Trading Mechanism Specification

The following document is the current working version of the trading specification; it is subject to change and all assertions should be critiqued and reviewed to ensure validity.

Preamble — Working Model

This document describes the **current working model** of the functionSPACE protocol. It is not final. Definitions, parameters, and algorithms may change as we complete simulations, audits, formal analysis, and live testing. Where trade-offs arise, we will revise the design to preserve the core invariants of the system: participant-funded solvency, determinism, path independence of minting, and transparent, rule-based settlement.

All parameters (e.g., $K, P_0, \mu, \gamma, \tau, \lambda_s, \lambda_d$) and numerical floors (ε_α, ε_dens, ε_sum) are **provisional** and subject to calibration. Text intended as explanation is **illustrative**, while the normative math appears in the referenced sections for state, minting, and settlement; if inconsistencies occur, the normative sections govern. Prior framings based on divergence/KL/Bregman are **deprecated**; the Dirichlet-potential model defined here supersedes them pending further evidence.

1. Introduction & Executive Summary

Today’s systems tax expression and truth in three ways. **Articulation cost**: participants must coerce rich beliefs into coarse, discrete choices, which discards probability mass and creates edge effects. **Confirmation cost**: settlement depends on late, brittle resolution paths that externalize risk and time; capital idles and incentives skew. **Interaction cost**: coordination through order books, auctions, and ad-hoc rules creates path dependence, leakage, and copy trading that dilutes information and rewards timing over accuracy.

These frictions compound on chain. Discretization makes outcomes gameable and liquidity siloed. Delayed confirmation pushes risk into governance and oracles. Frictional interaction makes the price a lagging indicator, not a real-time aggregator of beliefs. A viable mechanism must let beliefs be expressed continuously, keep solvency participant-funded, and make valuation rational and path-independent under strict computational bounds.

The guiding principles and architectural mandate are defined once in §3. This section references §3 and does not restate it.

The architecture in §5 and the mechanics in §6 operationalize these requirements into objects, parameters, and algorithms.

2. Problem statement

2.1 The Foundational Frictions (The "Why")

The mandate for a new economic primitive stems from a deep and pervasive inefficiency in the global economic system. This inefficiency manifests as a significant systemic burden—a hidden tax paid for the market's structural inability to efficiently price and manage complex, continuous risk. This is not a single problem, but a crisis that can be formally deconstructed into a synthesis of three foundational market failures.

1. **The Coasean Friction (The Cost of Transacting):** The work of Ronald Coase demonstrates that when transaction costs are prohibitive, markets fail to reach efficient outcomes. In the context of modern information markets, these costs are rampant. They arise from ill-defined and incomplete property rights over nuanced informational states, forcing participants to incur significant **search costs** to find liquidity, **bargaining costs** in the form of bid-ask spreads and impermanent loss, and massive **enforcement costs** to secure the link between on-chain assets and off-chain reality via oracles.
2. **The Akerlofian Friction (The Cost of Asymmetry):** George Akerlof's work on the "Market for Lemons" shows that acute information asymmetry—where one party knows more than another—can degrade and even collapse markets through adverse selection. This friction is endemic to decentralized finance. It manifests when uninformed liquidity providers are systematically exploited by informed arbitrageurs, and it is the root cause of oracle manipulation, where malicious actors exploit a temporary information advantage to sell the protocol a fraudulent price feed.
3. **The Arrow-Debreu Friction (The Cost of Incompleteness):** The general equilibrium theory of Kenneth Arrow and Gérard Debreu defines a "complete" market as one where a tradable security exists for every possible future state of the world, allowing for the perfect allocation of risk . Any market that fails to provide this full set of state-contingent claims is, by definition, an **incomplete market**, which results in a quantifiable and significant **welfare loss** for its participants. This cost of "in-expression"—the inability to hedge against the full, continuous spectrum of possibilities—is the direct economic damage inflicted by low-capacity information channels.

3. System ethos and goals

This document serves as the technical working specification for the **Expressible Information-Forecasting Surface**, the core trading layer of the functionSPACE protocol. Its purpose is to translate the philosophical "why" articulated in the Mission Document into a concrete architectural "how". This specification provides the rigorous, first-principles-driven blueprint for the mechanism, intended to guide its implementation, mathematical proofing, and validation (Normative math in §7; invariants and tests in §10.8–10.9).

All design decisions contained herein are governed by a strict adherence to the foundational principles of the venture. They are not arbitrary choices but are the logical consequences of the system's overarching vision and ethos.

3.1 The Core Mandate

The **North Star Goal** for this trading layer is to unlock and capitalize the full spectrum of economic expression. The architecture must produce a single, unified surface where any numerical belief can be articulated in its true, continuous form. This surface must be entirely participant-funded, intrinsically fair, and seamlessly fluid, thereby definitively solving the **Cost of State Articulation**.

3.2 The Design Ethos

The architecture of the trading layer is a direct expression of three constitutional commitments that ensure its integrity, neutrality, and resilience .

- **Neutral Simplicity:** The core protocol must be a lean, powerful, and unopinionated primitive. We will aggressively pursue simplicity in the base-layer mechanics, minimizing the protocol's "opinion" on how it should be used. This ensures it can serve as a credibly neutral foundation for a vast ecosystem of applications, pushing complexity to the edges, in line with the End-to-End Principle.
- **Incentives and Emergence:** We will elicit desired system properties through emergent, incentive-driven dynamics rather than enforcing them with deterministic rules. The mechanism is designed as a transparent game where the self-interested, profit-maximizing strategy for any rational agent is the precise action that contributes to the health, accuracy, and liquidity of the overall system. This follows the principles of Incentive Compatibility and spontaneous order.
- **Self-Contained Coherence:** The system's core functions must be sovereign, deriving their integrity from within the system itself. The trading layer's solvency and valuation logic must be **endogenous**, flowing from its own internal economic game without reliance on external capital or dependencies. This reduces the attack surface and ensures the system can make a credible, sovereign promise of its integrity.

3.3 Architectural Imperatives

To fulfill the Core Mandate in alignment with the Design Ethos, the trading layer's mechanism **must** achieve the following six specific and measurable system goals. These imperatives form the pass/fail criteria for the success of the architecture.

3.3.1 Participant-Funded Solvency

The mechanism must be entirely self-funding, with all liquidity and payouts sourced exclusively from participant collateral. The protocol's internal accounting must guarantee bounded loss for this collective pool under all market conditions without reliance on external capital. This is a direct response to the Coasean and Akerlofian frictions that plague subsidized liquidity models.

- **Rationale:** This imperative is a direct response to the **Coasean and Akerlofian frictions** that plague externally subsidized liquidity models. Such models often create new information asymmetries between traders and liquidity providers and introduce their own transaction costs and dependencies, failing to solve the core problem. A self-funding system internalizes liquidity as an emergent property of belief expression itself.

3.3.2 Two-Sided, Symmetric Liquidity

Participants must be able to exit positions symmetrically via the core protocol at any time before final resolution. The redemption of a position must be governed by the same universal laws as its creation. This is a prerequisite for mitigating the Coasean friction of a "capital trap".

- **Rationale:** This is a prerequisite for mitigating the **Coasean friction** of a "capital trap," where the inability to exit imposes a prohibitive transaction cost on participants. It also addresses the **Akerlofian friction** that arises when agents cannot exit positions, causing their "stale" beliefs to degrade the quality of the market's aggregate information.

3.3.3 Rational & Continuous Valuation

The mechanism must be constructed upon a valuation framework that is **rational, transparent, and continuous**. It must provide clear incentives for the revelation of accurate high-fidelity information and a coherent valuation pathway for any position, from its creation to its final settlement.

- **Rationale:** The incentive for **rational participation** is the primary solution to the **Akerlofian friction** of information asymmetry. A **continuous valuation framework** is essential for solving the **Arrow-Debreu friction** of incomplete markets, as it eliminates unpriceable temporal gaps and risks, making a new class of complex, function-based assets rationally manageable.

3.3.4 Fair Pricing of Endogenous Risk

The mechanism must fairly and continuously price the full spectrum of endogenous risks, including **Consensus Risk** and **Metastate Risk** . By pricing this risk spectrum, the protocol creates a complete market for different risk profiles, ensuring that agents with different risk tolerances can all participate on rational terms .

- **Rationale:** By fairly pricing this full risk spectrum, the protocol creates the **Continuum of Certainty Preference**. This continuum is what provides a new and more robust form of **incentive compatibility**, ensuring that agents with different risk tolerances—from early, risk-seeking speculators to late, risk-averse hedgers—can all participate on rational terms. This dynamic is the active ingredient that incentivizes both market creation and sustained participation, solving a deeper layer of the **Arrow-Debreu friction** by creating a complete market for different risk profiles.

3.3.5 Computational & Cognitive Tractability

The mechanism must be executable within the practical economic constraints of a public blockchain, and its state must be comprehensible to a rational agent. Computational intractability manifests as a prohibitive Coasean transaction cost (gas fees), while cognitive intractability creates severe Akerlofian friction between expert and novice users. Both must be solved for the system to be a viable economic primitive.

- **Rationale:** Computational intractability manifests as a literal **Coasean transaction cost** in the form of prohibitive gas fees. Cognitive intractability, as identified in the **Cost of User Interaction**, creates a severe **Akerlofian friction** between expert and novice users, undermining the goal of a fair and accessible system. Both must be solved to create a viable economic primitive.

3.3.6 Truthful Belief Expression: The mechanism must be **incentive-compatible**, ensuring that for any rational agent, the optimal trading strategy to maximize their expected utility is to articulate their true, unbiased probabilistic belief. The system's reward structure must not offer any profitable avenues for the strategic misrepresentation or "gaming" of one's belief.

- **Rationale:** This imperative is the practical implementation of the system's core **epistemic goal**: to produce a maximally clear and **incentive-auditable signal** of collective belief . It is the primary defense against the **Akerlofian frictions** that degrade market quality by ensuring that the information aggregated is of the highest possible fidelity. By making truthfulness the most profitable strategy, we ensure the protocol's resulting price signal is a robust and reliable instrument for societal risk management and capital allocation. This imperative transforms the philosophical commitment to **Incentive Compatibility** into a non-negotiable architectural requirement.

4. Solution overview

The Design Imperatives laid out present a formidable challenge. No mechanism built upon the existing architecture of discrete markets can satisfy them, as that architecture is the very source of the economic frictions we aim to solve. To meet these imperatives, a solution cannot be an incremental improvement; it must be a fundamental **phase change** in the nature of what is being traded.

4.1 The Principle of Continuous Expression

The key architectural unlock for the functionSPACE protocol is the **Principle of Continuous Expression**. This principle mandates a transition from a market of discrete, fungible outcome-tokens to a market where the fundamental traded asset is an information claim expressed as a **continuous, non-fungible function**—a Probability Density Function (PDF).

This is the direct, first-principles answer to the frictions defined in Chapter 1. By enabling the trade of a full PDF, the system creates a new, more complete, and high fidelity **informational property right**. This directly attacks the root of the **Coasean Friction**; instead of forcing participants to incur high transaction costs to approximate a nuanced belief with clumsy, discrete instruments, the protocol allows them to define, own, and trade their precise belief directly.

Furthermore, this transition is the necessary precondition for addressing the **Arrow-Debreu Friction**. A market on PDFs is, by its nature, more complete than one on discrete outcomes. It vastly increases the "channel capacity" of the market, allowing participants to express views and hedge against the entire continuum of possibilities, not just a few predefined points.

This architectural shift from points to functions is what unlocks a richer design space, making it possible to escape the accounting paradoxes of fungible shares. It is the foundational change that enables the new game-theoretic contract detailed in the following section.

4.2 The Fluid Entitlement Contract

The architectural phase change to **Continuous Expression** is not an end in itself; it is the necessary precondition that unlocks a new, more sophisticated game-theoretic contract with participants. This new contract, the **Fluid Entitlement**, is the core conceptual solution that dissolves the paradoxes inherent in prior models.

To understand its significance, we must first contrast it with the rigid structure it replaces. Prior participant-funded markets, such as Pennock's Dynamic Pari-Mutuel (DPM), operate on a contract of **Fixed Entitlement**. In this model, a participant purchases a static, proportional claim on the final payout pool. This rigidity is an unavoidable consequence of a market built on discrete, fungible shares; fairness dictates that every identical share must have an equal claim, a promise of "inter-asset equality". It is this fixed contract that inevitably leads to the "Pennock Paradox"—the inability to offer symmetric, two-sided liquidity without breaking the system's solvency promise.

The functionSPACE protocol is engineered around a fundamentally different promise. By leveraging the unique, non-fungible nature of the **Information Claim** made possible by continuous expression, the protocol offers a **Fluid Entitlement** contract.

- **Definition: The Fluid Entitlement Contract** In this new game, a participant's **Information Claim** does not represent a fixed entitlement to a proportion of the final pool. Instead, it represents a **dynamic entitlement**, where its potential value is continuously appraised against the evolving market aggregate state (ψagg). The protocol's promise is not one of inter-asset equality, which is meaningless for inherently unique assets, but one of **procedural fairness**: the guarantee that every claim will be continuously and fairly appraised against the market consensus via a universal, transparent law.

By abandoning the rigid accounting constraints of fungible shares, the system is freed from the logic-lock that defined the Pennock Paradox. This new, more dynamic contract is the game-theoretic foundation that allows for the construction of a perpetually solvent, participant-funded market with symmetric, two-sided liquidity.

4.3 The Rationality and Purity of the Fluid Entitlement Game

A contract where a participant's entitlement is fluid—subject to change by the subsequent actions of others—could at first appear inherently irrational. Why would a rational agent enter a game where the value of their position could be arbitrarily changed? The answer lies in a new dimension of game theory enabled by operating in the continuous **function space**. The system's rationality is guaranteed by a set of mechanics that transform this apparent flaw into a powerful and equitable feature, creating a richer, higher-fidelity market.

The rationality of this game rests on two pillars: The first: Instantaneous Pricing and Early-Contrarian Advantage, and the second: Metastate Fairness.

Pillar 1: Instantaneous Pricing and Early-Contrarian Advantage

The protocol prices each new position against the live potential gradient of the market state. When a trader commits collateral to underweighted regions of the consensus, the gradient is steep there, so more **Information Claim** is minted per unit collateral than if the same trade were executed after others have shifted the state. As flow moves probability toward those regions the gradient flattens and the same collateral mints fewer claims, so acting first locks a larger integral of that gradient along the trade path. This behavior arises from the curvature of the **Dirichlet potential**, not from any subjective reward for novelty. Crucially, mint depends on both the belief's placement and the size of incoming collateral relative to the current pool: a large commitment into a shallow market can earn a high mint even if the belief is only moderately contrarian, because it materially updates the state. The **Mint Multiplier** is the incremental claims secured by time priority; Settlement weighs each ticket by the claims it minted and how much probability it put on the realized outcome.

Pillar 2: Metastate Fairness via the Continuum of Certainty Preference

The deeper source of rationality emerges over time, through the dynamic nature of the **Unified Collateral Pool**. Because the final size of the pool is unknown at the time of trade, the absolute value of any claim is also unknown. This introduces a new, priceable dimension of risk: **Metastate Risk**, defined precisely as the uncertainty regarding the future size of the Unified Collateral Pool.

