Categorical Insurance: Learners and Comonadic Governance

J. Michael Constans and Claude Opus 4.7

***Abstract*—We describe the mathematical structure underlying a Haskell library for machine-learning–driven design of insurance contracts. Learners, in the sense of Fong–Spivak–Tuyéras and Cruttwell–Gavranović–Ghani–Wilson–Zanasi, appear as morphisms in a symmetric monoidal category that uniformly captures both classical actuarial estimators (we exhibit Bühlmann credibility) and gradient-based methods (we exhibit linear regression). Governance is carried by the Env comonad over a monoid of rule sets, so federal, state, and internal regulations compose via the monoid operation. Contract construction is a co-Kleisli arrow factoring through this comonadic context, making the rule *"no contract may be made that violates governance"* a static guarantee rather than a runtime convention. Layered regulation reduces, by conjunctivity of satisfaction over the monoid sum, to a single co-Kleisli composite, and switching jurisdiction becomes a substitution at the carrier level that is functorial in $\mathsf{Mon}(\mathsf{Set}) \to \mathsf{CoMon}(\mathsf{Set})$.**

***Index Terms*—Category theory, comonad, parametric learner, lens, governance, insurance pricing, Bühlmann credibility, gradient descent, Para construction, co-Kleisli.**

I. Introduction

THE two recurring problems in machine-learning–driven insurance pricing are heterogeneity and governance. Classical actuarial methods (credibility theory, generalised linear models, Kalman filters) and modern machine learning (gradient-trained networks, kernel methods) are usually treated as disjoint toolkits, even though they share a common operational structure: state, prediction, update. At the same time, insurance contracts cannot be written freely; they must comply with regulatory, internal, and contractual constraints that themselves compose. Treating governance as a runtime check bolted onto a model leaves open the possibility of constructing contracts that should never have existed.

This paper gives a mathematical account of one answer: *learners are morphisms in a symmetric monoidal category, governance is a comonadic context, and contracts are the audited outputs of co-Kleisli pipelines.* The development is exhibited in the `categorical-insurance` library; a working implementation in Haskell is available at `github.com/soulstrop/categorical-insurance`.

Notation.: We work informally in Set. Categories are denoted $\mathcal{C}, \mathcal{D}$; specific categories of interest in $\mathsf{sans\text{-}serif}$. We write $\otimes$ for the monoidal product (in Set, the Cartesian product $\times$), $\mathbf{1}$ for the monoidal unit, and $\mathrm{inl}, \mathrm{inr}$ for the coproduct injections.

J. M. Constans is with Feature Software, LLC. E-mail: mikeco@constans.dev.

Claude Opus 4.7 is a large language model developed by Anthropic (model identifier `claude-opus-4-7`). This paper was prepared through interactive collaboration with that model: the human author directed the scope, theorems, and exposition; the model drafted prose, formalised statements, and produced the `tikz-cd` diagrams under continuous human review.

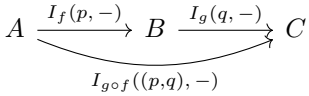

Fig. 1. Forward pass of $g \circ f$.

II. The Category of Learners

Definition 1 (Learner)**.** A *learner* from A to B is an equivalence class of quadruples (P, I, U, R) where P is a set (the *state* or *parameter space*) and

$$\begin{aligned} I &\colon P \times A \longrightarrow B && \text{(implementation)},\\ U &\colon P \times A \times B \longrightarrow P && \text{(update)},\\ R &\colon P \times A \times B \longrightarrow A && \text{(request)}. \end{aligned}$$

Two quadruples (P, I, U, R) and (P', I', U', R') are equivalent if there is a bijection $\varphi \colon P \to P'$ intertwining all three structure maps in the obvious sense.

We write $\mathsf{Learn}(A, B)$ for the set of learners from A to B. We call I the *prediction*, U the *update*, and R the *request*: it propagates a target signal upstream during sequential composition.

Example 2 (Identity learner)**.** $\mathrm{id}_A := (\mathbf{1},\ I,\ U,\ R)$ where $I(*, a) = a$, $U(*, a, a') = *$, $R(*, a, a') = a'$.

