificate.logPartition} \leq \mathsf{Real.log}\ \mathsf{confidence}^{-1})$
$\mathsf{FEP.VariationalDuality.expectation}\ \mathsf{posterior}\ \mathsf{populationLoss}$
$\leq \mathsf{FEP.VariationalDuality.expectation}\ \mathsf{posterior}\ \mathsf{empiricalLoss}$
$+\ (\mathsf{FEP.FiniteInformation.finiteKL}\ \mathsf{posterior}\ \mathsf{prior} + \mathsf{Real.log}\ \mathsf{confidence}^{-1})$
$/\mathsf{inverseTemperature}$

{Hypothesis : Type*} [Fintype Hypothesis] (prior : FiniteLaw Hypothesis)
(priorPositive : ∀hypothesis, 0 < prior hypothesis) (hypothesis : Hypothesis)
prior hypothesis ≠ 0

14.117 fep-117 — Posterior-Odds Multiplicative Recursion

{Hypothesis Evidence : Type*} [Fintype Hypothesis] [Fintype Evidence]
(prior : FiniteLaw Hypothesis) (likelihood : FiniteKernel Hypothesis Evidence)
(evidence : Evidence) (evidencePositive : 0 < likelihood.predictive prior
evidence) (favored reference : Hypothesis)
(referencePriorPositive : 0 < prior reference) (referenceLikelihoodPositive : 0
< likelihood reference evidence)
FEP.LearningTheory.posteriorOdds prior likelihood evidence evidencePositive
favored reference = (prior favored/prior reference) · (likelihood favored
evidence/likelihood reference evidence)

{Hypothesis Evidence : Type*} [Fintype Hypothesis] [Fintype Evidence]
(prior : FiniteLaw Hypothesis) (likelihood : FiniteKernel Hypothesis Evidence)
(evidence : Evidence) (evidencePositive : 0 < likelihood.predictive prior
evidence) (hypothesis : Hypothesis) (priorZero : prior hypothesis = 0)
likelihood.posterior prior evidence evidencePositive hypothesis = 0

14.118 fep-118 — Exponential Posterior Concentration from a Likelihood Gap

(priorGood priorBad likelihoodGood likelihoodBad likelihoodGap : ℝ)
(sampleCount : ℕ) (priorGoodPositive : 0 < priorGood)
(priorBadNonnegative : 0 ≤ priorBad)
(likelihoodGoodPositive : 0 < likelihoodGood)
(likelihoodBadNonnegative : 0 ≤ likelihoodBad)
(likelihoodGapNonnegative : 0 ≤ likelihoodGap) (gapBound : likelihoodBad
≤ Real.exp (−likelihoodGap) · likelihoodGood)
FEP.LearningTheory.twoHypothesisPosteriorBad priorGood priorBad likelihoodGood
likelihoodBad sampleCount ≤ (priorBad/priorGood) · Real.exp
(−((sampleCount : ℝ) · likelihoodGap))

FEP.LearningTheory.twoHypothesisPosteriorBad (1/2) (1/2) (3/4) (1/4) 2
= 1/10

14.119 fep-119 — Bayesian-Mixture Log-Loss Regret Bound

{Hypothesis : Type*} [Fintype Hypothesis]
(prior : FiniteLaw Hypothesis) (likelihood : Hypothesis ⇒ ℝ)
(likelihoodNonnegative : ∀hypothesis, 0 ≤ likelihood hypothesis)
(selected : Hypothesis)
prior selected · likelihood selected ≤ FEP.LearningTheory.mixtureEvidence prior
likelihood

$\{\text{Hypothesis} : \text{Type}^*\}\ [\text{Fintype Hypothesis}]$
$(\text{prior} : \text{FiniteLaw Hypothesis})\ (\text{likelihood} : \text{Hypothesis} \Rightarrow \mathbb{R})$
$(\text{likelihoodNonnegative} : \forall \text{hypothesis}, 0 \le \text{likelihood hypothesis})$
$(\text{selected} : \text{Hypothesis})\ (\text{priorPositive} : 0 < \text{prior selected})$
$(\text{likelihoodPositive} : 0 < \text{likelihood selected})$
$-\ \text{Real.log}\ (\text{FEP.LearningTheory.mixtureEvidence prior likelihood}) \le -\text{Real.log}$
$(\text{likelihood selected}) - \text{Real.log}\ (\text{prior selected})$

