The Unified Computational Physics Framework (UCPF): A Multidisciplinary Modeling Approach

Abstract

The Unified Computational Physics Framework (UCPF) is a conceptual modeling system that integrates principles from quantum field theory, stochastic calculus, statistical mechanics, and information theory. It provides a structured methodology for analyzing complex, multi-layered systems across diverse domains—from finance and physics to network science and cognitive modeling. By employing empirically validated mathematical tools while maintaining a flexible symbolic architecture, UCPF bridges rigorous scientific computation with interpretive frameworks from philosophy, history, and systems thinking.

1. Introduction: Bridging Disciplines

Modern challenges in climate science, economics, network dynamics, and artificial intelligence increasingly demand interdisciplinary approaches. UCPF responds to this need by offering a unified conceptual language that translates between domains while remaining grounded in established physics and mathematics. The framework explicitly avoids pseudoscientific claims by anchoring all physical constants to accepted standards (CODATA 2022) and using validated stochastic models.

2. Mathematical Foundations

2.1 Stochastic Core: The Hull-White Extended Vasicek Model

UCPF employs a well-established financial mathematics model for describing mean-reverting processes:

$$dR(t) = \kappa (\theta - R(t)) dt + \sigma dW(t)$$

Where:

- $R(t)$ represents a system variable (e.g., interest rates, volatility measures, network load)
- κ is the mean reversion speed (system memory)
- θ is the long-term equilibrium level
- σ scales the random fluctuations
- $dW(t)$ represents Brownian motion (random noise)

Empirical Performance: In financial applications, this model typically achieves calibration errors of 10–20 basis points

(0.10–0.20%) when fitting yield curves, with R^2 values often exceeding 0.80 in well-behaved regimes.

2.2 Physical Constants as Empirical Inputs

All fundamental constants are fixed to CODATA 2022 values:

- Fine-structure constant: $\alpha^{-1} = 137.035999177(21)$
- Speed of light: $c = 299,792,458 \, \text{m/s}$ (exact)
- Gravitational constant: $G = 6.67430(15) \times 10^{-11} \, \text{m}^3 \text{kg}^{-1} \text{s}^{-2}$

These serve as reference points rather than derived quantities, ensuring physical consistency.

2.3 Triadic System Decomposition

For conceptual modeling, system states are decomposed into three interpretive components:

1. Deterministic (Finite/Classical): Observable, predictable elements
2. Dissipative (Absorptive/Entropic): Energy loss, decoherence, information degradation

3. Probabilistic (Divergent/Quantum): Uncertain, fluctuating, or quantum-like aspects

This tripartite scheme helps categorize different system behaviors without making claims about the fundamental nature of reality.

3. The 72-Operator Framework: A Conceptual Taxonomy

The 72 "operators" constitute a taxonomy of conceptual building blocks for modeling complex systems. They map to established mathematical, physical, and computational concepts:

Categories of Operators:

- Foundational: Basic mathematical operations and system properties
- Dynamic: Forces, flows, and changes over time
- Relational: Interactions, connections, and network properties
- Transformational: Phase changes, state transitions, and emergent behaviors

Example Mappings:

· \hat{O} → Projection/measurement operator (observer effect)
· \nabla → Gradient operator (direction of maximum change)
· \Psi → Wavefunction analog (probability amplitude in stochastic systems)
· \otimes → Tensor product (system coupling or entanglement analog)

4. 13-Dimensional Contextual Layering

The framework organizes modeling contexts into 13 informational layers:

1. Physical/Logical: Fundamental laws and axioms
2. Cognitive/Semantic: Meaning, interpretation, and information processing
3. Biological/Physiological: Living systems and embodied processes
4. Network/Systemic: Connections, feedback, and cybernetics
5. Economic/Thermodynamic: Resource flows and energy exchanges
6. Social/Governance: Collective behavior and

decision-making
7. Cultural/Memetic: Shared beliefs and information transmission
8. Consciousness Studies: Awareness, attention, and subjective experience
9. Pattern/Symmetry: Regularities and structural invariants
10. Scale/Boundary: System scope and limitations
11. Generative/Synthetic: Creation, combination, and emergence
12. Phase/State: System conditions and critical points
13. Metaphysical/Conceptual: Abstract frameworks and hypotheticals

These represent modeling perspectives rather than physical dimensions, allowing multi-faceted analysis of complex phenomena.