Definition 3 (Sequential composition)**.** For $f = (P_f, I_f, U_f, R_f) \in \mathsf{Learn}(A, B)$ and $g = (P_g, I_g, U_g, R_g) \in \mathsf{Learn}(B, C)$, the composite $g \circ f \in \mathsf{Learn}(A, C)$ is defined by

$$\begin{aligned} P_{g\circ f} &= P_f \times P_g,\\ I_{g\circ f}((p,q), a) &= I_g\big(q,\ I_f(p,a)\big),\\ U_{g\circ f}((p,q), a, c) &= \big(U_f(p, a, b^*),\ U_g(q, I_f(p,a), c)\big),\\ R_{g\circ f}((p,q), a, c) &= R_f\big(p, a, b^*\big), \end{aligned}$$

where $b^* := R_g(q, I_f(p,a), c)$ is the target produced by the downstream request map.

The forward direction reads as the composite of Figure 1. The request map of g produces, given the prediction $b = I_f(p,a)$ and the desired output c, a target b^* for f; the update of f then trains against this internal target rather than against c directly. This is what makes Learn a category rather than a mere semigroup.

Theorem 4 ([1], [2])**.** *Composition in Theorem 3 is associative and unital with respect to Theorem 2. Hence* Learn *is a category whose objects are sets and whose hom-sets are* $\mathsf{Learn}(A, B)$.

Definition 5 (Parallel product)**.** For $f \in \mathsf{Learn}(A, B)$ and $g \in \mathsf{Learn}(C, D)$, the parallel product $f \otimes g \in \mathsf{Learn}(A \times C, B \times D)$ has parameter space $P_f \times P_g$ and acts componentwise on each factor.

Theorem 6 ([2])**.** $(\mathsf{Learn},\ \otimes,\ \mathbf{1})$ *is a symmetric monoidal category.*

Remark 7 (Para and lenses). The triple (I, U, R) has the type of a parametrised optic: I and U together resemble a lens get/put, and R provides the residual that makes sequential composition functorial. The construction factors through the bicategory $\mathsf{Para}(\mathsf{Optic})$ of parametrised optics [3], which puts lens machinery directly at the disposal of learner state.

III. Two Instances

The point of Theorems 4 and 6 is that wildly different estimators inhabit the same algebraic universe. We exhibit one instance of each flavour and observe that they compose without mediation.

A. Bühlmann credibility as a Bayesian learner

Let $X \sim \mathcal{N}(\theta, \sigma^2)$ with σ^2 known and prior $\theta \sim \mathcal{N}(\mu_0,\ 1/\kappa_0)$. The Normal–Normal conjugate posterior after observation x has parameters

$$\kappa_1 = \kappa_0 + \frac{1}{\sigma^2}, \quad \mu_1 = \frac{\kappa_0\,\mu_0 + x/\sigma^2}{\kappa_1}.$$

This defines $\mathrm{Cred}_{\mu_0,\kappa_0,\sigma^2} \in \mathsf{Learn}(\mathbf{1},\ \mathbb{R})$:

$$\begin{aligned} P &= \{(\mu, \kappa) \in \mathbb{R} \times \mathbb{R}_{>0}\}, \\ I((\mu,\kappa),\ *) &= \mu, \\ U((\mu,\kappa),\ *,\ x) &= \left(\frac{\kappa\,\mu + x/\sigma^2}{\kappa + 1/\sigma^2},\ \kappa + \frac{1}{\sigma^2}\right), \\ R((\mu,\kappa),\ *,\ x) &= *. \end{aligned}$$

Proposition 8 (Credibility from the recursion)**.** *After n observations $x_1, \ldots, x_n$ the implementation returns*

$$I = Z \cdot \bar{x} + (1 - Z) \cdot \mu_0, \quad \bar{x} := \tfrac{1}{n}\textstyle\sum_i x_i, \quad Z := \frac{n}{n + \kappa_0\,\sigma^2}.$$

Proof sketch. Induct on n. The recursion $\kappa_n = \kappa_0 + n/\sigma^2$ is solved in closed form, and the recursive μ_n rearranges to the stated convex combination, with Z identifying as the classical Bühlmann credibility factor of [4]. □

