A Higher Effective Topos for Certified Stochastic Optimization

Anonymous

Abstract. We present the higher regular exact completion of Hyland's effective topos via bounded coherent objects in a Grothendieck envelope and recover Hyland's topos as its category of zero-types. Equivariant rational optimizer runs descend through action-groupoid quotients. The examples include stochastic gradient descent, sampled curvature, a third-derivative Halley step, and an undercomplete autoencoder with trainable encoder and decoder. The autoencoder is evaluated on synthetic images and held-out Iris measurements. A Lean 4 artifact checks typed arithmetic and autoencoder compilation, finite stochastic law equivariance, directional jets, and certificates for explicit 1- and 2-cell presentations. The categorical construction and its interpretation are proved separately in the manuscript.

Keywords. Effective topos, realizability, higher exact completion, action groupoids, stochastic optimization, jets, Lean.

1 Introduction

Hyland's effective topos interprets maps on its natural numbers object as recursively tracked operations [1]. Its categorical structure distinguishes an arbitrary externally described transformation from one admitting uniform realizers. The higher question considered here is how to retain that distinction while admitting nontrivial paths between presentations of data, state and algorithms. In the examples, a path renames coordinates and transports the input samples, state and output together. A fixed state can have a nontrivial loop. Recording its orbit alone erases the corresponding automorphism.

Higher realizability has several established formulations. Awodey and Emmenegger construct a candidate effective 2-topos from coherent groupoids and recover the usual effective topos on zero-types [2]. Anel, Awodey and Barton describe a higher completion using partitioned assemblies [3]. Maruyama develops a higher tripos framework and a Kleene-assembly instance with an explicit universe hierarchy [4]. We use the higher regular exact completion of the ordinary effective topos, identifying it with a bounded coherent part of a sheaf category. The method keeps the sheaf category, its ordinary sheaves and the effective objects distinct. The construction applies a general theorem of Stefanich [5].

Our contribution is to connect this categorical presentation to tracked stochastic computations and to supply executable certificates for a fragment. On representatives, an optimizer is an arithmetic program with explicitly supplied finite samples. Equivariance is the condition that this program and its arithmetic output can be transported on the quotient. A second condition concerns the sampling law. It is possible for the program to commute with every relabeling while a choice of weights breaks that symmetry.

Three interpretations appear below. The outer category provides effective groupoid quotients and higher identities. The optimization semantics uses finite rational distributions and quotient descent. The Lean artifact verifies expression and path compilation against explicit models and proves structural identities about executable optimizers. In particular, a finite derivation of an equation about path labels and a proof of correctness of the entire higher exact completion are different verification tasks.

2 The bounded higher effective completion

Write $\mathbf{Eff}_{\mathcal{U}}$ for Hyland's effective topos with carriers in a Grothendieck universe $\mathcal{U}$ and take a larger universe $\mathcal{V}$ containing a presentation of this category. We use ordinary exactness of $\mathbf{Eff}_{\mathcal{U}}$, its finite extensive coproducts, and the effective natural-number object N_{eff}. Recall that an assembly is a set A with a nonempty set $\rho_A(a) \subseteq \mathbb{N}$ for each $a \in A$. A function $f : A \to B$ is a map of assemblies if some index e satisfies

$$n \in \rho_A(a) \Longrightarrow \varphi_e(n) \downarrow \in \rho_B(f(a)).$$

Here φ_e is Kleene application. One can obtain $\mathbf{Eff}_{\mathcal{U}}$ by exact completion of the regular category of assemblies, or by the exact/lex completion of partitioned assemblies [1, 2]. The assembly N_{eff} has $\rho(n) = \{n\}$, so all its endomorphisms in $\mathbf{Eff}_{\mathcal{U}}$ are exactly the total recursive functions.

2.1 An ambient site and the higher completion

Equip $C = \mathbf{Eff}_{\mathcal{U}}$ with the topology J_{fin} whose covering sieves contain a finite family $(U_i \to U)_i$ with $\coprod_i U_i \to U$ a regular epimorphism. The empty family covers the initial object. This is a Grothendieck topology because regular epimorphisms and finite coproducts are stable

under pullback, and composites of finite jointly epimorphic families are finite jointly epimorphic [7, Prop. A.3.2.1]. We use this specified finitary topology with no assumption about arbitrary external covering families. Set

$$\begin{aligned}\mathcal{X} &= \mathrm{Shv}_{J_{\mathrm{fin}}}(C;\mathcal{S}_{\mathcal{V}}),\\ \mathcal{E} &= \{X \in \mathcal{X} :\ X \text{ is coherent and}\\ &\qquad\qquad n\text{-truncated for some finite } n\}.\end{aligned}$$

The site gives an accessible left exact sheafification of space-valued presheaves at the covering-sieve inclusions [6, Prop. 6.2.2.7]. An object X is coherent when $\mathcal{X}_{/X}$ is coherent in every degree [7, Def. A.2.1.6]. In particular, X is quasi-compact, meaning every family jointly covering it by effective epimorphism has a finite subfamily with that property. The truncation bound can vary with X.

Lemma 2.1. *The sheaf Yoneda functor $j : C \to \mathcal{X}$ is fully faithful and preserves finite limits, finite coproducts, and regular epimorphisms, with regular epimorphisms viewed as effective epimorphisms in $\mathcal{X}$. Its image consists of coherent 0-truncated objects.*

Proof. For each finite covering family with copairing $p : \coprod_i U_i \to U$, the coproduct is disjoint and its pullbacks distribute over the summands. Consequently ordinary representable matching families are exactly the maps out of the domain of p equalizing its kernel pair. Their amalgamations are unique by effectiveness of p. Discrete spaces are closed under limits, so the same representables satisfy descent as space-valued presheaves. Full faithfulness and preservation of finite limits follow from Yoneda. A covering family induces a jointly effective epimorphic family of representables: its covering sieve is the presheaf image of the copairing, and sheafification identifies that sieve with its target. The inclusion of each summand of a finite coproduct in C is such a cover. The resulting map $\coprod_i jU_i \to j(\coprod_i U_i)$ is also monic by disjointness, finite-limit preservation, and $j0 \simeq 0$ from the empty covering. It is therefore an equivalence. Coherence of representables follows from the finitary site theorem [7, Prop. A.3.1.3]. □

Theorem 2.2 (Recovery and realization)**.** *The full subcategory $\mathcal{E} \subseteq \mathcal{X}$ is a bounded exact ∞-category with finite disjoint universal coproducts. The functor j induces equivalences*

$$C \simeq \mathcal{E}_{\leq 0}, \qquad \mathrm{Ex}^{\mathrm{reg}}_{\infty}(C) \simeq \mathcal{E},$$

where the second equivalence is the higher exact completion of the regular category C in the sense of Stefanich. Finite limits and groupoid quotients in $\mathcal{E}$ are computed in $\mathcal{X}$.

