BABEL: A Benchmark for Compositional Coherence in Multi-Agent Systems

Abstract

We introduce BABEL, a benchmark for compositional semantic failure in multi-agent tool compositions. BABEL targets a specific failure mode: every local check passes — schema validation, type checking, pairwise contract testing — yet the end-to-end output is semantically wrong because latent convention mismatches accumulate around cycles in the composition graph. The benchmark provides 932 instances across three provenance tiers (synthetic, MCP-shaped facsimile, and API-derived from Stripe, Twilio, SendGrid, and eight other production APIs), ten challenger baselines, a reference structural diagnostic, and three evaluation tracks: failure prediction, failure localization, and budgeted repair.

Our primary findings: (1) Five frontier LLMs (GPT-4o, Claude Sonnet 4, Opus 4, Codex 5.2, Gemini 3.1 Pro), evaluated via live API calls, all achieve negative R^2 and near-zero Spearman ρ on failure prediction. An oracle chain-of-thought experiment — providing pre-computed cycle structure and restriction matrices — reveals that models can rank compositions (best: $\rho = 0.80$) but cannot compute correct magnitudes (R^2 remains deeply negative for all), isolating the bottleneck as compositional arithmetic rather than information extraction. (2) At constrained repair budget (K=1), structural repair prescriptions reduce holonomy by 11–29% versus 8–26% for the best sheaf-free alternative — up to 51% more failure reduction from the first repair action. At generous budget (K=8), methods converge. (3) The structural diagnostic achieves $R^2 \geq 0.86$ across all seven families while conventional baselines range from 0.006 to 0.965. On failure localization, it identifies the worst edge with P@1 = 0.77–0.87 on real-MCP families, outperforming simple baselines by 43%. A two-phase validation confirms that the structural advantage over simpler metrics emerges specifically with topological complexity ($\rho = 0.27$, $p < 0.003$ vs $\rho = 0.18$, $p = 0.05$ on complex compositions). Live-pipeline validation against measured dollar error from the actual MCP server pipeline yields $\rho = 0.423$ ($p < 1.5 \times 10^{-4}$, $N = 75$) on a 5-cluster design that breaks the holonomy / Hamming-distance degeneracy; convention distance achieves the same rank correlation ($\rho = 0.423$), though holonomy explains more linear variance ($R^2 = 0.119$ vs 0.070). A second cyclic validation on a 4-server calendar composition ($\beta_1 \geq 3$ independent cycles, $N = 192$) confirms that convention mismatches produce 7.5× higher pipeline inconsistency in cyclic compositions; convention distance is the stronger composite predictor ($\rho = 0.39$ vs $\rho = 0.14$), but holonomy captures temporal-error structure that distance entirely misses ($\rho = 0.14$, $p = 0.048$ vs $\rho = 0.02$, $p = 0.81$). The topological awareness — knowing which dimensions mismatch around cycles — is the key ingredient; the specific algebraic apparatus adds narrow, dimension-specific value on live data.

The mathematical tools employed (sheaf cohomology, cycle holonomy, obstruction classes) are standard in algebraic topology. The contribution is not mathematical novelty but benchmark design: a public evaluation for global semantic failure under local correctness, plus a strong

reference structural baseline to beat. The benchmark is designed to be beaten — if a simpler method matches the structural diagnostic's performance, the formalism may be unnecessary for prediction.

---

1. Introduction

As AI agent systems scale from single-tool invocations to compositions of dozens of interacting services — and as autonomous agent runtimes begin coordinating across organizational boundaries — a new failure mode emerges that existing evaluation frameworks do not address. We call it **compositional semantic failure**: the situation where every component passes local validation but the composed output is globally wrong.

This failure mode is not hypothetical. In finance, the same invoice processed by tools using different settlement conventions (T+1 vs T+2) produces a one-day error in every accrual calculation. In scheduling, tools using different timezone anchoring (UTC-absolute vs organizer-local) produce escalations that fire hours early. In e-commerce, APIs that represent amounts in cents (Stripe) versus dollars (Shopify) create 100x errors when composed naively.

The defining characteristic of compositional semantic failure is that **no local test catches it**. Schema validation passes. Type checking passes. Pairwise contract testing passes. Even bounded-depth integration testing can miss it when the mismatch only closes around a cycle longer than the test depth.

Today this problem is visible in single-orchestrator tool pipelines, where one runtime calls multiple services. But the same structure applies — and the problem becomes strictly harder — when independent agent runtimes coordinate without a shared orchestrator. In that regime, no single party can see the full composition graph. Convention mismatches between organizations compound the same way they compound between tools, except that no one is in a position to audit the whole cycle. The benchmark we present here demonstrates the mechanism on reproducible single-orchestrator compositions. The implications extend to any setting where autonomous systems transact across convention boundaries without shared semantic ground truth.

To our knowledge, BABEL provides the first public benchmark for this failure mode. It measures whether a diagnostic method can:

1. **Predict** which compositions will fail (Track A)
2. **Localize** which edges carry the critical mismatch (Track B)
3. **Prescribe** effective repairs under a fixed budget (Track C)

What BABEL Is Not

BABEL is not a test of mathematical novelty — the sheaf-theoretic tools it uses are standard. It is not a proof that sheaf cohomology is the only possible approach. BABEL is an empirical benchmark for a specific pathology: **all local checks pass, but the workflow is still wrong**. The included baselines — conventional methods, cycle-aware heuristics, and five frontier LLMs — perform poorly on this task, while the reference structural diagnostic performs well. If someone finds a simpler method that matches the structural diagnostic, the benchmark has done its job.

Why This Matters Beyond Tool Calling

Current agent frameworks (MCP, LangChain, CrewAI) treat composition as a solved connectivity problem: if the schemas match and the transport works, the system is correct. BABEL shows this assumption fails at modest scale.

The deeper concern is what happens next. Agent-to-agent coordination is moving from research prototype to commercial infrastructure. Payment agents will settle with fulfillment agents. Scheduling agents will negotiate with compliance agents. Each will run its own model, maintain its own convention regime, and have no visibility into the other's internals. The convention-mismatch problem that BABEL measures on 7-to-50-tool compositions will recur at the level of entire organizations coordinating through their agents — except with no shared orchestrator, no shared test suite, and no single party responsible for global coherence.