B. Linear regression as a gradient learner

Let $A = \mathbb{R}^d$, $B = \mathbb{R}$, learning rate $\eta > 0$. Define $\mathrm{Lin}_\eta \in \mathsf{Learn}(\mathbb{R}^d,\ \mathbb{R})$ by

$$\begin{aligned} P &= \mathbb{R}^d, \\ I(w,\ x) &= w \cdot x, \\ U(w,\ x,\ y) &= w - \eta\,(w \cdot x - y)\,x, \\ R(w,\ x,\ y) &= x - \eta\,(w \cdot x - y)\,w. \end{aligned}$$

The update is the standard gradient step on squared loss $L = \frac{1}{2}(w \cdot x - y)^2$. The request is $\partial L/\partial x = (w \cdot x - y)\,w$, returned in the same form an upstream learner expects: a corrected input.

Remark 9. Cred and Lin are objects of the same category. A composite of one with the other is again a learner whose update integrates Bayesian and gradient steps without ad-hoc glue. The abstraction is doing real work: heterogeneity at the algorithmic level disappears at the categorical one.

IV. Comonadic Governance

A. The Env comonad over a monoid

Definition 10 (Env comonad)**.** For any monoid $(M, \otimes_M, e_M)$ in Set, the *Env comonad* on Set is the endofunctor $W_M(A) := M \times A$ with counit $\varepsilon\colon W_M \Rightarrow \mathrm{id}_{\mathsf{Set}}$ and comultiplication $\delta\colon W_M \Rightarrow W_M W_M$ given on components by

$$\varepsilon_A(m, a) = a, \quad \delta_A(m, a) = (m,\ (m, a)).$$

The comonad axioms are the commuting diagrams of Figure 2: the two upper diagrams express that ε is a counit for δ, and the lower diagram expresses coassociativity. For $W = W_M$ each square is an immediate unfolding of Theorem 10 together with the unit and associativity laws of the monoid M.

Remark 11 (Functoriality of $W_{(-)}$). The assignment $M \mapsto W_M$ extends to a functor $W_{(-)}$ from the category $\mathsf{Mon}(\mathsf{Set})$ of monoids in Set to the category $\mathsf{CoMon}(\mathsf{Set})$ of comonads on Set: a monoid homomorphism $h\colon M \to N$ induces a comonad morphism $W_M \Rightarrow W_N$ given pointwise by $h \times \mathrm{id}$. This is the sense in which choosing a richer carrier monoid (more rules, layered context) is functorial in the carrier itself.

B. Governance objects as a monoid

Definition 12 (Governance)**.** For a fixed proposal type P, set

$$\mathsf{Governance}(P) := \mathrm{List}(\mathsf{Rule}_P) \times \mathrm{List}(\mathrm{Tag})$$

where $\mathsf{Rule}_P := P \to (\mathbf{1} + \mathsf{Violation})$ is the type of governance rules ($\mathrm{inl}(*)$ encodes "no violation" on a proposal, $\mathrm{inr}(v)$ a specific violation v).

Lemma 13. $\mathsf{Governance}(P)$ *is a monoid under componentwise list concatenation, with identity $(\varnothing, \varnothing)$. We denote the operation $\oplus$.*

We write $\mathsf{Governed}_P := W_{\mathsf{Governance}(P)}$ for the Env comonad over this monoid: a value of $\mathsf{Governed}_P A$ is a pair of a governance environment and a focal value of type A.

C. Validation under composed governance

Definition 14 (Satisfaction)**.** A proposal $p \in P$ *satisfies* a governance object $G \in \mathsf{Governance}(P)$, written $p \models G$, when every rule $r \in G$ has $r(p) = \mathrm{inl}(*)$.