Proof. Coherent objects of $\mathcal{X}$ contain the terminal object and are closed under finite limits, finite coproducts, and geometric realization of groupoid objects [7, Props. A.3.1.3 and A.6.1.6]. These operations preserve boundedness. For groupoid realization this last assertion follows by pulling the diagonal of $Y = |G_\bullet|$ back along the effective epimorphism $G_0 \times G_0 \to Y \times Y$: its pullback is the map $G_1 \to G_0 \times G_0$, which is finitely truncated, and truncation descends along effective epimorphisms [5, Rem. 2.3.10]. Thus $\mathcal{E}$ is exact, and its finite coproducts are disjoint and universal. A map between objects of $\mathcal{E}$ has the same truncation in either category. Indeed, truncatedness of a map is detected inductively via its diagonals, which are computed by finite limits, and equivalences are reflected by the full inclusion.

Representables generate sheaves under colimits and give an effective epimorphism $\coprod_i jU_i \to Y$ for any $Y \in \mathcal{X}$. For example, take one arrow for each component of each $Y(U_i)$. If Y is coherent this cover reduces to a finite one by quasi-compactness, and its source can be combined into one jU by Lemma 2.1. In particular every object of $\mathcal{E}$ has a 0-truncated effective epimorphic cover.

Let $Y \in \mathcal{E}$ also be 0-truncated and choose $jU \twoheadrightarrow Y$. Its kernel pair $R = jU \times_Y jU$ is coherent and is a subobject of $j(U \times U)$. Choose $jV \twoheadrightarrow R$ as above. The composite $jV \to j(U \times U)$ comes from a morphism of C, and its image is representable because j preserves monos and regular epimorphisms. That image is R. The resulting relation $R_0 \hookrightarrow U \times U$ in C is an equivalence relation by full faithfulness and finite-limit preservation. Its quotient $q : U \twoheadrightarrow Q$ in C has kernel pair R_0. Since jq is effectively epimorphic and has kernel pair R, uniqueness of the quotient gives $Y \simeq jQ$. This identifies the full category of 0-types of $\mathcal{E}$. Finally, the exact, bounded category $\mathcal{E}$ satisfies the 0-complicial hypothesis of [5, Prop. 2.3.11(2)], which gives the equivalence via the regular embedding of its 0-types. □

The ambient 0-types are the ordinary sheaves of sets on (C, J_{fin}). They include $\coprod_{k\in\mathbb{N}} 1$, a 0-type whose coproduct injections form a cover with no finite subcover. Thus the ambient object is outside $\mathcal{E}$. The same coproduct gives the ambient natural-number object. In C, any countable coproduct of 1 would be a natural-number object via the shift and recursion on the summands. Its endomorphisms would then include every external function $\mathbb{N} \to \mathbb{N}$, contradicting effective tracking [2, §1]. This obstruction persists when the sheaf universe is enlarged. For fixed C, the equivalence of Theorem 2.2 identifies the intrinsic higher completion independently of the choice of a larger sheaf universe. Changing the carrier universe changes the input category C.

Corollary 2.3. *Let P denote the category of partitioned assemblies in $\mathcal{U}$. The composite $P \to C \to \mathcal{E}$ exhibits $\mathcal{E}$ as the higher exact completion of the finite-limit category P for left exact functors.*

Proof. For an exact ∞-category $\mathcal{D}$, its 0-truncated part $\mathcal{D}_{\leq 0}$ is ordinary exact. For instance the quotient of an ordinary equivalence relation $R \hookrightarrow A \times A$ is still 0-truncated, since the pullback of its diagonal along the square of its atlas $A \to |R_\bullet|$ is the monomorphism $R \hookrightarrow A \times A$. A left exact functor from the ordinary category P factors through $\mathcal{D}_{\leq 0}$ by the diagonal criterion for truncation, and regular functors from C to $\mathcal{D}$ factor regularly through the

same part. Hence we have the natural equivalences

$$\begin{aligned}\mathrm{Fun}_{\mathrm{reg}}(\mathcal{E},\mathcal{D}) &\simeq \mathrm{Fun}_{\mathrm{reg}}(C,\mathcal{D}_{\leq 0})\\ &\simeq \mathrm{Fun}_{\mathrm{lex}}(P,\mathcal{D}_{\leq 0})\\ &\simeq \mathrm{Fun}_{\mathrm{lex}}(P,\mathcal{D}).\end{aligned}$$

The first equivalence is Theorem 2.2 with [5, Thm. 2.2.10]. The second is the ordinary exact/lex universal property of C [2, §2]. These equivalences act by restriction along the displayed inclusions. □

2.2 Logical structure used by the computational fragment

For $f : A \to B$ and $g : Z \to A$ in C, the ambient dependent product of jg along jf is represented by $j(\Pi_f Z \to B)$. In fact for every represented test object $jT \to jB$,

$$\begin{aligned}\mathrm{Map}_{/jB}(jT, \Pi_{jf} jZ) &\simeq \mathrm{Map}_{/jA}(j(T \times_B A), jZ)\\ &\simeq \mathrm{Hom}_{C/B}(T, \Pi_f Z).\end{aligned}$$

These test objects detect equivalences of sheaves over jB. Since $\mathcal{E}$ is full, it also has these dependent products on represented maps. Represented products, equalizers, images and dependent products therefore interpret the corresponding first-order operations on represented predicates. Arithmetic in the computational fragment uses jN_{eff} and recursion on represented data from C. In particular,

$$\mathrm{End}_{\mathcal{E}}(jN_{\mathrm{eff}}) \cong \mathrm{End}_C(N_{\mathrm{eff}}),$$

with effective tracking as above. Bounded coherent objects also admit the ambient truncations by [7, Cor. A.2.4.4]. The ambient topos supplies the usual slice semantics by pullback, including dependent types, and the higher completion provides the exact subcategory used for the examples. This gives a categorical interpretation of represented dependent products. The Lean implementation checks the concrete numerical and cell syntax.

2.3 Action quotients and isotropy

Let a group object H of C act on A by computably tracked operations. Its action nerve is a groupoid object with degree m given by $j(A \times H^m)$. Its realization $[jA/jH]$ belongs to $\mathcal{E}$ and is 1-truncated. To see the truncation bound, the diagonal pulls back along the product of the atlas $jA \twoheadrightarrow [jA/jH]$ with itself to $j(A \times H) \to j(A \times A)$, a 0-truncated map. The same pullback identifies the loop object at a global state a with the represented stabilizer

$$\Omega_{[a]}[jA/jH] \simeq j\{h \in H \mid ha = a\}.$$

An equivariant update u gives a map on the quotient by applying u to the object coordinate and leaving the group coordinate unchanged in every simplicial degree. Those maps commute with the face and degeneracy maps by equivariance and the group laws. In particular the induced arrow on loops at a state sends the stabilizer of a into the stabilizer of $u(a)$, preserving composition of symmetries. This is how finite symmetry witnesses behave in the optimization semantics.