14.120 fep-120 — Bayes-Factor Multiplicativity and Model-Evidence Update

$(\text{priorOdds firstFavored firstReference secondFavored secondReference} : \mathbb{R})$
$(\text{firstReferenceNonzero} : \text{firstReference} \neq 0)$
$(\text{secondReferenceNonzero} : \text{secondReference} \neq 0)$
$\text{FEP.LearningTheory.bayesFactor}\ (\text{firstFavored} \cdot \text{secondFavored})$
$(\text{firstReference} \cdot \text{secondReference}) = \text{FEP.LearningTheory.bayesFactor firstFavored}$
$\text{firstReference} \cdot \text{FEP.LearningTheory.bayesFactor secondFavored secondReference}$
$\wedge\ \text{FEP.LearningTheory.updatedModelOdds priorOdds}\ (\text{firstFavored} \cdot \text{secondFavored})$
$(\text{firstReference} \cdot \text{secondReference}) = \text{FEP.LearningTheory.updatedModelOdds}$
$(\text{FEP.LearningTheory.updatedModelOdds priorOdds firstFavored firstReference})$
$\text{secondFavored secondReference}$

$\text{FEP.LearningTheory.bayesFactor}\ (3/4)\ (1/4) = 3$
$\wedge\ \text{FEP.LearningTheory.updatedModelOdds}\ 1\ (3/4)\ (1/4) = 3$

$(\text{favored} : \mathbb{R})$
$\text{FEP.LearningTheory.bayesFactor favored}\ 0 = 0$

14.121 fep-121 — Laplace-Smoothing Error Identity

$(\text{successes sampleCount} : \mathbb{N})\ (\text{target} : \mathbb{R})\ (\text{sampleCountPositive} : 0 < \text{sampleCount})$
$(\text{successesAtMost} : \text{successes} \le \text{sampleCount})$
$(\text{targetBounds} : \text{target} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1)$
$\text{laplaceEstimate successes sampleCount} - \text{target} = \text{shrinkage sampleCount}$
$\cdot\ (\text{empiricalRate successes sampleCount} - \text{target}) + \text{laplaceBias sampleCount}$
target

$(\text{sampleCount} : \mathbb{N})$
$\text{shrinkage sampleCount} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1$

14.122 fep-122 — Laplace-Smoothing Bias Bound

$(\text{sampleCount} : \mathbb{N})\ (\text{target} : \mathbb{R})\ (\text{targetBounds} : \text{target} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1)$
$|\text{laplaceBias sampleCount target}| \le 1/((\text{sampleCount} : \mathbb{R}) + 2)$

$\text{laplaceBias}\ 2\ 0 = 1/4 \wedge \text{laplaceBias}\ 2\ 0 \neq 0$

14.123 fep-123 — Absolute-Error Transfer through Laplace Smoothing

$$(\text{successes sampleCount} : \mathbb{N})\ (\text{target error} : \mathbb{R})$$
$$(\text{sampleCountPositive} : 0 < \text{sampleCount})$$
$$(\text{successesAtMost} : \text{successes} \le \text{sampleCount})$$
$$(\text{targetBounds} : \text{target} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1)\ (\text{empiricalErrorBound} :$$
$$|\text{empiricalRate successes sampleCount} - \text{target}| \le \text{error})$$
$$|\text{laplaceEstimate successes sampleCount} - \text{target}| \le \text{shrinkage sampleCount} \cdot \text{error}$$
$$+ 1/((\text{sampleCount} : \mathbb{R}) + 2)$$

$$(\text{successes sampleCount} : \mathbb{N})\ (\text{target error} : \mathbb{R})\ (\text{empiricalErrorBound} :$$
$$|\text{empiricalRate successes sampleCount} - \text{target}| \le \text{error})$$
$$0 \le \text{error}$$