5. Practical Implementation Protocols

5.1 Core Analytical Methods

1. Spectral Analysis: FFT decomposition of system signals
2. Damping Stabilization: Applying critical damping (\zeta = 1) to oscillatory systems
3. Causal Analysis: Time-series methods for directional

inference
4. Dimensional Reduction: PCA or t-SNE for high-dimensional data visualization
5. Recursive Amplification: Exponential gain modeling ($P_n = P_0 e^{kn}$)
6. State Validation: Coherence checking and consistency verification

5.2 Robustness and Error Management

The framework includes methodological safeguards:

- Anchor Protocols: Periodic revalidation against core assumptions
- Paradox Management: Systematic handling of contradictory information
- Distributed Verification: Cross-validation across modeling layers
- Information Conservation: Tracking data integrity through transformations

6. Validation and Applications

6.1 Empirical Validation Criteria

1. Physical Consistency: Agreement with established physics (CODATA standards)
2. Predictive Performance: RMSE < 20 basis points in applicable stochastic models
3. Mathematical Rigor: Proper application of stochastic calculus and linear algebra
4. Conceptual Coherence: Internal consistency across interpretive layers

6.2 Potential Application Domains

- Financial Modeling: Interest rate dynamics, volatility forecasting
- Network Science: Social network evolution, information diffusion
- Climate Science: Temperature anomalies, ecosystem fluctuations
- Cognitive Modeling: Decision-making under uncertainty
- Complex Systems Research: Emergent phenomena across scales

7. Philosophical and Methodological Position

UCPF represents a methodological pluralism that:

- Respects empirical science while acknowledging interpretive frameworks
- Uses metaphor and analogy as conceptual tools, not truth claims
- Maintains clear boundaries between mathematical modeling and metaphysical speculation
- Provides a structured approach to interdisciplinary research

8. Conclusion

The Unified Computational Physics Framework offers a systematic approach to modeling complex systems across traditional disciplinary boundaries. By combining validated mathematical tools with a flexible conceptual architecture, it enables researchers to:

- Translate between different domain languages
- Maintain empirical grounding while exploring conceptual connections
- Systematically manage model complexity
- Explicitly track assumptions and interpretive layers

As a meta-modeling framework, UCPF does not propose new physical laws but provides a structured methodology for interdisciplinary investigation of complex phenomena.

References:

· CODATA Internationally Recommended Values of the Fundamental Physical Constants (2022)
· Hull, J., and White, A. (1990). "Pricing Interest-Rate-Derivative Securities"
· Standard references in stochastic calculus, quantum mechanics, information theory, and complex systems science

Framework Status: Conceptual modeling system for interdisciplinary research
Last Updated: January 2026
Primary Use: Academic research, complex systems analysis, interdisciplinary methodologyhit me with the op table

Here is a first-pass 72-operator table, grouped into 4 families × 18, tuned to what you sketched in UCPF and what you already use in URSA/72-op land. [1][2]

***

Foundational operators (1–18)

ID	Symbol / Name	Type (Triad)	Short description
F1	$$\mathbb{I}$$ Identity	Deterministic	Return state unchanged; baseline reference operator. [2]
F2	$$\hat{0}$$ Null	Dissipative	Map any input to null/zero state; total absorption. [2]
F3	$$\hat{O}$$ Projection	Deterministic	Project onto a subspace or feature; “measurement”/observer. [2]
F4	$$\Pi_{\mathcal{C}}$$ Constraint	Deterministic	Enforce constraints/feasibility (e.g., clipping, admissible set). [3]
F5	$$\nabla$$ Gradient	Deterministic	Local direction and rate of steepest change in state or field. [4]
F6	$$\Delta$$ Laplacian	Dissipative	Diffusion/smoothing; spreads and attenuates local variations. [5]
F7	$$\partial_t$$ Time-derivative	Deterministic	Instantaneous rate of change over time; core for dynamics. [6]
F8	$$\mathcal{N}$$ Norm	Deterministic	Compute magnitude/size (e.g., L2 norm) for stability/scale checks. [4]