At $A = 1$ and $H = C_2$, the quotient is BC_2 and $1 \times_{BC_2} 1 \simeq jC_2$. Ordinary orbit collapse yields $\tau_{\leq 0} BC_2 \simeq 1$. In particular reflection at all maps inducing an isomorphism on 0-truncation would retain only the ordinary 0-types. That reflection fails to preserve the displayed pullback, so it cannot furnish the left exact localization used above.

The bounded completion supports higher types in every finite dimension. For a discrete abelian group object M, start with jM and iterate the bar construction. Each bar realization of a commutative group object inherits the commutative group structure since finite products commute with these realizations in $\mathcal{X}$, hence also in $\mathcal{E}$. The identification $\Omega BK \simeq K$ follows from groupoid effectiveness. For $M = C_2$ this constructs an n-type $B^n jC_2$ in $\mathcal{E}$ with $\Omega^n B^n jC_2 \simeq jC_2$ for every finite $n \geq 1$. The executable optimization examples below use the first delooping level through action groupoids.

2.4 Position in higher realizability

Awodey and Emmenegger recover **Eff** as the 0-types of a candidate effective 2-topos presented by coherent groupoids [2, Cor. 18]. Anel, Awodey, and Barton describe a bounded higher exact/lex completion of partitioned assemblies in terms of coherent presheaves of spaces using truncated groupoid presentations [3]. The regular exact completion used in Theorem 2.2 comes with the universal property of Stefanich's construction. Maruyama's preprint identifies that higher exact completion of **Eff** with the higher exact completion of Kleene assemblies, and for a version with inaccessible carrier bounds proves further local cartesian closure and universe results [4, §5]. These results place the calculation of quotients in the higher effective-topos program. The presentation here uses a coherent sheaf envelope to compute the bounded effective completion and relate its action quotients to explicit checked optimization programs. Cubical assemblies supply a different approach to univalence; Church's thesis for their natural-number object requires a further modality in the model of Swan and Uemura [8].

3 Optimization with effective samples and transport

Let $j : C \to \mathcal{E} \subseteq \mathcal{X}$ be the embedding from Theorem 2.2. We write Q for the image under j of the effective rational ring, presented by reduced fractions with a positive denominator. Its arithmetic and order are realized by integer algorithms. Write $V = Q^d$ for a fixed finite coordinate space. The derivatives below are formal derivatives of rational polynomials and the expectations are finite rational sums. These choices give exact computations stable under pullback.

3.1 Finite stochastic execution

Let I be an inhabited finite sample set, presented by a coproduct of copies of 1. A specified weight function $p : I \to Q$ satisfies $p_i \geq 0$ and $\sum_i p_i = 1$. Given a state object S and an update $T : I \times S \to S$, define

$$R_0(s) = s,$$
$$R_{t+1}(\mathbf{i}, i; s) = T(i, R_t(\mathbf{i}; s)).$$

For a fixed horizon n, the update is a deterministic map $R_n : I^n \times S \to S$. The stochastic law is obtained by assigning a schedule $\mathbf{i} \in I^n$ the weight $p_{\mathbf{i}} = \prod_{t<n} p_{i_t}$ and applying R_n. In particular, for an observable $o : S \to Q^m$ its expected value is

$$\mathbb{E}_p[o(R_n)] = \sum_{\mathbf{i} \in I^n} p_{\mathbf{i}}\, o(R_n(\mathbf{i}; s)).$$

This is a finite sum even when multiple schedules produce the same state. One can supply an observed schedule to replay a run or enumerate all schedules to calculate the finite law. The weights specify the experiment, and a physical random source would supply its draws. Schedule enumeration grows as $|I|^n$.

Let $\ell_i : V \to Q$ be polynomial losses, $L = \sum_i p_i \ell_i$, and $\eta \in Q$ a step size. The sample-gradient update is

$$T_i(x) = x - \eta \nabla \ell_i(x).$$

At every history $\mathbf{i} \in I^t$, set $z = R_t(\mathbf{i}; x_0)$. The prescribed law for the next draw yields

$$\sum_{i \in I} p_i \nabla \ell_i(z) = \nabla L(z),$$
$$\sum_{i \in I} p_i T_i(z) = z - \eta \nabla L(z).$$

These identities follow by linearity of formal differentiation. They express unbiasedness at a specified state and apply even to a history of weight zero, since the conditional expression here is the explicitly supplied next-draw law. This formulation uses the finite algebra of stochastic approximation [9, 11].

3.2 Quadratic steps and finite moment laws

Write a rational sample loss as

$$\ell_i(x) = \tfrac{1}{2}(x - a_i)^{\mathsf{T}} H_i (x - a_i), \quad H_i^{\mathsf{T}} = H_i.$$

Its gradient is $g_i(x) = H_i(x - a_i)$ and its Hessian is H_i. Our three transition rules are ordinary SGD, heavy-ball momentum [10], and damped sample Newton. In the displayed momentum convention v stores an accumulated gradient:

$$\begin{aligned} \text{SGD}: &\quad x' = x - \eta g_i(x), \\ \text{Momentum}: &\quad v' = \beta v + g_i(x), \qquad x' = x - \eta v', \\ \text{Newton}: &\quad x' = x - \alpha K_i g_i(x). \end{aligned}$$

Here $K_i = H_i^{-1}$ for an invertible sample Hessian. A sample may also declare an active coordinate subspace. In that case the projected Newton rule takes K_i to be the inverse of the restriction of H_i on the active subspace and zero on its orthogonal complement. A diagonal Hessian with positive rational entries on the active coordinates admits this rule by division. For $\alpha = 1$, the update sets the active coordinates to their sample targets and leaves the other coordinates fixed. For momentum, setting $\beta = 0$ recovers SGD on the position and eliminating v gives the genuine two-step relation

$$x_{t+1} = (1 + \beta)x_t - \beta x_{t-1} - \eta g_{i_t}(x_t) \qquad (t \geq 1).$$

Proposition 3.1 (Mean and covariance of finite-horizon SGD)**.** *Put $A_i = I_d - \eta H_i$, $b_i = \eta H_i a_i$, $\bar{H} = \sum_i p_i H_i$, and $u = \sum_i p_i H_i a_i$. Let m_t and C_t be the finite-law mean and covariance of the SGD position at time t. Then*

$$m_{t+1} = (I_d - \eta \bar{H}) m_t + \eta u,$$
$$C_{t+1} = \sum_i p_i A_i C_t A_i^{\mathsf{T}} + \sum_i p_i d_i d_i^{\mathsf{T}},$$
$$d_i = A_i m_t + b_i - m_{t+1}.$$