The protocol's unlock is that it allows participants to price their own tolerance for this risk. This creates a **Continuum of Certainty Preference**, a new and powerful alignment lever for information markets.

- An agent with a **low demand for certainty** (e.g., a venture-style speculator, an early evangelist) is willing to accept high Metastate Risk. They can enter a nascent market when the pool is small, securing an **Information Claim** with a high first mover dividend. Their primary incentive is the potential for their claim's absolute value to be magnified by future capital inflows that validate their belief in the *importance* of the market itself.
- An agent with a **high demand for certainty** (e.g., a traditional arbitrageur, a late-stage hedger) will only enter a mature market with a large and stable collateral pool. They pay an implicit premium for this certainty in the form of lower dividend, as their conforming belief will have lower impact on the established consensus.

This dynamic directly addresses a form of the **Arrow-Debreu Friction** that plagues prior models. Fixed Entitlement markets, by offering a static reward function, fail to price Metastate Risk. This creates a "chicken-and-egg" problem where no rational actor is incentivized to bootstrap a market until its future importance is already assured. By creating a mechanism to explicitly compensate participants for taking on the risk of a market's nascence, functionSPACE incentivizes market creation and early information revelation that would otherwise never occur, thus helping to complete the market.

Ultimately, the purity of this model lies in its capacity to cleanly decouple a participant's epistemic expression (the shape of their PDF) from their economic risk profile (the Principal committed against a given metastate). In doing so, it creates a higher-fidelity market that aggregates not only *what* its participants believe will happen, but also the collective conviction, risk tolerance, and belief in the importance of the question itself.

4.4 The Core User Action and its Digital Manifestation

To execute the Fluid Entitlement game, the protocol must translate the conceptual act of expressing a belief into a concrete, on-chain transaction. The core user action is the atomic submission of a **(Belief, Principal)** pair, which results in the minting of a unique digital asset that embodies their position.

The submission transaction contains two distinct inputs:

1. **The Belief Vector (ψindiv):** The user’s belief is not an abstract concept but a precise data structure: a **vector of Bernstein Coefficients (c_0, c_1, ..., c)**. This vector must satisfy the intrinsic validity conditions of the basis (i.e., all coefficients are non-negative, and their sum equals 1), which the protocol deterministically verifies upon submission. This is the practical implementation of expressing a full PDF.
2. **The Collateral Principal (C):** The user specifies an amount of the market's designated fungible asset (e.g., USDC). This collateral is transferred from the user and deposited into the market’s single, shared **Unified Collateral Pool**.

Upon successful submission of these two components, the protocol mints and assigns to the user the central asset of the entire system: the **Information Claim**.

Definition: The Information Claim A non-fungible position ticket (NFT) that binds three facts: the submitted belief vector **p**, the committed collateral **C**, and the fixed number of claims **m** minted at entry. Claims are non-transferable and remain attached to the ticket; they move only if the ticket transfers. The ticket’s payout is appraised at settlement by a single public rule using accuracy and the finalized market state; the quantity mmm never changes. It is not a fungible share; it is the container for the holder’s Fluid Entitlement.

5. Architecture and preliminaries

To understand the mechanics of the functionSPACE trading layer, we must first establish a formal vocabulary. This chapter defines the core data structures and mathematical concepts that constitute the market's state and a participant's belief. Each concept is presented with its technical definition, followed by a non-technical explanation of its role and purpose. A firm grasp of these foundational objects—α, P, q, p, and A(α)—is essential for understanding the dynamic processes detailed in subsequent chapters.

5.1.1 The State Vector (α): Pseudo-counts as Cumulative Evidence

Technical Definition. The market state is a vector $\alpha \in R^{K+1}_{>0}$ with components $\alpha_k > 0$ for $k = 0, ..., K$, where K is the fixed Bernstein degree chosen at market creation (see §9.1). The state is initialized by a Dirichlet-style prior of strength $P_0 > 0$. The default is uniform,

$$\alpha_{0,k} = \frac{P_0}{K+1} \quad for\ all\ k,$$

and the implementation must enforce a strict positivity floor $\varepsilon_\alpha > 0$ so that $\alpha_k \geq \varepsilon_\alpha$ at all times (see §10.2,6 for recommended values). Let $P = \sum_k \alpha_k$ and $q = \alpha/P$ (see §5.2).

Trades update the state linearly (see §6.1): a buy of collateral C with belief p applies $\alpha_{new} = \alpha_{old} + C\,p$; a full close applies the exact inverse update.

Conceptual Explanation. Think of α as $K + 1$ evidence buckets. Each trade pours weighted belief into these buckets. When the market is young and uncertain, the buckets are small and even. As conviction accumulates, some buckets grow large, reflecting where participants have focused probability mass. α is the protocol's internal state of this accumulated evidence; public views like the consensus curve derive from it, but α itself is the single source of truth.

Justification. A positive pseudo-count state makes updates simple, safe, and auditable. Linear updates $(\alpha \leftarrow \alpha + Cp)$ preserve path independence at the state layer and keep the domain of special functions used later (e.g., $log\Gamma$, digamma) strictly valid. The prior P_0 sets initial depth, tuning early incentives without any external pricing. The explicit floor ε_α prevents numerical singularities and guarantees deterministic behavior in fixed-point arithmetic.

5.1.2 Total Mass (P) and The Public Consensus (q)

Technical Definition: From the state vector α, we derive two critical public-facing metrics:

- **Total Mass (P):** The total informational mass or pseudo-mass of the market, calculated as the sum of the components of the state vector:

  $$P = \sum_k \alpha_k \text{ and } q = \alpha/P$$

- **Public Consensus (q):** The normalized, public-facing consensus PDF, calculated by dividing the state vector by the total mass: q = α / P. The vector q is a valid coefficient vector, as its components are non-negative and sum to 1.

  The consensus PDF is $f_q(x) = \frac{K+1}{H-L} \sum_{k=0}^{K} q_k B_{k,K}(u(x))$. q are the coefficients; f_q is the density users see.

Conceptual Explanation: If α is the raw vote count, then P is the total number of votes cast. It represents the market's maturity and the total weight of conviction behind the consensus. A small P signifies a young, uncertain market, while a large P signifies a mature, information-rich market. The vector q is the final poll result—the percentage of votes for each outcome. This is the consensus curve that traders see and interact with, representing the market's current best guess of the probability distribution for the final outcome.

Justification: This separation of the internal state (`α`) from the public consensus (`q`) is a cornerstone of the system's robustness. The minting physics are driven by changes in `α` and P (the informational mass), not the total dollar value of the collateral pool. This critical design choice decouples the economic incentives for information contribution from the treasury mechanics, preventing a host of potential manipulation vectors and ensuring the integrity of the consensus signal.

5.1.4 The Dirichlet Potential (`A(α)`): A Measure of Information Content

Technical Definition: The core of the protocol's economic engine is the **Dirichlet Potential**, `A(α)`.

$A(\alpha)$ equals the **negative log-partition** of the Dirichlet:

$A(\alpha) = log\Gamma(\sum_k \alpha_k) - \sum_k log\Gamma(\alpha_k) =- logB(\alpha)$. A is concave

where Γ is the Gamma function, the generalization of the factorial to real and complex numbers (§7.6). This function takes the market's internal state vector `α` as input and outputs a single scalar value representing the total information content of that state.

5.1.4.1 In Plain English: What is "State Energy"?

Think of the Dirichlet Potential `A(α)` as a measurement of the market's total "certainty" or "informational energy."

- A **low energy** state corresponds to an uncertain, low-information market. When a market is new, the `α_k` values are small and evenly spread, reflecting high uncertainty. This state has a low `A(α)` value.
- A **high energy** state corresponds to a certain, information-rich market. As traders add capital and conviction, the `α_k` values grow larger and more concentrated, reflecting increasing certainty about the outcome. This state has a high `A(α)` value.

A trade is valued based on how much it **increases the market's energy**. By submitting a belief, a trader pushes the market from a lower energy state (`α_old`) to a higher energy state (`α_new`). The protocol measures this change in energy (`ΔA = A(α_new) - A(α_old)`) and rewards the trader in direct proportion to their contribution.

Justification: This specific function is the lynchpin of the mechanism's fairness and security. It is a function whose change (`ΔA`) depends only on the start and end states (`α_old` and α_new) concave, not the path taken between them. This **path-independent** property, known as telescoping, makes it impossible to gain an advantage by slicing a large trade into many smaller ones, neutralizing a critical manipulation vector. Its mathematical properties, which will be explored in the next chapters, ensure that adding information is always rewarded and that the reward is proportional to the novelty of the contribution.

The concavity of the Dirichlet Potential is not a flaw; it is the fundamental mathematical reason the mechanism has its most important incentive properties.

Because the function is concave, the increase in potential (ΔA) from adding a fixed amount of evidence (C*p) is much larger when the existing evidence (α_old) is small. As the market matures and the α vector grows, the function flattens out, and the same trade produces a much smaller ΔA.

5.2 The Trader's Belief (p): Representation via Bernstein Polynomials

A trader's belief is encoded as a vector of Bernstein coefficients **p**, where $p \in R^{K+1}$. This vector must satisfy two simple but strict constraints:

1. **Non-negativity:** $p_k \geq 0$ for all $k = 0, ..., K$.
2. **Partition of Unity:** $\sum_{k=0}^{K} p_k = 1$.

These coefficients define a continuous function. However, for this function to be a valid Probability Density Function (PDF) that integrates to 1 over the market's domain, a normalization constant is required. The correctly normalized PDF, $f(x)$, is defined as:

- **Formula:** Define $u(x) = \frac{x-L}{H-L} \in [0, 1]$; all PDF evaluation and settlement use this normalization. If any $p_k = 0$, its log term is $-\infty$; implementations apply a small density floor $\varepsilon_{dens} > 0$ when computing log-densities to avoid underflow while preserving ranking.

$$f(x) = \frac{K+1}{H-L} \sum_{k=0}^{K} p_k B_{k,K}(u(x))$$

  where $B_{k,K}$ are the Bernstein basis polynomials and $u(x)$ is the outcome normalized to $[0, 1]$.

- **Justification:** This normalization constant is mathematically essential. The integral of a single Bernstein basis polynomial $B_{k,K}(u)$ over the domain $[0, 1]$ is $1/(K + 1)$. Since our coefficients p_k are constrained to sum to 1, the integral of the unscaled sum $\sum_k p_k B_{k,K}(u)$ would also be $1/(K + 1)$. Therefore, we must multiply by $(K + 1)$ to ensure the function integrates to 1 over $[0, 1]$. The further division by $(H - L)$ is the standard change-of-variables scaling to correctly normalize the PDF over the market's specific domain $[L, H]$. This construction guarantees that any valid p vector corresponds to a valid PDF without requiring complex on-chain validation.

Conceptual Explanation: Instead of choosing a single outcome, a trader expresses their belief by "drawing" a complete probability curve. The vector p represents the set of "control points" that define the shape of this curve. While a user will interact with intuitive tools like sliders or drag-and-drop interfaces to shape their belief, these interfaces translate the visual curve into a precise p vector for the protocol to process. This allows for an immense variety of shapes, from sharp, confident peaks to broad, uncertain distributions or even multi-modal beliefs.

Justification: Bernstein polynomials were chosen as the basis for representing beliefs due to their exceptional numerical stability and intuitive properties. The constraints on the coefficients provide a computationally simple and foolproof way to ensure that any belief submitted by a user is a valid PDF, eliminating the need for complex on-chain validation and making the system inherently robust against malformed inputs.

5.3 Environment constraints

The Fluid Entitlement game in continuous function space only works if it lives inside a physically coherent, computationally viable system. This chapter shows how we achieve that. The stack has two layers: a **constraint layer** (what the blockchain allows, cheaply and deterministically) and a **physics layer** (how the protocol converts beliefs and collateral into state changes and payouts). Together they satisfy the design imperatives from Chapter 3.3.

5.3.1 The Foundational Constraint: The Blockchain Environment & The Bernstein Basis

Public blockchains are not neutral canvases. They impose hard costs and determinism: every read, write, and arithmetic operation consumes gas and must produce the same result on every node. Any mechanism that relies on heavy numerics, iterative solvers, or unconstrained data structures will fail in practice. This directly addresses our need for **computational and cognitive tractability**.

Constraint-informed choice. To trade beliefs as continuous objects without breaking cost or determinism, we encode each belief as a fixed-length **Bernstein coefficient vector** of degree K (set at market creation). The vector has $K + 1$ entries that are non-negative and sum to one. The contract can verify these two checks cheaply and deterministically on every submission.

Why Bernstein for this environment.

- **Validity by construction.** Non-negative, sum-to-one coefficients guarantee the submitted curve is a proper probability density after a known normalization(§5.2); no on-chain integration or shape repair is needed.

- **Linear aggregation.** Market state updates are simple additions of weighted coefficient vectors, so consensus updates and reversals are predictable and audit-friendly.

- **Numerical stability.** The basis is shape-preserving and well-behaved on [L,H], avoiding oscillations and edge artifacts that would force expensive safeguards. (Bernstein basis is variation-diminishing on L,H]; no Gibbs oscillations.)

- **Cost predictability.** All core paths scale linearly in K. Creators choose K once, which fixes per-trade and per-settlement workload ex ante.

- **Protocol alignment.** This representation meshes with the protocol's physics: minting depends on a state potential that reads the same state vector, and settlement evaluates densities at a single outcome using stable, closed-form routines using uses log-sum-exp with precomputed $log\binom{K}{k}$ table.

What we avoid. We do not accept arbitrary functions, adaptive meshes, or per-trade numerical integration. We do not depend on floating nondeterminism. We keep arithmetic in fixed-point and push any heavy calibration off-chain.

5.3.2 Design Imperative (Computational & Cognitive Tractability)

A continuous-belief market must run inside a hostile environment: every node computes the same result, computation costs gas, and nondeterminism is unacceptable. Tractability here means two things: **the math runs cheaply and deterministically on-chain**, and **the mental model is simple enough for users and auditors to follow**.

Representation. Each belief is a fixed-length Bernstein **coefficient vector** of degree K set at market creation. Components are non-negative and sum to one. The contract checks these two conditions deterministically on submission. A known normalization turns the vector into a valid PDF over $[L, H]$. No on-chain integration, no adaptive meshes.

State and updates. The market state is a positive vector α\alpha. A buy adds weighted coefficients to state; a full close removes the same mass. These are linear, audit-friendly updates. No solvers, no iterative fitting.

Minting physics. The protocol mints claims from a single scalar: the **change in Dirichlet potential** ΔA between the before/after states. Both evaluations use the same deterministic implementation of $log\Gamma$ and the same rounding, so the mint **telescopes**: slicing or reordering a trade cannot change the total. Complexity is linear in K.