14.124 fep-124 — Squared-Risk Transfer through Laplace Smoothing

$$(\text{successes sampleCount} : \mathbb{N})\ (\text{target} : \mathbb{R})\ (\text{sampleCountPositive} : 0 < \text{sampleCount})$$
$$(\text{successesAtMost} : \text{successes} \le \text{sampleCount})$$
$$(\text{targetBounds} : \text{target} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$(\text{laplaceEstimate successes sampleCount} - \text{target})^2 \le 2 \cdot \text{shrinkage}$$
$$\text{sampleCount}^2 \cdot (\text{empiricalRate successes sampleCount} - \text{target})^2 + 2$$
$$\cdot (1/((\text{sampleCount} : \mathbb{R}) + 2))^2$$

$$\{\Omega : \text{Type}^*\}\ [\text{Fintype}\ \Omega]\ (\text{sampling} : \text{FiniteLaw}\ \Omega)\ (\text{successes} : \Omega \Rightarrow \mathbb{N})$$
$$(\text{sampleCount} : \mathbb{N})\ (\text{target} : \mathbb{R})\ (\text{sampleCountPositive} : 0 < \text{sampleCount})$$
$$(\text{successesAtMost} : \forall \text{outcome}, \text{successes outcome} \le \text{sampleCount})$$
$$(\text{targetBounds} : \text{target} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\text{FEP.VariationalDuality.expectation sampling}\ (\lambda\ \text{outcome} \mapsto (\text{laplaceEstimate}$$
$$(\text{successes outcome})\ \text{sampleCount} - \text{target})^2) \le 2 \cdot \text{shrinkage sampleCount}^2$$
$$\cdot\ \text{FEP.VariationalDuality.expectation sampling}\ (\lambda\ \text{outcome} \mapsto (\text{empiricalRate}$$
$$(\text{successes outcome})\ \text{sampleCount} - \text{target})^2) + 2$$
$$\cdot (1/((\text{sampleCount} : \mathbb{R}) + 2))^2$$

14.125 fep-125 — Bernoulli Brier Excess-Risk Identity

$$(\text{target forecast} : \mathbb{R})$$
$$\text{bernoulliBrierScore target forecast} - \text{bernoulliBrierScore target target}$$
$$= (\text{forecast} - \text{target})^2$$

$$(\text{target} : \mathbb{R})$$
$$\text{bernoulliBrierScore target target} - \text{bernoulliBrierScore target target} = 0$$

14.126 fep-126 — Finite-Law Laplace Brier-Risk Bound

$$\{\Omega : \text{Type}^*\}\ [\text{Fintype}\ \Omega]\ (\text{sampling} : \text{FiniteLaw}\ \Omega)\ (\text{successes} : \Omega \Rightarrow \mathbb{N})$$
$$(\text{sampleCount} : \mathbb{N})\ (\text{target} : \mathbb{R})\ (\text{sampleCountPositive} : 0 < \text{sampleCount})$$
$$(\text{successesAtMost} : \forall \text{outcome}, \text{successes outcome} \le \text{sampleCount})$$
$$(\text{targetBounds} : \text{target} \in \text{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\text{brierExcessRisk sampling}\ (\lambda\ \text{outcome} \mapsto \text{laplaceEstimate}\ (\text{successes outcome})$$
$$\text{sampleCount})\ \text{target} \le 2 \cdot \text{shrinkage sampleCount}^2$$
$$\cdot\ \text{FEP.VariationalDuality.expectation sampling}\ (\lambda\ \text{outcome} \mapsto (\text{empiricalRate}$$
$$(\text{successes outcome})\ \text{sampleCount} - \text{target})^2) + 2$$
$$\cdot (1/((\text{sampleCount} : \mathbb{R}) + 2))^2$$

$$\{\Omega : \mathrm{Type}^*\}\ [\mathrm{Fintype}\ \Omega]\ (\mathrm{sampling} : \mathrm{FiniteLaw}\ \Omega)$$
$$\sum \mathrm{outcome},\ \mathrm{sampling}\ \mathrm{outcome} = 1$$