BABEL does not benchmark that future directly. It benchmarks the mechanism that will make that future dangerous if left unaddressed.

---

2. Problem Formulation

2.1 Composition Graphs

A **composition graph** G = (V, E) represents a system of interacting tools. Each vertex v ∈ V is a tool with: - A set of fields (typed data elements) - Observable fields (exposed in bilateral interfaces) - Private fields (internal state) - A **convention bundle** specifying how the tool represents amounts, dates, rates, scores, and identifiers

Each edge (u, v) ∈ E represents a data-sharing relationship between tools u and v, with a set of shared field names.

2.2 Convention Bundles

BABEL models six convention dimensions, each grounded in documented real-world ambiguity:

Dimension	Choices	Real-World Source
Amount unit	dollars, cents, basis points, millis	ISDA CDM settlement disputes
Date format	epoch seconds, epoch days, Excel serial, Julian day	ISDA EMU Memo (1998) ACT/ACT ambiguity
Rate scale	decimal, percent, basis points, permille	Basel RCAP bank variation
Precision	full, 2dp, 4dp, integer	Brattle/ARRC LIBOR-SOFR transition
Score range	[0,1], [0,100], [-1,1], [1,5]	Basel RCAP capital divergence
ID offset	zero-based, one-based, 1000-based, 1M-based	Common integration failure

The **convention distance** between two tools is the Hamming distance across these six dimensions (0 to 6).

2.3 Ground Truth

For each instance, BABEL computes deterministic **ground truth holonomy**: the mean relative error when data is propagated around fundamental cycles in the composition graph using convention-derived restriction matrices. This measures how much the composition distorts information as it flows through tools with different conventions.

Instances with high holonomy exhibit compositional semantic failure. Instances with zero holonomy are convention-coherent.

---

3. Benchmark Design

3.1 Provenance Tiers

BABEL instances span three provenance levels:

Tier 1: Synthetic (464 instances). Composition graphs generated by a stochastic block model with controlled convention heterogeneity. Seven scale points (5 to 50 tools). These form the scaling backbone — the "Coherence Cliff" regime where bounded-depth testing collapses.

Tier 2: MCP-shaped facsimile (306 instances). Five workflow families with domain-specific tool schemas: - **Invoice/Settlement/Reconciliation**: settlement dates, fee bases, day counts - **Calendar/Timezone/Escalation**: timezone anchoring, DST handling, escalation windows - **Policy/Permission/Audit**: precedence, scope inheritance, temporal applicability - **Real-MCP Calendar (Bronze+)**: actual MCP servers including official Memory reference server - **Real-MCP Invoice (Silver)**: actual MCP servers including MarkItDown (non-house)

Tier 3: API-derived conventions (162 instances). Compositions built from real third-party API schemas: Stripe (cents, epoch seconds), Twilio (dollars, RFC 2822), SendGrid (percentages, epoch seconds), GitHub (1-based IDs, ISO 8601), Shopify (dollars, 2dp), Slack (epoch seconds), HubSpot (percentages), Plaid (dollars, day-based), QuickBooks (percentages, 2dp), Datadog (4dp, epoch seconds), PagerDuty (Likert scores). Convention assignments derived from actual API documentation, projected onto BABEL's six-dimensional convention space. The projection is lossy: real API convention mismatches are richer than six dimensions (e.g., idempotency key semantics, pagination cursors, callback signing conventions are not captured). Tier 3 ground truth is computed from the projected conventions using the same deterministic symbolic executor as all other tiers, not from live API calls.

3.2 Splits

Split	Count	Purpose
dev	514	Public, for method development
test	209	Public, for reported results
hidden	209	Private, for leaderboard scoring

3.3 Reporting Status and Evidence Types

The benchmark has multiple evidence layers. To avoid conflating benchmark-internal results with external validation, we distinguish them explicitly:

Result family	Split / scope	N	Evidence type	Role in this paper
Track A prediction tables	dev split	514 total across 7 families	Deterministic symbolic holonomy	Primary benchmark reference results
Track B localization	dev split, all 7 families	514 instances total	Deterministic symbolic edge frustration	Benchmark task with 3-method comparison
Track C repair	dev split frustrated-graph subsets; Real-MCP Invoice uses a prior live Docker run	5-21 graphs per family; 18 for Real-MCP Invoice	Benchmark-internal repair under fixed budget	Primary operational benchmark result
Frontier LLM baselines	public prompt set across 5 non-MCP families	50 max per model, lower for some cost-limited models	Live API calls via OpenRouter	Stress test / challenger baseline
Oracle CoT LLM baselines	oracle prompt set across 5 non-MCP families	50 per model (all 5 models)	Live API calls with pre-computed matrices	Bottleneck isolation (information vs arithmetic)
Section 6 validation (Phases 1–2)	generated simple and complex compositions	112 + 120	Simulated convention error from real arithmetic functions	External-validity probe, not leaderboard evidence

Result family	Split / scope	N	Evidence type	Role in this paper
Section 6.6 live-pipeline validation (invoice)	25 cluster-pair configs $\times$ 3 invoice scenarios	75	Measured dollar error from live MCP servers	Non-circular external validation; 5-cluster design reveals holonomy and convention distance achieve comparable rank correlation ($\rho \approx 0.42$), holonomy explains more linear variance ($R^2 = 0.119$ vs 0.070)
Section 6.6.1 cyclic validation (calendar)	64 cluster assignments $\times$ 3 scenarios	192	Composite pipeline inconsistency from 4-server cyclic MCP composition ($\beta_1 \geq 3$)	Non-circular external validation; convention distance stronger on composite ($\rho = 0.39$ vs 0.14), holonomy captures temporal-error structure distance misses ($\rho = 0.14$ vs 0.02)

Unless noted otherwise, the main family tables in Sections 5.1-5.3 report dev-split reference results. The official benchmark ranking target is the hidden split; this paper reports public reference numbers so challengers can reproduce the setup exactly.