Lemma 15 (Conjunctivity of governance composition)**.** *For all $p \in P$ and $G_1, G_2 \in \mathsf{Governance}(P)$,*

$$p \models G_1 \oplus G_2 \quad \Longleftrightarrow \quad p \models G_1 \text{ and } p \models G_2.$$

Proof. The rule list of $G_1 \oplus G_2$ is the concatenation of the rule lists of G_1 and G_2; the universal property "every rule returns inl" distributes over concatenation. □

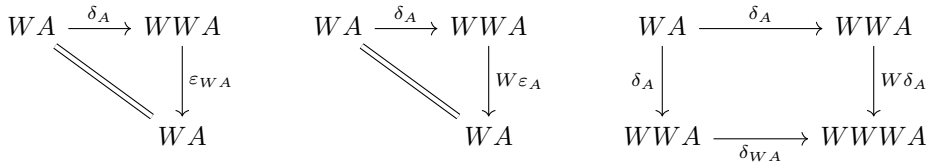

Fig. 2. Comonad axioms for W: left counit (left), right counit (centre), and coassociativity (right).

V. Validation as a Co-Kleisli Arrow

Recall that the co-Kleisli category $\mathsf{coKl}(W)$ of a comonad W on $\mathcal{C}$ has the same objects as $\mathcal{C}$, while a morphism $A \to B$ in $\mathsf{coKl}(W)$ is a map $WA \to B$ in $\mathcal{C}$. Composition lifts via δ, and the identity at A is ε_A.

Let $\mathsf{Contract}(P)$ be a type whose data constructor is private and whose only public introduction rule is the partial function

$$\mathsf{validate}\colon \mathsf{Governed}_P P \longrightarrow \mathrm{List}(\mathsf{Violation}) + \mathsf{Contract}(P). \tag{1}$$

That is, validate is a morphism $P \to \mathrm{List}(\mathsf{Violation}) + \mathsf{Contract}(P)$ in $\mathsf{coKl}(\mathsf{Governed}_P)$. Concretely, $\mathsf{validate}(g)$ applies each rule of $\mathrm{governance}(g)$ to $\varepsilon(g)$. When every rule returns $\mathrm{inl}(*)$ the focal proposal is wrapped as a contract; otherwise the collected failing-rule violations are returned in the left summand.

Theorem 16 (Static governance)**.** *Suppose* $\mathsf{Contract}(P)$ *has its data constructor hidden and is exposed only via* validate*. Then for every* $c \in \mathsf{Contract}(P)$ *there exists* $g \in \mathsf{Governed}_P P$ *with* $\varepsilon_P(g) \models \mathrm{governance}(g)$*. That is: every contract that exists has been admitted by some governance environment that it satisfies.*

Proof. By construction, each value $c \in \mathsf{Contract}(P)$ arises as the right summand of $\mathsf{validate}(g)$ for some $g \in \mathsf{Governed}_P P$. By Equation (1), this is precisely the case in which every rule of $\mathrm{governance}(g)$ returns $\mathrm{inl}(*)$ on $\varepsilon_P(g)$, i.e. when $\varepsilon_P(g) \models \mathrm{governance}(g)$. □

Theorem 16 is the categorical content of the abstraction barrier on $\mathsf{Contract}(P)$: the impossibility of constructing a contract outside of validate becomes a property of the type system rather than a runtime invariant. The development of this section treats decisions as valued in $\mathbf{1} + \mathsf{Violation}$ — a binary admit/violate alternative; Section VI generalises this to decisions valued in an arbitrary monoid, and recovers the present treatment as the case of the free monoid on violations.

VI. Decision Systems Parameterised by a Monoid

The development of Sections IV and V treats the carrier of the governance comonad as a list of *rules* valued in $\mathbf{1} + \mathsf{Violation}$. This is the case the production implementation most closely tracks. The categorical structure is considerably more general, however, and recognising the generalisation unifies governance with adjacent industrial notions — particularly the *guardrails* of insurance underwriting, in which decisions yield risk scores, severity flags, or graded recommendations rather than binary admit/reject outcomes.

A. Decisions valued in a monoid

Definition 17 (Decision)**.** Let $(M, \otimes_M, e_M)$ be a monoid in Set. A *decision valued in* M on proposals of type P is a function $d\colon P \to M$.