If $\bar{H}$ has an inverse with $x_* = \bar{H}^{-1} u$, then

$$m_t = x_* + (I_d - \eta \bar{H})^t (x_0 - x_*).$$

Proof. Expand $R_{t+1} = A_i R_t + b_i$ and sum first over the histories, then over the last label. This gives the mean recurrence. In the centered outer product use $R_{t+1} - m_{t+1} = A_i(R_t - m_t) + d_i$. Both mixed terms sum to zero over histories. This yields the covariance recurrence using the product weight of each extended history. Subtract the fixed point x_* from the mean recurrence and iterate to get the final identity. □

When $\bar{H}$ is invertible, a full undamped Newton step for L returns x_*. When each H_i is invertible, the mean of an undamped sample Newton step is $\sum_i p_i a_i$. The two values coincide exactly when $\sum_i p_i H_i a_i = \bar{H} \sum_i p_i a_i$. For example, two equally likely scalar samples with centers $0, 2$ and curvatures $1, 3$ give a mean sample-Newton output of 1 and a full Newton output of $3/2$. Unbiased gradients therefore give no automatic unbiasedness guarantee for the inverse-Hessian direction.

Example 3.2 (Exact rational coordinate optimization)**.** Use three samples

	axis	curvature	target
A	x_1	2	1
B	x_2	4	-2
C	x_1	1	3

with law $(1/4, 1/2, 1/4)$, initial position and momentum zero, and $\eta = 1/4$, $\beta = 1/2$, $\alpha = 1$. We report positions

for a fixed three-draw schedule and for the exact weighted enumeration of all nine two-draw schedules:

Rule	Schedule ABC	$\mathbb{E}[X_2]$
SGD	$(9/8, -2)$	$(145/256, -3/2)$
Momentum	$(23/16, -3)$	$(185/256, -2)$
Newton	$(3, -2)$	$(3/2, -3/2)$

For instance, on ABC the momentum position after AB is $(3/4, -2)$ and its accumulated gradient is $(-1, 8)$. The last gradient is $(-9/4, 0)$, so the accumulated gradient becomes $(-11/4, 4)$, giving the displayed position. For SGD, $\bar{H} = \mathrm{diag}(3/4, 2)$ and $u = (5/4, -4)$. Proposition 3.1 first gives $m_1 = (5/16, -1)$, then the displayed m_2. For Newton, an axis is selected on at least one of the two draws with probability $3/4$. Conditional on such a selection, the last target has mean 2 on the first axis and value -2 on the second axis. The objective's stationary point is $(5/3, -2)$.

3.3 Axis changes and paths of optimizer runs

Let $G = C_2$, acting on $V = Q^2$ by the identity and the coordinate transposition $P(x_1, x_2) = (x_2, x_1)$. A sample transforms by

$$(a_i, H_i) \longmapsto (Pa_i, PH_iP^{\mathsf{T}}).$$

Momentum transforms by $(x, v) \mapsto (Px, Pv)$. Sample labels can also be reindexed by an involution $\sigma : I \to I$, acting componentwise on schedules. Precisely, the action sends the sample data at i to the transformed data at $\sigma(i)$, and sends weight p_i to the weight at $\sigma(i)$. This gives an action on the object D of sample tables, together with an action on $I^n \times S$. The labelled input retains the transformed table, so an equivariant run is well-defined even for asymmetric data.

Proposition 3.3 (Run equivariance and descent)**.** *Each of the three quadratic algorithms gives a G-equivariant map $D \times I^n \times S \to S$ for every finite n. Consequently each induces a map over BG between the corresponding homotopy quotients in $\mathcal{E}$. The finite law, as a weighted indexed family, and the expected position commute with the same action.*

Proof. Write $P^{\mathsf{T}}P = I_2$. The gradient at Px for the transformed sample is $Pg_i(x)$, because $PH_iP^{\mathsf{T}}(Px - Pa_i) = PH_i(x - a_i)$. For an invertible Hessian, $(PH_iP^{\mathsf{T}})^{-1} = PH_i^{-1}P^{\mathsf{T}}$. For projected Newton, the same formula applies to K_i because P permutes the active and complementary coordinate subspaces. Substitution in the three step rules gives equivariance. Induction over the schedule proves the run statement. The construction is also valid for the identity action element.

The action groupoid of an object Y has k-simplices $G^k \times Y$. The equivariant run induces a map of these nerves by retaining the G-coordinates and applying the run to the object coordinate. Geometric realization gives the map on homotopy quotients. The projections to the action nerve of 1 make this a map over $BG = 1 /\!/ G$. These quotients are computed in $\mathcal{X}$. The projections $G^k \times Y \to G^k$ form a Cartesian transformation of simplicial diagrams, since the action maps are invertible. Descent [6, Theorem 6.1.3.9(4)] gives $1 \times_{BG} (Y /\!/ G) \simeq Y$ by pulling the augmented transformation back to simplicial degree zero. Relabelling permutes the schedule indices and preserves the matched products of weights. Linearity of P then commutes with the finite sum for expected position. □

The arrow on optimizer runs transports every intermediate state, since equivariance holds for every prefix. For the three-sample SGD run of Example 3.2, transport with the corresponding transformed schedule gives $(-2, 9/8)$. In a chart with one isotropic sample, $a = (1, 1)$, $H = 2I_2$, $x = (2, 2)$, and $\eta = 1/4$, SGD gives $x' = (3/2, 3/2)$. The axis transposition is a nonidentity loop at the input and yields a nonidentity loop at the output. Indeed a point fixed by G has loop object G in the action quotient: the loop object is the fiber of the source–target map $G \times Y \to Y \times Y$ at that point. The orbit set records a single point at either diagonal state, so its induced map would discard the loop. Componentwise proofs of the rational equality and explicit composition of the two path labels serve different roles here.

A fixed source law descends through reindexing by σ exactly when $p_i = p_{\sigma(i)}$ for all i. For instance, take two samples exchanged by P, one on each axis with curvature 2 and target 1. From the origin with $\eta = 1/4$, their SGD outputs are $(1/2, 0)$ and $(0, 1/2)$. Keeping weights $(3/4, 1/4)$ tied to the old labels gives mean $(3/8, 1/8)$, which the exchange sends to $(1/8, 3/8)$. Transporting the weights together with the samples restores equivariance.

3.4 A third-derivative update

The degree-k rational jet of a polynomial loss at x is the class of $\ell(x + u)$ in $Q[u_1, \ldots, u_d]/(u)^{k+1}$. An invertible linear chart change P acts by substitution on the truncated polynomial. In particular, for $\ell^P = \ell \circ P^{-1}$ there is an identity

$$J_k\ell^P(Px)(Pu) = J_k\ell(x)(u).$$

It follows directly by substituting $P^{-1}(Px + Pu) = x + u$ before truncation. Any equivariant rational rule constructed from these jets can replace the quadratic step in the action-groupoid construction of Proposition 3.3.