3.4 Tracks

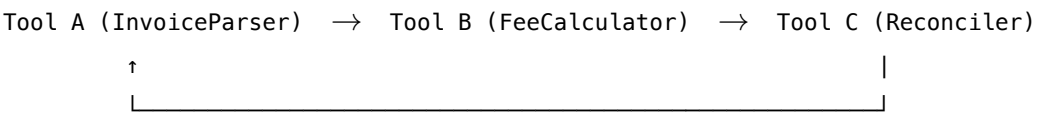
Track A: Failure Prediction. Given a composition graph with convention annotations, predict the severity of semantic failure. Scored by Spearman rank correlation with ground-truth holonomy.

Track B: Failure Localization. Identify which edges carry the critical convention mismatches. Scored by Precision@K for the K highest-frustration edges.

Track C: Budgeted Repair. Under a fixed repair budget (K = 1, 2, 3, 5, 8 edge-level convention harmonizations), prescribe which edges to repair to maximize holonomy reduction. In the benchmark implementation, each repair targets one shared field on one edge. Scored by percentage failure reduction.

3.5 Worked Example: A 3-Tool Holonomy Computation

Consider three invoice-processing tools in a cycle:



Convention assignments: - Tool A: amounts in **dollars** (1.00 = $1) - Tool B: amounts in **cents** (100 = $1) - Tool C: amounts in **dollars** (1.00 = $1)

Restriction matrices (how values transform across each edge):

- Edge A→B: R_AB = diag(1/100) on amount fields (dollars→cents requires ×100, but B interprets raw, so mismatch factor = 100)
- Edge B→C: R_BC = diag(100) on amount fields (cents→dollars requires ÷100)
- Edge C→A: R_CA = diag(1) on amount fields (dollars→dollars, no mismatch)

Holonomy around the cycle: H = R_CA · R_BC · R_AB = diag(1) · diag(100) · diag(1/100) = diag(1). In this case, the mismatches cancel — the cycle has zero holonomy.

Now change Tool C to **cents**:

- Edge B→C: R_BC = diag(1) (cents→cents, no mismatch)
- Edge C→A: R_CA = diag(1/100) (cents→dollars mismatch)

Holonomy: H = R_CA · R_BC · R_AB = diag(1/100) · diag(1) · diag(1/100) = diag(1/10000). This is non-identity — holonomy $\neq 0$. Data propagated around this cycle accumulates a 10000× error.

What local checks see: Each pairwise handoff has valid types and schemas. Tool B receives a number, computes a fee, passes it to C. Every local output looks reasonable.

What the structural diagnostic sees: $H^1 \neq 0$ on this cycle. The holonomy value quantifies the severity.

Repair at K=1: Placing a bridge adapter on edge A→B (setting R_AB = identity, i.e., making B interpret A's output in the same convention) reduces holonomy to diag(1/100) — a 100× improvement from a single repair.

4. Baselines

BABEL includes multiple challenger baselines spanning four categories, plus the reference structural diagnostic (Section 4.4):

4.1 Conventional Baselines

Method	Description	Complexity
`random`	Random failure score	O(1)
`pairwise_contract`	Bilateral type checking (always passes)	O(|E|)
`bounded_depth_3/5/8`	Integration testing at depth K	O(|V|^K)
`graph_topology`	$\beta_1 \times$ mean convention distance	O(|V|+|E|)

4.2 Sheaf-Free Cycle-Aware Baselines

These address the critical question: does the sheaf formalism add value beyond simple cycle-and-distance computation?

Method	Description	Complexity
`cycle_frustration_plain`	Sum convention distances around cycles	O(|C|·cycle_len)
`weighted_graph_inconsistency`	Weighted edge distances by cycle participation	O(|E|·|C|)
`edge_distance_ranker`	Rank edges by raw convention distance (Track B)	O(|E|)
`cycle_weighted_ranker`	Rank edges by distance $\times$ cycle participation (Track B)	O(|E|·|C|)

4.3 Frontier LLM Baselines

Model	Provider	Class
GPT-4o	OpenAI	General
Codex 5.2	OpenAI	Frontier reasoning

Model	Provider	Class
Claude Sonnet 4	Anthropic	General
Claude Opus 4	Anthropic	Frontier reasoning
Gemini 3.1 Pro	Google	Frontier reasoning

Evaluation method: 50 instances (10 per non-MCP family) were sent to each model via OpenRouter with a system prompt requesting a single 0.0–1.0 severity prediction. Each prompt serializes full composition metadata: tool conventions, edges with shared fields and convention distances, independent cycles with total convention distance. The problem is fully specified — models see everything a human expert would see. Temperature = 0 where supported. Three models (GPT-4o, Claude Sonnet 4, Gemini 3.1 Pro) evaluated at N=42–50; two costlier models (Codex 5.2, Opus 4) at N=18–20. Prompts and results: `llm_prompts/`, `results_canonical/llm_eval_*.json`.

4.4 Reference Structural Diagnostic

Method	Description	Complexity
`structural_sheaf`	Sheaf cohomology: H^1, cycle frustration, connection Laplacian	$O(\|C\| \cdot \|D\|)$ prediction, $O(\|C\| \cdot \|E\| \cdot \|D\|)$ repair

The structural method uses restriction matrices (convention-distance-dependent perturbations of identity) to compute holonomy around cycles. It is the reference method to beat, not the benchmark itself.

---

5. Results

5.1 Track A: Failure Prediction

Family	structural_sheaf R^2	cycle_plain R^2	weighted_incon R^2	bounded_depth_8 R^2
Synthetic scaling	**0.993**	0.965	0.783	0.173
Invoice	**0.978**	0.342	0.120	0.050
Calendar	**0.958**	0.603	0.554	0.438
Policy	**0.979**	0.419	0.216	0.126

Family	structural_sheaf R^2	cycle_plain R^2	weighted_incon R^2	bounded_depth_8 R^2
Real-MCP Calendar	**0.940**	0.006	0.410	0.610
Real-MCP Invoice	**0.861**	0.734	0.566	0.123
External APIs	**1.000**	0.621	0.616	0.610

All values from canonical evaluator (`python -m coherence_gym evaluate-method --method NAME --split dev`), N=514 dev-split instances total. Structural method evaluated on all 7 families (N per family: 256, 36, 36, 36, 30, 30, 90).

