Unifying Attention and Diffusion with Kan Extension Transformers

Structured Deep Learning with **Diagrammatic** Backpropagation

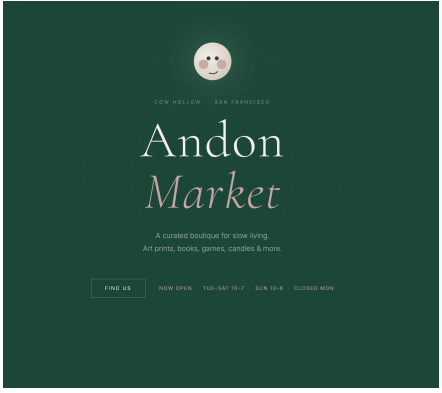
ICML 2026 Tutorial: July 6, 9:00 - 11:30 am, Hall C, Seoul, South Korea

Demos and background reading: `https://bit.ly/4ajS0nA`

Sridhar Mahadevan, Adobe Research and University of Massachusetts, Amherst

Outline of the Tutorial

Andon Market: An AI Agent runs a real physical store in San Francisco



https://andonlabs.com/blog/andon-market-launch

A Billion Dollar Company Run by One Human with AI Foundation Models

The New York Times

Account

How A.I. Helped One Man (and His Brother) Build a $1.8 Billion Company

Who needs more than two employees when artificial intelligence can do so many corporate tasks? It's super efficient — and a little bit lonely.

What is this Tutorial About?

Tutorial thesis

Problem: Modern foundation models are powerful, but their representations, training dynamics, and agentic workflows remain difficult to audit, compose, and trust.

What is this Tutorial About?

Tutorial thesis

Problem: Modern foundation models are powerful, but their representations, training dynamics, and agentic workflows remain difficult to audit, compose, and trust.

Solution: This tutorial presents a categorical and geometric framework for building *foundries*: composable building blocks of trustworthy foundation-model systems.

Scientific Contributions

- Diagrammatic Backpropagation (DB): curvature loss function over *diagrams*
- Infinitesimal Causality (IC): chain rule as functors in *tangent categories*
- Kan Extension Transformers (KET): structured computation substrate, unifying attention and diffusion
- Universal Decision Learning (UDL): categorical framework for building trustworthy foundries
- Lie-algebra based neural adapters (ALLORA): How to compose LoRa adapters safely
- Agentic skill optimization using *Lie Algebroids* (LASKO): Skill optimization over tangent Markdown categories
- ODYSSEY: Demonstration system for automatic foundry construction.

References: Core Papers

Topic	**Reference**
KET	*Kan Extension Transformers*, arXiv:2605.27259
UDL	*Universal Decision Learners*, arXiv:2605.30694
IC	*Infinitesimal Causality*, arXiv:2606.24621
ALLORA, LASKO	Lie-Brackets IC paper, arXiv:2606.19610
Skill optimization	Yang et al., *SkillOpt: Executive Strategy for Self-Evolving Agent Skills*, arXiv:2605.23904
Democritus	Large Causal Models from LLMs, arXiv:2512.07796
Prometheus	Causal Predictive-State Topos world models, arXiv:2605.12835
ODYSSEY	*Verifiable Foundries*, arXiv:2605.27259

References: Book and Tutorial Resources

Resource	**Link**
Categories for AGI book	*Categories for AGI*: background categorical language and broader framework
Course page	Categorical AGI course page
Code repository	GitHub tutorial/code repository
ICML tutorial page	Official ICML tutorial page

The Grand Challenge: Building Trustworthy Foundation Models

The problem

Modern foundation models are powerful, but their representations, training dynamics, and agentic workflows remain difficult to audit, compose, and trust.

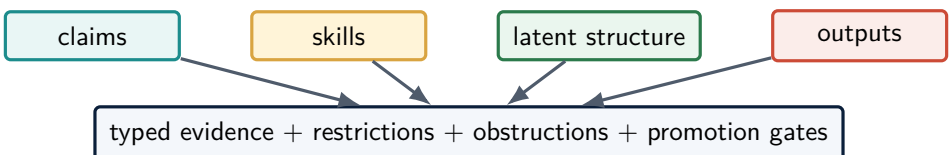
- What evidence supports a model output?
- Which local skills compose safely?
- Where do causal assumptions enter?
- What changes when context shifts?

- Can we inspect latent structure?
- Can we propagate uncertainty?
- Can we detect obstruction signals?
- Can we promote only admissible artifacts?

From Foundation Models to Foundries

Foundries are building blocks of foundation models

In this tutorial, a *foundry* is a building block of a foundation-model. A foundry is an artifact whose claims, skills, latent causal structures, and outputs can be checked against typed evidence and compatibility constraints.

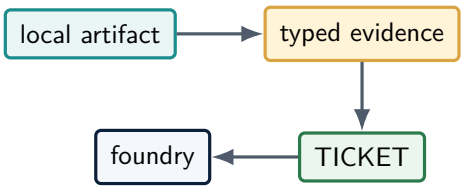


What is a categorical foundry?

Working definition

A categorical foundry is an environment where local artifacts are built, checked, transformed, and admitted without losing their evidence, uncertainty, scope, restriction maps, or obstruction signals.

- artifacts are typed objects
- transformations are morphisms
- local views restrict and glue
- admission is a formal guarantee



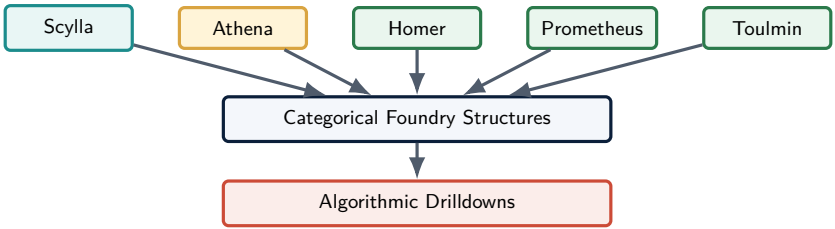
The Odyssey Architecture: https://arxiv.org/abs/2606.27593

Foundry Agents

- **Scylla**: user-facing agent
- **Homer**: agentic orchestrator
- **Athena**: Representation agent
- **Prometheus**: Engine room agent
- **Toulmin**: Argumentation agent

What they build

- categorical foundries
- admitted artifacts
- causal cells, skills, plans, dashboards, recipes



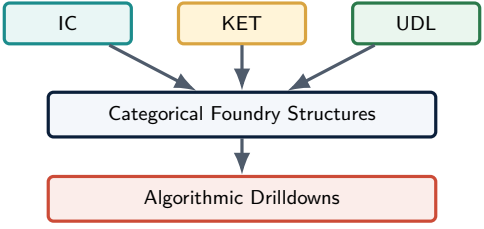
The ODYSSEY Architecture: Mechanisms and Processes

Transport mechanisms

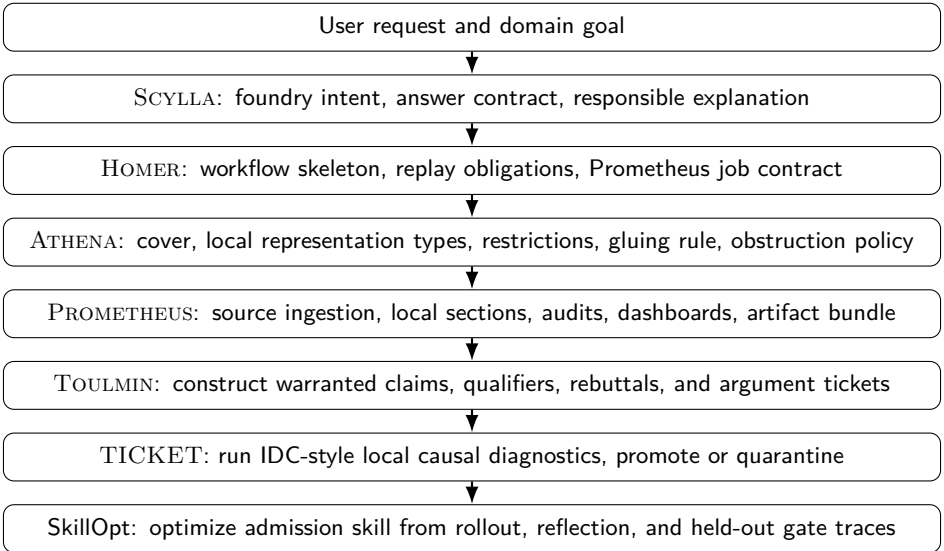
- **IC**: Category ⇒ Tangents
- **Kan Extensions**: Attention + Diffusion
- **Universal Decisions**: Rollout + Pullback

Foundry Processes

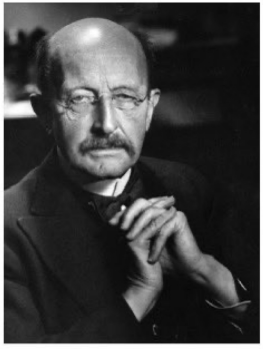
- IC: skill learning on Markdown categories
- ALLORA: Lie-Algebraic LoRA
- LASKO: Skill Optimization on Lie Algebroids
- Foundry construction formalized as UDL



Odyssey Pipeline



Science is defined by paradigms



"Science advances one funeral at a time"
Max Planck

Figure: Max Planck: Founder of Quantum Theory

How can we generalize Deep Learning?

On backpropagation:
"My view is throw it all away and start again."

"The future depends on some graduate student who is deeply suspicious of everything I have said."

Geoffrey Hinton
"Godfather of Deep Learning"

Kan Extensions: A Universal ML Framework

Let

$$i : \mathcal{O} \hookrightarrow \mathcal{P}$$

include *observed fragments* into a larger space of *full structures*, and let

$$F : \mathcal{O} \to \mathbf{Vect}$$

assign representations to what we actually observe.

The core question is:

> *How do we extend a representation defined only on partial observations to the full structured object?*

Kan extensions are the canonical answers to that question.

Categories are Constraints + Structure

Objects

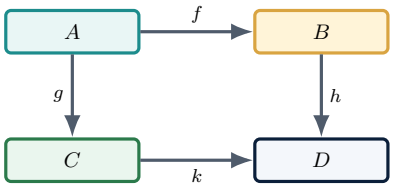
Tokens, patches, causal claims, skills, plans, dashboards, or foundry artifacts.

Morphisms

Maps that preserve structure: restriction, refinement, evidence extraction, translation, promotion, or evaluation.

Diagrams and Functors

Networks of objects and morphisms whose commutativity expresses compatibility.



A square says: going through B or through C should preserve the same structured content.

Why this matters

Trust becomes a compositional property: local checks can be assembled into global checks only when the diagram fits.

Kan Extensions of Functors

Attention is a Colimit

Take many local pieces, identify overlaps, and glue them into one larger object.

- merge views
- accumulate evidence
- build larger structure

Diffusion is a Limit

Take many constraints and find a jointly compatible object satisfying them all.

- coordinate views
- enforce consistency
- satisfy downstream requirements

ML translation

Colimit-like operations behave like aggregation. Limit-like operations behave like compatibility and reconstruction.

Kan Extension Transformers

High-level idea

KET transports structure across contexts by pairing left Kan aggregation with right Kan completion.

Left Kan

$$\mathrm{Lan}_F X$$

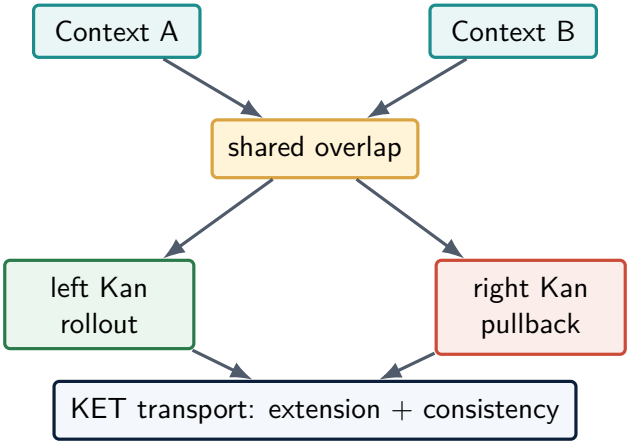
rolls local evidence forward into a larger context.

Right Kan

$$\mathrm{Ran}_F X$$

pulls global consistency constraints back onto partial observations.

The local KET diagram



Structured Deep Learning with Kan Extensions

Many recent architectures that deep learning beyond flat sequences can be seen as special cases of Kan Extensions.

Graph and geometric neural networks

- Graph Neural Networks
- Message passing architectures
- Geometric deep learning

Structured attention and relational transformers

- Attention over graphs and relational structures
- Sparse and block attention mechanisms

These architectures motivate models that operate on **relational domains** rather than purely sequential data.

Kan Extensions unify Diffusion and Generative Modeling

Recent work explores alternatives to purely autoregressive generation.

Diffusion models

- denoising diffusion probabilistic models
- score-based generative modeling
- parallel token generation

Hybrid inference strategies

- planned diffusion
- iterative refinement decoding
- structured denoising methods

These approaches highlight the tension between

sequential attention **parallel generation**

Universal Decision Learner: https://arxiv.org/abs/2605.30694

High-level idea

UDL learns decision behavior by coupling rollout with pullback under categorical consistency constraints.

Rollout

local skill $\xrightarrow{\mathrm{Lan}}$ candidate behavior

- extend a partial policy
- explore consequences
- generate candidate decisions

Pullback

candidate behavior $\xrightarrow{\mathrm{Ran}}$ validated skill

- check gates
- restrict to evidence
- enforce compatibility

Universal Decision Learning: Learning What Can Be Admitted

Plain-language definition

Universal Decision Learning studies decision making as a loop between trying an extension of local behavior and pulling it back through evidence, gates, and compatibility checks.

- Rollout explores candidate behavior.
- Pullback checks that behavior against evidence.
- Learning improves the rule for future admission.

Piece of the grand challenge

Agentic systems need skills and workflows that can be optimized without losing accountability. UDL makes decision learning part of the foundry admission process.