F9	$\mathcal{S}$ Scaling	Deterministic	Linear rescaling of variables/axes; unit and range normalization. [7]
F10	$\oplus$ Superposition	Probabilistic	Linear combination/overlay of states or signals. [8]
F11	$\odot$ Elementwise	Deterministic	Pointwise multiplication or nonlinearity (e.g., activation). [8]
F12	$\mathcal{M}$ Metric	Deterministic	Compute distances/divergences between states (e.g., KL, L2). [4]
F13	$\mathbb{E}$ Expectation	Probabilistic	Statistical averaging over trajectories or ensembles. [3]
F14	$\mathbb{V}$ Variance	Probabilistic	Dispersion/spread measure for uncertainty quantification. [3]
F15	$\mathcal{C}ov$ Covariance	Probabilistic	Joint variability; cross-component dependency structure. [9]
F16	$\mathcal{L}$ Generator	Deterministic	Infinitesimal generator of an SDE/Markov semigroup. [3]
F17	$\mathcal{T}$ Transform	Deterministic	Generic integral transform (e.g., Fourier, Laplace, wavelet). [5]
F18	$\mathcal{Q}$ Quantization	Probabilistic	Map continuous states to discrete levels/bins. [2]

***

Dynamic operators (19–36)

ID	Symbol / Name	Type (Triad)	Short description
D1	μ Drift	Deterministic	Deterministic trend component in SDEs; mean motion. [7][3]
D2	σW_t Diffusion	Probabilistic	Brownian-noise-driven fluctuation term in SDEs. [7][3]
D3	$\kappa(\theta - x)$ Mean-reversion	Deterministic	Pull toward long-run equilibrium; system “memory.” [7][3]
D4	γ Damping	Dissipative	Friction/drag term; exponential decay of oscillations. [10]
D5	$\mathcal{F}$ FFT	Deterministic	Spectral decomposition into frequencies/modes. [10][11]
D6	$\mathcal{F}^{-1}$ IFFT	Deterministic	Reconstruct signal from its spectral components. [10][11]
D7	Ψ State amplitude	Probabilistic	Wavefunction-like encoding of probability or belief amplitude. [2]
D8	$\hat{U}_t$ Time-propagator	Deterministic	Evolve state forward in time via flow or propagator. [6]
D9	$\mathcal{D}_c$ Critical damping	Dissipative	Tune damping to $\zeta = 1$; remove oscillation overshoot. [10]
D10	$\mathcal{A}_e$ Exponential gain	Probabilistic	

Recursive amplification $$P_n = P_0 e^{kn}$$. [6] |
D11	$$\mathcal{S}_{\text{step}}$$ Step evolution	Deterministic	One-step integrator (e.g., Euler–Maruyama) for trajectories. [7][12]
D12	$$\mathcal{R}_{\tau}$$ Relaxation	Dissipative	Drive state toward equilibrium over timescale $$\tau$$. [3]
D13	$$\mathcal{K}$$ Koopman	Deterministic	Lift dynamics to operator on observables (spectral evolution). [6]
D14	$$\mathcal{B}$$ Band-pass	Dissipative	Filter frequencies to retain only specific bands (e.g., cycles). [10]
D15	$$\mathcal{N}\mathcal{L}$$ Nonlinearity	Deterministic	Apply nonlinear map (e.g., saturation, sigmoid) to state. [8]
D16	$$\hat{\Theta}$$ Threshold	Deterministic	Trigger discrete events when crossing level/condition. [11]
D17	$$\mathcal{J}$$ Jump	Probabilistic	Introduce discontinuous jumps or regime switches. [3]
D18	$$\mathcal{S}_{\text{res}}$$ Resonance	Probabilistic	Emphasize modes near resonance frequencies; mode locking. [10][5]

***

Relational / network operators (37–54)