For a concrete third-derivative rule, use a separable sum of one-variable quartics with parameters $f(x) = \frac{h}{2}(x - a)^2 + \frac{r}{4}(x - a)^4$, where $h > 0$ and $r \geq 0$ are rational. Apply Halley's root rule [12, §3.8(v), Eq. 3.8.12] to the derivative f', one coordinate at a time. For $z = x - a$,

$$g = hz + rz^3, \quad g' = h + 3rz^2, \quad g'' = 6rz,$$
$$z' = z - \frac{2gg'}{2(g')^2 - gg''},$$
$$2(g')^2 - gg'' = 2h^2 + 6hrz^2 + 12r^2z^4 > 0.$$

Thus every sample step is defined over the rational numbers. Set $q = rz^2/h$. After simplification,

$$z' = z\frac{q(3q-1)}{1+3q+6q^2}.$$

For $q \geq 0$, $2|3q^2 - q| < 1 + 3q + 6q^2$: if $3q^2 - q \geq 0$, the difference is $1 + 5q$, and otherwise it is $1 + q + 12q^2$. Hence $|z'| \leq \frac{1}{2}|z|$, with strict inequality for $z \neq 0$, for every fixed sample coordinate. For sample center $(1, 0)$, pairs $(h, r) = (2, 1), (1, 2)$ in the respective coordinates, and input $(2, -1)$, the output is $(17/16, -10/31)$. Swapping the axes also swaps the two parameter pairs, so the transported output is $(-10/31, 17/16)$. This contraction concerns single-sample errors around their own centers. A stochastic sequence can move those centers between steps.

The executable jet example uses $f(x) = x^4 + 2x^2 - 3x + 1$. At $x = 1$, its jet coefficients through order four are $(1, 5, 8, 4, 1)$. The first three derivative values are $5, 16, 24$. The stationarity update gives $1 - 160/392 = 29/49$, computed exactly by the Lean artifact.

3.5 A two-dimensional coherence condition

We can impose a coherence condition on the optimizer of Proposition 3.3 one degree higher. Let $B = K(C_2, 2)$ denote the constant object obtained by applying the inverse image $\mathcal{S} \to \mathcal{X}$ of the global sections geometric morphism to the indicated Eilenberg–Mac Lane space. The represented finite group C_2 lies in $\mathcal{E}$. Its classifying object BC_2 is the image of the abelian group object $K(C_2, 1)$ in spaces. The second classifying object B lies in $\mathcal{E}$ by two applications of closure under groupoid realization. The basepoint map $q : 1 \to B$ is an effective epimorphism and its homotopy fiber over the basepoint is $K(C_2, 1)$. These claims follow in spaces because the inverse image preserves finite limits and colimits. Since $\Omega^2 B \simeq C_2$ and $\mathcal{X}$ is nondegenerate, B fails to be terminal. The map q admits no global section: a section would make q an equivalence because its source is terminal.

For a Newton step with fixed sample, consider the extra condition that a genuine change of position carry a point of q's fiber over the chart $b \in B$. Formally, over $B \times V \times V$, the certificate object is terminal in each fiber over the decision locus $x' = x$. Over the complementary locus $x' \neq x$, it is the pullback of q along the projection to B. Decidable rational equality gives this partition, and finite coproducts and pullbacks put the certificate object in $\mathcal{E}$. Locally on q, every step admits a certificate. If one input has a nonzero Newton correction, a global certificate for the optimizer family over B would give a section of q upon restricting to that input. The preceding argument rules this out. Thus exact agreement of all coordinate computations still leaves a higher gluing problem for the decorated output. The Lean artifact checks the obstruction on the two distinct 2-cell labels of the strict presentation. The categorical proof above establishes the corresponding obstruction for the certificate family.

4 A trained encoder and decoder

An undercomplete autoencoder learns to reconstruct its input through a latent space of smaller dimension [15, Ch. 14]. Here each input is a signed 2×2 image flattened in row-major order as $x \in X = Q^4$. The coding space is $H = Q^2$. Both are represented rational objects of the effective completion. The two hidden units have input weights $w_i \in Q^4$, bias $b_i \in Q$, and decoder column $v_i \in Q^4$, with a shared output bias $c \in Q^4$. Thus the parameter object is

$$\Theta = (Q^4 \times Q \times Q^4)^2 \times Q^4.$$

Define the following operations, with $\cdot$ the coordinate dot product:

$$\phi(a) = \begin{cases} a & a \geq 0, \\ a/4 & a < 0, \end{cases}$$
$$E_\theta(x)_i = \phi(w_i \cdot x + b_i),$$
$$D_\theta(z) = c + z_1 v_1 + z_2 v_2,$$
$$R_\theta(x) = D_\theta(E_\theta(x)).$$

The decoder output is a signed rational intensity vector. The reconstruction loss is $L_x(\theta) = \frac{1}{2}\|R_\theta(x) - x\|_2^2$. This architecture trains 22 scalars, including both the encoder and the decoder. The visible input and target are the same image.

4.1 Exact backpropagation on an activation region

Fix an input x and write $a_i = w_i \cdot x + b_i$, $z_i = \phi(a_i)$, and $r = R_\theta(x) - x$. Define the implemented slope λ_i to be 1 if $a_i \geq 0$ and $1/4$ otherwise. Backpropagation first computes $s_i = \lambda_i(v_i \cdot r)$. The sample update field has components

$$g_{w_i} = s_i x, \quad g_{b_i} = s_i, \quad g_{v_i} = z_i r, \quad g_c = r.$$

At $a_i = 0$ the chosen slope is part of the update rule. The following derivative result assumes preactivations are nonzero.

Proposition 4.1 (Local loss polynomial)**.** *At a parameter θ with $a_1 a_2 \neq 0$, let $\dot\theta = (\dot w_i, \dot b_i, \dot v_i, \dot c)$ be any rational parameter direction. The linear coefficient of $L_x(\theta + t\dot\theta)$ near $t = 0$ is*

$$\dot c \cdot r + \sum_{i=1}^{2} \left(\dot w_i \cdot g_{w_i} + \dot b_i g_{b_i} + \dot v_i \cdot g_{v_i} \right).$$

The loss in this interval is a rational polynomial of degree at most four.