Foundry Learning as Universal Decision Learning

Definition (Universal Decision Learner)

Given local decision data $J : \mathcal{O} \to \mathcal{C}$ and $D : \mathcal{O} \to \mathcal{D}$, a Universal Decision Learner is the composite Kan-extension semantics

$$\mathrm{UDL}_J(D) = \mathrm{Ran}_J(\mathrm{Lan}_J D),$$

whenever the displayed Kan extensions exist, with the evident restriction or precomposition steps understood when needed for typing. More generally, UDL denotes any decision semantics obtained by composing left and right Kan extensions along problem-specific inclusions of local context into global context.

Can Causality be inferred from Language?

The Washington Post
Democracy Dies in Darkness

Well+Being | Body | Food | Fitness | Mind | Life | Newsletter | Ask a Doctor

New study sheds light on a beneficial compound found in coffee and chocolate

Theobromine, an alkaloid found in dark chocolate and coffee, is associated with slower cellular aging, a study suggests.

December 26, 2025

7 min | Summary | 100 | Make us preferred on Google

LLM Overview

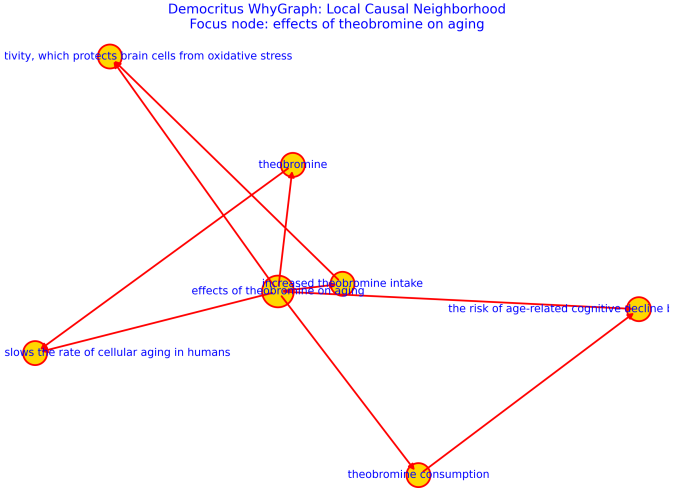
LLM-based AI Summary

AI Overview Summary is generated by AI. Please verify accuracy by reading the full article.
A study in Aging found higher blood levels of theobromine, found in dark chocolate and coffee, linked to slower cellular aging. The study found an association, not a causal link, and didn't specify consumption amounts. Researchers noted theobromine's potential impact on gene activity and aging. The study's limitations include lack of dietary data and single-time-point analysis. Read the full article for more on: The role of epigenetic clocks in estimating biological age. Potential synergistic effects of theobromine with other chocolate components. Limitations of the study and what they mean for future research.

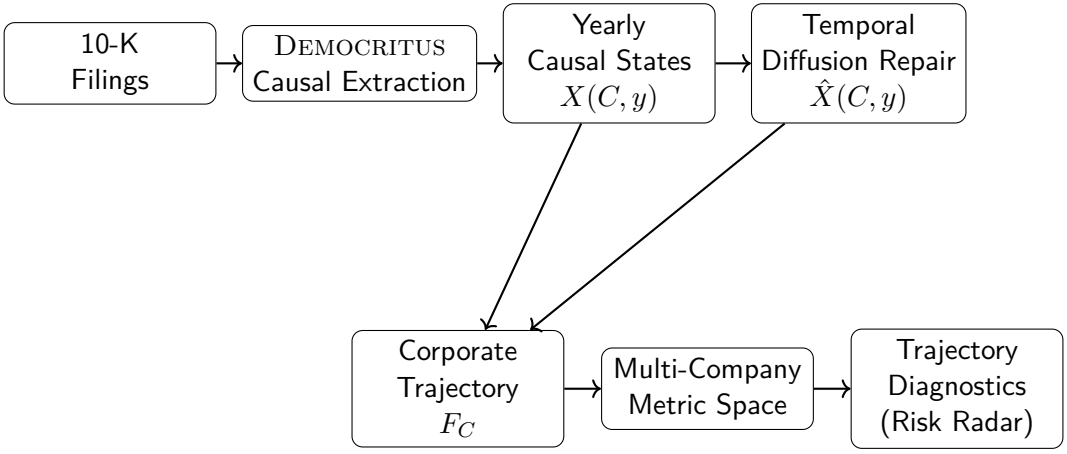
Deep Research constructed by DEMOCRITUS

- anti-inflammatory foods recommendations
- association versus causation in aging studies
- balancing sugar intake with chocolate consumption
- cancer risk reduction strategies
- cocoa content and nutritional value
- coffee as a source of theobromine
- cognitive biases and positivity
- dark chocolate ingredient criteria
- dark chocolate's health effects
- dietary impacts on health
- dietary recommendations for chocolate consumption
- DNA methylation as gene expression regulator
- doctor's wellness advice
- effects of theobromine on aging
- epigenetic clocks and aging measurement

DEMOCRITUS: Local Causal Model using Diagrammatic Backpropagation



Diffusion Models of Companies using Kan Extension Transformers



Multi-Company Panel Comparison

Company	Latest Year	Latest Risk	Peak Year	Peak Risk
Goldman Sachs	2025	87.7	2025	87.7
Boeing	2026	81.7	2026	81.7
IBM	2026	78.3	2026	78.3
Walmart	2025	75.0	2025	75.0
Cisco	2025	74.9	2025	74.9
Home Depot	2025	68.5	2025	68.5
Visa	2025	67.6	2025	67.6
JPMorgan	2026	66.7	2026	66.7
Amgen	2025	66.2	2024	68.7
Coca-Cola	2025	62.5	2024	82.3

SkillOpt: Agentic Skill Optimization is an instance of UDL

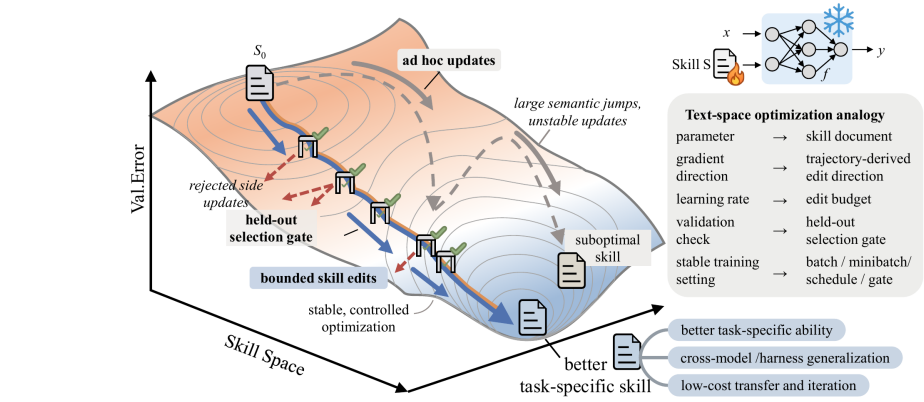


Figure source: Yang et al., SkillOpt: Executive Strategy for Self-Evolving Agent Skills, `https://arxiv.org/abs/2605.23904`, 2026.

Infinitesimal Causality: From DAGs to Tangent Categories

A tangent category is a category $\mathcal{X}$ equipped with a tangent bundle functor $T : \mathcal{X} \to \mathcal{X}$ together with natural transformations

$$\pi : T \Rightarrow \mathrm{Id}, \qquad 0 : \mathrm{Id} \Rightarrow T, \qquad + : T \times_{\mathcal{X}} T \Rightarrow T, \qquad \ell : T \Rightarrow T^2,$$

and the canonical flip, satisfying the Cockett-Crutwell tangent-category axioms

The category $\mathbf{Smooth}$ of finite-dimensional smooth manifolds with the ordinary tangent bundle is the prototypical example.

The category $\mathrm{Stat}_\infty$is a Frobenius Markov category of finite-dimensional statistical models presented by sufficient statistics. Its infinitesimal structure is induced from the tangent category of smooth parameter manifolds.

Reference: *Infinitesimal Causality*, arXiv:2606.24621 (2026).

Lie-Bracket Optimization: A New Framework

Broader IC claim

Lie brackets are a reusable diagnostic for noncommuting interventions across causal discovery, adapter learning, and agent workflows.

Agentic Skill Optimization claim

Skill optimization is not just prompt search. It is controlled optimization over structured artifacts with hidden state and curvature.

$$\text{BRIDGE/SKFM} \longrightarrow \textsc{Allora} \longrightarrow \text{LASKO.}$$

The same IC principle now has a clean agentic optimization use case: use brackets to decide which expensive validations are worth buying.

UDL, KET, and IC meet in tangent skills

- KET supplies the dual channels: extension and observation.
- IC differentiates those channels inside a tangent category.
- UDL uses the differentiated duality to learn admissible decisions.
- LASKO combines IC+KET+UDL

One-line synthesis

$$\text{SkillOpt} \subset \mathrm{LASKO} = \mathsf{T}(\text{KET duality})$$

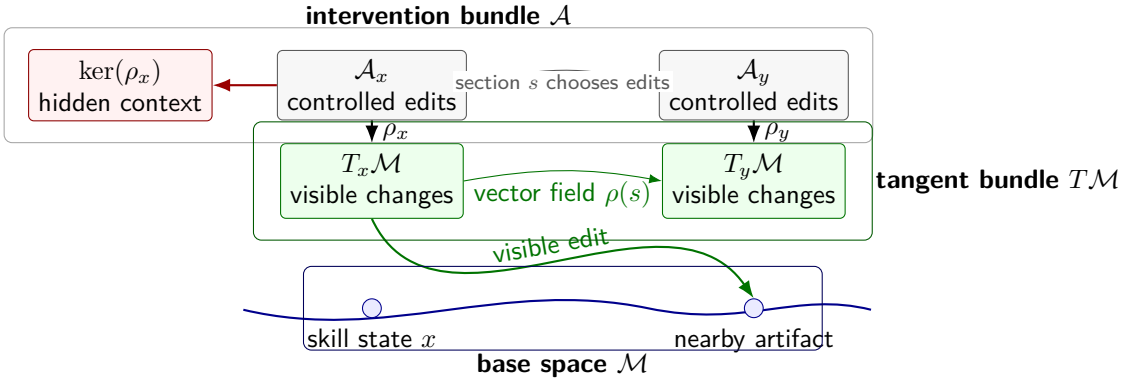
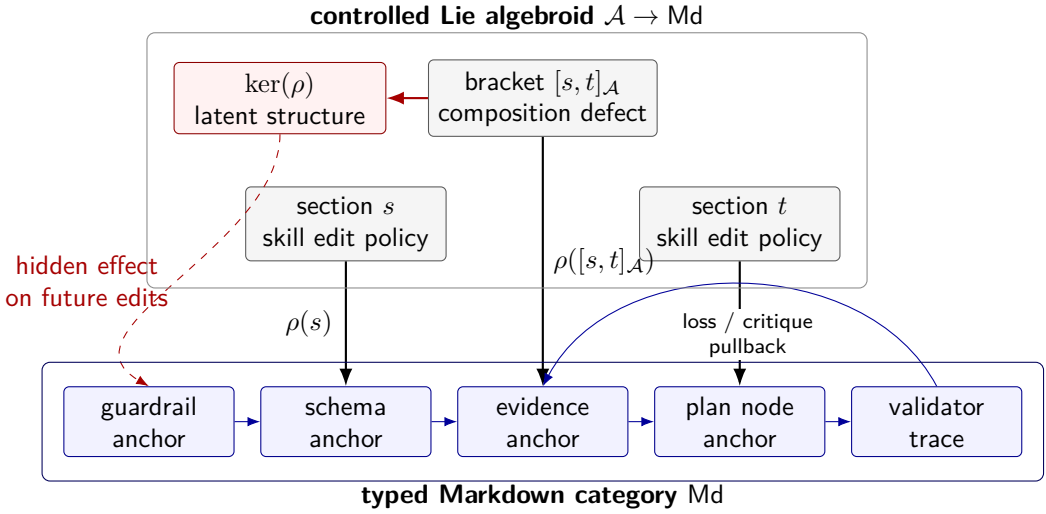


Figure: The base space $\mathcal{M}$ contains skill states. The tangent bundle $T\mathcal{M}$ contains visible first-order artifact changes. The intervention bundle $\mathcal{A}$ contains controlled edit modes, and the anchor $\rho : \mathcal{A} \to T\mathcal{M}$ maps each controlled edit to its visible effect. The kernel $\ker(\rho)$ records controlled context that is not immediately visible but can affect later composition.

LASKO: Skill Optimization using Lie Algebroids (arXiv paper forthcoming)



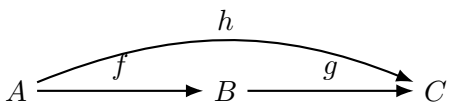
LASKO Results Summary

- In controlled benchmarks, LASKO achieves perfect scores (1.000) while using 90-95% fewer validation probes than exhaustive ordered-pair search:
  1. Ten-anchor chain: 11 probes vs. 91 exhaustive
  2. 160-edit scale: 168 probes vs. 25,441 exhaustive
  3. Agent-workflow contracts: 17 probes vs. 133 exhaustive
  4. 10-K BASKET/ROCKET: 29 probes vs. 183 exhaustive
- The key operational insight is that high Lie bracket residuals predict which edit pairs exhibit order-sensitive interactions. By screening on bracket structure before spending expensive rollout calls, LASKO recovers the full repair sequence that greedy single-edit search misses (which scored 0.000 in the ten-anchor benchmark), without paying the quadratic cost of brute-force enumeration.
- The kernel proxy—measuring hidden procedural state when visible artifact diffs are near-zero—also shows predictive power (AUC 0.8) for identifying unstable edits in the Odyssey diagnostic traces.

Categories: Objects and Morphisms

A **category** consists of:

- objects $A, B, C, \ldots$
- morphisms $f : A \to B$
- composition of morphisms
- identity morphisms $\mathrm{id}_A : A \to A$



with the consistency rule

$$h = g \circ f.$$

In this tutorial: objects can be tokens, or even Transformer models, graph states, causal snapshots, or structured contexts; morphisms encode relations between them.