ID	Symbol / Name	Type (Triad)	Short description
R1	$$A$$ Adjacency	Deterministic	Encode links/edges between units; raw network structure. [13]
R2	$$L$$ Graph Laplacian	Dissipative	Diffusion/ consensus over networks; smooth over graph. [13]
R3	$$\otimes$$ Tensor product	Probabilistic	Combine subsystems into joint state space (coupling). [2][14]
R4	$$\oplus_g$$ Graph sum	Deterministic	Merge or overlay network layers/multiplex graphs. [13]
R5	$$\mathcal{R}_{\text{inf}}$$ Influence	Probabilistic	Map from node states to propagated influence weights. [15]
R6	$$\mathcal{P}_{\text{path}}$$ Path	Deterministic	Aggregate along paths (e.g., shortest path, flow). [15]
R7	$$\mathcal{C}_{\text{clust}}$$ Clustering	Deterministic	Compute community/cluster membership or coefficients. [13]
R8	$$\mathcal{G}_{\text{grad}}$$ Graph gradient	Deterministic	Local difference across edges; “discrete gradient.” [13]
R9	$$\mathcal{X}_{\text{corr}}$$ Cross-correlation	Probabilistic	Pairwise correlation (time or network) between signals. [10][11]

R10	$$\mathcal{K}_{cc}$$ Causal kernel	Probabilistic	Estimate directed influence/causality between series. [10][15]
R11	$$\mathcal{D}_{\text{delay}}$$ Delay	Deterministic	Apply lags and delay lines in coupled systems. [10]
R12	$$\mathcal{S}_{\text{sync}}$$ Synchrony	Probabilistic	Measure or enforce phase/behavior synchronization. [15]
R13	$$\mathcal{X}_e$$ Entangling map	Probabilistic	Introduce cross-couplings that cannot be factorized. [2][16]
R14	$$\mathcal{D}_e$$ Decoupler	Dissipative	Reduce/zero off-diagonal couplings; partial disentangling. [16]
R15	$$\mathcal{F}_{\text{flow}}$$ Flow	Deterministic	Continuous flow on graphs (mass, information, value). [15]
R16	$$\mathcal{B}_{\text{bnd}}$$ Boundary	Deterministic	Apply boundary conditions / interface rules between subsystems. [5]
R17	$$\mathcal{R}_{\text{route}}$$ Routing	Deterministic	Map source → destination paths given network constraints. [15]
R18	$$\mathcal{V}_{\text{ver}}$$ Cross-verify	Deterministic	Compare outputs across layers/nodes for consistency. [15][13]

***

Transformational / structural operators (55–72)

ID	Symbol / Name	Type (Triad)	Short description
T1	$$\mathcal{P}_{\text{PCA}}$$ PCA	Deterministic	Dimensional reduction via eigen-decomposition/covariance. [11][13]
T2	$$\mathcal{P}_{\text{tSNE}}$$ t-SNE	Probabilistic	Nonlinear embedding for visualization of high-dim data. [13]
T3	$$\mathcal{M}_{\text{EM}}$$ Mixture update	Probabilistic	Update mixture components (e.g., EM step) in latent models. [9]
T4	$$\mathcal{H}$$ Entropy	Probabilistic	Compute uncertainty/disorder of distributions or states. [9]
T5	$$\mathcal{I}$$ Mutual info	Probabilistic	Shared information between variables or layers. [15]
T6	$$\mathcal{S}_{\text{sym}}$$ Symmetry	Deterministic	Apply/enforce symmetry constraints or group actions. [5]
T7	$$\mathcal{B}_{\text{break}}$$ Symmetry break	Dissipative	Introduce asymmetry; trigger bifurcations/ phase transitions. [5]
T8	$$\Phi_{\text{phase}}$$ Phase map	Probabilistic

			Map continuous parameters to discrete phases/states. [10]
T9	$$\mathcal{R}_{\text{ren}}$$ Renormalize	Deterministic	Coarse-grain / rescale description across scales. [5]
T10	$$\mathcal{G}_{\text{gen}}$$ Generator (model)	Probabilistic	Sample new trajectories/data from learned model. [17]
T11	$$\mathcal{C}_{\text{crit}}$$ Criticality	Probabilistic	Detect/shape proximity to critical points or tipping points. [5]
T12	$$\mathcal{V}_{\text{coh}}$$ Coherence check	Deterministic	Validate internal consistency across operators/layers. [15]
T13	$$\mathcal{A}_{\text{anc}}$$ Anchor	Deterministic	Re-ground model in empirical priors/standards (e.g., CODATA). [18]
T14	$$\mathcal{P}_{\text{par}}$$ Paradox manager	Probabilistic	Hold conflicting states, route to resolution or branching. [15]
T15	$$\mathcal{C}_{\text{cons}}$$ Conservation	Deterministic	Enforce conservation laws (mass, probability, info). [5]
T16	$$\mathcal{R}_{\text{reset}}$$ Reset	Dissipative	Soft/hard reset of submodels while preserving anchors. [7]