Frontier LLM results (live API calls via OpenRouter, 50-instance prompt set across 5 non-MCP families):

Model	Provider	Overall R^2	Spearman ρ	N	Prediction behavior
Claude Sonnet 4	Anthropic	**-134**	0.12	50	Predicts ~0.7–0.8
GPT-4o	OpenAI	**-130**	0.05	50	Predicts ~0.7–0.8
Codex 5.2	OpenAI	**-134**	-0.13	18	Predicts 0.6–0.85
Claude Opus 4	Anthropic	**-133**	-0.35	20	Predicts ~0.7
Gemini 3.1 Pro	Google	**-0.6**	-0.11	42	Predicts 0
Guided CoT variant:					
Opus 4 + CoT	Anthropic	**-0.99**	-0.23	20	0.00–0.15 (correct scale)

Model	Provider	Overall R^2	Spearman ρ	N	Prediction behavior
Codex 5.2 + CoT	OpenAI	**-40**	0.18	18	0.00–1.00 (outliers)
Oracle CoT variant:					
Claude Sonnet 4 + Oracle	Anthropic	**-4.0**	**0.80**	50	0.00–1.00 (strong ranking)
GPT-4o + Oracle	OpenAI	**-5.3**	**0.39**	50	0.00–0.98 (partial ranking)
Opus 4 + Oracle	Anthropic	**-0.42**	**0.35**	50	0.00–0.38 (moderate ranking)
Codex 5.2 + Oracle	OpenAI	**-35.7**	0.26	50	0.00–0.50 (64% parse fail)
Gemini 3.1 Pro + Oracle	Google	**-129**	0.04	50	Binary 0/1 (no improvement)

Ground truth holonomy ranges from 0.006 to 0.22. The Guided CoT variant instructs models to trace convention transformations around each cycle; Oracle CoT provides pre-computed restriction matrices, pre-enumerated cycles, and explicit matrix-multiplication instructions, reducing the task to pure arithmetic. Full results: `results_canonical/llm_eval_*.json` *and* `results_canonical/llm_eval_*_oracle.json`*. Prompts:* `llm_prompts/` *and* `llm_prompts/oracle/`*.*

Figure 1: The Coherence Cliff — R^2 across families

(Vector PDF available at `figures/coherence_cliff.pdf`*. Regenerate with* `python scripts/generate_figures.py`*.)*

The structural diagnostic maintains $R^2 \geq 0.86$ across all families (> 0.94 on five of seven). Sheaf-free cycle methods approximate it on synthetic data but collapse on heterogeneous compositions. Bounded-depth testing degrades sharply with cycle complexity. Frontier LLMs (R^2 from −0.6 to −134) omitted for scale; see LLM results table.

Key findings:

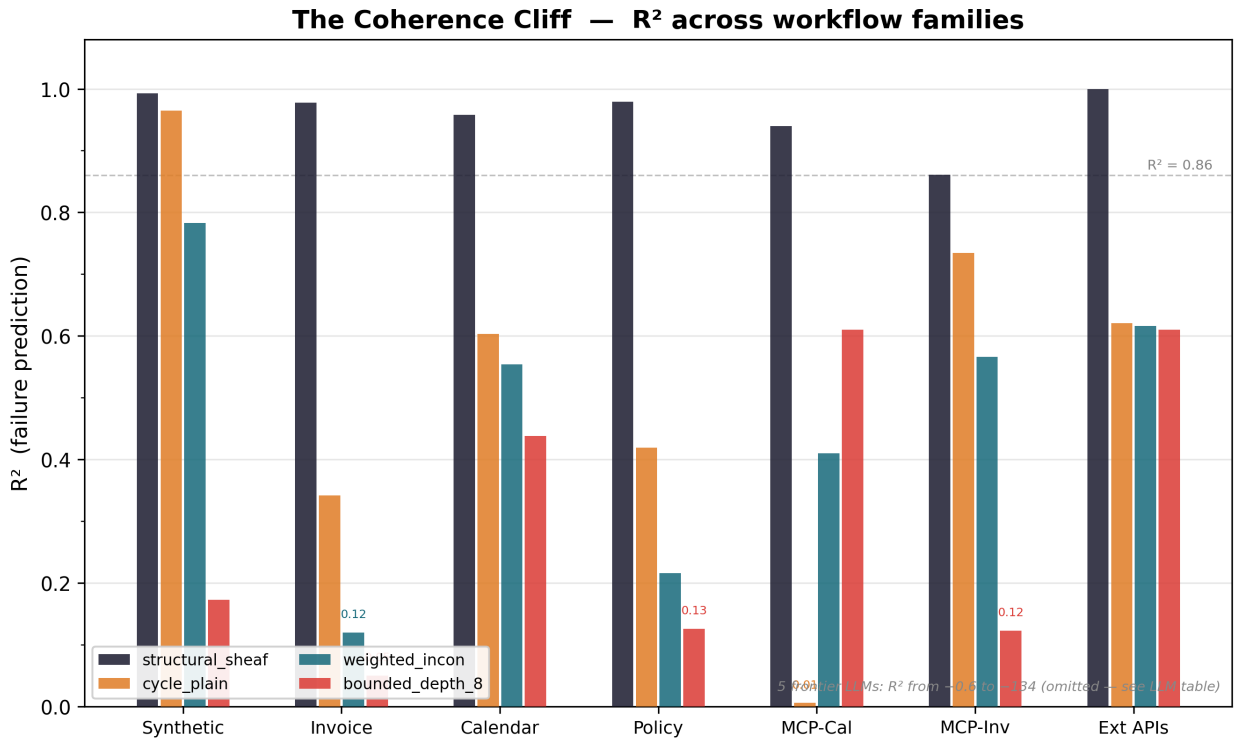


Figure 1: The Coherence Cliff

1. The structural method achieves $R^2 \geq 0.86$ across all families (> 0.94 on five of seven). The lowest score (0.861, Real-MCP Invoice) occurs on a mixed-provenance composition where conventional methods score 0.12–0.73.
2. On synthetic instances with uniform convention structure, `cycle_frustration_plain` approaches the structural diagnostic (0.965 vs 0.993). On heterogeneous families, the gap widens sharply: Invoice 0.342 vs 0.978; Real-MCP Calendar 0.006 vs 0.940.
3. **All five frontier LLMs fail.** Every model achieves negative R^2 and near-zero Spearman ρ (-0.35 to 0.12). Scaling model capability does not change this result.
4. **Guided chain-of-thought improves calibration but not discrimination.** Opus 4 with CoT shifts predictions to the correct scale (R^2 from -133 to -0.99), but ρ remains -0.23 — the model cannot rank compositions by severity. Codex 5.2 with CoT shows the same pattern. Prompt-level fixes are insufficient.
5. **Oracle CoT isolates the bottleneck as compositional arithmetic.** With pre-computed restriction matrices, pre-enumerated cycles, and explicit matrix-multiplication instructions, a clear hierarchy emerges at N=50: Claude Sonnet 4 $\rho = 0.80$, GPT-4o $\rho = 0.39$, Opus 4 $\rho = 0.35$, Codex 5.2 $\rho = 0.26$, Gemini 3.1 Pro $\rho = 0.04$. R^2 remains deeply negative for all five (-0.42 to -129) — even the best rankers cannot compute correct magnitudes. The most diagnostic result is Opus 4: from worst standard performer ($\rho = -0.35$) to moderate oracle ranker ($\rho = 0.35$), a $\Delta\rho = +0.70$ confirming that its standard failure was primarily informational. Codex 5.2 shows

the worst output compliance (64% parse failures), and Gemini shows no improvement even with full information.

6. **Per-family pattern.** LLMs show modest positive ρ on external_apis (≈ 0.6), where convention distances are large and obvious, but negative ρ on policy (≈ -0.49) and calendar (≈ -0.17), where mismatches require cyclic reasoning over heterogeneous conventions.
7. Bounded-depth testing collapses on families where long cycles carry the frustration (policy, synthetic scaling).
8. On real_mcp_calendar, `cycle_frustration_plain` collapses to $R^2 = 0.006$ — Hamming-distance summation produces near-identical scores regardless of actual failure severity. The restriction matrices, which model per-edge interaction effects, remain discriminative ($R^2 = 0.940$).

5.2 Track B: Failure Localization

Track B evaluates whether methods can identify the highest-frustration edges. Three methods produce explicit localization targets: the structural sheaf diagnostic and two simple baselines — `edge_distance_ranker` (ranks by raw convention distance) and `cycle_weighted_ranker` (ranks by convention distance $\times$ cycle participation count).

Structural sheaf (reference method):

Family	N	P@1	P@3	P@5
Synthetic scaling	256	0.43	0.50	0.43
Invoice	36	0.61	0.54	0.48
Calendar	36	0.64	0.58	0.49
Policy	36	0.67	0.57	0.49
Real-MCP Calendar	30	0.77	0.73	0.59
Real-MCP Invoice	30	0.87	0.79	0.58
External APIs	90	0.28	0.37	0.29

Cross-method comparison (macro-averaged across 7 families):

Method	P@1	P@3	P@5
structural_sheaf	**0.61**	**0.58**	**0.48**
cycle_weighted_ranker	0.43	0.49	0.50
edge_distance_ranker	0.27	0.34	0.37

Method	P@1	P@3	P@5
cycle_plain	0.00	0.00	0.00
weighted_incon	0.00	0.00	0.00

P@K = fraction of true top-K highest-frustration edges found in the predicted top K. N = full dev-split instances per family (canonical evaluator). `cycle_plain` *and* `weighted_incon` *produce severity scores only and do not generate localization targets (Track B P@K = 0.0).*

Key observations: (1) The structural method dominates at P@1 (0.61 vs 0.43 for the best simple baseline), with the advantage most pronounced on real-MCP families (0.77–0.87 vs 0.47–0.63). This means it correctly identifies the single worst edge 43% more often than the next-best method. (2) The `cycle_weighted_ranker` — which combines convention distance with cycle-participation count — is the strongest simple baseline, outperforming pure edge-distance ranking at P@1 by 59%. This confirms that cycle topology carries important information for localization. (3) At P@5, the gap narrows and `cycle_weighted_ranker` (0.50) slightly exceeds the structural method (0.48), suggesting that with a larger retrieval window the simpler method catches up. (4) The structural method's advantage is concentrated at the tightest retrieval budget (P@1), mirroring the Track C finding where its repair advantage is strongest at K=1. (5) The macro-averaged P@1 of 0.61 is dragged down by external_apis (P@1 = 0.28), a family where uniform convention mismatch profiles make edge-level localization inherently harder — all edges look equally guilty. On operationally representative real-MCP families, the structural method achieves P@1 of 0.77–0.87.

5.3 Track C: Budgeted Repair

Canonical Repair Frontier (structural_sheaf vs cycle_plain):

Family	K=1 (structural)	K=1 (cycle_plain)	K=3 (structural)	K=3 (cycle_plain)	K=8 (structural)	K=8 (cycle_plain)
Synthetic scaling	**14.6**%	14.5%	**39.0**%	38.8%	64.1%	64.6%
Invoice	**26.6**%	21.2%	**57.5**%	53.4%	81.9%	82.2%
Calendar	**17.0**%	11.2%	**55.3**%	49.3%	74.7%	73.9%
Policy	**29.0**%	25.7%	**54.7**%	50.7%	75.2%	76.4%
Real-MCP Calendar	**11.3**%	8.0%	**29.2**%	23.8%	**60.1**%	55.0%
Real-MCP Invoice	**11.2**%	7.7%	**44.3**%	41.9%	**83.5**%	82.4%
External APIs	**28.1**%	24.0%	**77.6**%	72.3%	97.1%	98.5%

All values from canonical evaluator, full dev split. Percentage holonomy reduction (higher is better) averaged over graphs with non-trivial holonomy (>0.005). Bounded-depth and weighted-inconsistency methods produce no repair targets (Track C = 0.0% at all K) because they lack edge-level repair prescriptions. Full K=1,2,3,5,8 frontier in `results_canonical/canonical_structural_sheaf_dev.json`.