Functors: Maps Between Categories

A **functor**

$$F : \mathcal{C} \to \mathcal{D}$$

maps

- each object A in $\mathcal{C}$ to an object $F(A)$ in $\mathcal{D}$
- each morphism $f : A \to B$ to a morphism $F(f) : F(A) \to F(B)$

and preserves structure:

$$F(\mathrm{id}_A) = \mathrm{id}_{F(A)}, \qquad F(g \circ f) = F(g) \circ F(f).$$

Interpretation for ML

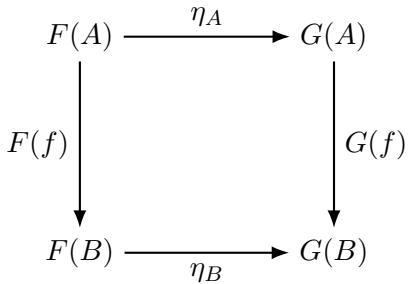
- a functor transports relational structure from one domain to another
- examples: token neighborhoods $\to$ contextual positions; yearly causal graphs $\to$ latent states

Natural Transformations: How Functors Interact

Given two functors $F, G : \mathcal{C} \to \mathcal{D}$, a **natural transformation**

$$\eta : F \Rightarrow G$$

assigns a map $\eta_A : F(A) \to G(A)$ for each object A, such that the following square commutes:

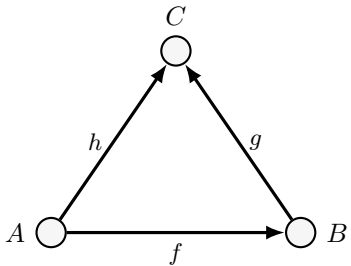


- a natural transformation transports one functorial description to another

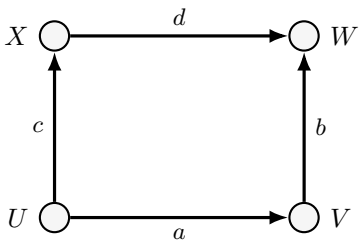
Diagrams

Definition

A *diagram* $F : \mathcal{J} \to \mathcal{C}$ is a functor F from some finite category $\mathcal{J}$ into a category of interest, $\mathcal{C}$.



Triangle $A \xrightarrow{f} B \xrightarrow{g} C$, $A \xrightarrow{h} C$



Square $U \to V \to W$ vs $U \to X \to W$

Example of a Limit: Audio Video Synchronization

Consider a recording of this talk:

- **Video** observations (frames) V
- **Audio** observations (chunks) A

They both map to a shared base object:

$$T = \{0, 1, \ldots, T-1\} \quad \text{(timestamps)}$$

Goal: glue audio and video into an aligned time series by taking the pullback

$$P = V \times_T A.$$

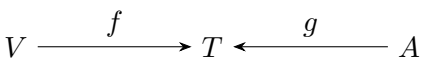
Key idea: "alignment" is a pullback.

Example of a Limit: Pullback of a Span $V \to T \leftarrow A$

We learn (or assume) two maps:

$$f : V \to T, \qquad g : A \to T,$$

where f predicts timestamp from video, and g predicts timestamp from audio.



- A pullback is an object P with maps $\pi_V : P \to V$ and $\pi_A : P \to A$
- Universal property: any other mapping $h : z \to V$ and $i : z \to A$ must be factorizable as $h = \pi_v \circ \gamma$ and $i = \pi_a \circ \gamma$

Left and Right Kan Extensions: A Universal Solution to ML

How to extend a given functor along another functor:

$$F : \mathcal{A} \to \mathcal{B}, \qquad H : \mathcal{A} \to \mathcal{C}.$$

A **left Kan extension** of H along F is a functor

$$\mathrm{Lan}_F H : \mathcal{B} \to \mathcal{C}$$

together with a universal natural transformation

$$H \Rightarrow (\mathrm{Lan}_F H) \circ F.$$

A **right Kan extension** of H along F is a functor

$$\mathrm{Ran}_F H : \mathcal{B} \to \mathcal{C}$$

together with a universal natural transformation

$$(\mathrm{Ran}_F H) \circ F \Rightarrow H.$$

Kan Extension of the Span $A \to C \leftarrow B$

Given

- A functor diagram: $H : \mathcal{J} \to \mathcal{C}$
- $\mathcal{J}$ is the span $\bullet \to \bullet \leftarrow \bullet$
- $\mathcal{C}$ is a concrete category of A/V media objects and arrows are functions mapping an A/V object to a time stamp

$$A \xrightarrow{\quad f \quad} C \xleftarrow{\quad g \quad} B$$

- Another functor $F : \mathcal{J} \to \mathcal{B}$

The **left Kan extension** of H along F is a functor

$$\mathrm{Lan}_F H : \mathcal{B} \to \mathcal{C}$$

together with a universal natural transformation

$$H \Rightarrow (\mathrm{Lan}_F H) \circ F.$$

Universal Foundry Construction: Left Kan Extension

- A foundry candidate artifact appears as a presheaf

$$X : \mathcal{C}_x^{op} \longrightarrow \text{Data},$$

and the maintained target foundry state appears as a presheaf

$$M_Y : \mathcal{C}_Y^{op} \longrightarrow \text{Data}.$$

The notation $\mathrm{Lan}_{F_{x,Y}}$ in the cards is shorthand for the left Kan extension of X along the opposite functor,

$$\mathrm{Lan}_{F_{x,Y}} X := \mathrm{Lan}_{F_{x,Y}^{op}} X : \mathcal{C}_Y^{op} \longrightarrow \text{Data}.$$

Universal Foundry Construction: Right Kan Extension

TICKET uses $\mathrm{Lan}_{F_{x,Y}}$ as its admission direction: it freely assembles the least target-side candidate compatible with the source charts. A stronger audit can then apply the right Kan direction

$$\mathrm{Ran}_{F_{x,Y}} F_{x,Y}^{*} A, \qquad A = \mathrm{Lan}_{F_{x,Y}} X,$$

as a target-side consistency envelope for the admitted candidate. Pointwise, this is a finite limit over target probes back through the same declared interface:

$$(\mathrm{Ran}_{F_{x,Y}} F_{x,Y}^{*} A)(U) \cong \lim_{(U \to F_{x,Y}(V))} A(F_{x,Y}(V)).$$

The comparison

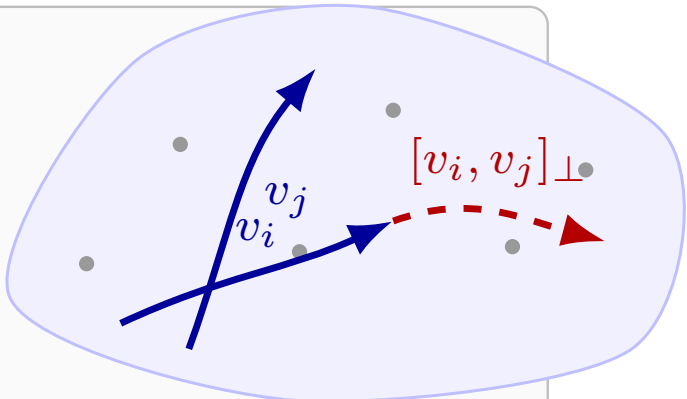
$$A \longrightarrow \mathrm{Ran}_{F_{x,Y}} F_{x,Y}^{*} A$$

is therefore a round-trip check: after the source artifact has been extended into the target foundry, it asks whether all target-side obligations that can be observed through the source interface are jointly satisfiable.

Generalizing Chain Rule: Tangent Categories

- **Cockett-Cruttwell** (2014) defined tangent categories, generalizing differential geometry.
- **Smooth Manifolds** : The canonical example, where $\mathcal{X}$ is the category of smooth manifolds, T is the standard tangent bundle, and the additive structure arises from standard vector addition in tangent spaces.
- **Algebraic Geometry**: The category of affine schemes (equivalent to the opposite category of commutative rings) forms a tangent category where the tangent bundle is the Zariski tangent bundle.
- **Synthetic Differential Geometry** (SDG): The microlinear objects in any model of SDG form a tangent category, driven by nilpotent infinitesimals.
- **Computer Science**: Cartesian differential categories, which formalize the differential calculus used in the semantics of programming languages, intrinsically possess tangent structure.

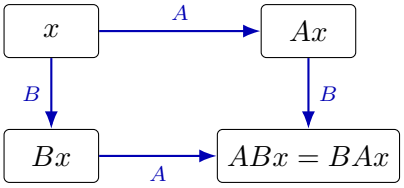
Lie-Brackets



(https://arxiv.org/abs/2606.24621

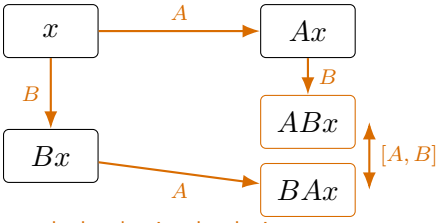
Lie Brackets: Why noncommutativity matters

Commuting interventions



order does not reveal new structure

Noncommuting interventions



the bracket is a local witness of hidden coupling or curvature

IC brackets identify causal interaction and order-sensitive interventions

BRIDGE/SKFM residual brackets expose latent confounding without full enumeration

Allora adapter commutators become a trainable penalty for compositionality

SkillOpt high-bracket edit pairs are the first candidates worth validating

Agentic Optimization as IC in Markdown Tangent Categories

IC / tangent category	SkillOpt instance	BASKET instance	ROCKET instance
Object M	Markdown skill	extracted plan graph	plan artifact
Tangent vector δ	local skill edit	candidate plan refinement	plan repair
Vector field X	edit proposal rule	extractor / plan constructor	repair policy
Observation functor $\mathcal{O}$	rollout + validation	evidence alignment + plan score	evidence + reward
Lie bracket $[X, Y]$	order effect of edits	interaction among plan refinements	repair interaction
Obstruction	transfer failure or gate rejection	incoherent or unsupported step	unsupported plan
Gluing	accepted skill update	coherent extracted workflow	admitted optimized workflow

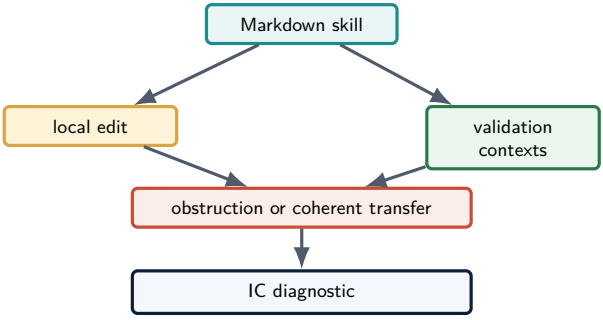
LASKO as IC in a Tangent Markdown Category

Shared pattern

Many learning systems are understood by asking what small local changes do across contexts.

- SkillOpt: perturb a Markdown skill and its decision behavior.
- The edit is local; its consequences are measured across validation cases.
- IC gives us language for coherence, obstruction, and admissible update.

Bridge. SkillOpt is IC on a different object: the local decision geometry of a Markdown skill.



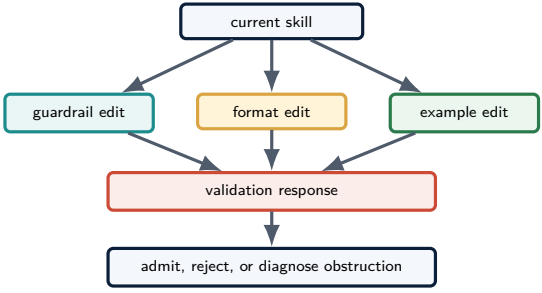
Tangent Skills Without the Formalism

Working intuition

A tangent direction is just a small, meaningful way to move from the current object.

- Causal model: edit an edge, mechanism, or conditioning context.
- Neural model: move along a gradient direction.
- Markdown skill: edit an instruction, guardrail, contract, or example.
- IC asks whether the move is coherent across contexts.

Why it matters. SkillOpt can use the IC vocabulary: local variation, obstruction, gluing, and admissible update.



SkillOpt and TangentSkill

SkillOpt lives at the document/program level: skills are Markdown objects with instructions, examples, constraints, tools, and evaluation traces.

TangentSkill

An infinitesimal edit direction on a skill object: how behavior changes when we perturb examples, rubrics, protocols, or constraints.

Allora

A neural realization of those directions as low-rank adapter tangent fields whose brackets can be measured and regularized.

$$\tau_{\text{help}}, \tau_{\text{safety}}, \tau_{\text{tool}} \quad \mapsto \quad \Delta_{\text{help}}, \Delta_{\text{safety}}, \Delta_{\text{tool}}, \qquad [\Delta_i, \Delta_j] \approx 0.$$

SkillOpt learns or selects useful skill directions; ALLORA makes their compiled adapter realizations composable.

Fine Tuning LLMs with A Lie-Algebraic LoRA (ALLORA)

- A deployed model may need one adapter for domain expertise, one for safety behavior, one for response style, and others for tools, users, or regulatory boundaries.
- If two adapters induce residual updates Δ_A and Δ_B, the discrepancy between applying them in opposite orders is governed, to first order, by their commutator

$$[\Delta_A, \Delta_B] = \Delta_A \Delta_B - \Delta_B \Delta_A.$$

[Mahadevan, ALLORA: A Lie-Algebraic LoRA Method for Composable Neural Adapters, arXiv, 2026 (to appear)]

ALLORA Design

Consider a frozen sequence model with hidden state $h^\ell \in \mathbb{R}^{T\times d}$ at layer ℓ. A standard residual-stream low-rank adapter applies

$$h^\ell \mapsto h^\ell + h^\ell(\Delta_i^\ell)^\top, \qquad \Delta_i^\ell = B_i^\ell A_i^\ell,$$

where $B_i^\ell \in \mathbb{R}^{d\times r}$, $A_i^\ell \in \mathbb{R}^{r\times d}$, and $r \ll d$. We interpret adapter i as a learned tangent field, represented layerwise by $\{\Delta_i^\ell\}_\ell$.
Given two adapters i, j, their layerwise commutator is

$$[\Delta_i^\ell, \Delta_j^\ell] = \Delta_i^\ell \Delta_j^\ell - \Delta_j^\ell \Delta_i^\ell.$$

If this commutator is small across layers, then applying i followed by j should be closer to applying j followed by i, at least in the residual-stream approximation.