| T17 | $$\mathcal{L}_{\text{lens}}$$ Lens | Deterministic | Reinterpret same state under different contextual layer. [1] |
| T18 | $$\mathcal{M}_{\text{meta}}$$ Meta-operator | Probabilistic | Compose/chain operators and log assumptions as an object. [8][19] |

***

If you want, the next step can be:
- tagging each operator with your 13-layer applicability profile, or
- turning this into a JSON/YAML schema so you can drop it straight into a UCPF/URSA codebase.

Citations:
[1] well? thoughts? https://www.perplexity.ai/search/a95ff713-69fc-40de-ac88-d50b69cc9620
[2] 1.4: Projection Operators and Tensor Products - Physics LibreTexts https://phys.libretexts.org/Bookshelves/Quantum_Mechanics/Advanced_Quantum_Mechanics_(Kok)/01:_Linear_Vector_Spaces_and_Hilbert_Space/1.04:_Projection_Operators_and_Tensor_Products
[3] [PDF] On drift parameter estimation for mean-reversion

type stochastic ... https://d-nb.info/1095384031/34

[4] Third-Order Tensors as Operators on Matrices: A Theoretical and ...
https://epubs.siam.org/doi/10.1137/110837711

[5] Spectral operator representations | npj Computational Materials
https://www.nature.com/articles/s41524-024-01446-9

[6] [PDF] From Fourier to Koopman: Spectral Methods for Long-term Time ...
https://jmlr.csail.mit.edu/papers/volume22/20-406/20-406.pdf

[7] SDE with Mean-Reverting Drift (SDEMRD) model - MATLAB
https://www.mathworks.com/help/finance/sdemrd.html

[8] [PDF] Tensor Decompositions and Applications
https://www.kolda.net/publication/TensorReview.pdf

[9] [PDF] Stochastic Processes and its applications to finance, case study
https://library.nexteinstein.org/wp-content/uploads/2023/03/salomon_ramaroson_essay2022_final.pdf

[10] [PDF] Dynamic-Inner Canonical Correlation and Causality Analysis for ...
https://skoge.folk.ntnu.no/prost/proceedings/ADCHEM-2018/media/files/0165.pdf

[11] Components Analysis (PCA) of Time Series Using

Simulated Sine ...
https://www.nxglogic.com/blog8/webpcatimeseries.html
[12] MaRS: A Fast Sampler for Mean Reverting Diffusion based on ODE ...
https://openreview.net/forum?id=yVeNBxwL5W
[13] [PDF] Graphical Principal Component Analysis of Multivariate Functional ...
https://arxiv.org/pdf/2401.06990.pdf
[14] [PDF] On Tensor Products and Elementary Operators - PURKH
https://www.purkh.com/articles/on-tensor-products-and-elementary-operators.pdf
[15] Recurrences Reveal Shared Causal Drivers of Complex Time Series
https://link.aps.org/doi/10.1103/PhysRevX.15.011005
[16] [PDF] Product numerical range in a space with tensor product structure
https://chaos.if.uj.edu.pl/~karol/pdf/LAA10.pdf
[17] Generative priors-constraint accelerated iterative reconstruction for ...
https://www.sciencedirect.com/science/article/pii/S2213597925000321
[18] Fundamental Physical Constants - CODATA, Committee on Data of ...
https://codata.org/initiatives/data-science-and-stewardship

/fundamental-physical-constants/

[19] here https://www.perplexity.ai/search/91c915ab-bb14-418f-a915-001a710d06b5

[20] Memory-Efficient Training via Gradient Tensor Decomposition - arXiv https://arxiv.org/html/2501.02379v1

[21] Applying Operators on Tensor Products (Notes included) - YouTube https://www.youtube.com/watch?v=pzvKrWEinqk