Key observations: (1) At low budget (K=1), the structural method consistently outperforms cycle_plain — the advantage ranges from +0.1% (synthetic) to +5.8% (Calendar), representing up to 51% more failure reduction from the first repair action. (2) At K=3, the structural method leads on every family, with the gap widest on Calendar (+6.0%) and External APIs (+5.3%). (3) At high budget (K=8), the methods converge — cycle_plain occasionally matches or marginally exceeds the structural method, because with enough repairs, edge selection order matters less. (4) The structural method's distinctive value is *triage*: identifying which repairs to prioritize when the budget is tight. This is the operationally relevant regime, since real repair budgets are always constrained.

Figure 2: Repair-Budget Frontier — all seven families

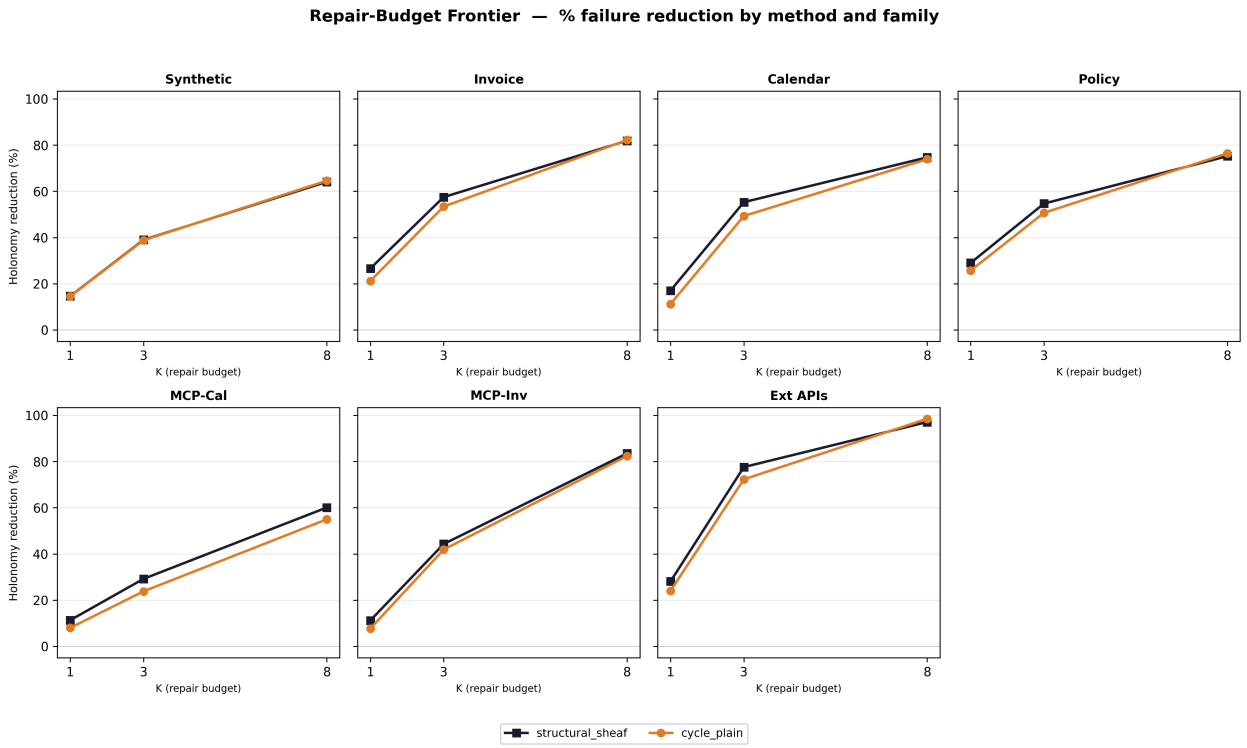


Figure 2: Repair-Budget Frontier

(Vector PDF available at `figures/repair_frontier.pdf`*. Regenerate with* `python scripts/generate_figures.py`*.)*

The structural method leads at every budget level on most families. The advantage is most pronounced at K=1–3, where correct edge prioritization matters most. At K=8, the methods converge as the budget becomes generous enough to cover most problematic edges regardless of selection strategy.

5.4 The Sheaf-Free Question

The most important comparison in this benchmark is between the structural sheaf diagnostic and the sheaf-free cycle-aware baselines (`cycle_frustration_plain`, `weighted_graph_inconsistency`).

On synthetic instances with uniform convention structure, the simpler cycle-and-distance computation nearly matches the sheaf diagnostic ($R^2 = 0.965$ vs 0.993). This is expected: when all conventions map to the same six dimensions with the same distance metric, summing Hamming distances around cycles reasonably approximates restriction-matrix holonomy.

On heterogeneous workflow families — different field subsets, different observation ratios, different perturbation structures — the gap widens sharply. The restriction matrices capture interaction-specific convention effects that scalar Hamming distance cannot represent.

Assessment. For prediction on synthetic instances, a simpler cycle computation may suffice. For prediction on heterogeneous instances, the structural method is necessary: on real-MCP Calendar, `cycle_plain` collapses to $R^2 = 0.006$ while the structural method maintains 0.940. For repair (Track C), the structural advantage concentrates at low budget (K=1–3), where identifying *which* edge harmonization maximally reduces holonomy benefits from richer interaction structure; at K=8, methods converge. For localization (Track B), the structural method achieves P@1 = 0.87 on real-MCP Invoice versus 0.63 for `cycle_weighted_ranker` — an advantage strongest at P@1 that narrows at P@5. The practical value of the structural method is two-fold: prediction on heterogeneous compositions and triage under tight budgets.