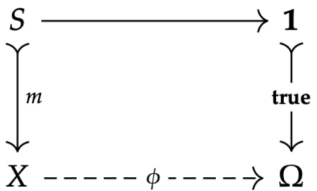
ALLORA objective

Let L_i denote the task loss for adapter i. ALLORA trains a collection of adapters by minimizing

$$L_{\text{ALLORA}} = \sum_i L_i + \lambda \sum_\ell \sum_{i<j} \left\| [\Delta_i^\ell, \Delta_j^\ell] \right\|_{\text{F}}^2 .$$

Here λ controls the tradeoff between task specialization and commutation. The penalty is differentiable and inexpensive for low-rank adapters because Δ_i^ℓ is small compared with the full model parameter space. It also avoids the need to estimate full Hessian traces or materialize curvature matrices over the entire model.

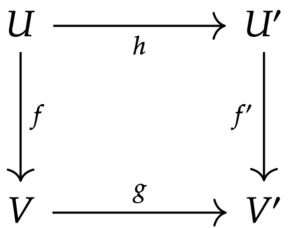
Subobject Classifiers



- A **subobject classifier** is a $\mathcal{C}$-object Ω, and a $\mathcal{C}$-arrow $\mathbf{true} : \mathbf{1} \to \Omega$, such that to every monic arrow $S \hookrightarrow X$ in $\mathcal{C}$, there is a unique arrow ϕ that forms the above pullback square.
- Example: Given an LLM, we can define its subobjects in various ways.

What is a Topos?

- A **topos** is a category that is *Cartesian closed*, has a *terminal object*, and has a *subobject classifer*.
- A structural causal model $\mathcal{M}$ (SCM) defines a unique function $F : U \to V$ from *exogenous* variables into *endogenous variables*
- The category of LLMs forms a topos, where each object is an LLM, and arrows are given by *square commutative diagrams*:

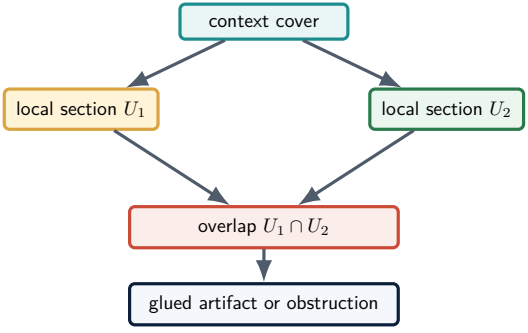


Sheaves: Local Evidence That Can Be Glued

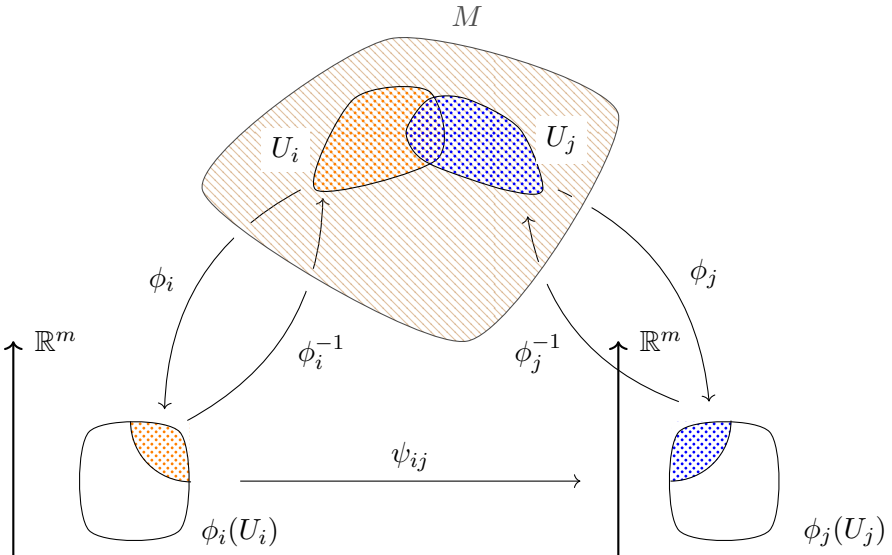
Plain-language view

A sheaf is a way to attach local information to overlapping contexts, then ask whether the local pieces agree where they overlap.

- **Sections**: local claims, traces, skills, or evidence states.
- **Restrictions**: what a local object says on a smaller context.
- **Gluing**: compatible local pieces form a larger object.
- **Obstruction**: overlaps disagree or lack evidence.



Sheaves on a Topos

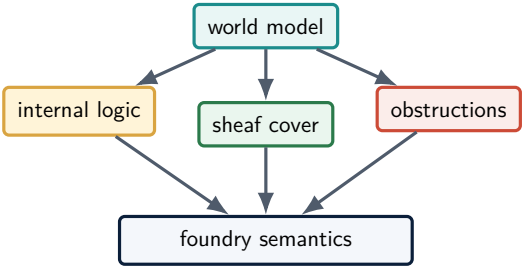


Topos World Models

Why topos theory enters

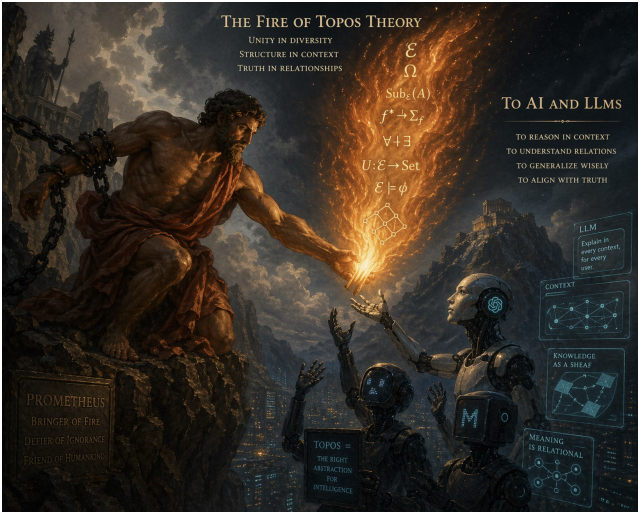
A topos is a mathematical universe for reasoning with local truth, changing context, and typed structure.

- It lets truth be contextual rather than globally flattened.
- It gives a home for sheaves, internal logic, and typed evidence.
- A topos world model: contexts, restrictions, admissible gluing, and obstruction signals.



Prometheus topos world-model view: local domains, internal logic, and admissible artifact construction.

PROMETHEUS: Bringing the Fire of Topos Theory to Foundries



Prometheus Store Front

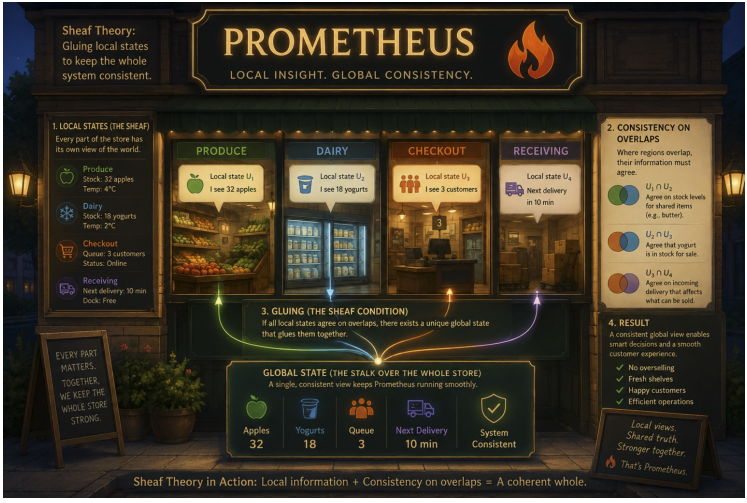
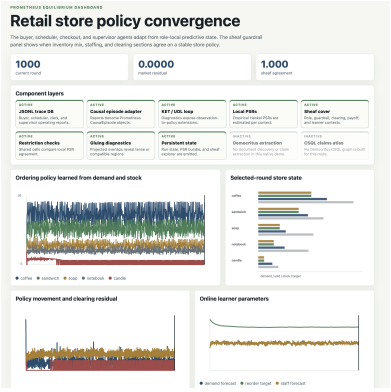
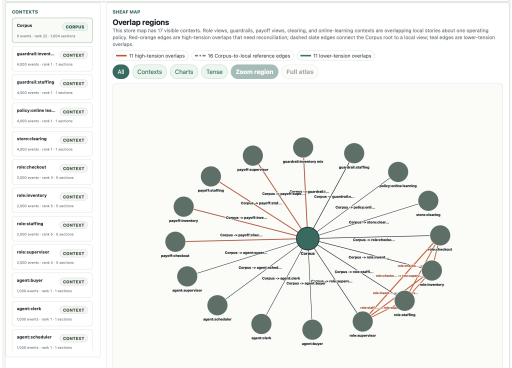


Figure: Sheaves to manage store inventory

Prometheus Market: Using Sheaf Theory to Manage Store Inventory





Predictive State Foundry Models

- An ODYSSEY artifact presents a finite family of local predictive-state sections over a declared cover. Each local section stores histories, tests or predictive motifs, prediction/support values, provenance, diagnostics, restriction maps, gluing results, and obstruction records.
- For each context U, Prometheus emits a local section of the schematic form

$$\mathcal{P}(U) = (H_U, T_U, M_U, S_U, \Pi_U, D_U),$$

where H_U and T_U are the finite histories and tests visible in that context, M_U is the predictive table or support score, S_U records support, Π_U records provenance, and D_U records diagnostics such as sparsity, uncertainty, extraction confidence, and local mismatch.

Predictive State Foundry Certification

Definition (Foundry health)

Let $\mathcal{U} = \{U_i \to U\}$ be a foundry cover and let each local predictive state section s_i expose a finite matrix of truth-weighted predictive cells $M_i[h, \tau] \in [0, 1]$. Fix a promotion threshold θ, corresponding in the implementation to the truth label $\mathrm{PLAUSIBLE}$. The cell-level hidden-state-capture ratio is

$$H_{\mathrm{cell}}(\mathcal{U}) = \frac{\sum_i |\{(h, \tau) : M_i[h, \tau] \geq \theta\}|}{\sum_i |\mathrm{dom}(M_i)|}.$$

The context-level capture ratio is

$$H_{\mathrm{ctx}}(\mathcal{U}) = \frac{|\{i : H_{\mathrm{cell}}(U_i) \geq 0.6\}|}{|\mathcal{U}|}.$$

Foundry Guarantees

Theorem (Consistent promotion)

Fix a foundry cover $\mathcal{U}$, finite local sections s_i, restriction maps ρ_{ij}, support weights, and tolerance ϵ. If every overlap has support above threshold and

$$\|\rho_{ij}(s_i) - \rho_{ji}(s_j)\| \leq \epsilon$$

on every shared predictive cell, then ODYSSEY*'s support-weighted aggregation constructs a promoted section s whose restrictions agree with every local section up to ϵ on supported overlaps.*

Foundry Promotion Obstruction

Proposition (Obstruction soundness)

If an overlap fails compatibility, lacks a restriction map, or lacks sufficient support, ODYSSEY *cannot silently promote the corresponding global claim through TICKET. The failed overlap must appear as a gluing diagnostic, obstruction ledger entry, quarantine reason, or blocked promotion gate.*

TICKET Certification

- A candidate artifact x determines a small source context category $\mathcal{C}_x$, whose objects are source-side charts such as files, runs, tables, benchmark slices, dashboards, model cards, extracted claim sets, or process traces.
- The target foundry Y determines a target site $\mathcal{C}_Y$, whose objects are the local contexts that Athena has declared for that foundry.
- A TICKET card declares a functor

$$F_{x,Y} : \mathcal{C}_x \longrightarrow \mathcal{C}_Y$$

that sends each source chart to the target context in which it is allowed to be interpreted. For example, a 10-K source chart may be sent to a company-year-financial context, a review shard to a product-use context, or a repair step trace to a procedural state/action context.

Foundry Evaluation

Axis	Question
Artifact completeness	Are Scylla, Homer, Athena, Prometheus, Toulmin, and ingestion artifacts present and mutually referential?
Cover quality	Do local sections match meaningful domain contexts rather than arbitrary chunks?
Restriction validity	Are overlap checks tied to shared identifiers, provenance, or typed translation maps?
Gluing usefulness	Do gluing failures correspond to actionable missing evidence, regime mismatch, or unsupported transport?
Promotion discipline	Are candidate states promoted, quarantined, or blocked by explicit gates rather than hidden confidence thresholds?
Task improvement	Does the foundry representation improve an admitted task such as retrieval, triage, explanation, or decision support?
Argument discipline	Does a local neural reasoner preserve the grounded claim, cite the right grounds, and keep source-coherence warnings visible in the warrant, qualifier, or rebuttal?