Proof. Put $\dot a_i = \dot w_i \cdot x + \dot b_i$. If $|t| < \min_i |a_i|/(1 + |\dot a_i|)$, each activation retains its branch and the code is $z_i + t\lambda_i \dot a_i$. The updated residual is $r + tA + t^2 B$, where

$$A = \dot c + \sum_i (z_i \dot v_i + \lambda_i \dot a_i v_i),$$
$$B = \sum_i \lambda_i \dot a_i \dot v_i.$$

The coefficient of t in $\frac{1}{2}(r + tA + t^2B) \cdot (r + tA + t^2B)$ is $r \cdot A$. Substituting $\dot{a}_i$ gives the displayed formula. The coefficients of degrees two through four are

$$\tfrac{1}{2}A \cdot A + r \cdot B, \qquad A \cdot B, \qquad \tfrac{1}{2}B \cdot B.$$

These are local directional jet coefficients of the parameterized loss. □

Write $U_x(\theta) = S(\theta - \eta g_x(\theta))$, where the same scalar function $S : Q \to Q$ is applied to each new parameter. Taking $S = \mathrm{id}$ gives exact rational SGD. The grid variant with denominator $N > 0$ uses

$$S_N(u) = \mathrm{trunc}(Nu)/N$$

with truncation toward zero. Each step evaluates the update field in rational arithmetic before storing the updated parameters on this grid. It differs in each coordinate from the corresponding unrounded update by less than $1/N$. This controls denominator growth and specifies the training rule used when grid updates are enabled. The branch test, quotients, and rounding are all computable on the reduced-fraction representation of Q.

The Lean program passes runtime parameter vectors and features to the typed arithmetic compiler of Section 5. Its bytecode computes affine preactivations, decoder outputs, and the arithmetic inside each hidden unit's gradient. The Lean training driver selects activation branches and applies the update and grid rule. The kernel checks equality of the compiled training run and the direct specification for every sample schedule and grid function. The derivative calculation above supplies the mathematical interpretation away from activation breakpoints.

In the artifact's `ChartExample`, the preactivations at $t = 0$ are $(1/4, -1/2)$ and change by $(t, 0)$. The verified coefficients of the loss along that parameter line are

$$c_0 = 16205/8192, \quad c_1 = -121/512,$$
$$c_2 = 163/128, \qquad c_3 = 7/8, \qquad c_4 = 1.$$

The directional Newton correction $-c_1/(2c_2) = 121/1304$ stays in the same activation region. For this correction the artifact checks that the actual reconstruction loss decreases to $5690182520071/2891414573056$. The comparison uses the full loss at the new parameter, including the cubic and quartic terms.

4.2 Latent paths and stochastic training

Let $P(z_1, z_2) = (z_2, z_1)$ denote the nonidentity element of $G = C_2$. It acts on Θ by exchanging the two triples (w_i, b_i, v_i) and on H by permuting the code coordinates. It acts trivially on X. For the swap $\sigma : \Theta \to \Theta$,

$$E_{\sigma\theta} = PE_\theta, \qquad D_{\sigma\theta} = D_\theta P^{-1}.$$

The second equation expresses contravariant transport of the decoder with respect to the latent coordinate change. In particular, $R_{\sigma\theta} = R_\theta$ and $L_x(\sigma\theta) = L_x(\theta)$.

Proposition 4.2 (Equivariance of the learning rule)**.** *The sample field g_x, update U_x, and update along any finite sample schedule commute with the latent exchange. For a finite input law fixed under that exchange, the law of the trained parameters transports by the same exchange, and the final reconstruction law at each fixed input is invariant.*

Proof. Reconstruction invariance fixes r. Swapping the units exchanges z_i, λ_i, and v_i, hence exchanges both encoder gradient components and the decoder-column gradient in each unit. The shared output-bias gradient remains r. This argument also covers a zero preactivation with the prescribed slope. Rational arithmetic and uniform coordinate rounding commute with the same permutation. Induction over a sample list proves the statement for a fixed schedule. Each finite-law branch carries the same schedule weight under transport, since the group acts on the parameters and leaves the data unchanged. Summation or pushforward gives the claims about the laws. □

Lean checks the above equivariance for finite schedules and finite independent-draw laws. Its path transport retains the exchange label and preserves path identities and composition.

The parameter exchange also sends the line $\theta + t\dot{\theta}$ to $\sigma\theta + t\sigma\dot{\theta}$. Reconstruction invariance gives identical local loss polynomials on these lines. When the directional Newton correction exists and stays in the activation region, the correction scalar is identical and the two updated parameters differ by the same exchange.

Form the action quotients of $\Theta \times X$ and $\Theta \times H$ over $[\Theta/G]$ in $\mathcal{E}$. The encoder and decoder give arrows

$$[\Theta \times X/G] \xrightarrow{\overline{E}} [\Theta \times H/G] \xrightarrow{\overline{D}} [\Theta \times X/G]$$

over the parameter quotient. The source is canonically $[\Theta/G] \times X$. The intermediate object has the code transport P along exchange arrows. At a parameter with two identical hidden-unit triples the exchange is a genuine automorphism of the parameter. Its action on the fiber sends the code $(1, 0)$ to $(0, 1)$, even though all actual encoder outputs at that parameter are diagonal. Thus this automorphism is visible on a latent fiber even when neither the parameter nor any input reconstruction changes. A quotient by the set of parameter orbits discards the automorphism.

4.3 A reproducible bars and stripes experiment

The training set consists of the four signed images $u = (1, 1, -1, -1)$, $v = (1, -1, 1, -1)$, $-u$, and $-v$, in that order. The held-out images are mixtures with coefficients

$$(3/4, 1/4), \quad (-1/4, 3/4), \quad (1/2, -1/2), \quad (-5/8, -3/8)$$

in the basis (u, v). These are new images in the same two-dimensional structured family. The zero-output baseline

has mean losses 2 and 73/64 on the training and held-out sets respectively. The bottleneck can in principle represent the whole test family exactly. For a reference parameter, use input weights $u/4, v/4$, hidden biases $2, 2$, decoder columns u, v, and output bias $-2(u+v)$. On inputs of the form $\alpha u + \beta v$ with $|\alpha|, |\beta| \le 1$, the two preactivations are $2+\alpha, 2+\beta$. Their positivity makes the resulting reconstruction exact.

The initialization used in training has zero output bias, both hidden biases equal to $1/2$, and the input weights and decoder columns

i	w_i	v_i
1	$(1/4, 0, 0, -1/4)$	$(1/4, 1/8, -1/8, -1/4)$
2	$(0, 1/4, -1/4, 0)$	$(1/4, -1/8, 1/8, -1/4)$.

We replay a deterministic pseudorandom sample schedule. Start with $s_0 = 104729$, update $s_{k+1} = 48271 s_k \bmod 2147483647$, and select index $\lfloor s_{k+1}/16384 \rfloor \bmod 4$ using zero-based indexing. Use 96 one-example steps with learning rate $1/16$ and grid denominator 4096. The four training inputs occur 21, 26, 23, and 26 times. The uniform law provides a separate stochastic interpretation for the four sample choices. This run is a fixed replay and the table reports means over the four images in each split at each checkpoint.

Updates	Training loss	Held-out loss
0	1.97656	1.08429
24	1.31537	0.88664
48	0.27825	0.09126
72	0.13284	0.04454
96	0.07170	0.05814

Lean checks the five checkpoints as rational equalities. Decimals are rounded for display. The held-out samples are used only for reporting. The last held-out loss increases relative to step 72 even as the training loss decreases.