Settlement physics. At resolution the contract evaluates each ticket's log-density at the realized outcome using **log-sum-exp** with a small set of precomputed constants (e.g., a table of log-binomials for the chosen K). This keeps arithmetic in stable log-space and avoids overflow/underflow. Scores are tempered and normalized in fixed-point; totals use tiny floors to prevent divide-by-zero without changing rankings.

Numerical discipline. Everything runs in fixed-point with a single rounding rule (ties-to-even). State stays strictly positive; densities apply a tiny floor only for logging.

Special functions use a published, bounded approximation. Identical inputs produce identical outputs on all nodes.

$$\hat{log}\Gamma(x) = \{Lanczos(x),\ x \le x_{sw},\ Stirling(x),\ x > x_{sw},\ |\hat{log}\Gamma_{Lanczos}\left(x_{sw}\right) - \hat{log}\Gamma_{Stirling}\left(x_{sw}\right)| \le 2\,ulp.$$

You get a fast, stable $log\Gamma$ with low error for small–medium x (Lanczos) and correct asymptotics for large x (Stirling), without a jump at x_{sw}. That prevents client divergence and rounding mismatches that would break telescoping and determinism.

Predictable cost. All hot paths are $O(K)$ with no hidden branches:

- Submission checks are constant per coefficient.
- Minting is two potential evaluations plus vector adds.
- Settlement per ticket is one pass over $K + 1$ terms.

Creators pick K once, which fixes per-trade and per-settlement workload up front.

Cognitive tractability. The user model is short: draw a belief, choose collateral, get a fixed number of claims; later, payout is computed by a published rule that blends **claim share** and **accuracy** at the realized outcome. Path independence guarantees that execution tactics do not change economics. Auditors can re-compute every step from public state.

Deliberate exclusions. No arbitrary function uploads, no divergence-based pricing, no floating-point, no on-chain quadrature, no iterative solvers, no per-trade matrix inversions, ~~no midstream re-appraisa~~l. The protocol trades flexibility and some raw performance for simplicity, determinism, and verifiability.

Result. Bernstein coefficients give enough expressivity for real forecasts while keeping submission, minting, and settlement **gas-bounded, deterministic, and explainable**—meeting the computational and cognitive tractability imperative.

5.3.3 The Emergent Physics: Information Geometry & The Three Core Principles

The decision to use the **Bernstein Basis** is more than an engineering solution for computational tractability; it is the choice that enables a new, elegant, and powerful physics to emerge. Information Geometry … a statistical manifold, and to describe the evolution of the market as motion within that landscape. This framework is organized by three foundational laws that explain how the architecture works in practice.

Information Geometry is the branch of mathematics that applies the tools of differential geometry to the study of probability distributions. It allows us to treat the space of all possible market beliefs as a tangible, structured landscape, a **statistical manifold**, and to describe the evolution of the market as motion within that landscape. This framework is governed by three foundational laws, which supersede all prior axiomatic models and provide the rigorous proof for how our architectural solution is possible(§7.*, §10.*).

Principle I: The Manifold Postulate

The complete state of a continuous information market is the positive vector $\alpha \in R^{K+1}_{>0}$; the public consensus uses the normalized coefficients $q = \alpha/P$, which lie in the interior of the K-simplex. The Bernstein coefficients are the coordinates on this manifold.

- **Significance & Analogy (The Navigator's Chart):** This principle asserts the market’s state is a precise object with explicit coordinates, not a vague notion. We are operating on a high-resolution **nautical chart** (the statistical manifold). The aggregate market belief (**q**) is the ship's exact **position** on this chart, and the value of our protocol lies in knowing this position with geometric precision. This provides a measurable state, a prerequisite for satisfying **Design Imperative 3.3.5(Cognitive Tractability)**.

Principle II: Potential

A trade induces a linear transition of the state-point α in the manifold's coordinate space. Because the Dirichlet potential A is concave, ΔA is larger when you add mass to underweighted regions and when the incoming collateral is large relative to the total mass P; the same refinement mints less in a deep market. The ex-ante reward for this transition,the trader's informational contribution,is quantified not by a divergence or distance metric, but by the exact change in the system's total potential energy, ΔA, generated by the Dirichlet Potential A(α).

- **Significance & Analogy (Charging the Battery):** This principle replaces prior, more abstract models with the concrete physics of the implemented system. It establishes that a participant's reward is directly proportional to the increase in the market's total "certainty" or "informational energy." This provides the rigorous, first-principles justification for the claim minting formula (`m = μ * ΔA`) and the core incentive for contributing novel information (§7.2.2).
  We can visualize the market's informational state as a battery, whose charge level is measured by the potential `A(α)`. A new, uncertain market is a battery with a low charge; a mature, high-conviction market is a nearly full one. A trader's collateral is the power source, and their belief directs how that power is applied. A trade is the act of charging the battery. The reward minted to the trader is directly proportional to how much their action increases the battery's total charge (`ΔA`). This explains why adding information to a "low-charge" state—either early in a market's life or against the consensus—yields a greater reward, fulfilling the imperative to incentivize price discovery. This is the direct justification for the mint rule $m = \mu \times \Delta A$ in §6.2.

Principle III: Two-Part Valuation

The total value of a position is determined by two distinct and mathematically asymmetric functions. The first part, the Claim Share (`s_i`), is an *ex-ante* measure of a trader's cumulative contribution to the system's potential energy (`ΔA`) over time. The second part, the Accuracy Share (`a_i`), is an *ex-post*, competitive evaluation of a ticket's final accuracy relative to the true outcome and all other participants.

- **Significance & Analogy (The Logbook and the Finish-Line Photo):** This principle formally replaces the outdated "Duality Principle" with the more accurate principle of **Separation of Concerns**. It provides the foundational logic for the final payout algorithm ($w_i = (s_i)^{\lambda_s}(a_i)^{\lambda_d}$), which is explicitly non-dualistic. The system uses two different "rulers" to measure a participant's merit: one for their contribution to the journey and one for their precision at the destination. This is analogous to a motorsport event where the final prize is determined by two separate phases.
-

The **Claim Share (`s_i`)** is determined in a high-risk, high-reward "qualifying" phase where each trade is a single attempt to make the most significant impact on the market. In our racing analogy, this is the one qualifying lap where a driver can take a risky and **aggressive, contrarian** line—a gamble that could lead to a record-breaking time. In the protocol, such an early or contrarian trade creates a large **impact (`ΔA`)** on the market's state, which in turn mints a disproportionately large number of **claims (`m_i`)** . The risk, however, is that this contrarian belief is simply wrong, meaning the trader has secured a massive starting advantage for a race they are destined to lose. This advantage is your final **Claim Share (`s_i`)**, calculated as your minted claims (`m_i`) relative to the total claims from all other participants (`M_total`).
The **Accuracy Share (`a_i`)** is the final, decisive phase of the payout, represented by the **Photo Finish** of the race. This phase disregards your starting advantage and judges one thing only: your exact position at the single, pre-determined finish line, which represents the true outcome (`x*`). The protocol acts like a race official examining this photo finish, evaluating the probability density of each ticket's belief curve assigned to this outcome. Having the highest density gives a massive, disproportionate advantage. The system's **tempered softmax** calculation ensures this is a **winner-take-most** contest; the difference between first and second place in the photo is rewarded exponentially, while the shares for those further back diminish rapidly . Your final**Accuracy Share** is this calculated portion of the prize, a pure measure of your final, relative precision at the moment of truth.

- ~~First is the **Logbook (Claim Share)**. Throughout the voyage (the trading phase), your logbook meticulously records every significant and novel course correction you initiated (ΔA), proving your influence on the fleet's trajectory. A thick logbook with many impactful entries gives you a large claim on the final prize pool. Second is the **Finish-Line Photo (Accuracy Share)**. Upon arrival (settlement), a photo is taken. Your score here depends solely on your final position. You are judged on how close you were to the exact finish line (x*) *relative to everyone else in the photo*. Being the closest gives you the highest score, but another racer's superior accuracy diminishes yours. A large payout requires both a well-documented journey of contribution and a prominent place in the final photo. This law provides the rigorous foundation for our two-part valuation framework, guaranteeing that both early conviction and final accuracy are rewarded in a balanced and incentive-compatible manner.~~

6. The Market Lifecycle: A Conceptual Overview

Purpose: To provide a high-level narrative of a market's life, introducing core concepts from a trader's perspective before the detailed technical breakdown.

This chapter walks through the lifecycle of a functionSPACE market to build an intuitive understanding of its core dynamics. We will follow a market's journey from its creation by a single user, through a dynamic trading phase where a collective consensus is shaped, and finally to settlement, where the pool of capital is redistributed based on participants' contributions and accuracy. This narrative introduces the fundamental concepts and illustrates the incentive structures that emerge from the protocol's mechanics, grounding the "how" in a relatable "why."

6.1 Market Creation

Note - Market creation detail is TBD

Every market begins as a question about a future numerical reality. A **Market Creator**, who could be any user, instantiates this question on-chain with a single transaction. They start by defining an objectively verifiable, continuous outcome, such as "What will the price of Gold per ounce be on September 30, 2025?".

To make the market tradable, the creator sets two key parameters:

- **The Outcome Range `[L, H]`:** This defines the lower and upper bounds of possible outcomes (e.g., $2,000 to $4,000) - ([L < H]).
- **Resolution Criteria:** The specific source of truth that will be used to determine the final outcome.

The creator's final step is to become the market's first participant. They submit their own belief about the outcome, expressed as a complete Probability Density Function (PDF) and backed by their own capital. This inaugural trade seeds the unified collateral pool and establishes the market's **genesis state**—the first, tentative consensus from which all subsequent activity will evolve.

This process directly reflects the design ethos of **Neutral Simplicity** and **Incentives and Emergence**. The protocol provides a simple, unopinionated, and permissionless primitive for market creation. It does not dictate what can be traded; rather, it empowers creators to bootstrap markets they have conviction in, with their own capital serving as the first signal of belief.

6.2 The Trading Phase: Shaping the Consensus

Once a market is live, it becomes a dynamic surface where any participant can express their belief. The collection of all beliefs, weighted by the capital behind them, forms the public **consensus curve (q)**—a continuously updated PDF representing the market's collective wisdom. Every new trade "pushes against" this consensus, and the nature of that push determines a trader's potential reward.

Consider a few archetypal traders:

- **The Early Contrarian:** An early participant examines the market and finds the creator's initial consensus too conservative. She holds a strong, differing view and submits her own PDF backed by significant collateral. Because her belief adds substantial **new information** to a market with very little existing capital (a "shallow" pool), the mechanism rewards her contribution significantly. For each dollar she commits, she mints a large number of non-transferable **claims** (`m`), which represent her ownership stake in the final payout pool.
- **The Consensus Follower:** Later, the market has attracted more capital and the consensus has stabilized. A new trader arrives who generally agrees with the current consensus. He submits a belief curve that is very similar to the public q. His trade still contributes valuable capital to the unified pool, but because his belief is not novel, it provides little new information. As a result, his trade mints far fewer claim tokens per dollar than the early contrarian's did.
- **The Late, High-Conviction Expert:** As the resolution date approaches, new, high-quality information becomes available. An expert in the field develops a very precise forecast. She enters the now-mature market with a large trade and a very "sharp" or "peaky" PDF, confidently asserting that the outcome will fall within a narrow range. Even though she is "late" and the pool is large, her high-conviction trade can still materially shift the consensus and earn a meaningful number of claim tokens with diminishing returns (§7.6). More importantly, her sharp, accurate belief positions her perfectly to win the "accuracy" component of the payout at settlement.

This entire phase is a clear expression of **Incentives &Emergence**. The protocol doesn't command traders to be truthful; it creates a transparent economic game where the dominant

strategy is to express one's true belief. The physics of the minting mechanism naturally reward informative, differentiated reports while making lazy strategies like blindly copying the consensus economically weak.
Design intent; proof obligation in §7.6/Appendix: show truthful reporting is a best response under $m = \mu\Delta A$ and w_i

6.3 The Settlement Phase: From Consensus to Payout

The trading phase ends when the event occurs and the true outcome,x*, is provided by the protocol's Integrity Layer. At this point, the entire, unified collateral pool is distributed among the eligible ticket holders. The payout calculation is not a simple "winner-takes-all" lottery; it's a nuanced process that honors both a trader's informational contribution and their final accuracy.

A ticket's final payout is determined by two factors:

1. **Claim Share (s_i):** This is the proportion of total claim tokens held by a ticket. It represents the owner's share of influence—a measure of how much valuable information they contributed to shaping the consensus throughout the trading phase.
2. **Accuracy Share (a_i):** This is determined by how "correct" a ticket's PDF was. It is measured by the probability density the curve assigned to the true outcome `x\*`. The system functions as a sharp, contest-style game: an eligibility gate (`α-gate`) ensures that only tickets with a competitive density at the outcome can receive a payout, preventing "participation trophies" for wildly inaccurate beliefs.

The final payout is a function of both shares, ensuring a balance between rewarding those who built the consensus and those who were most accurate at the end.

This lifecycle is governed by the ethos of **Self-Contained Coherence** and the imperative of **Participant-Funded Solvency**. The entire process is endogenous. The liquidity is sourced from participants, the rewards are determined by internal mechanics, and the final payouts are distributed from the same pool, guaranteeing the system is always solvent and its integrity flows from within.

7. Mechanics (single source of truth for formulas)

7.1 Market Creation

Note - Market creation detail is TBD

Every market begins as a question about a future numerical reality. A **Market Creator**, who could be any user, instantiates this question on-chain with a single transaction. They start by defining an objectively verifiable, continuous outcome, such as "What will the price of Gold per ounce be on September 30, 2025?".

To make the market tradable, the creator sets two key parameters:

- **The Outcome Range `[L, H]`:** This defines the lower and upper bounds of possible outcomes (e.g., $2,000 to $4,000).
- **Resolution Criteria:** The specific source of truth that will be used to determine the final outcome.

The creator's final step is to become the market's first participant. They submit their own belief about the outcome, expressed as a complete Probability Density Function (PDF) and backed by their own capital. This inaugural trade seeds the unified collateral pool and establishes the market's **genesis state**—the first, tentative consensus from which all subsequent activity will evolve.

This process directly reflects the design ethos of **Neutral Simplicity** and **Incentives and Emergence**. The protocol provides a simple, unopinionated, and permissionless primitive for market creation. It does not dictate what can be traded; rather, it empowers creators to bootstrap markets they have conviction in, with their own capital serving as the first signal of belief.

7.2 Buy transaction

The fundamental interaction with the functionSPACE trading layer is the "buy" transaction. This is the atomic operation through which a participant submits their belief, commits collateral, influences the market consensus, and receives a reward for their contribution. This chapter dissects this operation into its sequential, immutable steps. Understanding this flow is key to understanding how the protocol translates the abstract concepts from Chapter 3 into a concrete, functioning economic mechanism.

7.2.1 Step 1: Updating the State (Superposition)

The process begins when a trader submits their belief vector p and commits an amount of collateral C. The protocol's first action is to update its internal state vector α by additively blending this new information. This is the principle of superposition, where new evidence is layered on top of all existing evidence.