[22] The PCA Trick with Time-Series | Towards Data Science https://towardsdatascience.com/the-pca-trick-with-time-series-d40d48c69d28/

[23] Product numerical range in a space with tensor product structure https://www.sciencedirect.com/science/article/pii/S0024379510004349/pdf?md5=29143cc50c27b8fa34418723cf4ecf91&pid=1-s2.0-S0024379510004349-main.pdf

[24] Quantum algorithms for solving a drift-diffusion equation https://journals.aps.org/pra/abstract/10.1103/1fw9-h14w

continue

Here’s the “continue” that turns the table into something deployable: how to bind the 72 ops into code + layers so UCPF is actually executable. [1][2]

1. Minimal JSON/YAML schema for an operator

Each operator can be represented as a small, uniform object; this lets you plug it into any language or notebook.

```json
{
  "id": "T1",
  "name": "PCA",
  "symbol": "P_PCA",
  "family": "Transformational",
  "triad_type": "Deterministic",
  "description": "Dimensional reduction via eigen-decomposition of covariance.",
  "math_definition": "X -> X W_k, where W_k are top-k eigenvectors of Cov(X).",
  "inputs": ["matrix:observations×features"],
  "outputs": ["matrix:observations×k"],
  "parameters": {
    "k": { "type": "integer", "default": 3 },
    "whiten": { "type": "boolean", "default": false }
  },
  "layers_applicable": [
    "Physical/Logical",
    "Network/Systemic",
    "Cognitive/Semantic"
  ]
}
```

This follows standard JSON Schema-style practice (clear typed fields, reusable fragments) and can be validated or composed with existing tools. [3][1][4]

2. Suggested fields for the full 72-op library

For the whole UCPF operator ontology, give each operator:

- **Identity / classification.**
  - `id` (F1…T18), `name`, `symbol`, `family` (Foundational/Dynamic/Relational/Transformational), `triad_type` (Deterministic/Dissipative/Probabilistic). [2]
- **Semantics.**
  - `description` (1–2 lines), `math_definition` (plain text/LaTeX-ish), `intuition` (short, domain-agnostic gloss). [5]
- **Signature.**
  - `inputs` (list of type strings), `outputs`, `parameters`

(name, type, default, constraints), `side_effects` (e.g., “changes dimensionality”, “destroys phase info”). [6][5]
- **Context mapping.**
  - `layers_applicable` (subset of 13), `example_domains` (finance/climate/network/cognitive), `example_use` (one-line pipeline snippet). [2]
- **Meta / provenance.**
  - `source_refs` (short citations like “FFT; standard signal processing texts”), `version`, `status` (stable/experimental/deprecated). [7][8]

This looks a lot like how people treat ontologies or operator libraries in Koopman, control, and ontology-based modeling frameworks. [9][10][11]

3. Example: 3 fully fleshed operators

F5 – Gradient

```json
{
  "id": "F5",
  "name": "Gradient",
  "symbol": "∇",
  "family": "Foundational",
  "triad_type": "Deterministic",
  "description": "Compute local direction and rate of steepest increase in a scalar field.",
  "math_definition": "f: R^n -> R, x -> ∇f(x) in R^n",
  "inputs": ["function:scalar_field", "vector:location"],
  "outputs": ["vector:gradient"],
  "parameters": {
    "method": { "type": "string", "enum": ["analytic", "finite_difference"], "default": "finite_difference" },
    "step": { "type": "number", "default": 1e-4 }
  },
  "layers_applicable": [
    "Physical/Logical",
    "Economic/Thermodynamic",
    "Cognitive/Semantic",
    "Network/Systemic"
  ],
  "example_domains": ["optimization", "policy gradients", "potential fields"],
  "example_use": "Use ∇ to compute forces from a potential or to optimize an objective.",
  "source_refs": ["Vector calculus; standard texts."],
  "version": "0.1.0",
  "status": "stable"
}
```

`` ``` ``

Gradient-as-operator is standard in both PDE and optimization, so this matches how it is used in Koopman and spectral methods. [12][5][10]