At step 96 the held-out input $x = (1, 1/2, -1/2, -1)$ gives the code $(7109/4096, 6993/4096)$ and reconstruction

$$R_\theta(x) = (a, b, -b, -a), \quad a = \frac{15161647}{16777216}, \quad b = \frac{34149}{65536}.$$

Exchanging the hidden-unit parameters exchanges these two distinct code values and leaves the reconstruction exactly equal. The Lean artifact checks the code and reconstruction equations under model exchange for all rational parameters and inputs. It reproduces the sample schedule and loss table using the compiled Lean project.

Every initial decoder column and the output bias lie in the data plane $W = \{(a, b, -b, -a) : a, b \in Q\}$. On an input from W, the residual and the corresponding gradient components lie in W as long as these parameters do. The grid map is odd, so rounding preserves W. Induction over training shows that every reconstruction lies in W. For a checkerboard $y = (1, -1, -1, 1)$, orthogonality to W implies loss at least $\|y\|^2/2 = 2$. The measured generalization is interpolation on the bar and stripe family.

4.4 Reconstructing Iris measurements

The same four-feature architecture also compresses the four sepal and petal measurements in the corrected UCI Iris data [16]. The input and target of a training example are the same normalized measurement vector. We hold out every fifth row of the source file, giving 120 training and 30 held-out examples, with ten held-out examples per species. Each feature is centered at its training mean and divided by its training range. The baseline predicts the training mean, which is zero in the normalized coordinates.

The two initial input weight vectors are $(1, -1, 1, 0)/4$ and $(0, 1, 0, -1)/4$. The hidden biases are $(1/16, -1/16)$. Decoder columns are $(1, 0, 1, 1)/4$ and $(0, 1, -1, 0)/4$, with zero output bias. With zero-based indices, epoch e visits training rows in the order $(17i + 13e) \bmod 120$ for $i = 0, \ldots, 119$. Since 17 is coprime to 120, each epoch uses every training example exactly once. There are 16 epochs with learning rate $1/8$ and grid denominator 16384. The means of the half-squared losses are

Epoch	Training loss	Held-out loss
Baseline	0.139442	0.128459
0	0.115408	0.108481
1	0.021482	0.021393
4	0.012202	0.011594
8	0.007521	0.007558
16	0.006241	0.007093

The artifact represents each measurement as an exact tenth and all training arithmetic is rational. Lean checks the split counts by reduction and proves that the epoch index map is injective. The executable reports the losses for this split and schedule. The decoder always takes values in the affine plane $c + \mathrm{span}(v_1, v_2)$. A two-component PCA model would select a plane by minimizing training projection error [15, §14.1]. The reported comparator is the constant training-mean predictor.

5 Tracked realization and compilation

5.1 Numerical objects and tracking

Give each rational its reduced signed numerator and positive denominator as a numeral code. A standard computable pairing operation encodes products and finite lists. These representations give a rational assembly and its finite powers. Normalization, addition, subtraction, multiplication, decidable equality and division with an explicit zero convention all have recursive trackers. In field-theoretic statements with a restricted domain, the nonzero denominator is a hypothesis. This gives a tracked arrow on the subobject cut out by that hypothesis. A particular Lean constant computing total division and a paper theorem about a Newton step with invertible curvature can therefore use the same arithmetic and have different domains.

For a fixed source term, prove tracking by induction on its expression tree. Loading, projection and arithmetic have trackers, and pairing and composition combine them. The stack semantics also operates on recursively coded finite environments and stacks. The general compiler works on finite syntax trees, so compiling and running valid code are uniform recursive operations on encodings. Running an update over a supplied finite list adds primitive recursion on the list. Exact enumeration of all length-T schedules generated by a finite alphabet is a bounded computation, even when its output is too large to be useful for statistical estimation. This logical fact is independent of its time and bit complexity.

The arithmetic equalities certified in the artifact hold on these numerical assemblies because their morphisms are tracked functions and their equalities are equalities of the underlying functions. The fully faithful embedding of the ordinary effective topos carries these arrow equalities into the completion. If an executable map commutes with the two elements of the coordinate-exchange action, it gives a functor on the associated internal action groupoids, and hence a map on their realizations. Its action on a path is labelled by the same group element. This is the external interpretation of the checked equivariance proofs.

5.2 Arithmetic bytecode

The source is intrinsically typed. Write $\Gamma \vdash e : A$ for a rational expression or a pair of source values in context Γ. Variables use typed de Bruijn indices. A stack instruction list has an input shape S and output shape T. Arithmetic consumes two scalar cells and returns a scalar. Pairing consumes cells of the appropriate types, and projection selects one component of a pair. The compiler is continuation-passing at the level of instruction lists. Given continuation k accepting $A :: S$, $\mathrm{comp}(e, k)$ accepts S. For every environment ρ and typed stack s, Lean proves

$$\mathrm{run}(\mathrm{comp}(e, k), \rho, s) = \mathrm{run}(k, \rho, [\![e]\!]_\rho :: s).$$

The proof proceeds by expression induction with the continuation and stack universally quantified. In the binary case the compiler emits instructions for the left term, the right term and then the operation. The right term is consequently on top when the operation executes. The ordering is significant for subtraction and division. Running with the empty continuation gives the source evaluation theorem. A further theorem transports any given extensional source equation to a verified equation about bytecode executions.

5.3 Path certificates

In the coordinate-exchange model, a path from x to y consists of $g \in C_2$ and evidence that $gx = y$. A path expression supports identity, an exchange generator, endpoint-typed composition and reversal. It compiles to an arrow retaining g. Two expressions with the same endpoints denote equal arrows if and only if their labels agree. Proof terms allow label normalization, symmetry and transitivity. The checker is a decision procedure for equality in this presentation. The kernel proves that successful checks establish semantic path equality, and that equality has a certificate in this grammar.

For example, at $d = (2, 2)$ both the identity and the exchange are paths $d \rightsquigarrow d$. Their labels differ, so the checker distinguishes them. Two exchanges compile to the identity. Lean's ordinary equality is proof-irrelevant [13]. The groupoid path sits in a data type, so the two labels are retained. Inside the higher exact completion it describes the corresponding loop in the realized action groupoid.

The artifact also presents a strict two-groupoid with one object, one 1-cell and 2-cells labelled by the abelian group C_2. Both compositions use group multiplication. The 2-cell compiler normalizes vertical composites, horizontal composites and inverses. Its proofs check interchange for arbitrary terms and distinguish the generating 2-cell from the identity. Factoring the identity on these 2-cells through the trivial 2-cell set is impossible, which is the corresponding algebraic calculation behind the manuscript's section obstruction. The calculation is in a strict presentation. The internal comparison and the sheaf-level obstruction use the categorical construction of Section 2.