- **Internal State Update Formula:** $P_{old} = \sum_{k} \alpha_{old,k}$
- **Public Consensus Update Formula:** $q_{new} = \left(P_{old} q_{old} + Cp\right) / \left(P_{old} + C\right)$, where `P = Σα_old`
- . Then

- **Application:** The trader's belief vector p is scaled by their committed collateral `C`. This product, `C * p`, represents the trader's weighted contribution to the market's evidence pool. This vector is then added to the market's existing state vector, `α_old`, to produce the new state, `α_new` . This change is reflected in the public consensus q, which updates as a collateral-weighted average of the previous consensus and the new trade. This contribution remains part of the market's state for as long as the position ticket remains open. If the ticket is fully closed before final settlement, this state change is precisely **reversed** (`α_old = α_new - C * p`), and any claims minted by the ticket are burned.[1]

7.2.2 Step 2: Minting Claim Tokens (m)

Technical Definition. After updating the state (see §7.1), the protocol mints claim tokens in exact proportion to the increase in the Dirichlet potential:

$$m \;=\; \mu \cdot \Delta A, \qquad \Delta A \;=\; A(\alpha_{after}) - A(\alpha_{before}),$$

with $A(\alpha) \;=\; log\Gamma\left(\sum_k \alpha_k\right) - \sum_k log\Gamma(\alpha_k)$ and $\mu > 0$ the market's mint scale (see §8.1). To preserve path independence under fixed-point arithmetic, implementations must compute ΔA as

$$\Delta A \;=\; \hat{A}(\alpha_{after}) - \hat{A}(\alpha_{before})$$

using the same deterministic approximation $\hat{A}$ for both evaluations, with identical rounding and parameterization (see §8.3).

Algorithm.

1. Read α_{before}, compute $\alpha_{after} = \alpha_{before} + Cp$.
2. Enforce domain safety: $\alpha_{k,before} \geq \varepsilon_\alpha$ and $\alpha_{k,after} \geq \varepsilon_\alpha$ for all k (see §8.3).
3. Evaluate $\hat{A}$ at both states and form $\Delta A = \hat{A}(\alpha_{after}) - \hat{A}(\alpha_{before})$.
4. Compute raw claims $m_{raw} = \mu \cdot \Delta A$. Quantize to the on-chain unit with deterministic rounding (tie-to-even).
5. If m rounds to zero, mint 0; otherwise record m on the ticket, increment(minimum trade gate likely) $M_{totalClaims} \leftarrow M_{totalClaims} + m$.

Properties.

- **Non-negativity**: $m \geq 0$. Each α_k increases, and the gradient $\partial A/\partial\alpha_k = \psi(P) - \psi(\alpha_k) > 0$ makes ΔA non-decreasing in C.
- **Diminishing returns**: A is concave, so marginal mint per added unit of mass falls as regions get crowded.
- **Telescoping**: For any sequence that takes $\alpha \rightarrow \alpha'$, $\sum_j m_j = \mu\,[\hat{A}(\alpha') - \hat{A}(\alpha)]$. Splitting or reordering the same net change does not alter total minted claims.

- **Budget isolation**: Claims are weighting units only; they do not create liabilities beyond the participant-funded pool distributed at settlement (see §9).

Application. The trade's m is bound to the position ticket and persists until settlement or a full close. A full close applies the inverse state update $\alpha_{after} \leftarrow \alpha_{after} - Cp$ and **burns** the ticket's claims, restoring $M_{totalClaims}$ accordingly (see §7.3 for ticket details).

Implementation Notes. Use high-precision fixed-point (e.g., Q64.64). Implement $log\Gamma$ with a bounded, deterministic approximation (e.g., Lanczos for small–medium inputs and Stirling-type for large inputs). Precompute and fix all constants used by $\hat{A}$. Publish max-error bounds and keep them identical across networks to ensure reproducibility (see §8.3).

7.3 Position Ticket (NFT)

The final output of a successful buy transaction is a **Position Ticket**, which takes the form of a non-fungible token (NFT). This ticket serves as the trader's immutable receipt and the sole representation of their ownership in the market. It encapsulates all critical information about the trade, creating an unbreakable link between the belief, the capital, and the influence earned. Claim tokens are non-transferable as fungible units; they are bound to the position ticket NFT and move only when the ticket is transferred.

The following data is permanently encoded within the NFT at the time of its creation:

- **Belief Coefficients (`p_i`):** The trader's original belief vector.
- **Collateral (`C_i`):** The amount of capital locked in the position.
- **Minted Claims (`m_i`):** The exact number of claim tokens generated by the trade.
- **Immutable Metadata:** Other essential data such as the `marketId` and `createdBlock`.

The use of an NFT is a crucial architectural choice that supports the ethos of **Self-Contained Coherence**. By binding the claims directly to the ticket that generated them, the protocol ensures that influence cannot be separated from the belief that created it, preventing claim reassignment and preserving the integrity of the system's accounting from trade to settlement. The NFT in its entirety can be sold back to the market and is transferable, enabling secondary-market sale.

7.4 Symmetric redemption (Sell)

(This section provides a high-level overview based on the conceptual framework. The precise mechanics are subject to further specification.)

In addition to entering positions, the protocol provides native, two-sided liquidity by allowing participants to exit a position at any time before the market's final resolution. This is accomplished through a

Symmetric Redemption mechanism.

This process enables a ticket holder to sell their position NFT back to the protocol. The redemption value is determined by the position's standing relative to the current, dynamic market consensus. This ensures the payout accurately reflects the market's valuation of the ticket's informational claim at the moment of exit. By providing a fair and predictable pricing mechanism for both entry and exit, the protocol guarantees there are no capital traps, solving a critical friction point of traditional, illiquid information markets.

Step 0 — Fetch ticket and live state

0.1 Decode the ticket record

From the NFT whose tokenId is supplied with the sell call, read the immutable payload:

Field	Symbol	Type / constraints	Purpose
Belief vector	$p = (p_0, ..., p_K)$	$p_k > 0,\ \sum p_k = 1$ (Q64.64)	Direction of both the original buy and the pending sell.
Minted claims	m	128-bit unsigned integer	Debt that the market must repurchase.
Owner	a_{owner}	address	Must match msg.sender to continue.

If any deserialisation fails or the caller is not the current owner, the transaction reverts with TicketNotAuthorised.

User sets `minCollateralOut` (slippage bound). Default = none. Trade reverts if payout t* < `minCollateralOut`.

0.2 Load the current market constants

Read once from contract storage:

- **State floor** $\varepsilon_\alpha > 0$.
- **Scale constant** $\mu > 0$.
- **Bucket count** $K + 1$.
- **Special-function tables** for $log\Gamma,\ \psi,\ \psi_1$ (versioned by checksum).

These constants are the same for every trade and never branch on user input.

RequireD on-chain logΓ and ψ. ψ₁ off-chain, only sign used.

0.3 Retrieve the live state vector

Compute or load from the on-chain accumulator:

$$\alpha := (\alpha_0, \ldots, \alpha_K), \qquad P := \sum_{k=0}^{K} \alpha_k.$$

Integrity checks:

1. **Domain guard** Assert $\alpha_k \geq \varepsilon_\alpha$ for all k.
2. **Consistency with history** Optionally verify

   $$hash(\alpha) \neq h_{pre}$$

   only to warn the front end; valuation itself depends solely on α *now*, not on the snapshot hash.

Admission records the sell request (ticket id, p, m, user slippage ε_sell). **Price is evaluated at execution** against the then-current state α_exec. If the computed payout t_exec violates the user's bound, the transaction reverts. No partial fills

Step 1 — Choose the exit path

1.1 Fix the direction vector

The belief vector $p = \left(p_0, \ldots, p_K\right)$ stored in the ticket is reused **unchanged** for the sell.

By construction $p_k > 0$ and $\sum_{k=0}^{K} p_k = 1$; it therefore defines a unique, unit-length ray in the $(K + 1)$-dimensional state space.

1.2 Parameterise the candidate post-trade states

Let $t \geq 0$ denote the **total collateral** to be removed along that ray.
For every tentative value of t we obtain a candidate state

$\beta(t) := \alpha - t p.$

Here:

- Each coordinate update is $\beta_k(t) = \alpha_k - t p_k$.
- When $t = 0$ the market is unchanged; when $t > 0$ every bucket is diminished in exact proportion to its p_k weight.

1.1 Fix the direction vector

The belief vector p = (p0,…,pK) stored in the ticket is reused unchanged for the sell.

By construction p_k ≥ 0 and ∑_k p_k = 1. This fixes the direction and scale of updates on the simplex.

Define the sell ray by β(t) = α − t · p for t ≥ 0, so each coordinate updates as β_k(t) = α_k − t p_k.

The only free variable is t; all other degrees of freedom are fixed by p.

1.3 Economic interpretation

- **Path identity** The exit direction is *identical* to the entry direction; no other path is allowed, ensuring path independence and eliminating arbitrage around curved trajectories.
- **Scalar unknown** All degrees of freedom except the single scalar t are fixed. The remainder of the sell algorithm will solve for the unique $t_\star$ that exactly balances the ticket's outstanding energy.

Outcome of Step 1
A one-parameter family of potential post-trade states, $\beta(t) = \alpha - tp$, is now established. The rest of the procedure will determine the admissible interval for t (Step 2), construct the energy-balance equation $g(t) = 0$ (Step 3), and solve for the unique root $t_\star$.

Step 2 — Determine the admissible interval for t

2.1 Compute the maximum safe withdrawal

For each coordinate k with $p_k > 0$ the bucket depth after removing collateral t is

$$\beta_k(t) = \alpha_k - t p_k.$$

Domain safety requires $\beta_k(t) \geq \varepsilon_\alpha$. Solving this inequality for t yields an individual upper bound

$$t_k^{max} = \frac{\alpha_k - \varepsilon_\alpha}{p_k}.$$

Define the global ceiling

$$t_{max} = \min_{k:p_k>0} t_k^{max}.$$

If $p_k = 0$ for some k the corresponding bucket is unaffected by the sell and imposes no constraint.

2.2 Form the admissible interval

All subsequent calculations must restrict t to

$$0 \leq t \leq t_{max}.$$

Within this interval every component of $\beta(t)$ remains strictly inside the domain of $\Gamma,\ \psi,\ \psi_1$(Proof only), and all potential evaluations are well-defined (reference X.7 for gamma details).

2.3 Edge cases

- **Immediate rejection** If $t_{max} = 0$ (i.e. at least one $\alpha_k = \varepsilon_\alpha$), no collateral can be withdrawn and the transaction reverts with StateAtFloor.
- **Numerical guard** All fixed-point divisions use Q64.64 arithmetic; the contract asserts t_{max} fits in 128 bits.

Outcome of Step 2 A closed, non-negative interval $[0,\ t_{max}]$ has been established. Any candidate t outside this range would breach the state floor and is therefore illegal. Step 3 constructs the energy-balance equation on exactly this interval.

Step 3 — Construct and analyse the energy-balance equation

3.1 Fix the immutable inputs

When a holder invokes **sell**, the engine reads exactly five items and nothing more:

1. **Belief direction** $p := (p_0, ..., p_K)$ with $\sum_{k=0}^{K} p_k = 1$ — the probability weights stored in the ticket.
2. **Minted claim units** m — the quantity of claims that were created at buy-time.
3. **Global scale constant** $\mu > 0$ — the conversion factor between changes in potential and claim units.
4. **State floor** $\varepsilon_\alpha > 0$ — the minimum legal value for every component of α.
5. **Alpha Vector at Execution** α_{exec} — The alpha vector state at execution of the sell

No external prices, order book data, or historical trades are consulted; valuation is entirely internal to these quantities. Current state of market - Alpha vector is established at time of execution of this trade.

3.2 Define the feasible ray back toward the floor

The contract will attempt to remove collateral along the *same* linear direction that was used when the position was opened. For a non-negative scalar t define

$$\beta(t) = \alpha - tp,$$

and restrict t to the closed interval

$$0 \le t \le t_{max}, \quad t_{max} := \min_{k:p_k>0} \frac{\alpha_k - \varepsilon_\alpha}{p_k},$$

so that every coordinate of $\beta(t)$ remains at or above the safety floor ε_α. Any t outside that interval would breach the domain of Γ and is therefore forbidden.

3.3 Construct the energy-gap equation

The protocol measures market “energy” with the Dirichlet potential

$$A(\alpha)=\log\Gamma\!\Bigl(\sum_{k=0}^{K}\alpha_k\Bigr)\;-\;\sum_{k=0}^{K}\log\Gamma(\alpha_k). \tag{3.1}$$

A(α)=logΓ (∑k=0Kαk) − ∑k=0KlogΓ(αk).(3.1)

$$A(\alpha) = \log\Gamma\left(\sum_{k=0}^{K} \alpha_k\right) - \sum_{k=0}^{K} \log\Gamma(\alpha_k). \qquad (3.1)$$

At mint, claim units were created by

$$m = \mu\,[A(\alpha_{after}) - A(\alpha_{before})].$$

To close the position completely, the sell must decrease the potential by exactly the same amount, but in the opposite direction. Define the one-variable function

$$g(t) := \mu\,[A(\alpha) - A(\beta(t))] - m, \qquad t \in [0, t_{max}].$$

The sell succeeds if there exists a scalar $t_\star \in [0, t_{max}]$ such that $g(t_\star) = 0$. Once such a $t_\star$ is found, the collateral returned to the user is $C_{close} = t_\star$; the public state is updated to $\alpha \leftarrow \beta(t_\star)$. $\alpha \leftarrow \alpha - t^* \times p$; and the ticket's m claims are burned.

3.4 Demonstrate strict monotonicity of $g(t)$

Differentiate g along the ray:

$$g'(t) = \mu\,\langle p, \nabla A(\beta(t))\rangle, \qquad \nabla A_k(\xi) = \psi(P_\xi) - \psi(\xi_k),$$

where $P_\xi := \sum_{i=0}^{K} \xi_i$ and ψ is the digamma function. Because the trigamma ψ_1 is strictly positive on $R_{>0}$, every component of ∇A increases (i.e.\ becomes less negative) as t grows, making $\langle p, \nabla A\rangle$ strictly increasing. Hence $g'(t) > 0$ everywhere on $[0, t_{max}]$; g is injective and has at most one root.

3.5 Establish the existence criterion for the root

Evaluate g at the endpoints:

- $g(0) =- m \leq 0$ — no collateral has been removed yet.
- $g(t_{max})$ — the potential drop achievable by draining to the floor.

Monotonicity implies two possibilities:

- If $g(t_{max}) \geq 0$, there exists a *unique* $t_\star \in (0, t_{max}]$ with $g(t_\star) = 0$.
- If $g(t_{max}) < 0$, even the maximum safe withdrawal cannot satisfy the energy balance, and the transaction **reverts** under the venue's all-or-nothing policy.

A conservative depth cap will be necessary to prevent $\alpha < 0$.