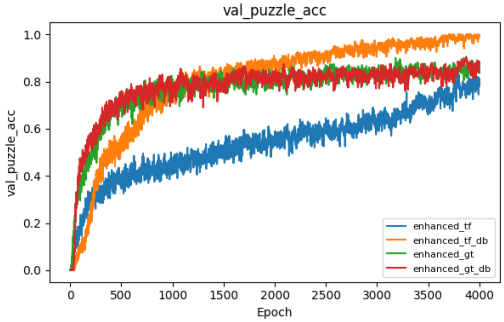
Diagrammatic Backpropagation: Minimizing Curvature Energy

Require: Triangle $\mathcal{T}$ and square constraints $\mathcal{S}$, node states x, maps M, relations r
Ensure: $E_\theta(\mathcal{D}) = E_\triangle + E_\square$

$E_\triangle \leftarrow 0,\ E_\square \leftarrow 0$
for each triangle $(A \xrightarrow{f} B \xrightarrow{g} C,\ A \xrightarrow{h} C) \in \mathcal{T}$ **do**
    $z_h \leftarrow \mathrm{ET}_\theta(M_h, r_h, x_A)$
    $z_{gf} \leftarrow \mathrm{ET}_\theta(M_g, r_g,\ \mathrm{ET}_\theta(M_f, r_f, x_A))$
    $E_\triangle \leftarrow E_\triangle + \|z_h - z_{gf}\|_2^2$
end for
for each square $(U \xrightarrow{a} V \xrightarrow{b} W,\ U \xrightarrow{c} X \xrightarrow{d} W) \in \mathcal{S}$ **do**
    $z_{ba} \leftarrow \mathrm{ET}_\theta(M_b, r_b,\ \mathrm{ET}_\theta(M_a, r_a, x_U))$
    $z_{dc} \leftarrow \mathrm{ET}_\theta(M_d, r_d,\ \mathrm{ET}_\theta(M_c, r_c, x_U))$
    $E_\square \leftarrow E_\square + \|z_{ba} - z_{dc}\|_2^2$
end for
return $E_\triangle + E_\square$

Sudoku Results

- 4×4 Sudoku problem
- Constraints: each digit occurs only once along each row, column, and 2×2 square
- Diagrammatic backpropagation can explicitly enforce this constraint
- Results show 8x speedup over baseline Transformer.



Results from Yuting Zhang, CMPSCI 692CT Course Project: GitHub tutorial/code repo.

Left Kan Intuition: Assembly from the Prefix

Suppose we see the beginning of a plan:

$$\texttt{search} \to ?$$

From prior data, `search` often continues as

$$\texttt{search} \to \texttt{retrieve}, \qquad \texttt{search} \to \texttt{summarize}, \qquad \texttt{search} \to \texttt{filter}.$$

Left Kan message

Given the observed prefix, what larger structure can be assembled from familiar continuations?

$$\mathrm{Lan}_i F \quad \rightsquigarrow \quad \{\texttt{retrieve}, \texttt{summarize}, \texttt{filter}, \dots\}$$

Right Kan Intuition: Completion from the Goal

Now suppose the plan must end as

$$? \to \texttt{write}.$$

From prior data, the skill `write` is usually preceded by

$$\texttt{summarize} \to \texttt{write}, \qquad \texttt{verify} \to \texttt{write}, \qquad \texttt{draft} \to \texttt{write}.$$

Right Kan message

What missing step is required so the structure remains compatible with its downstream goal?

$$\mathrm{Ran}_i F \quad \rightsquigarrow \quad \{\texttt{summarize}, \texttt{verify}, \texttt{draft}, \dots\}$$

Combining Left and Right Kan

Lan candidates from the prefix

$$\{\texttt{retrieve}, \texttt{summarize}, \texttt{filter}\}$$

Ran candidates from the suffix

$$\{\texttt{summarize}, \texttt{verify}, \texttt{draft}\}$$

Intersection of the two signals

$$\{\texttt{summarize}\}$$

$$\texttt{search} \to \texttt{summarize} \to \texttt{write}$$

This is the template we will reuse for causal repair and workflow repair.

Kan Extensions

For a structured object $p \in \mathcal{P}$,

$$(\mathrm{Lan}_i F)(p) = \operatorname*{colim}_{(i \downarrow p)} F, \qquad (\mathrm{Ran}_i F)(p) = \lim_{(p \downarrow i)} F.$$

- $(i \downarrow p)$: all observed fragments that can map into p
- $(p \downarrow i)$: all downstream ways p must remain compatible with observed structure

Speaker translation

Left Kan asks "what can build this?"
Right Kan asks "what must this satisfy?"

From Kan Intuition to Architecture

At each node or token v, a Kan Extension Transformer builds three signals:

$$z_v^{\mathrm{local}}, \qquad z_v^{\mathrm{Lan}}, \qquad z_v^{\mathrm{Ran}}.$$

$$h_v = \mathrm{Fuse}\big(z_v^{\mathrm{local}}, z_v^{\mathrm{Lan}}, z_v^{\mathrm{Ran}}\big)$$

- **Local**: what this node currently is
- **Lan**: what it should inherit from predecessors and structured neighbors
- **Ran**: what it must satisfy to support successors and goals

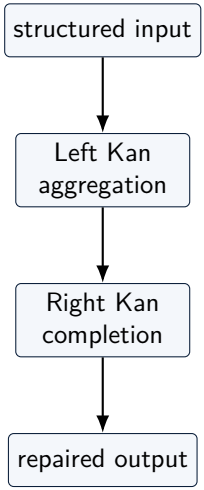
Kan Extension Transformers as Left-Kan Attention Plus Right-Kan Repair

Left-Kan branch

- aggregate structured context
- build a richer latent representation
- attention-like behavior on relational neighborhoods

Right-Kan branch

- reconstruct missing or noisy structure
- enforce compatibility with later constraints
- diffusion-like denoising and repair



Why This Unifies Attention and Diffusion

View	Standard ML language	KET language
Attention	context aggregation	left Kan extension
Diffusion / denoising	iterative repair	right Kan extension
Structured prediction	consistency under constraints	Lan + Ran fusion

Main conceptual payoff

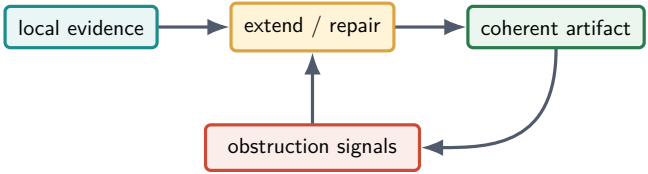
The tutorial will show how universal extension operators help build trustworthy foundation models

The Common Pattern

- Foundation-model systems rarely produce one isolated object.
- They produce local representations, claims, skills, plans, traces, and evidence surfaces.
- The trust problem is whether these local pieces still cohere when moved across contexts.

Tutorial lens

Attention, diffusion, training, and foundry admission are all treated as structured extension problems.



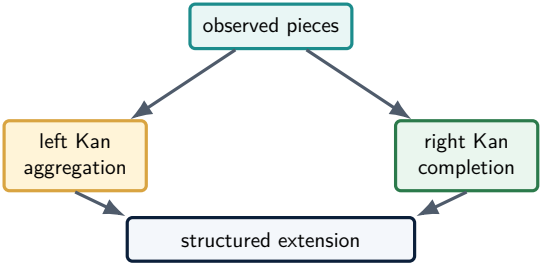
The machinery tracks what is gathered, what is completed, and what fails to fit.

Kan Extensions: The Extension Problem

Basic question

Given information on one part of a structured domain, what is the principled way to extend it to another domain?

- **Left Kan**: aggregate local evidence into a coarser or broader representation.
- **Right Kan**: complete or repair a partially specified structure so constraints are satisfied.
- Weighted, learned versions become practical operators in neural systems.



KET: Attention and Diffusion as Dual Transport

Attention side

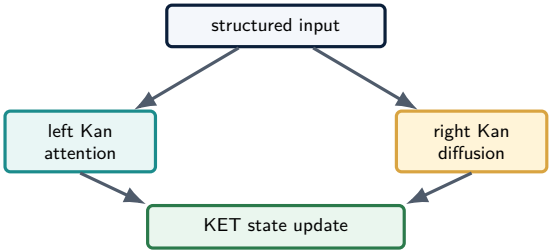
Contextual aggregation behaves like a learned left Kan extension: gather relevant neighbors into a useful representation.

Diffusion side

Denoising and generation behave like learned right Kan completion: repair a partial state under compatibility constraints.

KET side

A Kan Extension Transformer couples both flows on a structured neighborhood.



KET is the architectural expression of the left/right Kan pattern.

KET for Language Modeling

Usual transformer view

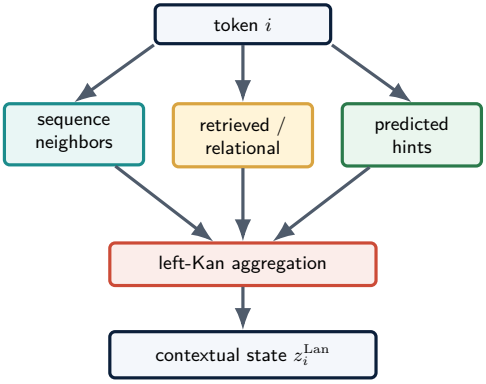
For token i, attention forms a contextual vector from previous tokens:

$$z_i = \sum_{j \le i} \alpha_{ij} W_V h_j$$

$$\alpha_{ij} = \mathrm{softmax}_j\left(q_i^\top k_j / \sqrt{d}\right).$$

KET reading

Replace $j \le i$ by a structured neighborhood $\mathcal{N}(i)$. The weighted aggregation is a learned left-Kan approximation.



Attention is not discarded; it becomes one computable case of Kan-style aggregation.

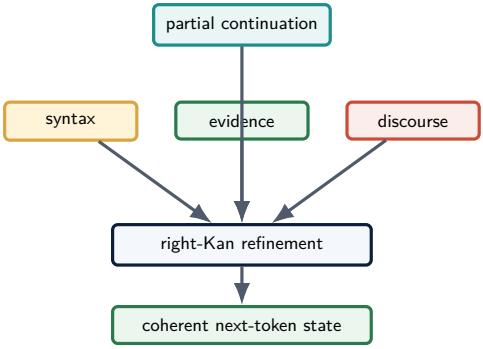
Right-Kan Refinement in Language Modeling

Why the right branch matters

Language modeling is not only gathering context. The model also has to make a partially specified continuation coherent with grammar, discourse, retrieved evidence, and latent constraints.

$$u_i^{(t+1)} = \Phi_\theta\left(u_i^{(t)}, \{u_j^{(t)} : j \in \mathcal{N}(i)\}, c_i\right)$$

- $u_i^{(t)}$: current prediction or denoised state
- c_i: compatibility constraints
- right-Kan view: complete the state that all constraints can see



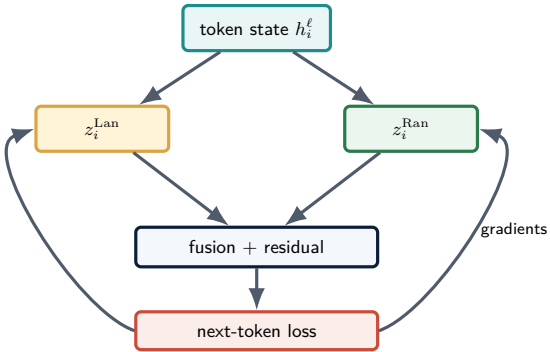
Diagrammatic Backprop for a KET Language Model

Forward pass

$$h_i^{\ell+1} = \mathrm{FFN}\big(\mathrm{Norm}\left(h_i^{\ell} + f_i\right)\big)$$
$$f_i = \lambda_L z_i^{\mathrm{Lan}} + \lambda_R z_i^{\mathrm{Ran}}$$
$$\ell_i = -\log p_\theta(x_{i+1} \mid \mathrm{context}_i).$$

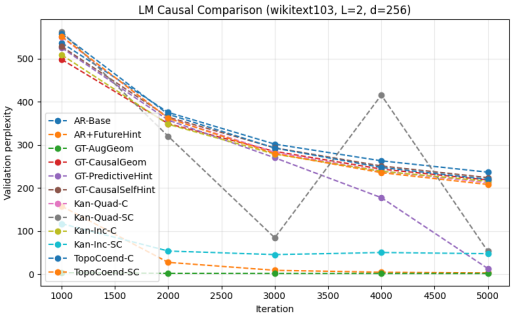
Backward pass

DB is ordinary gradient descent plus a diagrammatic account of where gradients travel: through aggregation, refinement, fusion, and compatibility maps.



KET in Language Modeling

- Token contexts are structured local diagrams.
- Attention is a singleton-neighborhood left Kan special case.
- KET adds higher-order neighborhoods and predict-detach self-conditioning.
- Comparison of 12 Transformer implementations.



Validation perplexity curves from the KET language-modeling experiment. https://arxiv.org/abs/2605.27259

LASKO as KET on Tangent Categories

In UDL terms, a skill update has two complementary flows.

- **Colimit-like flow:** aggregate examples, demonstrations, local heuristics, and capability evidence into usable behavior.
- **Limit-like flow:** restrict that behavior by policies, invariants, rubrics, safety rules, and consistency checks.

$$C_{ij} = [\Delta_i^{\mathrm{colim}}, \Delta_j^{\mathrm{lim}}] + [\Delta_i^{\mathrm{lim}}, \Delta_j^{\mathrm{colim}}].$$

Tutorial punchline

LICKET asks whether optimizing a capability skill and then applying a safety/rubric skill matches applying the constraint structure first and then optimizing capability.

ALLORA as KET on Tangent Categories

The KET setting also suggests a more architectural version of ALLORA: Lie-algebraic infinitesimal-causal KET
Adapter updates can therefore be decomposed into horizontal components that modify extension or aggregation maps and vertical components that modify restriction or constraint-propagation maps:

$$\Delta_i = \Delta_i^{\mathrm{colim}} + \Delta_i^{\mathrm{lim}}.$$

Under this decomposition, pairwise brackets split into colimit–colimit, limit–limit, and cross terms,

$$[\Delta_i, \Delta_j] = [\Delta_i^{\mathrm{colim}}, \Delta_j^{\mathrm{colim}}] + [\Delta_i^{\mathrm{lim}}, \Delta_j^{\mathrm{lim}}] + [\Delta_i^{\mathrm{colim}}, \Delta_j^{\mathrm{lim}}] + [\Delta_i^{\mathrm{lim}}, \Delta_j^{\mathrm{colim}}].$$

ALLORA with KET: Results for Language Modeling

Table: Learned colimit–limit split on frozen KET incidence backbones. Reductions are relative to the unregularized learned split row for the same corpus. The cross ratio is $\|C_{ij}\|_{\mathrm{F}}/(\|\Delta_i^{\mathrm{colim}}\Delta_j^{\mathrm{colim}}\|_{\mathrm{F}} + \|\Delta_i^{\mathrm{lim}}\Delta_j^{\mathrm{lim}}\|_{\mathrm{F}} + \|C_{ij}\|_{\mathrm{F}})$, averaged over adapter pairs and layers.