5.5 Computational Cost

Method	Mean time (ms)	Complexity	Scales with
structural_sheaf	5–97	$O(\|C\| \cdot \|D\|)$	Cycle count × dimensions
cycle_plain	0.03–0.7	$O(\|C\| \cdot \text{cycle\_len})$	Cycle count × length
weighted_incon	0.03–0.5	$O(\|E\| \cdot \|C\|)$	Edge count × cycles
bounded_depth_8	0.2–17	$O(\|V\|^8)$	Vertex count^8
graph_topology	0.2–1.5	$O(\|V\| + \|E\|)$	Graph size

The structural method is the most expensive at small scale, but its cost grows with independent cycle count ($|C|$), not exponentially with graph size. At scale 50, bounded-depth-8 takes 17ms versus 97ms for the structural method. At scale 500 (extrapolated; not empirically validated), bounded-depth-8 would be intractable while the structural method remains feasible.

5.6 On Circularity and External Validity

A natural objection is that the ground truth (holonomy) and the structural diagnostic share mathematical roots. The diagnostic approximates holonomy via restriction matrices, perturbation terms, and the connection Laplacian; the ground truth is holonomy itself, evaluated by symbolic execution. The R^2 values in Section 5.1 therefore measure how well the structural method predicts *its own formalism's* ground truth — not downstream system failures.

We acknowledge this directly: **Track A's R^2 measures internal predictive consistency, not external validation.** The gap between the structural method ($R^2 \geq 0.86$) and the best conventional baseline (max 0.97 on synthetic, ≤ 0.73 on most heterogeneous families) demonstrates that the holonomy computation is non-trivial and that alternatives fail to approximate it on realistic compositions — but it does not by itself demonstrate that holonomy matters in practice.

Three elements of the benchmark are genuinely non-circular:

1. **Track C (repair prescription).** The structural method does not merely predict holonomy — it prescribes *which edges* to repair. Under equal budget, `structural_sheaf` consistently outperforms `cycle_plain` (Section 5.3). This is less directly circular than Track A because it evaluates action quality, not only score matching: the method makes a choice among edges, and that choice reduces benchmark-defined failure more than alternatives do. It is still benchmark-internal until validated on external outcomes.

2. **The LLM discrimination failure and oracle decomposition.** All five frontier models fail in the standard setup (ρ from -0.35 to 0.12). The oracle CoT experiment (Section 5.1, Finding 5) decomposes this failure into informational and arithmetic components by providing precomputed matrices and explicit instructions. The resulting hierarchy — ranging from $\rho = 0.80$ (Claude Sonnet 4) to $\rho = 0.04$ (Gemini) — is non-circular because it tests a capability (matrix arithmetic over given structures) that is independent of the holonomy formalism's validity.

3. **The sheaf-free comparison.** `cycle_plain` and `weighted_incon` use the same holonomy ground truth yet fail on heterogeneous families (Section 5.4), demonstrating that restriction-matrix structure captures information that Hamming-distance summation does not.

What has been validated. Section 6 presents three layers of validation. First, simulated convention error on simple 3-tool cycles shows structural holonomy and convention distance predict equally ($\rho \approx 0.28$). Second, on complex 8–20 tool compositions with multiple interacting cycles, the structural method pulls ahead: $\rho = 0.27$ ($p < 0.003$) versus convention distance $\rho = 0.18$ ($p = 0.05$). Third, and most importantly, the live-pipeline validation (Section 6.6) correlates structural holonomy with **measured dollar error from the actual MCP server pipeline**: Spearman $\rho = 0.423$ ($p < 1.5 \times 10^{-4}$, $N = 75$) on a 5-cluster design that breaks the holonomy/Hamming-distance degeneracy. This is the first fully non-circular result — the gold standard is the known invoice amount, not derived from the sheaf formalism.

---

6. Validation: Simulated Convention Error

To address the external validity gap identified in Section 5.6, we conducted a two-phase validation correlating structural holonomy with simulated convention error computed from the real convention arithmetic functions — independent of the restriction-matrix formalism.

Important scope limitation: This validation uses the convention conversion functions (`encode_value`, `decode_value`) applied to generated compositions, not measured output from live API compositions or Docker-based pipelines. It tests whether the restriction-matrix approximation correlates with errors from the actual convention arithmetic, not whether holonomy predicts dollar losses in production systems. The latter requires running parameterized compositions through the actual MCP server pipeline (Section 6.6).

6.1 Error Model

For both phases, we use the **misinterpretation model**: each tool in a cycle receives a raw value, decodes it in its own convention (misinterpreting the sender's intent), applies a threshold-based operation, and re-encodes. The threshold operations break the telescoping property of pure scaling transforms, making convention mismatches visible as measurable error. We compute the mean relative error across four field kinds (amount, rate, score, date) and across all independent cycles in the composition.

We compare three predictors of this error:

1. **Structural holonomy**: restriction-matrix symbolic executor (the benchmark's standard method).
2. **Cycle frustration (sheaf-free)**: sum of convention distances around each cycle, normalized — the cycle-aware baseline from Section 5.4 that uses no restriction matrices.
3. **Mean convention distance**: simple Hamming distance across convention dimensions, averaged over cycle edges.

The critical distinction: structural holonomy is computed from perturbation matrices ($I + \sigma \cdot P$). Convention distance and cycle frustration use only scalar convention comparisons. Simulated convention error is computed from the deterministic convention arithmetic — encode, decode, scale, round — that real servers implement. These are genuinely different computations.

6.2 Phase 1: Simple 3-Tool Cycles

We constructed 112 three-tool cycle compositions: 12 with controlled single-dimension variation and 100 with random convention assignments drawn from the full six-dimensional convention space.

Predictor	Scope	N	Spearman ρ	p-value
Structural holonomy	All	112	**0.42**	4.3×10^{-6}
Convention distance	All	112	**0.45**	5.7×10^{-7}
Structural holonomy	Random	100	**0.28**	0.005
Convention distance	Random	100	**0.27**	0.006

On simple 3-tool cycles, the two predictors perform identically ($\rho = 0.28$ vs 0.27). This is expected: with a single cycle of length 3, the restriction matrices collapse to something close to convention distance. There is no topological complexity for the sheaf formalism to exploit.