14.127 fep-127 — Concentration-Event Transfer through Smoothing

$$\{\Omega : \mathrm{Type}^*\}\ [\mathrm{Fintype}\ \Omega]\ (\mathrm{successes} : \Omega \Rightarrow \mathbb{N})\ (\mathrm{sampleCount} : \mathbb{N})\ (\mathrm{target\ error} : \mathbb{R})$$
$$(\mathrm{sampleCountPositive} : 0 < \mathrm{sampleCount})\ (\mathrm{successesAtMost} : \forall \mathrm{outcome},\ \mathrm{successes}$$
$$\mathrm{outcome} \leq \mathrm{sampleCount})\ (\mathrm{targetBounds} : \mathrm{target} \in \mathrm{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$\mathrm{laplaceBadEvent\ successes\ sampleCount\ target\ error} \subseteq \mathrm{empiricalBadEvent\ successes}$$
$$\mathrm{sampleCount\ target\ error}$$

$$\{\Omega : \mathrm{Type}^*\}\ [\mathrm{Fintype}\ \Omega]\ (\mathrm{sampling} : \mathrm{FiniteLaw}\ \Omega)\ (\mathrm{successes} : \Omega \Rightarrow \mathbb{N})$$
$$(\mathrm{sampleCount} : \mathbb{N})\ (\mathrm{target\ error\ failure} : \mathbb{R})$$
$$(\mathrm{sampleCountPositive} : 0 < \mathrm{sampleCount})\ (\mathrm{successesAtMost} : \forall \mathrm{outcome},\ \mathrm{successes}$$
$$\mathrm{outcome} \leq \mathrm{sampleCount})\ (\mathrm{targetBounds} : \mathrm{target} \in \mathrm{Set.Icc}\ (0 : \mathbb{R})\ 1)$$
$$(\mathrm{rawProbabilityBound} : \mathrm{finiteEventProbability\ sampling}\ (\mathrm{empiricalBadEvent}$$
$$\mathrm{successes\ sampleCount\ target\ error}) \leq \mathrm{failure})$$
$$\mathrm{finiteEventProbability\ sampling}$$
$$(\mathrm{laplaceBadEvent\ successes\ sampleCount\ target\ error}) \leq \mathrm{failure}$$

14.128 fep-128 — Observation-Contingent Policy-Tree Recursion

$$\{\mathrm{Belief\ Action\ Observation} : \mathrm{Type}^*\}\ [\mathrm{Fintype\ Belief}]\ [\mathrm{Fintype\ Action}]$$
$$[\mathrm{Fintype\ Observation}]\ (\mathrm{model} : \mathrm{PolicyTreeModel\ Belief\ Action\ Observation})$$
$$\{\mathrm{depth} : \mathbb{N}\}\ (\mathrm{action} : \mathrm{Action})\ (\mathrm{continuation} : \mathrm{Observation} \Rightarrow \mathrm{PolicyTree\ Action}$$
$$\mathrm{Observation\ depth})\ (\mathrm{belief} : \mathrm{Belief})$$
$$\mathrm{policyTreeValue}\ (\mathrm{depth} := \mathrm{depth} + 1)\ \mathrm{model}\ (\mathrm{action}, \mathrm{continuation})\ \mathrm{belief}$$
$$= \mathrm{model.stageCost\ depth\ belief\ action} + \sum \mathrm{observation},\ \mathrm{model.observationLaw}$$
$$\mathrm{belief\ action\ observation} \cdot \mathrm{policyTreeValue\ model}\ (\mathrm{continuation\ observation})$$
$$(\mathrm{model.update\ belief\ action\ observation})$$

$$\{\mathrm{Belief\ Action\ Observation} : \mathrm{Type}^*\}\ [\mathrm{Fintype\ Belief}]\ [\mathrm{Fintype\ Action}]$$
$$[\mathrm{Fintype\ Observation}]\ (\mathrm{model} : \mathrm{PolicyTreeModel\ Belief\ Action\ Observation})$$
$$(\mathrm{belief} : \mathrm{Belief})$$
$$\mathrm{policyTreeValue}\ (\mathrm{depth} := 0)\ \mathrm{model}$$
$$(\mathrm{PUnit.unit} : \mathrm{PolicyTree\ Action\ Observation}\ 0)\ \mathrm{belief} = 0$$