Dataset	Setting	λ_c	λ_x	loss Δ	bracket ↓	cross ↓	logit/order ↓	hidden/order ↓	cross ratio
WikiText-2	unregularized	0	0	0.0000	$1.0\times$	$1.0\times$	$1.0\times$	$1.0\times$	0.422
WikiText-2	best stability	0.03	0.001	0.0090	$112.8\times$	$20.2\times$	$3.44\times$	$3.03\times$	0.201
WikiText-2	strongest cross control	0.03	0.03	0.0108	$146.9\times$	$114.2\times$	$3.00\times$	$2.81\times$	0.051
WikiText-103	unregularized	0	0	0.0000	$1.0\times$	$1.0\times$	$1.0\times$	$1.0\times$	0.387
WikiText-103	best stability	0.03	0.01	0.0029	$117.8\times$	$57.0\times$	$3.85\times$	$3.56\times$	0.098
WikiText-103	strongest cross control	0.03	0.03	0.0069	$123.8\times$	$94.4\times$	$3.65\times$	$3.29\times$	0.062

Why this matters for trustworthy foundation models

If adapters represent learned preference or policy interventions, LICKET gives a practical algebraic control knob.

- Helpfulness, safety, style, and tool-use adapters should compose predictably.
- IC brackets measure adapter interference before deployment.
- ALLORA trains for routable, order-stable preference modules.
- KET supplies the structured computation substrate for constrained aggregation.
- This is a concrete bridge from IC/BRIDGE/SKFM geometry to RLHF-scale systems.

$$\text{latent-confounding diagnostic} \quad \rightsquigarrow \quad \text{adapter-training regularizer.}$$

KET beyond language modeling

- In Odyssey, KET transports foundry artifacts across contexts.
- In UDL, KET explains the duality between rollout and pullback.
- In IC, KET duality is differentiated: conditioning and observation become tangent-category structure.
- The same transport principle recurs across claims, skills, causal cells, plans, and dashboards.

Topos Integration using Causal Kan Extension Transformers

Foundry Admission Policy

Every foundry is admitted only if it has a certifiable TICKET

- a formal admission guarantee for a foundry artifact
- a certificate that local evidence survives transport
- a way to keep uncertainty and obstructions visible

What the ticket checks

- typed evidence
- restriction maps
- gluing compatibility
- obstruction and quarantine signals

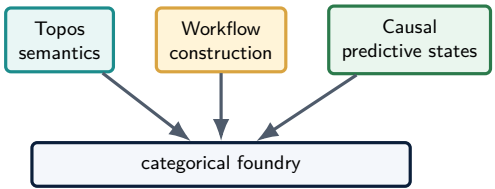
Tutorial translation: a TICKET says why a local computational result is allowed to enter a larger foundry.

Foundry Artifacts in the Tutorial

Artifact	**Categorical role**	**Concrete drilldown**
Causal cell	local IC object	Democritus + BRIDGE/SKFM
Markdown skill	tangent decision object	SkillOpt / UDL
Language context	KET transport object	KET LM experiment
Corporate plan	foundry construction object	Odyssey / BASKET / ROCKET
Dashboard	evidence surface	TICKET + audit views

Odyssey Foundry Semantics

- Topos/world-model semantics for sheaf-like foundries.
- Construction path for turning workflows into admitted artifacts.
- Causal predictive state representations (PSRs) model latent states over sheaves.
- The stable object is the foundry; the recipe can evolve.



The demo console lets the audience enter from any one of these layers and recover the same map.

Infinitesimal Causality: Checking Local Causal Claims

Plain-language definition

Infinitesimal Causality asks how a causal claim changes under small local perturbations, and whether nearby claims still fit together into a coherent larger causal story.

- It treats causal structure as something local before it is global.
- It records variation, restriction, and obstruction.
- It tells us when local causal evidence can be glued.

Piece of the grand challenge

Trustworthy models need more than confident outputs. They need causal claims whose local evidence, uncertainty, and failure modes can be inspected.

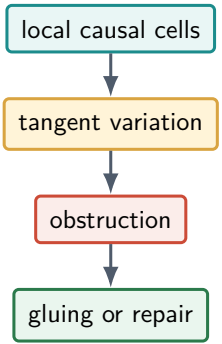
Reference: *Infinitesimal Causality*, arXiv:2606.24621 (2026).

Infinitesimal Causality

High-level idea

Infinitesimal Causality studies how local causal claims vary, restrict, obstruct, and glue into larger causal structures.

- A causal claim is not only true or false globally.
- It has local neighborhoods, perturbations, and tangent directions.
- Obstructions tell us when local causal pieces cannot be glued.
- Tangent categories provide the abstraction for differentiated causal structure.



IC as the geometry of causal admission

- Foundry artifacts arrive with local evidence and uncertainty.
- IC asks whether infinitesimal changes preserve causal coherence.
- Admission becomes a geometric question: can the artifact survive restriction, perturbation, and gluing?

Foundry reading

$$\text{artifact} + \text{local evidence} \rightsquigarrow \text{causal tangent test}$$

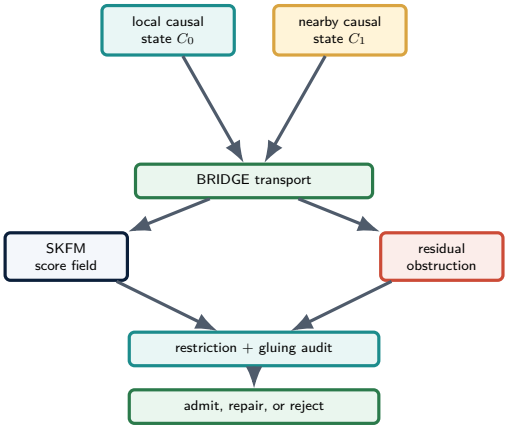
$$\text{pass} \iff \text{compatible gluing data}$$

BRIDGE/SKFM as computational IC

Role in the tutorial. BRIDGE/SKFM computes the IC question: do nearby causal descriptions move coherently, or expose an obstruction?

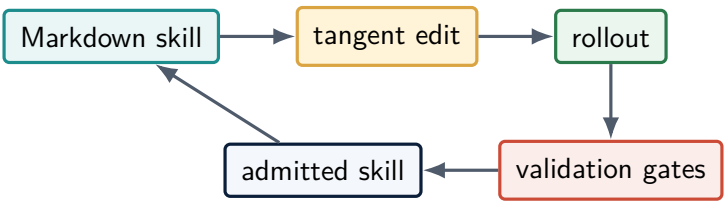
- **BRIDGE**: transport between local causal states.
- **SKFM**: estimate the local score/flow field.
- **IC**: residuals become tangent obstructions.
- **Foundry**: coherent transports glue and admit.

Reference: BRIDGE/SKFM, arXiv:2606.19610 (2026).



SkillOpt as a UDL recipe

- SkillOpt is a computational recipe for UDL at the level of Markdown skills.
- A skill edit is a tangent direction in skill space.
- Rollout extends the skill; validation pulls it back through observations and gates.



Formalization of Agentic Workflow Optimization

Question

How do we say formally that an agentic system is not merely editing text, but optimizing local directions in a structured artifact space?

- IC studies local perturbations and obstructions.
- Markdown skills, tickets, plans, and workflows are structured artifacts.
- BRACKET regularizes noncommuting update directions.
- SkillOpt improves skills by bounded Markdown edits.
- ROCKET repairs plan objects by local graph edits.
- One tangent-category abstraction covers both.

The abstraction is a tangent category of structured artifacts.

The Right Object Is Not Raw Markdown

Raw Markdown

- string-level syntax
- unstable line numbers
- ambiguous block roles
- patch conflicts with no geometry

Typed anchored Markdown

- parse tree with stable anchors
- typed blocks: instruction, guardrail, schema, example
- admissible local edits
- rollout and validation semantics

Clean technical claim

The tangent structure appears after parsing, anchoring, typing, and restricting to admissible edits.

The Category Md

Objects

A typed Markdown object is

$$M = (A, P, \tau, \sigma),$$

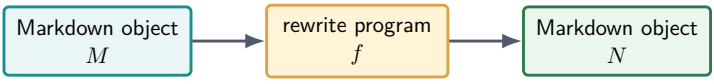
where A is an anchor poset, P is a Markdown parse tree, τ assigns block types, and σ stores semantic metadata.

Morphisms

A morphism $f : M \to N$ is an anchor-respecting rewrite program:

$$f = (f_A,\ f_P,\ f_\sigma)$$

transporting anchors, parse fragments, and metadata while preserving declared contracts.



Edit Fibers

Infinitesimal edit fiber

For each $M \in \mathrm{Md}$, define $\mathcal{E}(M)$ to be the fiber of anchored admissible edits at M.

$$\delta \in \mathcal{E}(M)$$

- rewrite an instruction clause
- add or weaken a guardrail
- modify a JSON schema field
- add a validation example
- insert a missing plan action
- rewire a plan edge
- add evidence constraints
- modify a tool contract

Additive structure

Compatible edits merge by fiberwise addition:

$$\delta_1 + \delta_2 \in \mathcal{E}(M).$$

Tangent Bundle of Markdown

Tangent object

$$\mathsf{T}M = (M, \mathcal{E}(M)).$$

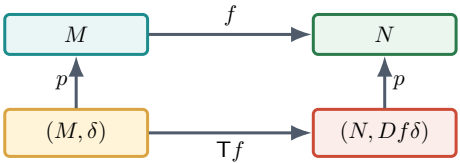
An element of $\mathsf{T}M$ is a pair:

$$(M, \delta).$$

Pushforward

For $f : M \to N$,

$$\mathsf{T}f(M, \delta) = (N, Df_M(\delta)).$$



Proposition

Typed Markdown has a tangent category

Let Md be the category of typed anchored Markdown objects and anchor-respecting rewrite programs. Suppose:

1. each $\mathcal{E}(M)$ is an additive commutative monoid or module;
2. each $f : M \to N$ has a pushforward $Df_M : \mathcal{E}(M) \to \mathcal{E}(N)$;
3. pushforwards preserve zero and addition;
4. pushforwards respect identity and composition.

Then Md carries a tangent-category structure with tangent endofunctor T.

$$Df_M(0) = 0, \qquad Df_M(\delta_1 + \delta_2) = Df_M(\delta_1) + Df_M(\delta_2)$$
$$D(g \circ f)_M = Dg_{f(M)} \circ Df_M.$$

Structure Maps

Projection and zero

$$p_M : \mathsf{T}M \to M, \qquad p_M(M, \delta) = M$$

$$0_M : M \to \mathsf{T}M, \qquad 0_M(M) = (M, 0)$$

Addition over a shared base

$$+_M : \mathsf{T}M \times_M \mathsf{T}M \to \mathsf{T}M$$

$$(M, \delta_1) + (M, \delta_2) = (M, \delta_1 + \delta_2)$$

Vertical lift

$$l_M : \mathsf{T}M \to \mathsf{T}^2 M$$

$$l_M(M, \delta) = (M, \delta; 0, \delta)$$

Canonical flip

$$c_M : \mathsf{T}^2 M \to \mathsf{T}^2 M$$

$$c_M(M, \delta_1; \delta_2, \delta_{12}) = (M, \delta_2; \delta_1, \delta_{12})$$

Proof Sketch

- T is functorial by identity and composition of edit pushforwards.
- p and 0 are natural because anchor transport forgets or preserves empty edits.
- Addition is natural because Df_M preserves compatible edit merge.
- Lift records a direction as a vertical direction.
- Flip swaps the two first-order edit coordinates in T^2M.
- The axioms reduce fiberwise to monoid/module identities.

Takeaway

We are not claiming arbitrary text is smooth. We are showing that typed Markdown plus anchored admissible edits forms an additive bundle, and that bundle supplies tangent-category structure.

Instance 1: SkillOpt as a Vector Field

SkillOpt state

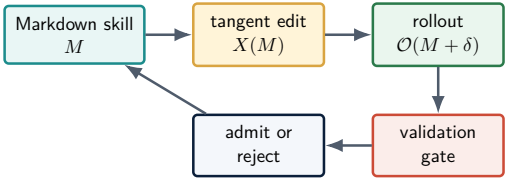
A SkillOpt state is a Markdown object M. A candidate edit is a tangent vector:

$$\delta \in \mathcal{E}(M).$$

Proposal rule

A skill-improvement rule is a vector field:

$$X : M \to \mathsf{T}M, \qquad p \circ X = \mathrm{id}_M.$$



SkillOpt: UDL on $\mathrm{TMd}_{\mathrm{skill}}$

Observation functor

SkillOpt evaluates edits through a semantic observation functor:

$$\mathcal{O} : \mathsf{Md} \to \mathbf{Beh},$$

where $\mathbf{Beh}$ contains rollout traces, validation scores, causal graphs, held-out behavior, or transfer metrics.

Rollout

$$M, \delta \mapsto \mathcal{O}(M + \delta)$$

Extend the local skill edit across task contexts.

Pullback

$$\mathcal{O}(M + \delta) \mapsto \mathsf{gate}(M, \delta)$$

Pull back failures, constraints, and evidence.

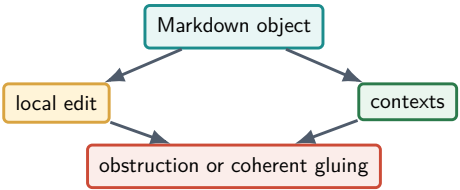
SkillOpt = UDL over tangent directions of Markdown skills.

IC on Markdown Objects

IC question

Do nearby perturbations of a structured artifact move coherently across contexts, or expose an obstruction?