SynthLean implements a general certifying frontend for a substantial fragment of Martin-Löf type theory and connects it to natural-model semantics [14]. Our proof language is specialized to arithmetic preservation and the displayed cell presentations. Its completeness statement quantifies over the paths written in the coordinate-exchange presentation.

6 Executable artifact and verification

6.1 Arbitrary finite-order directional jets

Let $x, v \in Q$ and let p be formed from constants and one variable by addition, subtraction, and multiplication. Define

$$J^k_{x,v} p = \big([\epsilon^j] p(x + v\epsilon)\big)_{0 \le j \le k}.$$

The coefficient evaluator and the compiler use the Cauchy rule

$$[\epsilon^j](ab) = \sum_{i=0}^{j} [\epsilon^i] a \, [\epsilon^{j-i}] b.$$

The coefficient compiler translates each finite sum into a scalar source expression in the environment (x, v). It then uses the typed bytecode compiler from Section 5. For every polynomial syntax tree, order, direction, basepoint and in-range coefficient, the kernel checks equality of bytecode execution to the coefficient semantics. The base case

for a variable has coefficients $x, v, 0, \ldots$. The multiplication proof lifts evaluation through the finite Cauchy sum. At degree zero the coefficient equals ordinary polynomial evaluation. Increasing the order leaves each previously compiled coefficient unchanged. The theorem is definitional because compilation indexes the expression by its coefficient degree. The reflection theorem gives

$$J^k_{-x,-v}(p \circ (-1)) = J^k_{x,v} p.$$

This is a symmetry of the loss chart as well as of the basepoint and direction. The executable jets are univariate along one direction. The coordinate-exchange transport theorem also covers every coefficient of a selected chart coordinate. At the diagonal point $(1, 1)$, the quartic example below has first coefficient 5 in the first coordinate with direction $(1, 2)$. Exchanging the coordinate without its direction gives 10. Transporting both keeps the coefficient 5. Thus a loop fixing the point can still act on differentiation data. The multivariate jet and permutation law used for the categorical model were established separately in Section 3.

For $v = 1$, the Taylor coefficients of degrees $1, 2, 3$ are $p', p''/2, p'''/6$. The Halley rule applied to p' has form

$$x' = x - \frac{4c_1c_2}{8c_2^2 - 6c_1c_3}.$$

The artifact guards a zero denominator, returning no update in that case. With $p(x) = x^4 + 2x^2 - 3x + 1$ at $x = 1$, it computes the exact fourth-order jet $(1, 5, 8, 4, 1)$ and the trial update $29/49$. The two outputs and the singular guard for a constant loss are checked by reduction. This exercise illustrates executable use of derivatives of order above the Hessian.

6.2 Finite executions and proof boundaries

The exact two-coordinate implementation uses sample records with an active axis, positive curvature on that axis in the worked example, and a target. Its Newton update restricts to the active axis, where the Hessian is a scalar. It leaves that coordinate unchanged when the Hessian is nonpositive. The optimizer uses compiled polynomial expressions to obtain the gradient and curvature, and compiled arithmetic for the update. The momentum state carries both velocity coordinates since gradients accumulated on one axis can influence later steps drawn on the other axis.

Each stochastic law is represented by a list of pairs of rational weights and data. The executable law enumerates ordered histories, so repeated outcomes are kept as separate summands. The kernel checks for all horizons that enumeration commutes with simultaneous relabeling of samples and state. The examples additionally check numerical outputs by kernel reduction. The closed-form moment identities, the higher exact-completion theorem and the 2-type gluing example are mathematical proofs in this article.

Lean module	**Verified content**
`Compiler`	Type-indexed bytecode and execution preservation.
`Paths` and `ProofLanguage`	Groupoid laws, path certificates, and interchange in the strict two-cell model.
`Optimization`	Step and run equivariance, path transport, and finite-law equivariance.
`Jets`	Arbitrary finite-order directional coefficients, reflection and guarded Halley execution.
`Autoencoder`	Backpropagation bytecode, training equivalence and transport of latent codes and sample laws.

Table 1: Main components of the Lean 4.19.0 artifact.

The companion archive includes the complete Lean sources, package configuration, build commands, manuscript sources, and the recorded output. The Lean build uses only its standard library. Computation tests concern exact finite inputs. Resource consumption grows with the number of histories and the size of rational numerators and denominators. The coefficient compiler expands products as ordinary trees, so it makes no general efficiency claim about high-order jets.

7 Conclusion

The exact completion retains recursively tracked arithmetic at dimension zero and provides groupoid quotients in which equivariant optimizer runs have meaningful path transport. Latent exchange in the autoencoder transports the learned code while preserving the decoded prediction and the training rule. This supplies an example of a learned intermediate representation with a nontrivial coordinate redundancy. The Lean artifact checks these transport equations and the implementations of arithmetic, directional jets and training. General univalent internal proof checking, arbitrary probability objects and convergence results for broad classes of losses require additional formalizations.

References

[1] J. M. E. Hyland. The effective topos. In A. S. Troelstra and D. van Dalen, editors, *The L. E. J. Brouwer Centenary Symposium*, pp. 165–216. North-Holland, 1982. DOI and author's copy.

[2] S. Awodey and J. Emmenegger. Toward the effective 2-topos. *Mathematical Structures in Computer Science*, 35:e32, 2025. DOI.

[3] S. Awodey. A higher effective topos. Category Theory 2025, Brno. Joint work with M. Anel and R. Barton. Slides.

[4] Y. Maruyama. Higher tripos theory. Preprint, 2026. arXiv:2610.27163.

[5] G. Stefanich. Derived ∞-categories as exact completions. Preprint, 2023. arXiv:2310.12925.

[6] J. Lurie. *Higher Topos Theory*. Annals of Mathematics Studies 170. Princeton University Press, 2009. Author's copy.

[7] J. Lurie. *Spectral Algebraic Geometry*. Draft, Appendix A, 2018. Author's copy.

[8] A. W. Swan and T. Uemura. On Church's thesis in cubical assemblies. *Mathematical Structures in Computer Science*, 31(10):1185–1204, 2021. DOI.

[9] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951. DOI.

[10] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. DOI.

[11] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018. DOI.

[12] F. W. J. Olver et al., editors. *NIST Digital Library of Mathematical Functions*, §3.8(v), Eq. (3.8.12). Halley's iteration.

[13] Lean developers. *The Lean Language Reference*, §4.2, Propositions. Reference manual.

[14] W. Nawrocki, J. Hua, M. Carneiro, Y. Xu, S. Woolfson, S. Rong, S. Hazratpour, and S. Awodey. A certifying proof assistant for synthetic mathematics in Lean. In *Certified Programs and Proofs* (CPP), pp. 88–103, 2026. DOI and archived artifact.

[15] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016, Chapter 14. Autoencoders.

[16] R. A. Fisher. Iris [Dataset]. UCI Machine Learning Repository, 1936. DOI, repository record, and CC BY 4.0.