3.6 Prepare the deterministic numerical solver

Because g is smooth, strictly increasing, and bracketed, a safeguarded Newton method with bisection fallback converges deterministically:

$$t_{k+1} = t_k - \frac{g(t_k)}{g'(t_k)}, \quad clamped\ to\ [0, t_{max}],$$

using Q64.64 fixed-point arithmetic and the same pre-tabulated approximations of $log\Gamma$, ψ, and ψ_1 employed on the buy side. Each probe costs $O(K)$ operations; convergence is likely in two to three iterations, typical for $K \leq 64$.

Clarification:$log\Gamma$ is used for buy, ψ is used for sell, and ψ_1 is used for sign only -not evaluated on chain.

3.7 Enumerate the roles of special functions

- **Γ (Gamma)** — shows up *only* inside the Dirichlet potential **A(α)**. In sell we evaluate **A** twice per Newton iteration to see how far the energy meter moved. It converts the whole multi-bucket α vector into a single scalar “state energy” that is path-independent.
- **ψ (digamma)** — the *first* derivative of log Γ. We need it for **g′(t)**, the slope in every Newton step. Economically it is the *instantaneous price* of adding or removing probability mass from each bucket.
- **ψ_1 (trigamma)** — the *second* derivative. We do **not** compute it on-chain each time; we just use its sign (always positive) in the formal proof that **g(t)** is monotone and the solver is safe.

Because these functions are deterministic and shared with the buy path, round-trip neutrality holds to within one half-ulp, and every replica of the engine will compute identical sell outcomes.

Result of Step 3: A single scalar equation $g(t) = 0$ has been formulated, its monotonicity proven, and its numerical solution strategy fixed. All remaining tasks in the sell flow reduce to finding the unique root $t_\star$ and applying the corresponding ledger updates.

Step 4 — Select a deterministic initial guess t_0

4.1 Motivation

The solver in Step 5 will apply safeguarded Newton iterations to the scalar equation $g(t) = 0$. A good interior starting point reduces both iteration count and the probability that the first Newton step must be bisected. The initial guess must satisfy

0 < t0 < tmax,

to stay inside the admissible interval derived in Step 2.

4.2 First-order Taylor estimate

Evaluate g and its derivative at the current state $t = 0$:

$$g(0) =- m, g^{'}(0) = \mu \langle p, \nabla A(\alpha)\rangle, \nabla A_k(\alpha) = \psi(P) - \psi\left(\alpha_k\right), P = \sum_{i=0}^{K} \alpha_i.$$

A linear approximation $g(0) + g^{'}(0)\,\delta \approx 0$ gives

$\delta^\star = \frac{m}{\mu \langle p, \nabla A(\alpha)\rangle}$.

Set

$t_0 = min\{\delta^*, \frac{1}{2} tmax\}$

The clamp at $\frac{1}{2} t_{max}$ guarantees interiority even in extreme states where the linear estimate overshoots.

4.3 Computational cost

- **Gradient evaluation** $O(K)$ digamma look-ups.
- **Dot product** $O(K)$ fixed-point multiplies and adds.
  Both operations use the same Q64.64 arithmetic and lookup tables already resident from earlier steps; no extra gas is consumed per bucket.

4.4 Correctness guarantees

1. **Positivity** Because $m > 0$ and $g^{'}(0) > 0$, $\delta^\star > 0$.

2. **Boundedness** The min-clamp enforces $t_0 < t_{max}$.
3. **Convergence aid** With t_0 in the interior and g strictly increasing, Newton's first step cannot jump outside $\left[0, t_{max}\right]$ unless $g^{'}\left(t_0\right)$=0, which is impossible under the ~~trigamma convexity~~ by strict monotonicity of g and the safeguard.

Outcome of Step 4
A single scalar t_0 has been computed, lying strictly within the admissible interval and informed by the first-order behaviour of g. The safeguarded Newton routine in Step 5 will now iterate from this starting value to the unique root $t_\star$.

Step 5 — Safeguarded Newton iteration

5.1 Initialisation

Technical step	Explanation
Bracket Set $lo = 0$ and $hi = t_{max}$. We already know $g(lo) \leq 0$ and $g(hi) \geq 0$ from Step 3.5.	Guarantees the true root lies inside $[lo, hi]$; every subsequent iterate will be forced to stay within these bounds.
Current iterate Initialise $t \leftarrow t_0$ (computed in Step 4).	Starts the solver at the first-order Taylor estimate, which is a good interior estimate; reduces expected iterations.
Number format All three scalars lo, t, hi are stored in Q64.64 fixed-point.	Uses the same deterministic arithmetic as the buy path; no conversions are needed later.

5.2 Iterative loop

Execute the following sequence for at approximately **three** passes(desired testing tbd); exit sooner if a stopping test is met.

For each pass *i* perform the six actions below; break early as soon as either convergence test is met.

1. **Evaluate the current gap and slope** *Technical*

$$g_t = \mu[A(\alpha) - A(\beta(t))] - m, \qquad s_t = \mu\langle p, \nabla A(\beta(t))\rangle,$$

with $\nabla A_k(\xi) = \psi(P_\xi) - \psi(\xi_k)$ and $P_\xi = \sum_j \xi_j$. *Explanation* g_t is the remaining energy to be repaid; s_t is the instantaneous price along the exit ray, derived from digammas.

2. **Form the raw Newton proposal**

$$t_{prop} = t - \frac{g_t}{s_t}.$$

   This is where the tangent hits the zero-line.

3. **Apply the safeguard**
   Condition If $t_{prop} \notin (lo, hi)$ **or** $s_t = 0$. *Action* Replace t_{prop} by the midpoint

$$t_{bisect} = \frac{lo+hi}{2}.$$

   The iterate now lies strictly inside the legal bracket $[lo, hi]$.

4. **Update the bracket**
   Evaluate $g(t_{prop})$. *If* $g \geq 0$ set $hi \leftarrow t_{prop}$; *else* set $lo \leftarrow t_{prop}$. This keeps the invariant $g(lo) \leq 0 \leq g(hi)$.

5. **Test for convergence**
   Residual test $|g(t_{prop})| \leq 2^{-64}$ μ. *Step-size test*
   $|t_{prop} - t| \leq 2^{-64} max\{1, t_{prop}\}$.
   Passing either test stops the loop; the root is now accurate to ≤ ½ ulp in Q64.64.

6. **Advance iterate** Set $t \leftarrow t_{prop}$ and continue to the next pass (unless already stopped).

Because g is strictly increasing and convex, the first Newton step typically requires 2-4 iterations with bisection fallback; a second step reaches Q64.64 precision; a third is the hard cap.

5.3 Deterministic termination guarantees

1. **Iteration bound** The Taylor start (Step 4) plus monotonicity ensures convergence in ≤ 3 rounds.

2. **Bracket integrity** The safeguard keeps every iterate within $[lo, hi]$; no overflow or domain breach is possible.

3. **Bit-exact reproducibility** All table look-ups and arithmetic paths are fixed; different implementations reach identical $t_\star$ and $g(t_\star)$ to ±½ ulp.

5.4 Final values for downstream steps

After the loop terminates, set

$t_\star = t, C_{close} = t_\star p.$

These outputs feed directly into:

- **Step 6** — ledger update $\alpha \leftarrow \alpha - C_{close}$ and claim burn $M_{total} \leftarrow M_{total} - m$;
- **Step 7** — event emission and invariants re-check.

The safeguarded Newton routine has now produced a root accurate to protocol tolerance with the minimal, bounded computational cost.

Step 6 — Apply ledger updates and settle the sell

6.1 Compute definitive cash and state deltas

- **Collateral vector** $C_{close} = t_\star p$ (from Step 5).
- **New state** $\alpha' = \alpha - C_{close}$.

Because $0 < t_\star \leq t_{max}$, every component $\alpha_k' \geq \varepsilon_\alpha$; hence all special-function domains remain valid.

6.2 Persist state changes

1. **State vector** Write $\alpha \leftarrow \alpha'$ to contract storage.
2. **Running total mass** $P \leftarrow P - t_\star$.
3. **State hash** $h_{post} \leftarrow hash\left(\alpha'\right)$ for downstream consistency proofs.

All writes use fixed-point encoding identical to the buy path.

6.3 Burn claims and update global supply

- **Global claims ledger** $M_{total} \leftarrow M_{total} - m$.

- **Ticket NFT** _burn(tokenId); any residual metadata is zeroed.

This restores **budget balance**: total outstanding claims again equal $\mu\left[A\left(\alpha_{\infty}\right) - A\left(\alpha'\right)\right]$.

6.4 Transfer collateral to seller

Execute an ERC-20 (or native-asset) transfer:

$POOL \rightarrow t\star \;\; \alpha_{owner}$

$$\mathrm{POOL} \xrightarrow{t_\star} \alpha_{\mathrm{owner}}.$$

Amount exactly equals the scalar $t_{\star}$; no fee or rounding adjustment is applied in this "pure" venue.

Revert if t* < minCollateralOut.

6.5 Post-trade invariant checks

Immediately after storage writes and before emitting events:

- **Floor guard** Assert $\alpha'_{k} \geq \varepsilon_{\alpha}$ ∀k
- **Pool solvency** Assert contract balance ≥ 0 (cannot underflow because of budget balance).
- **Claim supply** Assert $M_{total} \geq 0$ (guaranteed by Step 6.3).

Any failure reverts the entire transaction, leaving pre-sell state intact.

6.6 Emit settlement event - Overboard but left is as context

```
event SellExecuted(
   address indexed owner,
   uint256 tokenId,
   uint128 claimsBurned,        // m
   uint256 collateralReturned,  // t_star (scaled 1e18)
   bytes32 alphaPostHash        // h_post
);
```

Indexers can now reconstruct the full before/after state via the published hash and on-chain logs.

Outcome of Step 6
The market ledger, claim supply, and user balances have been synchronously updated to reflect a complete sell. All invariants hold, and a canonical event records the transaction for public audit.

Step 7 — Confirm invariants and broadcast completion

7.1 Round-trip neutrality verification

Technical statement	Explanation
The sell just executed and the corresponding buy that created the ticket together satisfy $\mu * [A(\alpha') - A(\alpha\_init)] = 0 \pm 2^{-64}$	Because both directions quantise the same potential jump with the same Q64.64 “ties-to-even” rule, any buy-then-immediate-sell cycle returns the trader’s cash to within one half-ulp. This closes the accounting loop and proves that no dust-farming arbitrage exists.

7.2 Final state-safety assertions

- **Floor integrity** Every bucket depth satisfies $\alpha_{k'} \geq \varepsilon_{\alpha}$.
- **Budget balance** Total on-chain collateral equals the potential-scaled claim supply, guaranteeing solvency.
- **Path independence** Because only α and α' enter the energy equation, the exact sequence of intermediate trades is irrelevant; the market can be replayed in any order and produce the same balances.

These checks ensure that the core tenets CT1–CT6 remain intact after the sell.

7.3 Public settlement signal

A single **SellExecuted** event is emitted containing:

- the seller’s address and ticket identifier,
- the number of claim units burned (m),
- the cash amount returned ($t_{\star}$),
- a hash of the post-trade state vector ($hash\left(\alpha'\right)$).

This immutable log entry lets off-chain indexers and auditors verify that:

1. the ticket disappeared exactly once,

2. the cash outflow matched the potential drop,
3. the published state hash chains cleanly into subsequent trades.

7.4 Outcome of Step 7

All economic and numerical invariants are now re-confirmed, and an auditable record has been broadcast. The sell lifecycle is fully closed; the engine is ready to process the next transaction with no residual side-effects.

Step 8 — Return final receipt and expose post-trade telemetry

8.1 Emit user-facing receipt

The smart-contract call returns true and exposes, via the ABI, the same values already logged in **SellExecuted**:

Field	Value	Purpose for the caller
collateralReturned	$t_\star$ (fixed-point)	Confirms the exact cash credited to the wallet.
claimsBurned	m	Verifies the ticket's liability is now zero.
alphaPostHash	$hash(\alpha')$	Lets front ends anchor their on-chain depth charts to the new state.

Because all numbers are final-state facts—no estimates or "pending" flags—the wallet can immediately update balances and P&L.

8.2 Surface real-time market metrics

After the transaction's state root is accepted in the next block, indexers and dashboards can:

1. **Re-price outstanding tickets** using the updated α'.
2. **Plot depth curves** that incorporate the just-executed withdrawal C_{close}.
3. **Publish liquidity statistics** (e.g., per-bucket slippage to the floor) that reflect the reduced pool mass.

These analytics rely solely on the public data emitted in Steps 6–7, so every participant sees the same numbers.

8.3 Confirm round-trip neutrality on chain

Auditors—or an automated invariant monitor—may replay the paired buy and sell:

$$\mu\,[A(\alpha_{after\ buy}) - A(\alpha_{before\ buy})] \;-\; \mu\,[A(\alpha_{before\ sell}) - A(\alpha_{after\ sell})] = \Delta_{round-trip}\varepsilon[-2^{-64}, -2^{-64}],$$

establishing that any dust is strictly within the half-ulp bound.
This serves as an on-chain certificate that the market's accounting loop is still perfectly closed.

8.4 Outcome of Step 8

- The caller receives a definitive, final receipt—no asynchronous steps remain.
- Third-party tools can now reconcile their order books and depth visualisations.
- Continuous audits have a self-contained data set to verify budget balance and round-trip neutrality.

The sell lifecycle is therefore complete from both the protocol's and the ecosystem's perspectives.

Appendix:
Sell Variables and Constants

- t **(aka C_{close})**
  Technical: Scalar payout to solve so the sell consumes the ticket's claim balance: find $t \geq 0$ with $\mu\,[A(\alpha) - A(\alpha - tp)] = ticket - claims$. Also the amount removed from state: $\alpha \leftarrow \alpha - tp$.
  Plain: The cash you get and the size taken out of the market in your shape.

- $g(t)$
  Technical: Root function $g(t) = \mu\,[A(\alpha) - A(\alpha - tp)] - ticket - claims$. Monotone on $\left[0, t_{max}\right]$; $g\left(t^{\backslash *}\right) = 0$ defines $t^{\backslash *}$.
  Plain: A meter that reads "too little," "too much," or "just right." We move t until the meter hits zero.

- $A(\alpha)$ **(Dirichlet potential)**
  Technical: $A(\alpha) = log\Gamma\left(\sum_k \alpha_k\right) - \sum_k log\Gamma\left(\alpha_k\right)$. Concave state function used for buy/sell valuation and path independence.
  Plain: A single number for the market's "energy." Buys add it; sells remove it.

- $\nabla A(\beta)$ **(gradient of the potential)**
  Technical: $\partial A/\partial \beta_k = \psi\left(\sum_j \beta_j\right) - \psi\left(\beta_k\right)$. Appears in $g'(t) = \mu\langle p, \nabla A(\alpha - tp)\rangle$ and Newton updates.
  Plain: The current slope of energy in each bucket; we use it to steer the solver.

- $r(\beta, p)$ **(directional rate)**
  Technical: $r(\beta, p) = \langle p, \nabla A(\beta)\rangle$, the claims-per-dollar at state β along direction p.
  Plain: How “expensive” it is, right now, to move the market in your shape.

- $\psi(x)$ **(digamma)**
  Technical: Derivative of $log\Gamma$. Required for ∇A and $g'(t)$. Monotone on $x > 0$.
  Plain: A helper function used to compute the slope precisely.