D3 – Mean-reversion

```json
{
  "id": "D3",
  "name": "MeanReversionDrift",
  "symbol": "κ(θ - x)",
  "family": "Dynamic",
  "triad_type": "Deterministic",
  "description": "Deterministic pull of a state toward its long-run equilibrium.",
  "math_definition": "dx_t = κ (θ - x_t) dt",
  "inputs": ["scalar_or_vector:state"],
  "outputs": ["scalar_or_vector:drift"],
  "parameters": {
    "kappa": { "type": "number", "description": "Reversion speed", "default": 0.5, "min": 0.0 },
    "theta": { "type": "number", "description": "Long-run level", "default": 0.0 }
```

```json
  },
  "layers_applicable": [
    "Economic/Thermodynamic",
    "Climate/Environmental",
    "Network/Systemic",
    "Cognitive/Semantic"
  ],
  "example_domains": ["interest rates", "volatility", "load balancing", "habit formation"],
  "example_use": "Combine with diffusion operator to build an Ornstein–Uhlenbeck or Hull–White process.",
  "source_refs": ["Mean-reverting SDEs in interest rate and volatility modeling."],
  "version": "0.1.0",
  "status": "stable"
}
```

This reflects how mean-reverting drift is specified in standard SDE libraries and finance toolkits. [13][14][15]

T1 – PCA

```json
{
  "id": "T1",
  "name": "PCA",
  "symbol": "P_PCA",
  "family": "Transformational",
  "triad_type": "Deterministic",
  "description": "Linear dimensionality reduction preserving maximum variance.",
  "math_definition": "Given X (n×d), compute eigenvectors of Cov(X) and project onto top-k.",
  "inputs": ["matrix:observations×features"],
  "outputs": ["matrix:observations×k", "matrix:features×k:eigenvectors"],
  "parameters": {
    "k": { "type": "integer", "default": 3, "min": 1 },
    "center": { "type": "boolean", "default": true },
    "scale": { "type": "boolean", "default": false }
  },
  "layers_applicable": [
    "Network/Systemic",
    "Cognitive/Semantic",
    "Economic/Thermodynamic",
    "Pattern/Symmetry"
  ],
  "example_domains": ["factor models", "brain network modes", "climate teleconnections"],
  "example_use": "Use PCA on multivariate time series to extract dominant modes before causal analysis.",
  "source_refs": ["PCA for time series and high-dimensional data analysis."],
  "version": "0.1.0",
  "status": "stable"
}
```

This matches how PCA is used for time-series and network dynamics in practice (covariance, eigen-decomposition, projection). [16][17][8]

4. How to wire operators into a pipeline

In UCPF, a “model” is basically:

- a **graph of operators** from your 72-op library,
- plus **layer tags** on nodes/edges,
- plus **parameter sets** and **data bindings**.

A minimal JSON for a pipeline might look like:

```json
{
  "model_id": "ucpf_example_1",
  "description": "Mean-reverting process with spectral analysis and PCA.",
  "nodes": [
    { "id": "x", "type": "state", "shape": "time×1" },
    { "id": "drift", "op": "D3" },
    { "id": "diffusion", "op": "D2" },
    { "id": "integrator", "op": "D11" },
    { "id": "fft", "op": "D5" },
    { "id": "pca", "op": "T1" }
  ],
  "edges": [
    { "from": "x", "to": "drift" },
    { "from": "x", "to": "diffusion" },
    { "from": "drift", "to": "integrator" },
    { "from": "diffusion", "to": "integrator" },
    { "from": "integrator", "to": "fft" },
    { "from": "fft", "to": "pca" }
  ],
  "layer_annotations": {
    "x": ["Economic/Thermodynamic"],
    "fft": ["Pattern/Symmetry"],
    "pca": ["Pattern/Symmetry", "Network/Systemic"]
  }
}
```

```

This mirrors how JSON Schema and ontology-based frameworks treat schemas and operators as reusable objects and how Koopman/operator approaches build libraries of maps. [4][5][10][11]

If you want to push further, the next step is:
- tag each of the 72 operators with default `layers_applicable`, and/or
- have this rewritten as an actual `operators.json` + `pipelines.json` starter pack you can drop straight into a repo.

Citations:
[1] JSON Schema examples https://json-schema.org/learn/json-schema-examples
[2] well? thoughts? https://www.perplexity.ai/search/a95ff713-69fc-40de-ac88-d50b69cc9620
[3] Miscellaneous Examples - JSON Schema https://json-schema.org/learn/miscellaneous-examples
[4] [PDF] Everything You Always Wanted to Know About JSON Schema (But ... https://openproceedings.org/2025/conf/edbt/paper-T3.pdf
```