14.129 fep-129 — Finite Policy-Tree Bellman Minimum

$$\{\mathrm{Belief\ Action\ Observation} : \mathrm{Type}^*\}\ [\mathrm{Fintype\ Belief}]\ [\mathrm{Fintype\ Action}]$$
$$[\mathrm{Fintype\ Observation}]\ [\mathrm{Nonempty\ Action}]$$
$$(\mathrm{model} : \mathrm{PolicyTreeModel\ Belief\ Action\ Observation})\ (\mathrm{depth} : \mathbb{N})$$
$$(\mathrm{belief} : \mathrm{Belief})$$
$$\mathrm{optimalTreeValue\ model}\ (\mathrm{depth} + 1)\ \mathrm{belief} = \mathrm{model.stageCost\ depth\ belief}$$
$$(\mathrm{optimalTreeAction\ model\ depth\ belief}) + \sum \mathrm{observation},\ \mathrm{model.observationLaw}$$
$$\mathrm{belief}\ (\mathrm{optimalTreeAction\ model\ depth\ belief})\ \mathrm{observation} \cdot \mathrm{optimalTreeValue}$$
$$\mathrm{model\ depth}\ (\mathrm{model.update\ belief}\ (\mathrm{optimalTreeAction\ model\ depth\ belief})$$
$$\mathrm{observation})$$

{Belief Action Observation : Type*} [Fintype Belief] [Fintype Action]
[Fintype Observation] [Nonempty Action]
(model : PolicyTreeModel Belief Action Observation) (depth : ℕ)
(belief : Belief) (alternative : Action)
model.stageCost depth belief (optimalTreeAction model depth belief) + ∑
observation, model.observationLaw belief (optimalTreeAction model depth belief)
observation · optimalTreeValue model depth (model.update belief
(optimalTreeAction model depth belief) observation) ≤ model.stageCost depth
belief alternative + ∑ observation, model.observationLaw belief alternative
observation · optimalTreeValue model depth
(model.update belief alternative observation)

14.130 fep-130 — Optimal Finite Policy-Tree Existence

{Belief Action Observation : Type*} [Fintype Belief] [Fintype Action]
[Fintype Observation] [Nonempty Action]
(model : PolicyTreeModel Belief Action Observation) (depth : ℕ)
(belief : Belief)
∃tree : PolicyTree Action Observation depth, policyTreeValue model tree belief
= optimalTreeValue model depth belief ∧ ∀alternative : PolicyTree Action
Observation depth, policyTreeValue model tree belief ≤ policyTreeValue model
alternative belief

{Action Observation : Type*} [Fintype Action] [Fintype Observation] (depth : ℕ)
Finite (PolicyTree Action Observation depth)

14.131 fep-131 — Open-Loop Plan Embedding into Policy Trees

{Belief Action Observation : Type*} [Fintype Belief] [Fintype Action]
[Fintype Observation] (model : PolicyTreeModel Belief Action Observation)
{depth : ℕ} (plan : OpenLoopPlan Action depth) (belief : Belief)
policyTreeValue model (openLoopEmbedding (Observation := Observation) plan)
belief = openLoopValue model plan belief

{Action Observation : Type*}
openLoopEmbedding (depth := 0) (Action := Action) (Observation := Observation)
(PUnit.unit : OpenLoopPlan Action 0)
= (PUnit.unit : PolicyTree Action Observation 0)

14.132 fep-132 — Closed-Loop Policy-Tree Dominance over Open Loop

{Belief Action Observation : Type*} [Fintype Belief] [Fintype Action]
[Fintype Observation] [Nonempty Action]
(model : PolicyTreeModel Belief Action Observation) {depth : ℕ}
(plan : OpenLoopPlan Action depth) (belief : Belief)
optimalTreeValue model depth belief ≤ openLoopValue model plan belief