Definition 18 (Decision system)**.** A *decision system* on P valued in M is a finite list $D = [d_1, \ldots, d_n]$ of decisions $d_i\colon P \to M$. Its *aggregate* is the decision

$$\langle D \rangle(p) := d_1(p) \otimes_M \cdots \otimes_M d_n(p),$$

with $\langle [\,] \rangle(p) := e_M$.

The set of decision systems on P valued in M, denoted $\mathcal{D}_M(P)$, is itself a monoid under list concatenation with identity $[\,]$ — exactly as in Theorem 13. The Env comonad $W_{\mathcal{D}_M(P)}$ carries a decision system as context for a focal proposal, and the co-Kleisli arrow

$$\mathrm{eval}_M\colon W_{\mathcal{D}_M(P)} P \longrightarrow M, \quad \mathrm{eval}_M(D, p) := \langle D \rangle(p),$$

evaluates the system on the focal proposal.

B. Admissibility

Definition 19 (Admissibility)**.** An *admissibility predicate* for decisions valued in M is a map $\mathrm{adm}\colon M \to \{\bot, \top\}$.

Definition 20 (Generalised validation)**.** Let $\mathsf{Contract}_M(P)$ be a type with private data constructor. Define

$$\mathsf{validate}_{M,\mathrm{adm}}\colon W_{\mathcal{D}_M(P)} P \longrightarrow M + \mathsf{Contract}_M(P)$$

to return $\mathrm{inr}(\mathsf{Contract}_M(p))$ when $\mathrm{adm}(\langle D \rangle(p))$ holds and $\mathrm{inl}(\langle D \rangle(p))$ otherwise.

Theorem 21 (Generalised static admissibility)**.** *If* $\mathsf{Contract}_M(P)$ *has its data constructor hidden and is exposed only via* $\mathsf{validate}_{M,\mathrm{adm}}$*, then for every* $c \in \mathsf{Contract}_M(P)$ *there exist a decision system* $D \in \mathcal{D}_M(P)$ *and a proposal* $p \in P$ *such that* $\mathrm{adm}(\langle D \rangle(p)) = \top$.

Proof. Each c arises as the right summand of $\mathsf{validate}_{M,\mathrm{adm}}(D, p)$ for some (D, p), which by Theorem 20 occurs exactly when $\mathrm{adm}(\langle D \rangle(p))$ holds. □

Theorem 21 subsumes Theorem 16: it is adm, not the structure of M, that gates the abstraction barrier on Contract.

C. Worked instances

Governance.: Take $M := \mathrm{List}(\mathsf{Violation})$ with $\otimes = +\!+$ and $e = [\,]$, and $\mathrm{adm}(m) := (m = [\,])$. A decision $d\colon P \to M$ takes the form $p \mapsto [\,]$ on success and $p \mapsto [v]$ on failure with violation v. The aggregate is the concatenation of all violations; adm asks whether the resulting list is empty. This recovers Theorems 12, 15 and 16 as the present specialisation.

Guardrails (additive risk score).: Take $M := \mathbb{R}_{\geq 0}$ with $\otimes = +$ and $e = 0$, and $\mathrm{adm}(s) := (s < s_{\max})$ for a fixed threshold $s_{\max}$. Each decision contributes a non-negative risk increment to a proposal; the aggregate is the total risk; admission requires the total to lie below the threshold. The carried monoid is abelian, so reordering decisions is behaviourally invisible — useful for parallel evaluation across warehouse partitions.

Guardrails (score and worst severity).: Take $M := \mathbb{R}_{\geq 0} \times \Sigma$ for a totally ordered severity lattice Σ with bottom $\bot$, with $\otimes((s_1, \sigma_1), (s_2, \sigma_2)) := (s_1 + s_2,\ \max(\sigma_1, \sigma_2))$ and $e := (0, \bot)$. Set $\mathrm{adm}((s, \sigma)) := (s < s_{\max}) \wedge (\sigma < \sigma_{\max})$. A single critical-severity decision now refuses the contract regardless of total score; soft-but-numerous decisions refuse it via the score threshold.