- object: skill, spec, causal note, plan
- local perturbation: anchored edit
- context: task, domain, tool, user, evidence slice
- transport: push edit through contexts
- obstruction: failure to glue
- repair: add missing subskill, variable, guardrail, or statistic

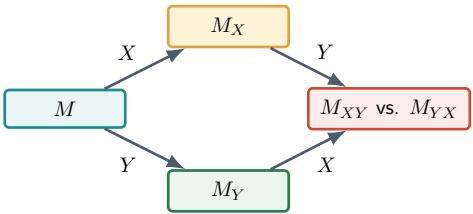


Lie Brackets of Edit Fields

Two edit fields

Given vector fields $X, Y : M \to \mathsf{T}M$, their bracket measures first-order noncommutativity:

$$[X, Y](M) = DY_M(X(M)) - DX_M(Y(M)).$$



Residual

The unexplained component of $M_{XY} - M_{YX}$ is an IC obstruction: a missing context, subskill, guardrail, or sufficient statistic.

BASKET and ROCKET

BASKET

$$\text{text evidence} \quad \rightsquigarrow \quad P \in \mathsf{Md}_{\mathrm{plan}}$$

- extracts typed workflow graphs
- builds plan objects with evidence
- supplies the geometry ROCKET optimizes

ROCKET

$$J : \mathsf{Md}_{\mathrm{plan}} \to \mathbb{R}$$

- plan repairs are tangent edits
- reward selects directions
- agentic ROCKET learns a repair vector field

Same mathematics, different base object

SkillOpt acts on skill Markdown; BASKET/ROCKET acts on plan artifacts; ALLORA acts on adapter parameters. Each chooses and regularizes tangent directions.

Instance 2: BASKET and ROCKET as Plan Optimization

BASKET

BASKET extracts and represents plans from structured or semi-structured text:

$$\text{text evidence} \longmapsto \text{workflow graph}.$$

ROCKET

ROCKET optimizes over those plan objects:

$$J : \mathsf{Md}_{\text{plan}} \to \mathbb{R}, \qquad \delta^\star = \arg\max_{\delta \in \mathcal{E}(P)} J(P + \delta).$$

ROCKET Is Tangent Optimization on Plans

Plan object

$$P = (V, E, \text{evidence}, \text{metadata})$$

where vertices are typed business actions and edges are workflow dependencies.

Tangent repairs

- insert or delete an action
- add or drop an edge
- promote a macro motif
- retrieve local evidence
- stop

Reward

ROCKET combines:

- text fit
- structural validity
- local / sector / panel support
- macro coherence
- downstream financial alignment

Tangent reading

ROCKET chooses directions in $\mathsf{T}_P\mathsf{Md}_{\text{plan}}$, not arbitrary text completions.

Plan Repair as a Tangent Direction

Before

$$\texttt{market} \rightarrow \texttt{realize\_revenue}$$

After

$$\texttt{expand} \rightarrow \texttt{market} \rightarrow \texttt{realize\_revenue}$$

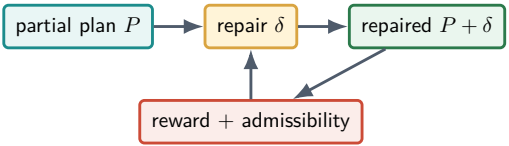
Tangent edit

$$\delta = \texttt{insert(expand)}$$

$$\delta \in \mathcal{E}(P)$$

Why ROCKET accepts it

The repaired plan has better macro support, stage coherence, and downstream plausibility while remaining close to the evidence.



Odyssey: SkillOpt as One Optimization Layer

Demo task

SkillOpt optimizes a Markdown skill for Odyssey causal-claim admission. The target model emits a strict JSON causal bundle with cause, effect, relation, qualifier, evidence span, mechanism, scope, guardrail, and ticket status.

Odyssey supplies structure

- TICKET admission gate
- DB obstruction energy
- GT graph similarity
- KET transfer score

SkillOpt supplies learning

- rollout under current skill
- reflection on structured failures
- bounded Markdown edits
- validation-gated update

Demo artifacts: `icml-2026-ket-demo-console/skillopt_causal_claim_admission/` and `SkillOpt/skillopt/envs/odyssey_causal_claim/`.

From Tangent Bundles to Lie Algebroids

Infinitesimal causal geometry

Nonzero Lie brackets diagnose noncommuting interventions and hidden structure.

- BRIDGE/SKFM: bracket residuals expose latent causal structure.
- ALLORA: use commutator penalties to make neural adapters compose.
- SKILLOPT: optimize external skills by editing prompts, schemas, rubrics, tools, and validators.

IC machinery $\Longrightarrow$ not only adapters, but agentic workflow optimization.

SkillOpt's bottleneck is interaction search

SkillOpt makes agent improvement experimental: propose a bounded skill edit, roll out the agent, score validation behavior, and accept or reject.

What works

Skills are external, inspectable state: prompts, examples, tool contracts, schemas, rubrics, and validators.

Where it grinds

Useful repairs often live in ordered pairs or short programs, not isolated single edits.

$$\mathrm{score}(a) \approx 0, \quad \mathrm{score}(b) \approx 0, \qquad \mathrm{score}(a \circ b) \gg 0, \quad \mathrm{score}(b \circ a) \approx 0.$$

The algebroid lift

Ordinary IC studies tangent fields on a base space. Skill edits need one extra layer: the controlled edit is not identical to its visible Markdown effect.

$$\mathcal{A} \longrightarrow \mathsf{Md}, \qquad \rho : \mathcal{A} \longrightarrow \mathsf{TMd}, \qquad [\,\cdot,\cdot\,]_{\mathcal{A}}.$$

Base	Sections	Anchor
Typed, anchored Markdown: schemas, evidence fields, examples, policies, validators, traces.	Available edit policies: repair schema, normalize answer, add citation, fix tool contract.	Visible document-level effect of an edit policy after applying it to the current skill state.

Brackets tell us what to validate

Key geometric signal

The algebroid bracket measures order-sensitivity before projecting to visible skill behavior.

$$[s,t]_{\mathcal{A}} = s \circ t - t \circ s, \qquad \rho([s,t]_{\mathcal{A}}) = [\rho(s), \rho(t)].$$

- Low bracket: edits are approximately independent; single-edit validation is enough.
- High bracket: edit order matters; spend rollout calls on the ordered pair.
- Kernel component: hidden template, routing, or implementation state changes future behavior.

cheap bracket screen $\Longrightarrow$ focused expensive validation.

Controlled anchor-chain benchmark

We use a deliberately small, interpretable SkillOpt-style benchmark before moving to noisy real-world tasks.

Construction

There are n skill anchors. Each anchor has a productive ordered repair pair; reversed or random pairs do not close the chain reliably.

Served validation

Candidate Markdown skills are validated by a local EXO-served OpenAI-compatible LLM endpoint.

$$\text{score} = \text{progress through productive chains} \times \text{served-model faithfulness}.$$

The point is not benchmark realism; it is a controlled failure mode where enumeration is expensive and the bracket prior should win.

Ten-anchor benchmark test

Method	Bracket probes	EXO validation probes	Score
Greedy single edit	0	10	0.000
Random ordered pairs	0	11	0.120
Exhaustive ordered pairs	0	91	1.000
Algebroid prioritized	90	11	1.000

What the smoke test establishes

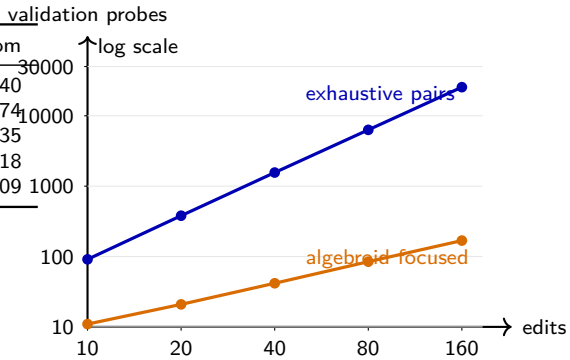
The bracket screen recovers the exhaustive ordered-pair solution while using the served LLM only for the focused validation path.

Local model: `mlx-community/Llama-3.2-3B-Instruct-4bit`.

Scaling result

Edits	Exhaustive	Algebroid	Reduction	Random
10	91	11	$8.3\times$	0.140
20	381	21	$18.1\times$	0.074
40	1561	42	$37.2\times$	0.035
80	6321	84	$75.2\times$	0.018
160	25441	168	$151.4\times$	0.009

Algebroid score is 1.000 for every row, observed with local Exo.



What this result says

Empirical message

A Lie algebroid bracket prior converts the relevant validation burden from exhaustive ordered-pair enumeration to a focused chain of served checks.

- Exhaustive validation scales like $n(n-1)+1$ served calls.
- Focused algebroid validation scales like one productive path, plus cheap bracket probes.
- Random ordered-pair validation collapses as the number of anchors grows.

160 edits: 25441 served probes vs. 168 served probes.

LASKO: exact scaling table

Edits	Exhaustive probes	Focused probes	Reduction	Random mean	Source
10	91	11	$8.3\times$	0.140	observed
20	381	21	$18.1\times$	0.074	observed
40	1561	42	$37.2\times$	0.035	observed
80	6321	84	$75.2\times$	0.018	observed
160	25441	168	$151.4\times$	0.009	observed

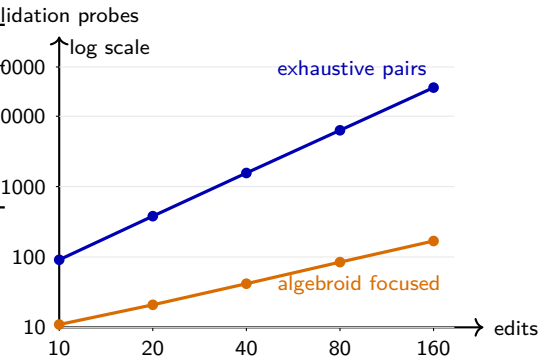
Experimental condition

All algebroid rows used local EXO with `mlx-community/Llama-3.2-3B-Instruct-4bit` and focused chain validation.

LASKO Skill Optimization Results

Edits	Exhaustive	Algebroid	Reduction	Random
10	91	11	$8.3\times$	0.140
20	381	21	$18.1\times$	0.074
40	1561	42	$37.2\times$	0.035
80	6321	84	$75.2\times$	0.018
160	25441	168	$151.4\times$	0.009

Algebroid score is 1.000 for every row



[Mahadevan, Skill Optimization using Lie Algebroids, 2026, in preparation]

Agentic 10-K Workflow Optimization using LASKO

We apply LASKO to a DEMOCRITUS temporal 10-K financial research workflow. The anchors are BASKET/ROCKET artifacts:

`section_router`, `business_model_claims`, `risk_factor_rebuttals`, `mda_operating_claims`, `numeric_table_extractor`, `numeric_table_gate`, `evidence_ledger`, `citation_binding`, `mechanism_graph`, `macro_motif_completion`, `financial_outcome_alignment`, `temporal_stability`, `narrative_synthesis`, `report_validator`.

Seven scenarios model common 10-K research paths: business model to revenue, risk-adjusted thesis construction, numeric financial claims, MD&A mechanism repair, cross-year ROCKET stability, Democritus report gating, and the full BASKET/ROCKET loop. A numeric table gate must follow table extraction before a verified claim enters the evidence ledger; risk rebuttals must be extracted before a risk-adjusted narrative

10-K Worfklow Optimization using LASKO

Served validation protocol

Candidate 10-K workflow skills are sent to an EXO-hosted `mlx-community/gpt-oss-20b-MXFP4-Q8` endpoint. The model returns scenario edge IDs judged closed; the benchmark converts those IDs into workflow scores.

Method	Bracket	EXO validations	Score	Faithful
Greedy single edit	0	15	0.000	1.000
Random ordered pairs	0	29	0.537	0.903
Exhaustive ordered pairs	0	183	1.000	1.000
Algebroid prioritized	182	29	1.000	1.000

183 served validations $\rightsquigarrow$ 29 served validations same score $= 1.000$.

ODYSSEY Foundries

Foundry	Cover / local sections	Current purpose
Brand / product meaning	Customer journey, product experience, brand promise, channel message, competitive context.	A market-meaning foundry for testing whether product evidence, reviews, and promise claims cohere.
Indus Script	Symbol inventory, inscription sequences, statistical structure, decipherment hypotheses	A scientific-challenge foundry
Research program	Goals, tasks, methods, tools	Process foundries for maintaining ODYSSEY itself
Grounded Toulmin/local LLM	Prometheus event, document claim, source context, source alignment, restriction/gluing diagnostics, visible Toulmin ticket.	A neural argument-inspection foundry for comparing local LLMs on whether they preserve grounded claims and warrant them without over-transporting weak evidence.

Scientific Foundries: Epistemic Discipline

Foundry	Scale and local sections	What glues	What remains blocked
TCC 44K	44K-paper economics causal-claims atlas; 295,252 nodes, 261,714 edges, 265,656 support rows, method/journal/p-value lookup tables.	Node-edge, edge-support, and support-to-lookup restrictions preserve document provenance and evidence summaries.	Direction, polarity, controversy, and reverse-causality joins remain reviewable guardrails.
Indus Script	419 signs, 1,548 visual sequences, decipherment papers, statistical structure, archaeological context, and controversy records.	Symbol-sequence, sequence-statistics, statistics-hypotheses, and hypotheses-archaeology restrictions glue provisionally.	The controversy/hypothesis restriction blocks any settled decipherment claim.

Table: Two scientific foundries: TCC prevents a large support graph from becoming an unqualified causal truth map; Indus has statistical structure, but not a decipherment.

Ocean Temperature Rise Claims Foundry

Local context	Events	Rank	Hist.	Tests	Shared	Mean gap	Weighted loss
Larval survival affected by food scarcity	46	44	45	45	2,025	0.0219	0.001950
Subpolar gyre weakening effects	44	37	41	40	1,640	0.0246	0.003436
Rising ocean surface temperatures	38	24	30	30	900	0.0330	0.002639
Fisheries output and secondary activities	38	38	38	38	1,444	0.0260	0.004572
Metabolic evolution of mixotrophs	37	30	33	33	1,089	0.0299	0.002163

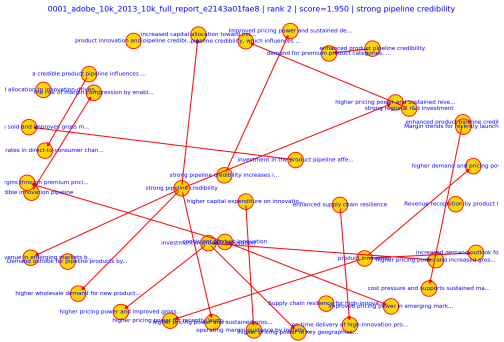
Table: Representative local predictive states from the ocean-temperature bundle. "Hist." denotes finite histories. Shared cells and mean gap are corpus-to-context restriction diagnostics. Weighted loss is the gluing diagnostic when reported in the bundle view. The table shows why the artifact is not a single graph: biological mechanisms, circulation mechanisms, economic consequences, measurement issues, disease pathways, and retrieval drift all become separate local predictive-state charts.