(depth : ℕ)
depth < depth + 1

14.133 fep-133 — Treewise Expected-Free-Energy Decomposition

{Belief State Action Observation : Type*} [Fintype Belief] [Fintype State]
[Fintype Action] [Fintype Observation] (model : EFEPolicyTreeModel Belief State
Action Observation) {depth : ℕ} (tree : PolicyTree Action Observation depth)
(belief : Belief)
policyTreeValue (efePolicyTreeModel model) tree belief = policyTreeValue
(riskAmbiguityPolicyTreeModel model) tree belief

{Belief State Action Observation : Type*} [Fintype Belief] [Fintype State]
[Fintype Action] [Fintype Observation] [Nonempty Action] (model :
EFEPolicyTreeModel Belief State Action Observation) (depth : ℕ)
(belief : Belief)
optimalTreeValue (efePolicyTreeModel model) depth belief = optimalTreeValue
(riskAmbiguityPolicyTreeModel model) depth belief

14.134 fep-134 — Strict Boolean Feedback Advantage

(action : Bool)
policyTreeValue boolFeedbackModel boolFeedbackTree false = 0 ∧ openLoopValue
boolFeedbackModel (boolOpenLoopPlan action) false = 1/2 ∧ policyTreeValue
boolFeedbackModel boolFeedbackTree false < openLoopValue boolFeedbackModel
(boolOpenLoopPlan action) false

(boolFeedbackTree.2 false).1 ≠ (boolFeedbackTree.2 true).1

14.135 fep-135 — Finite-Law Measure Embedding

{State : Type*} [Fintype State]
(law : FiniteLaw State) (state : State)
FEP.NativeBlanket.embeddedLaw law {state} = ENNReal.ofReal (law state)

{State : Type*} [Fintype State]
(law : FiniteLaw State)
FEP.NativeBlanket.embeddedLaw law Ω = 1

14.136 fep-136 — Finite-Law Embedding Injectivity and Expectation Transfer

{State : Type*} [Fintype State]
Function.Injective (FEP.NativeBlanket.embeddedLaw : FiniteLaw State ⇒ Measure
State)

{State : Type*} [Fintype State]
(law : FiniteLaw State) (observable : State ⇒ ℝ)
$\int$ state, observable state ∂ FEP.NativeBlanket.embeddedLaw law = $\sum$ state, law
state · observable state

14.137 fep-137 — Embedded Finite-Kernel Composition

{Input Output : Type*} [Fintype Input] [Fintype Output]
(prior : FiniteLaw Input) (kernel : FiniteKernel Input Output)
FEP.NativeBlanket.embeddedLaw (kernel.predictive prior)
= FEP.NativeBlanket.embeddedKernel kernel ∘ₘ FEP.NativeBlanket.embeddedLaw prior

14.138 fep-138 — Markov-Blanket Rectangle Factorization

{Internal Sensory Active External : Type*}
(model : StaticModel Internal Sensory Active External)
(blanket : Blanket Sensory Active) (internal : Internal) (external : External)
FEP.NativeBlanket.embeddedLaw (staticJoint model)
(FEP.NativeBlanket.blanketCoordinate^{-1}{blanket}∩
FEP.NativeBlanket.internalCoordinate^{-1}{internal}∩
FEP.NativeBlanket.externalCoordinate^{-1}{external}) = ENNReal.ofReal
(model.blanketLaw blanket) · ENNReal.ofReal
(model.internalGiven blanket internal) · ENNReal.ofReal
(model.externalGiven blanket external)

14.139 fep-139 — Native Markov-Blanket Conditional Independence

{Internal Sensory Active External : Type*}
(model : StaticModel Internal Sensory Active External)
CondIndepFun (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate
inferInstance) FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
FEP.NativeBlanket.internalCoordinate FEP.NativeBlanket.externalCoordinate
(FEP.NativeBlanket.embeddedLaw (staticJoint model))