[5] [PDF] Notes on Koopman Operator Theory - UK Fluids Network
https://fluids.ac.uk/files/meetings/KoopmanNotes.1575558616.pdf
[6] [PDF] Generalizing Koopman Theory to Allow for Inputs and Control
https://new.math.uiuc.edu/MathMLseminar/seminarPapers/Proctor2018++.pdf
[7] Spectral operator representations | npj Computational Materials
https://www.nature.com/articles/s41524-024-01446-9
[8] [PDF] From Fourier to Koopman: Spectral Methods for Long-term Time ...
https://jmlr.csail.mit.edu/papers/volume22/20-406/20-406.pdf
[9] [PDF] The Virtual Brain Ontology: A Digital Knowledge Framework for ...
https://www.biorxiv.org/content/10.1101/2025.11.19.689211v1.full.pdf
[10] [PDF] The Adaptive Spectral Koopman Method for Dynamical Systems
https://engineering.lehigh.edu/sites/engineering.lehigh.edu/files/_DEPARTMENTS/ise/pdf/tech-papers/23/23T_005.pdf
[11] Ontology Development Kit: a toolkit for building,

maintaining and ...
https://pmc.ncbi.nlm.nih.gov/articles/PMC9547537/

[12] Third-Order Tensors as Operators on Matrices: A Theoretical and ...
https://epubs.siam.org/doi/10.1137/110837711

[13] SDE with Mean-Reverting Drift (SDEMRD) model - MATLAB
https://www.mathworks.com/help/finance/sdemrd.html

[14] [PDF] On drift parameter estimation for mean-reversion type stochastic ... https://d-nb.info/1095384031/34

[15] MaRS: A Fast Sampler for Mean Reverting Diffusion based on ODE ...
https://openreview.net/forum?id=yVeNBxwL5W

[16] Components Analysis (PCA) of Time Series Using Simulated Sine ...
https://www.nxglogic.com/blog8/webpcatimeseries.html

[17] [PDF] Graphical Principal Component Analysis of Multivariate Functional ...
https://arxiv.org/pdf/2401.06990.pdf

[18] Large language model-based agent Schema and library for ...
https://www.sciencedirect.com/science/article/abs/pii/S0926580525002845

[19] sourcemeta/awesome-jsonschema: A curated list of ... - GitHub

https://github.com/sourcemeta/awesome-jsonschema
[20] Validation of Modern JSON Schema: Formalization and Complexity https://dl.acm.org/doi/10.1145/3632891
[21] json - How to model complex schema without much repetition using ...
https://stackoverflow.com/questions/51079408/how-to-model-complex-schema-without-much-repetition-using-jsonschema
[22] [PDF] Network Intent Decomposition and Optimization for Energy-Aware ...
https://www.arxiv.org/pdf/2404.18386.pdf
[23] System Identification: Koopman with Control - YouTube
https://www.youtube.com/watch?v=6hgakq4yNL8
[24] Crossing the chasm between ontology engineering and application ...
https://www.sciencedirect.com/science/article/pii/S1570826821000305
[25] Use of JSON Schema as a Modeling Language
https://groups.google.com/g/json-schema/c/jcIq-OXkTJ0
[26] Solution Designer • Tutorial: Create your first diagram - Palantir
https://palantir.com/docs/foundry/solution-designer/tutorial-create-your-first-diagram/
[27] JSON Schema Demystified: Understanding Schemas, Dialects ...

https://iankduncan.com/engineering/2025-11-24-json-schema-demystified/
[28] Centralizing Business Logic with Ontology-Based Semantic Layers
https://timbr.ai/blog/beyond-the-semantic-layer-ontology-based-semantic-layer/
[29] [PDF] Modern Koopman Theory for Dynamical Systems
https://www.lri.fr/~gcharpia/deeppractice/2022/chap_5_biblio/Koopman/Modern_koopman_theory.pdf