- $\psi_1(x)$ **(trigamma)**
  Technical: Derivative of ψ. Positive on $x > 0$. Used for stability analysis and safeguarded Newton behavior.
  Plain: A second helper that measures how quickly the slope itself changes.

- $\mu > 0$ **(scale)**
  Technical: Constant converting potential change to claims. Enters both mint and sell: $claims = \mu\Delta A$.
  Plain: A fixed unit converter so the same math mints and redeems fairly.

- α_{exec}, α_{post}
  Technical: State at execution (used in appraisal) and after execution (after removing tp). Must satisfy $\mu\left[A\left(\alpha_{exec}\right) - A\left(\alpha_{post}\right)\right] = claims\ burned.$
  Plain: exec-and-after snapshots of market weight.

- $p \in \Delta$
  Technical: Ticket’s belief direction used for the exit ray $\alpha - tp$ and all dot-products with ∇A.
  Plain: Your shape. It tells the system which parts of the state to remove.

- ε_α **(floor)**
  Technical: Componentwise lower bound; require $\alpha - tp \geq \varepsilon_\alpha$. Sets t’s domain and ensures special functions stay well-defined.
  Plain: A safety line so no bucket is drained to zero.

- t_{max}
  Technical: Upper bound for the root search: $t_{max} = min_{k:p_k>0}\left(\alpha_k - \varepsilon_\alpha\right)/p_k$. If $g\left(t_{max}\right) < 0$ reject (all-or-nothing).
  Plain: The most we could possibly pay while staying safe.

- $[a, b]$ **(solver bracket)**
  Technical: Monotone interval $\left[0, t_{max}\right]$ that contains the root. Used by bisection and to safeguard Newton steps.
  Plain: A fence we shrink until the right t is pinned down.

- **Newton step**
  Technical: Update $t \leftarrow t - g(t)/g^{'}(t)$ with fallback to bisection if the step leaves $[a, b]$ or doesn't reduce $|g|$.
  Plain: A smart jump toward zero using the local slope, with a safe backup if the jump isn't helpful.

- δ **(token unit) and rounding rule**
  Technical: Smallest cash quantum; quantize t to multiples of δ via min-$|g|$ with ties-to-even. Keeps round-trip neutrality.
  Plain: We round payouts to cents (or finer) in a way everyone reproduces exactly.

- τ_{t}, τ_{A} **(tolerances)**
  Technical: Stop criteria for the solver so post-quantization residual $|g\left(C_{close}\right)|$ is minimal and deterministic.
  Plain: How close is "close enough" before we round and pay.

- **Coefficient tables (for $log\Gamma, \psi, \psi_{1}$)**
  Technical: Piecewise minimax coefficients on $\left[\varepsilon_{\alpha}, a\right]$ plus asymptotic corrections for $x \geq a$. Enable fast, bounded-error evaluation.
  Plain: Pre-baked numbers so heavy math reduces to quick adds/mults with known accuracy.

- $order\_index$
  Technical: FIFO position at admission; determines which α_{pre} applies to the order's appraisal.
  Plain: your place in line only; price is evaluated against the live state at execution after all earlier-indexed trades are applied.

Correctness properties to validate

Round-trip neutrality. For any α, any p, any C, if you buy then immediately sell from the same α, the solver returns $C_{close} = C$ within rounding. Proof sketch. ΔA is path independent, so $m = \mu\,[A(\alpha + Cp) - A(\alpha)]$ and the sell inversion on the same path solves the same equation in reverse.

Path independence. Split a buy or sell into parts that sum to the same net shift. The total ΔA equals the one shot value. Numerically this requires bit identical evaluation of A and ψ families and consistent rounding.

Budget balance. On sell you debit the pool by exactly C_{close}. On buy you credit the pool by C. The pool evolves by user transfers only. No hidden terms exist in the pricing equations.

Monotonicity. C_{close} increases with m and decreases with tightening floors or deeper opposition along p. Formal. g is strictly increasing in t and strictly increasing in m with sign flipped in the root condition.

Safety. If any component would cross ε_α at the root, you must reject. In the all or nothing venue there is no partial execution. Under a venue that allows partials you would cap at t_{max} and burn the corresponding fraction of claims.

Interpreting profit and loss

Net PnL for a round trip that moves state is the difference of two integrals along the entry and exit rays. You gain if the average rate you pay on entry exceeds the average rate you receive on exit along your ray. You lose in the reverse case. The solver already prices the exit integral exactly from the pre-trade snapshot. There is no need to compute entry history to pay a sell. History only matters for your own accounting or analytics.

Risks and how the math addresses them

Self pricing. Disallowed because valuation reads the pre-trade snapshot only. The equations depend exclusively on α at admission and the stored ticket data.

Hidden spread from point pricing. Avoided because payout uses the integral of marginal prices along the ray. Large sells are priced by area, not the current gradient.

Unbounded loss. Impossible. $C_{close} \geq 0$ and is capped by t_{max}. You never owe the pool.

Numerical drift. Controlled by a single implementation of $log\Gamma$, ψ, and ψ_1 and fixed rounding rules. Telescoping then holds in practice.

Floor misconfiguration. Addressed by publishing ε_α and testing curvature near floors. Reject unsafe full closes in this venue.

This is the exact mathematical and numerical surface you need to implement and validate sell valuation and execution.

Bucketization policy (K = 30–60). Choose $K + 1$ buckets so that bucket width near the consensus peak is ≤ one standard deviation divided by 6–10 for bell-like markets. Publish bucket edges at market open. K sets resolution (finer shape capture) and runtime cost (linear in K) but **does not** change pricing physics.

Scale constant. Set $\mu = 10$ per market and keep it constant. μ scales claims linearly; it does not affect payout correctness, only diagnostic magnitudes.

Why iterate (one-dimensional root). There is no closed form for $t = C_{close}$. Define $g(t) = \mu[A(\alpha) - A(\alpha - tp)] - m$. Solve $g(t) = 0$ on $[0, t_{max}]$ with safeguarded Newton

(Newton step with bisection fallback). Stop when payout rounds deterministically to token units.

Tabled approximations (for on-chain viability). Implement $log\Gamma, \psi, \psi_1$ via piecewise minimax (Chebyshev/rational) approximations with uniform error bounds on $[\varepsilon_\alpha, a]$ and asymptotic expansions with two correction terms for $x \geq a$ (e.g., $a = 8$). Store the small coefficient tables in code; runtime uses adds/mults only.

Optimisations

- **Cache once:** compute $A(\alpha)$ a single time per order and reuse it.
- **Reuse vectors:** maintain $\beta(t) = \alpha - tp$ and update in place when t changes.
- **Difference form:** evaluate $A(\alpha) - A(\alpha - tp)$ as $log\frac{\Gamma(P)}{\Gamma(P-t)} - \sum_k log\frac{\Gamma(\alpha_k)}{\Gamma(\alpha_k - tp_k)}$ to avoid subtracting near-equal large numbers.
- **Fixed-point math:** use integer fixed-point with guard bits and ties-to-even to make results reproducible bit-for-bit.
- **Safe small-x math:** use $log1p(x)$ and $expm1(x)$ patterns where x is tiny to prevent precision loss.

7.5 Final settlement (algorithmic only)

Purpose. Convert all outstanding position tickets into payouts using a deterministic, single-pass algorithm. No governance, resolver, or oracle flow appears here; the outcome value is treated as a given input.

Scope. This procedure runs once per market after the outcome is fixed. It does not read order history. It uses only the final market state, the set of tickets, the current owner of each ticket at the settlement height, and the settlement parameters.

Inputs.

- Outcome $y \in [L, H]$.
- Final state α and any fixed constants defined in §5 and §9.
- Ticket set $\{T_i\}$ with stored belief shape and minted claim units.
- Parameter bundle $\Theta = \{\tau, \gamma, \lambda_s, \lambda_d\}$ from §9.
- Eligibility context: ticket.createdBlock/time and the **current NFT owner** at the settlement height.
- Numeric guards and arithmetic model from §10.

Transferability rule. The NFT may transfer. The claim units and Bernstein-shape components inside do not. Payouts go to the **owner of the NFT at the settlement height**. There is no transfer of internal units.

Contract. Settlement is path-independent and budget-closed. Participant-funded solvency holds: the treasury pays at most what was minted. All computations are deterministic under the fixed-point model in §10.

High-level flow (details in §§7.4.1–7.4.6).

1. Evaluate log-densities of each ticket's belief at y.
2. Apply the eligibility gate τ using on-chain creation time and settlement height ownership.
3. Compute accuracy shares $a_i(\gamma)$.
4. Compute claim shares s_i from each ticket's minted units and shape.
5. Weight tickets via λ_s, λ_d, apply floors/guards, and allocate payouts.
6. Finalize: record totals, burn claims, and close the budget.

Outputs.

- Per-ticket payout to the settlement-height owner.
- Final market accounting entries.
- Zero residual claim units and a closed treasury for the market.

Constraints. Use only §10 primitives (logΓ, log-sum-exp, guards). Do not introduce parameter values here; reference §9. No equations from other sections may be restated here.

7.5.1 Step 1: Compute log-densities at the outcome

Objective. For each eligible ticket i compute $\ell_i = log f_i\left(x^{\backslash *}\right)$ stably, where f_i is the ticket's PDF from §5.3.

Step 1.A — Normalize the outcome.

$$u^{\backslash *} = \frac{x^{\backslash *}-L}{H-L} \in [0, 1], use\ natural\ logs.$$

Step 1.B — Boundary branches.
If $u^{\backslash *} = 0$: $\ell_i = log\left(\frac{K+1}{H-L}\right) + log p_{i,0}$.
If $u^{\backslash *} = 1$: $\ell_i = log\left(\frac{K+1}{H-L}\right) + log p_{i,K}$.

Step 1.C — General case $0 < u^{\backslash *} < 1$ via log-sum-exp.

$$tmp_k = log p_{i,k} + log\binom{K}{k} + k log u^{\backslash *} + (K - k) log\left(1 - u^{\backslash *}\right),$$

$$s_{max} = \max_k tmp_k, log_\Sigma = s_{max} + log \sum_{k=0}^{K} exp\left(tmp_k - s_{max}\right), \ell_i = log\left(\frac{K+1}{H-L}\right) + log_\Sigma.$$

Step 1.D — Apply a density floor.

$$\ell_i \leftarrow max\left(\ell_i, log \varepsilon_{dens}\right) (see\ §10).$$

Set S. Let S be the set of **open tickets** with $createdBlock \leq eligibleCutoffBlock$. Only S participates in settlement.

7.5.2 Step 2: Eligibility gate (τ)

Peak. $\ell_{max} = max_{i\in S}\ell_i$.

Threshold. $\ell_{thr} = \ell_{max} + log\tau$.

Eligible set. $E = \left\{i \in S | \ell_i \geq \ell_{thr}\right\}$.

Empty-set fallback. If $E = \varnothing$, set $E = \left\{argmax_{i\in S}\ell_i\right\}$ using deterministic tiebreak (smallest ticketId).

Rationale. Filters non-competitive curves and prevents “participation trophies” while guaranteeing progress even in degenerate cases.

7.5.3 Step 3: Accuracy shares $\left(a_i\right)$

Tempered softmax. For $i \in E$,

$$z_i = exp\left(\gamma\left(\ell_i - \ell_{max}\right)\right), Z = \sum_{j\in E} z_j, Z \leftarrow max\left(Z, \varepsilon_{sum}\right), a_i = \frac{z_i}{Z}; a_i = 0 \, for \, i \notin E.$$

Rationale. $\gamma > 1$ yields winner-take-most while preserving continuity; ε_{sum} prevents divide-by-zero in fixed-point.

7.5.4 Step 4: Claim shares $\left(s_i\right)$

Scope. Claims are counted only over the frozen participant set S.

$$M_{totalClaims} = \sum_{j\in S} m_j, M_{totalClaims} \leftarrow max\left(M_{totalClaims}, \varepsilon_{sum}\right),$$

$$s_i = \frac{m_i}{M_{totalClaims}} \, for \, i \in S; s_i = 0 \, for \, i \notin S.$$

Rationale. Tickets created after the cutoff neither receive nor dilute payouts.

7.5.5 Step 5: Weights and payouts

Weights.

$$w_i = \left(s_i\right)^{\lambda_s}\left(a_i\right)^{\lambda_d}.$$

Budget.

$$W = \sum_{j\in S} w_j, W \leftarrow max\left(W, \varepsilon_{sum}\right).$$

Payouts. For $i \in S$,

$Payout_i = Pool \cdot \frac{w_i}{W}$.

Transfer rounding. Compute in Q64.64, then floor to the collateral token's decimals for transfer. Handle dust deterministically (largest-remainder or remit to MIB per market policy).

7.5.6 Step 6: Finalization

- Transfer payouts
- Burn or zero out m_i for all tickets and mark the market settled.
- Emit settlement events: $x^{\backslash *}$, ℓ_{max}, $|E|$, Z, W, per-ticket $\left(a_i, s_i, w_i, Payout_i\right)$.

Justification and guarantees.

- **Numerical stability.** Log-sum-exp in §10.6, floors ε_{dens}, ε_{sum} in §§9.1, 9.3–9.5 prevent overflow/underflow in fixed-point (§10.1-5).
- **Closed budget.** $\sum_{i \in S} Payout_i = Pool$ up to transfer rounding; dust is handled deterministically.
- **Fairness and freeze.** eligibleCutoffBlock seals the participant set before any resolver leakage (§8.2). τ removes non-competitive tickets; γ tunes contest sharpness.
- **Determinism.** All paths depend only on immutable inputs, precomputed tables, and fixed rounding; identical inputs yield identical outputs.

7.6 Dirichlet potential and mint properties

Technical Definition: The core of the protocol's economic engine is the **Dirichlet Potential**, A(α).
$A(\alpha)$ equals the **negative log-partition** of the Dirichlet:

$A(\alpha) = log\Gamma(\sum_k \alpha_k) - \sum_k log\Gamma(\alpha_k) =- logB(\alpha)$. A is concave

where Γ is the Gamma function, the generalization of the factorial to real and complex numbers. This function takes the market's internal state vector α as input and outputs a single scalar value representing the total information content of that state.

7.6.1 In Plain English: What is "State Energy"?

Think of the Dirichlet Potential A(α) as a measurement of the market's total "certainty" or "informational energy."

- A **low energy** state corresponds to an uncertain, low-information market. When a market is new, the $α_k$ values are small and evenly spread, reflecting high uncertainty. This state has a low $A(α)$ value.
- A **high energy** state corresponds to a certain, information-rich market. As traders add capital and conviction, the $α_k$ values grow larger and more concentrated, reflecting increasing certainty about the outcome. This state has a high $A(α)$ value.

A trade is valued based on how much it **increases the market's energy**. By submitting a belief, a trader pushes the market from a lower energy state ($α_{old}$) to a higher energy state ($α_{new}$). The protocol measures this change in energy ($ΔA = A(α_{new}) - A(α_{old})$) and rewards the trader in direct proportion to their contribution.