{Internal Sensory Active External : Type*}
letI : MeasurableSpaceBool

14.140 fep-140 — Conditional Independence under Measurable Coarsening

{Internal Sensory Active External : Type*}
{InternalImage ExternalImage : Type*} [MeasurableSpaceInternalImage]
[MeasurableSpaceExternalImage]
(model : StaticModel Internal Sensory Active External)
(internalImage : Internal ⇒ InternalImage)
(externalImage : External ⇒ ExternalImage)
(internalMeasurable : Measurable internalImage)
(externalMeasurable : Measurable externalImage)
CondIndepFun (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate
inferInstance) FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
(internalImage ∘ FEP.NativeBlanket.internalCoordinate)
(externalImage ∘ FEP.NativeBlanket.externalCoordinate)
(FEP.NativeBlanket.embeddedLaw (staticJoint model))

14.141 fep-141 — Native Blanket-Transition Closure

{Internal Sensory Active External : Type*}
(dynamics : Dynamics Internal Sensory Active External) (current : DynamicState
Internal Sensory Active External)
CondIndepFun (MeasurableSpace.comap FEP.NativeBlanket.blanketCoordinate
inferInstance) FEP.NativeBlanket.blanketCoordinate_measurable.comap_le
FEP.NativeBlanket.internalCoordinate FEP.NativeBlanket.externalCoordinate
(FEP.NativeBlanket.embeddedLaw (staticJoint (nextStaticModel dynamics current)))

14.142 fep-142 — Finite Exponential-Family Normalization and Support

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\sum \text{outcome}, \text{family.law parameter outcome} = 1$$

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})\ (\text{outcome} : \text{Outcome})$$
$$0 < \text{family.law parameter outcome}$$

14.143 fep-143 — Affine Exponential-Family Log-Density Ratio

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{left right} : \mathbb{R})\ (\text{outcome} : \text{Outcome})$$
$$\text{Real.log}\ (\text{family.law left outcome}/\text{family.law right outcome}) = (\text{left} - \text{right})$$
$$\cdot\ \text{family.statistic outcome}$$
$$-\ (\text{family.logPartition left} - \text{family.logPartition right})$$

14.144 fep-144 — Finite Log-Partition Gradient

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\text{HasDerivAt family.logPartition}\ (\text{family.mean parameter})\ \text{parameter}$$

14.145 fep-145 — Centered Exponential-Family Score Identity

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})\ (\text{outcome} : \text{Outcome})$$
$$\text{family.score parameter outcome} = \text{family.statistic outcome} - \text{family.mean}$$
$$\text{parameter}$$

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\sum \text{outcome}, \text{family.law parameter outcome} \cdot \text{family.score parameter outcome} = 0$$

14.146 fep-146 — Log-Partition Hessian–Variance–Fisher Identity

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\text{HasDerivAt family.mean}\ (\text{family.variance parameter})\ \text{parameter}$$

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{family} : \text{ScalarExponentialFamily Outcome})\ (\text{parameter} : \mathbb{R})$$
$$\text{family.fisher parameter} = \text{family.variance parameter}$$

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$0 < \text{ScalarExponentialFamily.threeStateFamily.variance}\ 0$$

$$\{\text{Outcome} : \text{Type}^*\}\ [\text{Fintype Outcome}]\ [\text{Nonempty Outcome}]$$
$$(\text{base} : \text{Outcome} \Rightarrow \mathbb{R})\ (\text{hBase} : \forall \text{outcome}, 0 < \text{base outcome})$$
$$(\text{constant parameter} : \mathbb{R})$$
$$(\text{ScalarExponentialFamily.constantStatisticFamily base hBase constant}).\text{fisher}$$
$$\text{parameter} = 0$$

14.147 fep-147 — Exponential-Family KL–Bregman Identity

{Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]
(family : ScalarExponentialFamily Outcome) (left right : ℝ)
finiteKL (family.law left) (family.law right) = family.logPartitionBregman left
right

{Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]
(family : ScalarExponentialFamily Outcome) (parameter : ℝ) (outcome : Outcome)
0 < family.law parameter outcome

14.148 fep-148 — Natural-to-Mean Coordinate Injectivity

{Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]
(family : ScalarExponentialFamily Outcome) {lower upper : ℝ} (hVariance : ∀
parameter ∈ Set.Icc lower upper, 0 < family.variance parameter)
StrictMonoOn family.mean (Set.Icc lower upper)