Joint governance and guardrails.: Take the product monoid $M := \mathrm{List}(\mathsf{Violation}) \times \mathbb{R}_{\geq 0}$ with componentwise operations. Set $\mathrm{adm}((vs, s)) := (vs = [\,]) \wedge (s < s_{\max})$. A proposal is admitted iff no hard rule has fired *and* its risk aggregate is within budget. The decision system holds both kinds of rule on equal algebraic footing; only the admissibility predicate distinguishes them. The framework requires neither prior commitment to which rules are "hard" nor architectural separation between the two flavours.

Visibility (per-query guardrail).: The four instances above gate contract construction at the moment of validation. Two further instances extend $\mathsf{validate}_{M,\mathrm{adm}}$ to operational concerns where "admit/refuse" is parametric in something beyond the monoid value.

Let Σ be a finite lattice of *visibility states* (e.g. Erased; $\mathrm{RestrictedTo}(R)$ for $R \subseteq \mathrm{Role}$; $\mathrm{Visible}$) ordered by restrictiveness. Take $M := \Sigma$ with $\otimes := \wedge$ (meet; the more restrictive of two states) and $e := \mathrm{Visible}$. Admission is parametrised by querying role: for each $r \in \mathrm{Role}$, $\mathrm{adm}_r(\sigma) := (\sigma \text{ admits } r)$. A decision system aggregates per-row visibility — an erasure rule contributes Erased for tombstoned rows; a litigation-hold rule contributes $\mathrm{RestrictedTo}(\ldots)$; the consumer-facing view for role r is $\mathsf{validate}_{M,\mathrm{adm}_r}$ restricted to admitted rows. Theorem 21 applies separately to each adm_r, and the abstraction barrier on the consumer-facing surface holds for that role.

In the production system this recovers the right-to-erasure mechanism: the view-layer filter `WHERE erased = false` is $\mathsf{validate}_{M,\mathrm{adm}_{\mathrm{consumer}}}$, with the privacy-officer role admitting the strictly more restrictive states for audit purposes.

Labelling (degenerate admission).: Take any monoid $(M, \otimes, e)$ and set $\mathrm{adm}(m) := \top$ for all m. $\mathsf{validate}_{M,\top}$ always returns $\mathrm{inr}(\mathsf{Contract}_M(p))$, with the carried $\langle D \rangle(p)$ persisted as *label* rather than *gate*.

The paradigmatic instance is field classification of personally identifiable information: take $M := \mathrm{Field} \to \mathrm{Sensitivity}$ with per-field most-restrictive merge as $\otimes$ and the constant-everywhere-non-sensitive function as e. Each classification rule is a decision $d\colon \mathrm{Schema} \to M$ contributing partial classifications; the aggregate is the merged classification of all fields across all rules. Theorem 21 holds vacuously and carries no admissibility content; the categorical structure justifies treating classification as a first-class decision system that composes with other systems and participates in the same replay arithmetic as Section VII's governance bundles.

$$\begin{array}{ccc} P & \xrightarrow{(\mathsf{Governance}^{\mathrm{CA}},\ -)} & \mathsf{Governed}_P P \\ & & \Big\downarrow{\scriptstyle \mathsf{validate}} \\ & & \mathrm{List}(\mathsf{Violation}) + \mathsf{Contract}(P) \end{array}$$

Fig. 3. Layered validation pipeline: the proposal is injected into the comonadic context carrying the composed governance, then validated.

D. Connection to industrial notations

The product-monoid construction of the previous paragraph is the categorical engine behind layered, scored decision-making in industrial settings. The OMG's Decision Model and Notation (DMN) expresses individual decisions as tables, with hit policies that correspond directly to choices of M: `Unique` is a partial function $P \rightharpoonup M$ for arbitrary M; `Collect` with sum aggregator is $M = \mathbb{R}$; `Output Order` is $M = \mathrm{List}(O)$ for some output type O. Decision Requirements Diagrams compose tables along the same lines as $\langle - \rangle$: a final decision aggregates the outputs of upstream decisions in a DAG, and the DAG's evaluation is a co-Kleisli composite. The Friendly Enough Expression Language (FEEL) supplies cell-level expressions; semantically these are total functions on the proposal record into M.