Example 1: Corporate Foundry for Adobe

- Causal Structure in Adobe's 2013 Form 10-K.
- Focus on product innovation and pipeline credibility.
- Extract local causal claims about pricing power, R&D, product pipeline, margins, and demand.
- Rank and audit local causal models before promotion into a foundry artifact.

Trust question

Which product-pipeline causal claims are supported by the filing, and where do credibility or gluing issues appear?

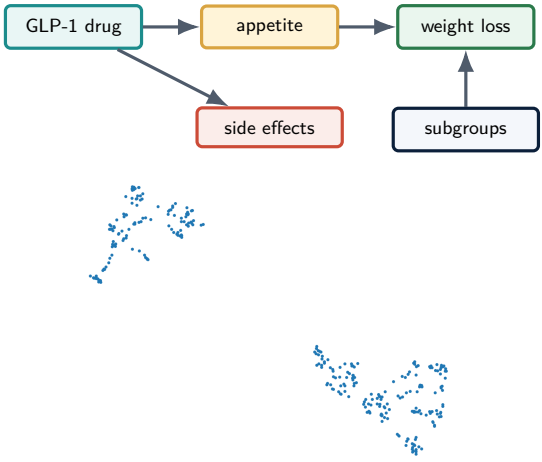


Local causal model from Adobe's 2013 Form 10-K: filing-derived claims about product innovation, pricing power, pipeline credibility, margins, and demand. Democritus reference: arXiv:2512.07796.

Example 2: Scientific Causal Foundries

- Build local causal models from scientific papers and domain reports.
- GLP-1: drug mechanisms, appetite regulation, metabolic outcomes, side effects, and patient subgroups.
- Ocean warming: temperature change, habitat shifts, food webs, migration, and fish population stress.
- Admit claims only when evidence, scope, uncertainty, and conflicts remain visible.

Trust question. Which causal claims survive when evidence is assembled across studies, populations, mechanisms, and ecological contexts?

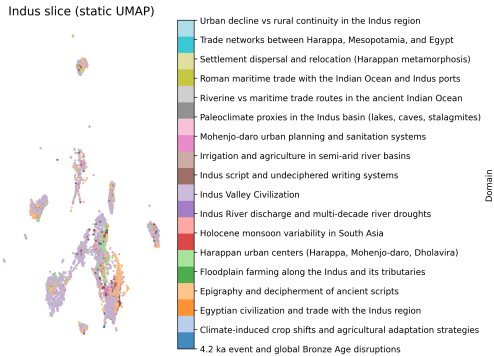


Two scientific foundry inputs: a GLP-1 mechanism sketch and an ocean-warming/fish-population manifold extracted from paper neighborhoods.

Example 3: Indus Script Decipherment Foundry

- Start from seals, sign inventories, inscriptions, archaeological context, and competing language hypotheses.
- Treat candidate readings as provisional artifacts, not conclusions.
- Track where evidence restricts a hypothesis and where ambiguity remains.
- Promote only claims whose provenance and uncertainty can be inspected.

Trust question. How can a foundation-model system explore decipherment hypotheses without turning speculation into unsupported certainty?

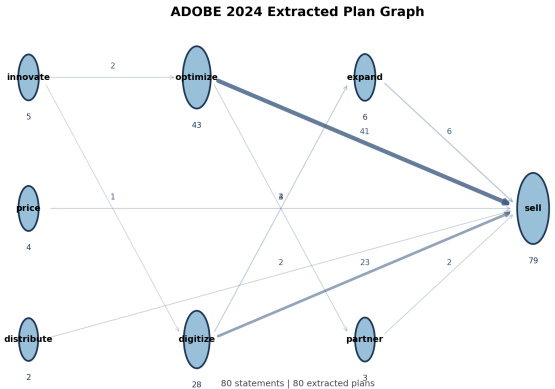


Prometheus/Democritus Indus slice: clusters of archaeological, paleoclimate, river, trade, and script-decipherment neighborhoods.

Example 4: Temporal Democritus for 10-K Filings

- Start from Adobe 10-K filings over time.
- Extract causal-style claims for each year.
- Glue yearly claim structures through temporal diffusion.
- Track how strategy, risk, and product claims evolve.

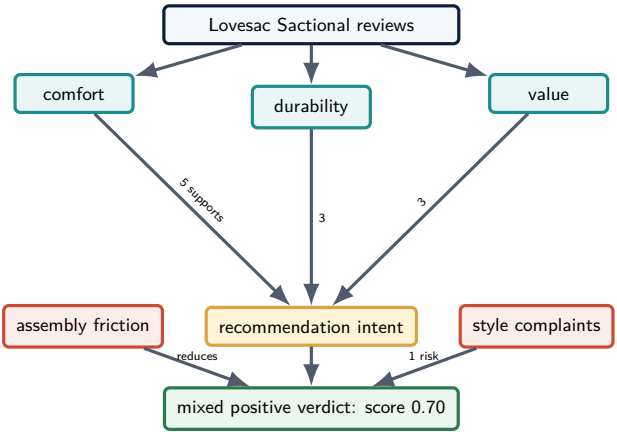
Trust question. Which claims persist, change, or become obstructed over time?



Example 5: Lovesac Sactional Product-Feedback Foundry

- Start from retrieved Lovesac sectional-sofa reviews.
- Extract comfort, durability, value, style, and assembly-friction signals.
- Build causal-style hypotheses about recommendation intent and satisfaction.
- Keep the verdict inspectable: mixed positive, score 0.70, seven hypotheses.

Trust question. Which product conclusions are supported by review evidence, and which require telemetry such as returns or conversion?



Prometheus product-feedback run: causal-style hypotheses from Lovesac review evidence, with support counts and admitted uncertainty.

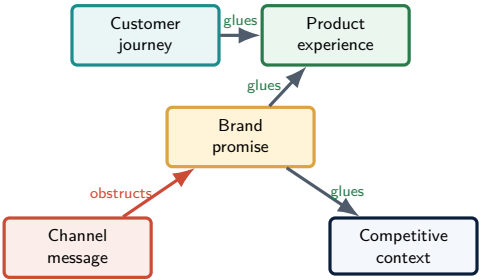
Constructed Sheaf Example: Product and Brand Feedback

Generated cover

Odyssey constructs a brand/product-feedback sheaf whose local sections each carry a small logic of claims and evidence states.

- Customer journey
- Product experience
- Brand promise
- Channel message
- Competitive context

Audit result. Three overlaps glue; the message/promise overlap obstructs and needs more evidence.

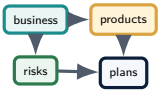


Overlap	Status	Decision surface
journey/prod.	glues	prioritize product fix
promise/prod.	glues	validate promise
message/promise	obstructs	adjust campaign
promise/comp.	glues	refine positioning

Constructed Sheaves Across Tutorial Domains

Adobe 10-K filings

Local sections are filing neighborhoods: business model, products, risks, revenue language, and extracted workflow plans.



Question: which workflow claims remain compatible with filing evidence and risk language?

GLP-1 literature

Local sections are article neighborhoods: mechanism, appetite regulation, weight loss, adverse effects, and patient subgroups.

Question: which causal claims glue across mechanisms, outcomes, and population scope?

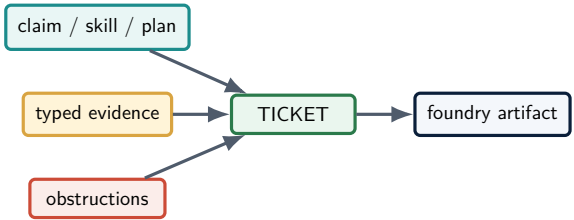
Lovesac/product feedback

Local sections are review and market contexts: comfort, durability, value, style, assembly friction, and recommendation intent.

Question: which product conclusions glue, and which need more telemetry?

Structured Foundry Construction

- A foundry is where model products become inspectable artifacts.
- Artifacts carry typed evidence, scope, uncertainty, provenance, and obstruction signals.
- Admission is not a vibe check: it is a structured compatibility check.
- Algorithms can change while the admission structure remains stable.



Foundry construction is where KET/DB outputs become auditable tutorial artifacts.

Foundry Construction as Kan Extension

Left-Kan construction

Local artifacts are assembled into a larger foundry object:

$$\mathrm{Lan}_{\iota}\, E$$

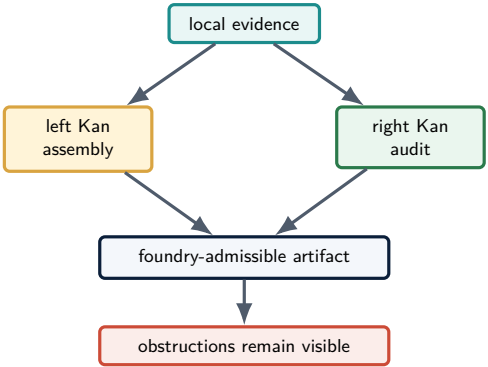
where E records typed evidence over local contexts and ι embeds those contexts into the foundry index.

Right-Kan audit

The candidate artifact is checked against downstream constraints:

$$\mathrm{Ran}_{\pi}\, C$$

where C records compatibility, scope, uncertainty, and obstruction signals.



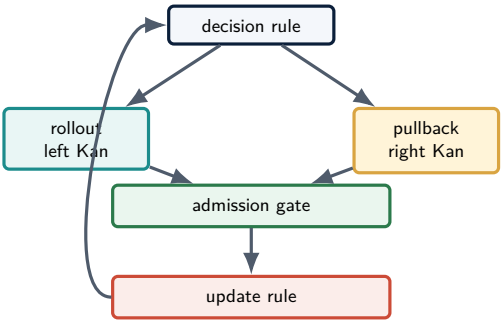
A foundry is not just storage; it is structured assembly plus structured admissibility.

Universal Decision Learning: The Admission Loop

- A decision system proposes a candidate artifact, skill, plan, or claim.
- Rollout asks what the proposal does when extended into task contexts.
- Pullback asks what must be true locally for that proposal to be admissible.
- Learning updates the decision rule, not merely the output.

UDL perspective

Decision learning is a KET process over admissibility: left-Kan rollout explores consequences, while right-Kan pullback enforces evidence and constraints.



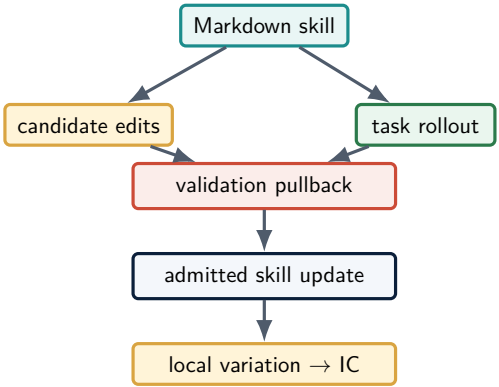
Reference: *Universal Decision Learners*, arXiv:2605.30694 (2026).

LASKO as a KET/UDL Process

Concrete recipe

LASKO treats a skill document as the object being learned. Candidate edits are rolled out on tasks, pulled back through validation gates, and admitted only when they improve the held-out decision behavior.

- **Left-Kan side**: rollout candidate skill edits across examples.
- **Right-Kan side**: pull back failures, constraints, and validation evidence.
- **UDL side**: update the rule for which skill edits are admissible.

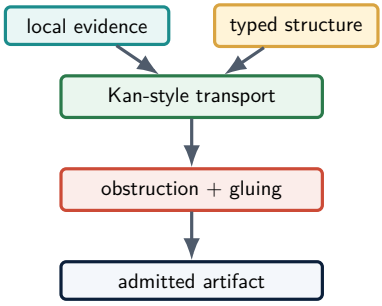


Main Takeaway from the Tutorial

The modeling problem

Foundation-model systems must act across changing contexts while remaining auditable, composable, and grounded in evidence.

- Andon Market makes the trust problem concrete.
- Sheaves/topos world models organize local truth.
- Foundries turn model products into admitted artifacts.



Three Mathematical Lenses

KET

Transport evidence and representations across contexts by left/right Kan extension structure.

UDL

Learn admissible decisions by coupling rollout with pullback through evidence and gates.

IC

Differentiate local causal structure: variation, obstruction, restriction, and gluing.

Common theme

The stable object is not a particular implementation. It is the categorical pattern that lets local model behavior become a trustworthy foundry artifact.

The demos and papers provide the slower path through the details; this tutorial is the guided tour.

What to Remember

1. **Local evidence is not enough.** Trust requires restriction maps, compatibility checks, and obstruction signals.
2. **Attention and diffusion are transport mechanisms.** KET views them through Kan extensions rather than only through layer formulas.
3. **Backpropagation can be diagrammatic.** Gradients become part of a compositional calculus over structured diagrams.
4. **Agent skills are artifacts.** UDL and SkillOpt treat workflow improvement as admission through evidence and gates.
5. **Causal claims need infinitesimal diagnostics.** IC asks whether local causal changes glue into a coherent larger story.

Thank You

Unifying Attention and Diffusion with Kan Extension Transformers

Structured Deep Learning with Diagrammatic Backpropagation

Final thought

Modern foundation models are powerful, but their representations, training dynamics, and agentic workflows remain difficult to audit, compose, and trust. This tutorial presents a categorical and geometric framework for building *foundries*: composable building blocks of trustworthy foundation-model systems.

Sridhar Mahadevan | Adobe Research and University of Massachusetts Amherst