Justification: This specific function is the lynchpin of the mechanism's fairness and security. It is a function whose change ($ΔA$) depends only on the start and end states ($α_{old}$ and $α_{new}$) concave, not the path taken between them. This **path-independent** property, known as telescoping, makes it impossible to gain an advantage by slicing a large trade into many smaller ones, neutralizing a critical manipulation vector. Its mathematical properties, which will be explored in the next chapters, ensure that adding information is always rewarded and that the reward is proportional to the novelty of the contribution.

The concavity of the Dirichlet Potential is not a flaw; it is the fundamental mathematical reason the mechanism has its most important incentive properties.

Because the function is concave, the increase in potential ($ΔA$) from adding a fixed amount of evidence ($C*p$) is much larger when the existing evidence ($α_{old}$) is small. As the market matures and the $α$ vector grows, the function flattens out, and the same trade produces a much smaller $ΔA$.

8. Properties and guarantees

8.1 Architectural & Economic Robustness

These properties relate to high-level design choices that ensure the system is safe, solvent, and logically sound, making it a reliable financial primitive.

8.1.1 Guaranteed Solvency

The protocol is a **closed-budget** system. The total payout distributed at settlement can never exceed the total collateral held within the market's unified pool.

- **How it Works:** The system is entirely participant-funded. All collateral from active positions is held in a single pool, and the settlement algorithm (detailed in Chapter 7.5) is simply a rule for distributing 100% of this pool among eligible participants.

There are no external liquidity providers, lenders, or mechanisms that could create liabilities in excess of the pool's assets.
- **Justification:** This provides an ironclad, mathematical guarantee of solvency, which is a non-negotiable requirement for a foundational economic primitive. It aligns with the **Self-Contained Coherence** ethos by ensuring the system's financial integrity is completely endogenous.

8.1.2 Decoupling State from Treasury

The protocol's internal mechanics are driven by dimensionless **pseudo-mass**, not by the cash value of the collateral pool.

- **Why it Matters:** The consensus state α and total mass P are used to calculate claim mints and update the consensus q. These values are derived from the amount of collateral C but are independent of the collateral token's external market price. If the price of the collateral asset (e.g., USDC, ETH) were to fluctuate, it would not retroactively alter the informational state of the market or the claims that have already been minted.
- **Justification:** This intentional decoupling insulates the protocol's internal incentive game from external market volatility. It ensures the logical consistency and integrity of the information market remain intact, regardless of fluctuations in the underlying treasury asset. This is a critical aspect of achieving **Self-Contained Coherence**.

8.1.3 Separation of Concerns

The protocol explicitly separates the reward for

informational contribution from the reward for **final accuracy**.

- **How it Works:**
  - **Ex-Ante Reward (Contribution):** Claim tokens (m) are minted at the time of the trade. This is a reward for the act of contributing information and moving the consensus.
  - **Ex-Post Reward (Accuracy):** The final payout at settlement is heavily weighted by a ticket's accuracy at the resolved outcome $x*$.
- **Justification:** This two-part reward structure prevents "double counting" and mitigates last-minute manipulation. A trader cannot simply wait until the outcome is nearly certain, place a large trade on the obvious result, and expect to receive the same reward as a participant who championed that belief from the beginning. It ensures that both early conviction and final precision are rewarded, creating a more balanced and fair incentive landscape over the entire market lifecycle.

8.2 Path Independence & Telescoping Mint

This is the protocol's most critical anti-manipulation feature. It guarantees that the total number of claim tokens minted depends only on the **net change in the market's state**, not on the order, size, or number of trades used to achieve that change.

- **How it Works:** The claim mint formula is a **telescoping sum**. For any sequence of trades that moves the market from an initial state to a final state, the sum of the claims minted at each step is identical to the claims that would have been minted if the state had changed in a single transaction.
    - **The Telescoping Identity:** For any sequence of `N` trades that takes the market from `α_initial` to `α_final`, the sum of the changes in potential is equal to the total change in potential.**Path Independence & Telescoping Mint.** Let $\alpha_0 \rightarrow \alpha_1 \rightarrow \cdots \rightarrow \alpha_N$ be the state path induced by a sequence of trades, and define $\Delta A_j = A(\alpha_j) - A(\alpha_{j-1})$. Then
        - $$\sum_{j=1}^{N} \Delta A_j \;=\; \sum_{j=1}^{N} \left[A(\alpha_j) - A(\alpha_{j-1}) \right] \;=\; A(\alpha_N) - A(\alpha_0).$$
    - **Implementation requirement.** Compute each increment as $\Delta A_j = \hat{A}(\alpha_j) - \hat{A}(\alpha_{j-1})$ using the **same** deterministic implementation $\hat{A}$ (identical constants and ties-to-even rounding). This preserves telescoping under fixed-point and prevents slice-dependent mint drift.

Justification: This property completely neutralizes strategies like **trade slicing**, where a user might otherwise try to gain an advantage by breaking up their execution. It ensures that the protocol rewards the **economic substance** of a trader's contribution, not the cleverness of their execution strategy. This aligns directly with the **Neutral Simplicity** ethos by embedding a powerful defense into the mechanism's core physics, eliminating the need for complex, opinionated rules to police trader behavior.

8.3 Core Incentive Dynamics

The protocol is designed as a transparent economic game. The following properties ensure that the rational, profit-maximizing strategy for any participant is to contribute valuable and truthful information to the market.

8.3.1 The Early/Contrarian Advantage

The mechanism is designed to explicitly reward participants who contribute information when it is most scarce and valuable: either early in a market's life or against the prevailing consensus. Acting before aligned flow arrives secures a **first-mover dividend**: the trade integrates a steeper potential gradient than if placed later, so the same collateral would mint fewer claims after consensus crowding flattens that gradient.

- **How it Works:** The "yield" on claim minting is highest in areas of greatest uncertainty. This behavior stems from the mathematical properties of the Gamma function's derivative, the digamma function (ψ). The gradient of the Dirichlet Potential is
  `∂A/∂α_k = ψ(P) - ψ(α_k).`
    - **Early Advantage:** When a market is new, the total mass P is small. Because the ψ function grows most steeply at small values, a trade of a given size `C`

will cause a much larger change in potential (ΔA) when P is small than when it is large. This results in more claims minted per dollar for early participants.
    - **Contrarian Advantage:** Similarly, if a trader adds mass to an unpopular outcome (a small α_k), the term -ψ(α_k) is larger, increasing the gradient. This means trades that challenge the consensus and add belief to unlikely outcomes are rewarded with more claim tokens.
- **Justification:** This dynamic is a direct implementation of the **Incentives & Emergence** ethos. It creates a powerful incentive for price discovery, rewarding pioneers who are willing to take risks to bootstrap new markets and encouraging the aggregation of diverse, non-consensus beliefs that are critical for a healthy information market.

8.3.2 The Unprofitability of Copying

Submitting a belief p that is identical or very similar to the current market consensus q is an economically weak strategy that generates minimal reward.

- **How it Works:** The mint gradient is $\partial A/\partial \alpha_k = \psi(P) - \psi(\alpha_k)$. The digamma ψ is **increasing and concave** (its derivative $\psi_1 > 0$ and decreasing). Therefore the gradient is steep where α_k is small and flatter where α_k is large. A trade of size C with belief p changes state by $C\,p$ and mints $m = \mu\,\Delta A$, where ΔA is the **integral of this gradient along the trade's path**. Two levers raise ΔA:
    - (i) **novelty**—adding mass to underweighted regions;
    - (ii) **precision**—concentrating mass to increase certainty where the market already focuses.

  Both levers scale with **depth**: the same C produces a larger ΔA when total mass P is small and a smaller ΔA in a deep market. Copying the consensus $p \approx q$ mostly pushes on already-large α_k, so ΔA and the mint are small.

- **Justification:** This property is essential for ensuring the protocol functions as a true **information aggregation** tool. It disincentivizes lazy or parasitic strategies where participants simply follow the herd. By rewarding novelty and conviction, the mechanism ensures that capital and influence flow to participants who contribute genuine, independent analysis to the market, rather than those who simply try to guess what others are thinking.

9. Parameters

9.1 Market Creation Parameters

These parameters are defined by a user at the time a new market is launched. They define the specific context and properties of that individual market.

Outcome Range [L, H]

- **Function:** A pair of real numbers that defines the immutable lower (L) and upper (H) bounds of the market's outcome variable. This range is used to normalize the outcome space for evaluation with Bernstein polynomials.
- **Guidance:** The creator must select a range that is reasonably expected to contain the final outcome. The protocol must have a clear, predetermined policy for outcomes that resolve outside this range, with the default policy being to void the market and refund all collateral.

Bernstein Degree (K)

- **Function:** A positive integer that sets the degree of the Bernstein basis polynomials used for encoding all PDFs within that market. A higher K allows for greater fidelity and more complex, "peaky" curve shapes, while a lower K enforces smoother, broader distributions .
- **Guidance:** The choice of K is a trade-off between expressive fidelity and computational cost. Higher degrees incur greater gas costs during both the minting and settlement phases. It is recommended that K be set as a global or chain-specific constant (e.g., between 48 and 64) to standardize gas costs and user experience.

Dirichlet Prior (α_0)

- **Function:** The initial state vector α of the market before any trades have occurred. It is defined by a **Prior Strength** parameter, P_0. The default is a uniform prior, where each component $\alpha_{0,k} = P_0 / (K+1)$. This represents a state of maximum initial uncertainty.
- **Guidance:** The prior strength P_0 sets the market's initial "depth." A smaller P_0 increases the claim mint rewards for the very first trades, creating a more volatile and high-reward environment for market pioneers. A larger P_0 dampens this effect, making the market more resilient to early, small trades. A common default is to set $P_0 = K$.

Mint Scale (μ)

- **Function:** A global scaling factor that converts the unitless change in Dirichlet Potential (ΔA) into the number of claim tokens (m) minted for a trade ($m = \mu * \Delta A$). It directly tunes the intensity of the early/contrarian incentive.
- **Guidance:** This parameter should be set globally and calibrated carefully. A higher μ amplifies the advantage of early trades, while a lower μ shifts the balance of rewards more toward final accuracy . The recommended heuristic is to choose μ such that the first doubling of a market's cumulative collateral mints approximately 40% of the eventual total claims in back-testing simulations.

Choose μ jointly with P_0. P_0 sets initial depth and flattens the potential gradient for early trades, so higher P_0 lowers ΔA and per-dollar mint; calibrate μ upward as P_0 rises to keep the initial mint rate within target bounds

9.2 Settlement Parameters (Protocol-level)

These parameters are global constants baked into the protocol. They are not set per-market but are fundamental to the system's game theory and overall economic balance.

Accuracy Temperature (γ)

- **Function:** The exponent γ (gamma) applied to the accuracy scores during settlement. It controls the "sharpness" of the payout contest. A value less than 1 creates a "winner-take-most" dynamic, while a value closer to 1 results in broader sharing among eligible tickets.
- **Guidance:** The global default is fixed at **`γ = 0.7`**. This value ensures that having a significantly more accurate belief results in a strongly amplified payout, while still allowing other highly accurate (but not top) participants to receive a meaningful share.

Eligibility Gate (τ)

- **Function:** Also referred to as the `α_gate`, τ (tau) is the minimum performance threshold required for a ticket to be eligible for a payout. A ticket is only eligible if its probability density at the true outcome `x*` is greater than or equal to τ percent of the maximum density achieved by any single ticket in the market.
- **Guidance:** The global default is fixed at **`τ = 0.03`** (i.e., 3%). This gate is a critical anti-spam and anti-copycat measure, ensuring that tickets with weak or irrelevant beliefs ("participation trophies") are excluded from the final payout distribution.

Payout Exponents (`λ_s`, `λ_d`)

- **Function:** The exponents `λ_s` (lambda-s) and `λ_d` (lambda-d) in the final payout weight formula, `w_i = (s_i)^λ_s * (a_i)^λ_d`. They balance the reward between a ticket's **s**hare of claims (`s_i`) and its **d**ensity-based accuracy (`a_i`).
- **Guidance:** These parameters should be globally configurable for research and testing but are fixed at a default of **`λ_s = 1`** and **`λ_d = 1`**. This default gives equal importance to both a participant's informational contribution during the trading phase and their final accuracy at settlement.

eligibleCutoffBlock

- **eligibleCutoffBlock** is the block height recorded at the resolver's commit; it freezes eligibility so that only tickets with `createdBlock ≤ eligibleCutoffBlock` can receive settlement payouts. It is stored in the market header and referenced in

§§9.1–9.2. The resolver sets `eligibleCutoffBlock = block.number - Δ_B` at commit, where `Δ_B` is a protocol constant that provides a small buffer to prevent same-block sniping and to ensure deterministic ordering. This cut-off seals the participant set before any potential leakage of x* and ensures fairness and reproducibility of the eligible set across nodes.

ε_{sum}

- Set a protocol constant $\varepsilon_{sum} > 0$ and apply $Z \leftarrow max(Z, \varepsilon_{sum})$ in §9.3 and $W \leftarrow max(W, \varepsilon_{sum})$ in §9.4 to avoid divide-by-zero under fixed-point underflow, with ε_{sum} chosen small enough to be economically negligible.

Justification: Extreme score separation can underflow Z or W to zero in fixed-point; a tiny floor preserves determinism without changing rankings.

10. Implementation & numerics

Goal. Deterministic, numerically stable, audit-ready computation for minting and settlement. All implementations must produce identical outputs given identical inputs.

10.1 Deterministic arithmetic model

- **Number format.** Use signed fixed-point Q64.64 for all reals stored or computed on-chain. Adopt a single library and rounding rule (ties-to-even) network-wide. Example reference: ABDKMath64x64. (GitHub)
- **Rounding contract.** Every transcendental call (log, exp, logΓ, ψ) and every sum/product in settlement must document: input domain, scaling, rounding direction, and overflow behavior. Store these in a "Numerics Manifest" alongside the contract bytecode (hash of constants, table checksums).
- **Overflow limits.** Prove that all intermediate magnitudes fit in 128-bit fixed-point given §8.1 bounds for K, P_0, μ, and collateral limits. Enforce require() guards where proofs rely on creator-set parameters.

10.2 Special functions: $log\Gamma$ (mandatory)

Goal. One deterministic implementation $\widehat{log}\Gamma$ used everywhere (minting and settlement). Fixed-point arithmetic. Bounded error. No reflection. Identical inputs ⇒ identical bits.

Numerics contract.

- Format: Q64.64, rounding **ties-to-even**. Natural logs.
- Floors: enforce $\alpha_k \geq \varepsilon_\alpha$ with $\varepsilon_\alpha = 2^{-32}$; set $x_{min} = \varepsilon_\alpha$.
- Switch: use two branches with published split $x_{sw} = 10$.
- Constants: precompute and embed $\frac{1}{2}ln(2\pi)$ as LN_SQRT_2PI.
- Determinism: publish all constants in a **Numerics Manifest** (checksums).