We do not pursue a formal categorical translation of DMN here. We note that the alignment is structurally tight: the monoid-parameterised formulation of this section is a faithful semantics for DMN's hit-policy algebra, and DMN tables are a notation for individual decisions in $\mathcal{D}_M(P)$.

VII. Layered Regulation

In practice, governance is layered. We assemble four bundles: $\mathsf{Governance}_{\mathrm{fed}}$ collects federal rules (protected-class rating prohibitions, ACA loss-ratio floor, etc.); $\mathsf{Governance}_{\mathrm{CA}}$ collects California rules (Prop 103 permitted auto factors, minimum auto liability); $\mathsf{Governance}_{\mathrm{int}}$ collects internal guardrails (consent, rate stability, explainability, reinsurance cession); and $\mathsf{Governance}_{\mathrm{base}}$ collects basic underwriting hygiene (positive premium, loss-ratio cap, coverage cap). The composed *California regime* is

$$\begin{aligned} \mathsf{Governance}^{\mathrm{CA}} := \mathsf{Governance}_{\mathrm{fed}} \oplus \mathsf{Governance}_{\mathrm{CA}} \\ \oplus \mathsf{Governance}_{\mathrm{int}} \oplus \mathsf{Governance}_{\mathrm{base}}. \end{aligned}$$

By Theorem 15, $p \models \mathsf{Governance}^{\mathrm{CA}}$ iff p satisfies each layer; by Theorem 16, the resulting contract is then admissible under the full layered regime. The end-to-end pipeline is the composite of co-Kleisli arrows shown in Figure 3.

Switching jurisdiction means substituting $\mathsf{Governance}_{\mathrm{NY}}$ for $\mathsf{Governance}_{\mathrm{CA}}$ at the carrier level. By Theorem 11 this is functorial in the carrier monoid: choices of governance form a diagram in $\mathsf{Mon}(\mathsf{Set})$, lifted by $W_{(-)}$ to a diagram in $\mathsf{CoMon}(\mathsf{Set})$, and $\mathsf{validate}$ is natural with respect to this lifting.

VIII. Outlook

Three directions extend the present work.

Para alignment.: Sequential composition of learners embeds into the bicategory of optics via the Para construction [3]; making this explicit puts lens combinators (composition lemmas, profunctor optics) directly at the disposal of learner state, and identifies the request map R with the optic's residual.

Probabilistic outputs.: Lifting the codomain of I from a point estimate $b \in B$ to a distribution over B allows governance to reason about posterior uncertainty rather than only the mean. The Giry monad on Set supplies the categorical setting, and rules can then phrase constraints such as "the expected loss has variance below $\sigma^2_{\max}$" instead of point estimates.

Layered comonads.: The composition in Section VII happens at the parameter level of a fixed comonad $W_{\mathsf{Governance}(P)}$. A genuine comonad *transformer* would let governance regimes nest as comonad layers (federal $\triangleright$ state $\triangleright$ product line), so that re-audit at each layer is itself a co-Kleisli arrow over a layered comonad. This is the natural place at which the comonadic structure of $\mathsf{Governed}$ begins to do work beyond what its Env-monoid carrier alone provides.

References

[1] B. Fong, D. I. Spivak, and R. Tuyéras, "Backprop as functor: A compositional perspective on supervised learning," *arXiv preprint arXiv:1711.10455*, 2017.

[2] G. S. H. Cruttwell, B. Gavranović, N. Ghani, P. Wilson, and F. Zanasi, "Categorical foundations of gradient-based learning," *arXiv preprint arXiv:2103.01931*, 2021.

[3] M. Capucci, B. Gavranović, J. Hedges, and E. Rischel, "Towards foundations of categorical cybernetics," *arXiv preprint arXiv:2105.06332*, 2021.

[4] H. Bühlmann and A. Gisler, *A Course in Credibility Theory and its Applications*, ser. Universitext. Springer, 2005.

[5] T. Uustalu and V. Vene, "Comonadic notions of computation," *Electronic Notes in Theoretical Computer Science*, vol. 203, no. 5, pp. 263–284, 2008.