{Outcome : Type*} [Fintype Outcome] [Nonempty Outcome]
(family : ScalarExponentialFamily Outcome) {lower upper : ℝ} (hVariance : ∀
parameter ∈ Set.Icc lower upper, 0 < family.variance parameter)
Set.InjOn family.mean (Set.Icc lower upper)

14.149 fep-149 — Two-State Continuous-Time Markov Kernel

(rates : TwoStateRates) (time : ℝ) (source : Bool)
∑ target, rates.transition time source target = 1

(rates : TwoStateRates) {time : ℝ} (hTime : 0 ≤ time) (source target : Bool)
0 ≤ rates.transition time source target

TwoStateRates.benchmarkRates.forward = 0.7
∧ TwoStateRates.benchmarkRates.backward = 0.3

14.150 fep-150 — Continuous-Time Identity at Zero

(rates : TwoStateRates) (source target : Bool)
rates.transition 0 source target = if source = target then 1 else 0

14.151 fep-151 — Two-State Chapman–Kolmogorov Semigroup

(rates : TwoStateRates) (left right : ℝ) (source target : Bool)
rates.transition (left + right) source target = ∑ middle, rates.transition left
source middle · rates.transition right middle target

14.152 fep-152 — Two-State Continuous-Time Master Equation

(rates : TwoStateRates) (time : ℝ) (source target : Bool)
HasDerivAt (λ candidate ↦ rates.transition candidate source target) (∑ middle,
rates.generator source middle · rates.transition time middle target) time
∧ HasDerivAt (λ candidate ↦ rates.transition candidate source target) (∑
middle, rates.transition time source middle · rates.generator middle target)
time

14.153 fep-153 — Continuous-Time Stationarity and Detailed Balance

$$(\text{rates} : \text{TwoStateRates})\ (\text{time} : \mathbb{R})\ (\text{target} : \text{Bool})$$
$$(\sum \text{source}, \text{rates.stationaryLaw source} \cdot \text{rates.transition time source target})$$
$$= \text{rates.stationaryLaw target}$$

$$(\text{rates} : \text{TwoStateRates})\ (\text{time} : \mathbb{R})\ (\text{source target} : \text{Bool})$$
$$\text{rates.stationaryLaw source} \cdot \text{rates.transition time source target}$$
$$= \text{rates.stationaryLaw target} \cdot \text{rates.transition time target source}$$

14.154 fep-154 — Exact Two-State Exponential Relaxation

$$(\text{rates} : \text{TwoStateRates})\ (\text{initial} : \text{FiniteLaw Bool})\ (\text{time} : \mathbb{R})$$
$$\text{rates.trueMass initial time} - \text{rates.stationaryTrue} = \text{rates.rho time}$$
$$\cdot\ (\text{initial true} - \text{rates.stationaryTrue})$$

$$\text{TwoStateRates.benchmarkInitial true}$$
$$\neq \text{TwoStateRates.benchmarkRates.stationaryTrue}$$

14.155 fep-155 — Exact Two-State Quadratic Lyapunov Decay

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{initial} : \mathsf{FiniteLaw}\ \mathsf{Bool})\ (\mathsf{time} : \mathbb{R})$$
$$\mathsf{rates.lyapunov\ initial\ time} = \mathsf{rates.rho\ time}^2$$
$$\cdot\ (\mathsf{initial\ true} - \mathsf{rates.stationaryTrue})^2$$

$$(\mathsf{rates} : \mathsf{TwoStateRates})\ (\mathsf{initial} : \mathsf{FiniteLaw}\ \mathsf{Bool})\ (\mathsf{time} : \mathbb{R})$$
$$\mathsf{HasDerivAt}\ (\mathsf{rates.lyapunov\ initial})$$
$$(-2 \cdot \mathsf{rates.decayRate} \cdot \mathsf{rates.lyapunov\ initial\ time})\ \mathsf{time}$$

$$-\ 2 \cdot \mathsf{TwoStateRates.benchmarkRates.decayRate}$$
$$\cdot\ \mathsf{TwoStateRates.benchmarkRates.lyapunov\ TwoStateRates.benchmarkInitial}\ 0 < 0$$