Piecewise scheme

Compute $\hat{log}\Gamma(x)$ as:

(a) Medium range $x \in [x_{min}, x_{sw}]$**: Lanczos**

Use the stabilized Lanczos form with fixed $(g, m, \{c_k\})$:

$$\hat{log}\Gamma(x) = \left(x - \frac{1}{2}\right)ln(x + g - \frac{1}{2}) - (x + g - \frac{1}{2}) + ln\left(c_0 + \sum_{k=1}^{m} \frac{c_k}{x+k-1}\right) + \frac{1}{2}ln(2\pi).$$

Published parameters. Provide $g, m, \{c_k\}_{k=0}^{m}$ as Q64.64 constants (recommend $g = 7, m = 9$ for Q64.64). Include a SHA-256 checksum of the table.

Notes.

- Evaluate the inner sum left-to-right with widened intermediates or compensated addition.
- No reflection needed since $x > 0$.

(b) Large range $x > x_{sw}$**: Stirling + Bernoulli tail**

$$\hat{log}\Gamma(x) = \left(x - \frac{1}{2}\right)lnx - x + \frac{1}{2}ln(2\pi) + \sum_{n=1}^{N} \frac{B_{2n}}{2n(2n-1)\,x^{2n-1}}.$$

Truncation rule. Choose N so the **next** term at $x = x_{sw}$ is below the target ulp in Q64.64. Recommend $N = 4$. Publish N.

Continuity check. Verify $|\hat{log}\Gamma_{Lanczos}(x_{sw}) - \hat{log}\Gamma_{Stirling}(x_{sw})| \leq$ target ulp. Publish the measured bound.

Domain and guards

- Reject $x < x_{min}$. Enforce $\alpha_k \geq \varepsilon_\alpha$ at all times (§8.3.6).
- Powers $x^{-(2n-1)}$: compute iteratively to avoid overflow; cap intermediates.
- All paths O(1): Lanczos m divisions; Stirling N terms. Gas is predictable.

Determinism requirements

- Use **the same** $\hat{log}\Gamma$ for all ΔA evaluations and settlement everywhere.
- Manifest MUST include: x_{min}, x_{sw}, N, g, m, $\{c_k\}$, LN_SQRT_2PI, rounding mode, and table checksums.
- Changing any constant requires a protocol version bump; telescoping (§7.1.1) relies on identical code paths.

Test vectors (ship with release)

- Points: x_{min}, $x_{sw} - \delta$, x_{sw}, $x_{sw} + \delta$, and large x.
- Compare to a high-precision reference offline; publish max abs/ulp error.

- Cross-client parity: identical Q64.64 outputs across at least two independent implementations.

Why two branches. Lanczos gives low bias and stability for small/medium x; Stirling is cheaper and well-conditioned for large x. A single formula cannot meet both the **error** and **cost** targets in Q64.64 across the full positive domain. The split at $x_{sw} = 10$ with $N = 4$ closes that gap and locks determinism.

10.3 Polygamma (optional, off-chain)

Minting uses only $log\Gamma$. Digamma ψ and trigamma ψ_1 appear in analysis and monitoring. If exposed, implement via:

- **Recurrence up** to $x \geq x_0$: $\psi(x + 1) = \psi(x) + \frac{1}{x}$; similarly for ψ_1.
- **Asymptotic** for large x: $\psi(x) = lnx - \frac{1}{2x} - \frac{1}{12x^2} + \cdots$, and $\psi_1(x) = \frac{1}{x} + \frac{1}{2x^2} + \cdots$. Publish error bounds. (Wikipedia, docs.tibco.com)

10.4 Precomputed tables (deterministic)

- $logBinom(K, k)$ for $k = 0..K$:

$$log\left(\frac{K}{k}\right) = log\Gamma(K + 1) - log\Gamma(k + 1) - log\Gamma(K - k + 1).$$

Compute off-chain with the same $log\Gamma$ scheme, round to Q64.64, embed as immutable constants, and publish a SHA-256 checksum. This eliminates repeated $log\Gamma$ at settlement and ensures bit-for-bit reproducibility. (Stan)

10.5 Stable mixture evaluation (log-sum-exp)

Settlement computes $\ell_i = logf_i\left(x^{*}\right)$ using log-sum-exp with a shift s_{max}:

$$\ell_i = log\left(\frac{K+1}{H-L}\right) + s_{max} + log\sum_{k=0}^{K} exp\left(tmp_k - s_{max}\right),$$

$$tmp_k = logp_{i,k} + log\left(\frac{K}{k}\right) + klogu^{*} + (K - k)log\left(1 - u^{*}\right).$$

This avoids overflow/underflow and matches best practice for softmax/log-sum-exp. Use boundary branches at $u^{*} \in \{0, 1\}$. (arXiv, School of Mathematics, gregorygundersen.com)

10.6 Floors, guards, and clamps

- **State floor:** $\alpha_k \geq \varepsilon_\alpha$. Choose ε_α such that $log\Gamma\left(\varepsilon_\alpha\right)$ and increments are representable in Q64.64 and far below any realistic trade size; document the value in the Numerics Manifest. Justification: $log\Gamma$ undefined at ≤ 0, steep curvature for tiny x. (Ele-Math)
- **Density floor:** apply $\ell_i \leftarrow max\left(\ell_i, log\varepsilon_{dens}\right)$ to keep log-densities finite when any $p_{i,k} = 0$. Choose ε_{dens} so it is economically negligible relative to contest scales. (arXiv)

- **Softmax floors:** for $Z = \sum_{j \in E} z_j$ and $W = \sum_j w_j$, apply $Z \leftarrow max\left(Z, \varepsilon_{sum}\right)$, $W \leftarrow max\left(W, \varepsilon_{sum}\right)$. Prevents divide-by-zero under underflow in fixed-point; does not change rankings if ε_{sum} is tiny. (arXiv)

- **Unit tests for floors:** demonstrate invariance of winner ranking for any ε choices within published bounds.

Rounding. Use Q64.64 with ties-to-even for all internal math. Keep claims and weights in fixed-point; do **not** floor per-trade. Floor only final token transfers to on-chain decimals; handle dust deterministically (largest-remainder or remit to MIB). This preserves telescoping and solvency.

10.7 Complexity and gas

- **Minting:** two $log\Gamma$ evaluations for P and each α_k aggregated by difference; O(K).

- **Settlement per ticket:** O(K) tmp terms plus a single log-sum-exp.

- **Tables:** O(K) storage for $logBinom(K, k)$. Avoid per-ticket $log\Gamma$ at settlement by using the table. These bounds are predictable and scale linearly in K. (Stan)

10.8 Security and invariants

- **State invariants:** $\alpha_k \geq \varepsilon_\alpha$; $\sum_k p_k = 1$ at submission; $C \geq 0$. Assert each before state updates.

- **No external calls during mint/settle.** Compute-then-write; no reentrancy windows.

- **Determinism invariant:** $\Delta A = \hat{A}\left(\alpha_{after}\right) - \hat{A}\left(\alpha_{before}\right)$ using the **same** $\hat{A}$ code path, constants, and rounding. This preserves telescoping in fixed-point.

- **Table integrity:** verify the $logBinom$ checksum at deployment; expose it via a view to aid auditors.

- **Eligibility freeze:** use eligibleCutoffBlock from §8.2; never evaluate tickets created after that block. Prevents same-block sniping and resolver leakage effects.

10.9 Conformance tests (publish with the release)

- **Vector tests:** ship JSON test vectors for $K \in \{48, 64\}$, random α, p, and outcomes $u^{\backslash *} \in \left\{0, 10^{-12}, 0.5, 1 - 10^{-12}, 1\right\}$; include reference ΔA and ℓ_i to 1 ulp.

- **Path independence:** compare 1-slice vs n-slice mints for identical net Δα.

- **Softmax stability:** construct cases with extremely peaked and flat tickets; verify identical winners across clients.

- **Cross-client parity:** run identical vectors across at least two independent implementations and compare Q64.64 outputs bit-for-bit.

Notes on sources. Lanczos/Stirling are standard for $log\Gamma$ with known accuracy–cost trade-offs; see MathWorld and Pugh's analysis. Accurate log-sum-exp and softmax require shifting; see Higham et al. and Gundersen. Fixed-point Q64.64 is widely used on EVM; see

ABDK. Log-binomial via $log\Gamma$ is canonical; Stan Math documents the identity. (MathWorld, Laplace, arXiv, School of Mathematics, GitHub, Stan)

A Critical Notes:

"Proper Scoring" and the functionSPACE Incentive Model

The preceding goals, taken together, define a mechanism that is axiomatically incompatible with the classic definition of a **proper scoring rule (MSR)**. It is critical for all contributors to understand that this is a deliberate and foundational design choice, not an oversight. The pursuit of our goals, particularly the combination of **Participant-Funded Solvency** and the **Fair Pricing of Endogenous Risk**, necessitates a departure from the MSR paradigm.

The core axiomatic conflict is this: a proper scoring rule requires a **fixed payoff mapping**, where a participant's reward is determined by a pre-defined contract with a sponsor and is independent of future market participation. Our goals, conversely, demand a **dynamic payoff** that is explicitly a function of the final, participant-funded pool size in order to fairly compensate for **Metastate Risk** and incentivize market creation and participation. A system's payoff cannot be simultaneously dependent on and independent of the final pool size; thus, we are making a principled trade-off, sacrificing the property of classic "proper scoring" to achieve the property of being a fully self-sustaining economic primitive.

This trade-off enables a fundamentally different and richer incentive structure. An MSR is designed to coordinate a single dimension of information: **epistemic belief** about an outcome. The functionSPACE model coordinates two distinct but interwoven signals: a participant's **epistemic expression** (the shape of their PDF) and their **economic expression** (the Principal committed against a given level of Metastate Risk). This creates a higher-fidelity market that aggregates not just *what* participants believe will happen, but the **collective, risk-adjusted conviction** behind those beliefs.

Assertion Sources:

- Definition and requirements of proper scoring rules. Forecaster is paid by a function of the reported distribution and realized outcome only. stat.washington.edu
- Market scoring rules (MSR) implement proper scoring via a subsidized market maker. Trader payoff equals a difference in a proper score; funding is from a patron, not other traders. MasonEconPapers Duke Computer Science
- Equivalence of MSRs and cost-function market makers. Truthful myopic incentives derive from convex cost functions tied to proper scores. Again, not self-funded. arXiv econcs.seas.harvard.edu Proceedings of Machine Learning Research
- Parimutuel mechanisms are self-funded by construction. Winners are paid from losers’ stakes. This is the opposite funding model from proper scoring/MSR. Nicholas Economides Stanford University

- Competitive or relative payouts break properness. When payoffs depend on others' reports, truthful reporting is no longer a dominant strategy. las.inf.ethz.ch arXiv

This implementation of the Dirichlet is Concave

In exponential-family theory, the log-partition function $A(\eta)$ is convex in the natural parameters. For Dirichlet, $A(\eta) = logB(\alpha) = \sum_k log\Gamma(\alpha_k) - log\Gamma(\sum_k \alpha_k)$ with $\eta_k = \alpha_k - 1$; this is the standard convex object. (people.eecs.berkeley.edu, cs.princeton.edu)

The spec defines the "Dirichlet potential" as

$$A_{spec}(\alpha) = log\Gamma\left(\sum_k \alpha_k\right) - \sum_k log\Gamma(\alpha_k) \quad = \quad - logB(\alpha).$$

This is the negative of the convex log-partition, hence concave on $R^{K+1}_{>0}$. Its Hessian is

$$\nabla^2 A_{spec}(\alpha) = \psi_1(P)\, 11^\top - diag\left(\psi_1(\alpha_k)\right), \quad P = \sum_k \alpha_k,$$

with ψ_1 the trigamma function ($\psi_1 > 0$), yielding negative quadratic forms on directions that conserve mass and producing diminishing mint rates as mass concentrates. (tminka.github.io)

- Why the signs check out: $log\Gamma(x)$ is convex on $x > 0$ (trigamma > 0). Therefore $logB(\alpha)$ is convex, so $- logB(\alpha) = A_{spec}(\alpha)$ is concave. This resolves the literature: sources calling the "Dirichlet potential" convex are referring to $logB(\alpha)$, not negative. (proofwiki.org, ScienceDirect)

$A_{spec}(\alpha) = log\Gamma(\sum\alpha) - \sum log\Gamma(\alpha)$ is concave.
log-partition, use $logB(\alpha)$ is convex.

Gradient, Hessian, and Concavity (normative)

Domain. Work on $\{\alpha \in R^{K+1}_{>0} : \alpha_k \geq \varepsilon_\alpha\}$ with ε_α per §10.6 (Q64.64). **Notation.** $P = \sum_k \alpha_k$. ψ is digamma, ψ_1 is trigamma. 1 is the all-ones vector.

Gradient.

$$\frac{\partial A}{\partial \alpha_k} = \psi(P) - \psi(\alpha_k).$$

Hessian.

$$\nabla^2 A(\alpha) = \psi_1(P)\, 11^\top - diag\left(\psi_1(\alpha_k)\right).$$

Mass-conserving curvature. For any v with $1^{\top}v = 0$,

$$v^{\top}\nabla^{2}A(\alpha)\, v =- \sum_{k} \psi_{1}(\alpha_{k})\, v_{k}^{2} \ < \ 0,$$

so A is strictly concave along directions that only redistribute mass. This yields diminishing ΔA as mass concentrates, matching §7.6’s incentive narrative.

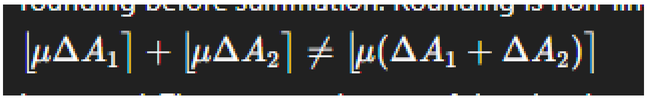
Global concavity. Since $A(\alpha) =- logB(\alpha)$ and the Dirichlet log-partition $logB$ is convex, A is concave on $\alpha_{k} > 0$. Use this as the single reference when asserting concavity.

Open Items

Sell

Path-independence vs. per-trade rounding

§7.2.2 asserts that slicing a trade cannot change total minted claims because ΔA telescopes. This is true in real arithmetic, but on-chain each slice is multiplied by μ and *quantised* to the claim unit with “ties-to-even” rounding before summation. Rounding is non-linear, so
⌊ μΔA1⌉+⌊ μΔA2⌉≠⌊ μ(ΔA1+ΔA2)⌉\bigl\lfloor\!\mu\Delta A_1\bigr\rceil+\bigl\lfloor\!\mu\Delta A_2\bigr\rceil \neq \bigl\lfloor\!\mu(\Delta A_1+\Delta A_2)\bigr\rceil⌊μΔA1⌉+⌊μΔA2⌉=⌊μ(ΔA1+ΔA2)⌉



$$\lfloor \mu\Delta A_1 \rceil + \lfloor \mu\Delta A_2 \rceil \neq \lfloor \mu(\Delta A_1 + \Delta A_2) \rceil$$

in general. The spec needs a proof that the chosen fixed-point precision and rounding rule bound the maximum drift or an explicit reconciliation step.

Simple round down can fix. Mu is likely to be 10 unless we find a perfect balance somewhere.

7.3 and 7.1 Not written properly yet

Rename first mover dividend, information claim/claim units/claims to be aligned