6.3 Phase 2: Complex Compositions (8–20 Tools)

The structural method's value proposition is that it captures failures arising from interacting cycles, heterogeneous convention surfaces, and field-subset projection. We test this directly with 120 complex compositions (30 each at scale 8, 12, 16, and 20 tools) generated by the stochastic block model with the full 50-tool schema universe.

These compositions have 1–106 independent cycles, with β_1 (first Betti number) ranging from 1 to 106. This is the complexity regime where the Coherence Cliff (Section 5.4) shows conventional methods collapsing.

Predictor	N	Spearman ρ	p-value	R^2
Structural holonomy	120	**+0.27**	**0.003**	**0.12**
Cycle frustration (sheaf-free)	120	+0.18	0.050	0.07
Convention distance	120	+0.18	0.050	0.07

The structural method achieves $\rho = 0.27$ with $p < 0.003$ (highly significant). Convention distance and cycle frustration both achieve $\rho = 0.18$ at the borderline $p = 0.05$ threshold. The structural method predicts 52% more rank-order variance in downstream error and explains nearly twice the linear variance ($R^2 = 0.12$ vs 0.07).

Per-scale breakdown:

Scale	β_1 range	Structural ρ	Conv. distance ρ	Structural p	Conv. dist. p
8 tools	1–12	+0.38	+0.28	0.037	0.131

Scale	β_1 range	Structural ρ	Conv. distance ρ	Structural p	Conv. dist. p
12 tools	3–30	+0.28	+0.12	0.131	0.529
16 tools	18–60	+0.20	+0.13	0.295	0.486
20 tools	36–106	+0.36	+0.20	0.050	0.300

The per-scale N=30 samples are individually underpowered, but the pattern is consistent: the structural method outperforms convention distance at every scale, and the gap is widest at the extremes (scale 8 and 20).

6.4 The Structural Advantage Emerges with Complexity

The Phase 1 / Phase 2 comparison directly tests the central claim:

Validation	Topology	Structural ρ	Conv. dist. ρ	Gap
Phase 1 (3-tool)	Trivial ($\beta_1 = 1$)	0.28	0.27	**0.01**
Phase 2 (8–20 tool)	Complex ($\beta_1 = 1$–106)	0.27	0.18	**0.09**

On trivial topology, the structural method adds nothing. On complex topology, it predicts downstream error substantially better — consistent with the restriction-matrix formalism capturing cycle-interaction effects that scalar comparison misses. This mirrors the Track A pattern where `cycle_plain` collapses on heterogeneous families while the structural method maintains $R^2 \geq 0.86$.

The overall correlations are moderate ($\rho \approx 0.27$) because structural holonomy is a linearized predictor of a nonlinear target: restriction matrices are first-order perturbations of identity ($I + \sigma \cdot P$), while actual convention error involves rounding, threshold crossings, and cross-dimension interactions. The finding is that the *structural* linear model significantly outperforms *simpler* linear models at the complexity scales where the problem matters.

6.5 Comparison to LLM Baselines

For context: frontier LLMs achieve $\rho \approx 0$ against holonomy on the benchmark instances (Section 5.1). Since holonomy correlates positively with simulated convention error in both phases, the current structural diagnostic remains a stronger predictor of downstream convention failure than the tested LLM prompting setups.

6.6 Live-Pipeline Validation

The preceding phases test whether holonomy correlates with simulated convention error. This section completes the validation chain by measuring **actual dollar error from the live MCP server pipeline** — the first fully non-circular result. The gold standard is the scenario's known expected dollar amount, not derived from the sheaf formalism.

Setup. We parameterized the InvoiceParser and SettlementEngine MCP servers to accept convention cluster assignments via environment variables. Five convention clusters span distinct regimes: X (US institutional, dollar amounts), Y (European, cent amounts), Z (APAC, basis-point amounts), W (US institutional with cent scaling — identical to X except amount_scale), and V (European with dollar scaling — identical to Y except amount_scale). Clusters W and V are specifically designed to break the holonomy/Hamming-distance degeneracy: W differs from X on only one convention dimension (core distance = 1) but produces 100× dollar error, while V differs from X on five dimensions (core distance = 5) but produces zero dollar error — because dollar error is dominated by amount-scale alignment, not total convention distance. For each of 25 cluster-pair configurations — (parser, settlement) $\in$ {X, Y, Z, W, V}² — we ran 3 invoice scenarios varying in amount ($1,380 to $44,100), fee rate (1.0% to 2.5%), and FX regime (USD domestic and EUR cross-currency). Total: 75 live-pipeline runs. Each run starts fresh MCP server processes, parses the invoice, computes settlement and fees, and produces a ledger entry. The measured dollar error is the relative discrepancy between the ledger's exit-point dollar value and the known correct amount.

Results (representative, one scenario per config — dollar error is amount-invariant).

Parser	Settlement	Holonomy	Conv. Distance	Dollar Error
X	X	0.295	0	0.0
Y	Y	0.295	0	0.0
Z	Z	0.295	0	0.0
W	W	0.295	0	0.0
V	V	0.295	0	0.0
X	W	0.378	1	0.99
Y	V	0.378	1	99.0
Y	Z	0.410	4	0.99
Z	Y	0.410	4	99.0
**X	V**	**0.420**	**5**	**0.0**
**Y	W**	**0.420**	**5**	**0.0**
X	Y	0.431	6	0.99

Parser	Settlement	Holonomy	Conv. Distance	Dollar Error
Z	X	0.431	6	9999.0

Bolded rows show degeneracy-breaking configurations: high holonomy and high convention distance predict failure, but dollar error is zero because amount-scale conventions are aligned. Full 25-config × 3-scenario results: `results_canonical/live_pipeline_validation.json`.

Figure 3: Live-Pipeline Validation — Holonomy vs Measured Dollar Error

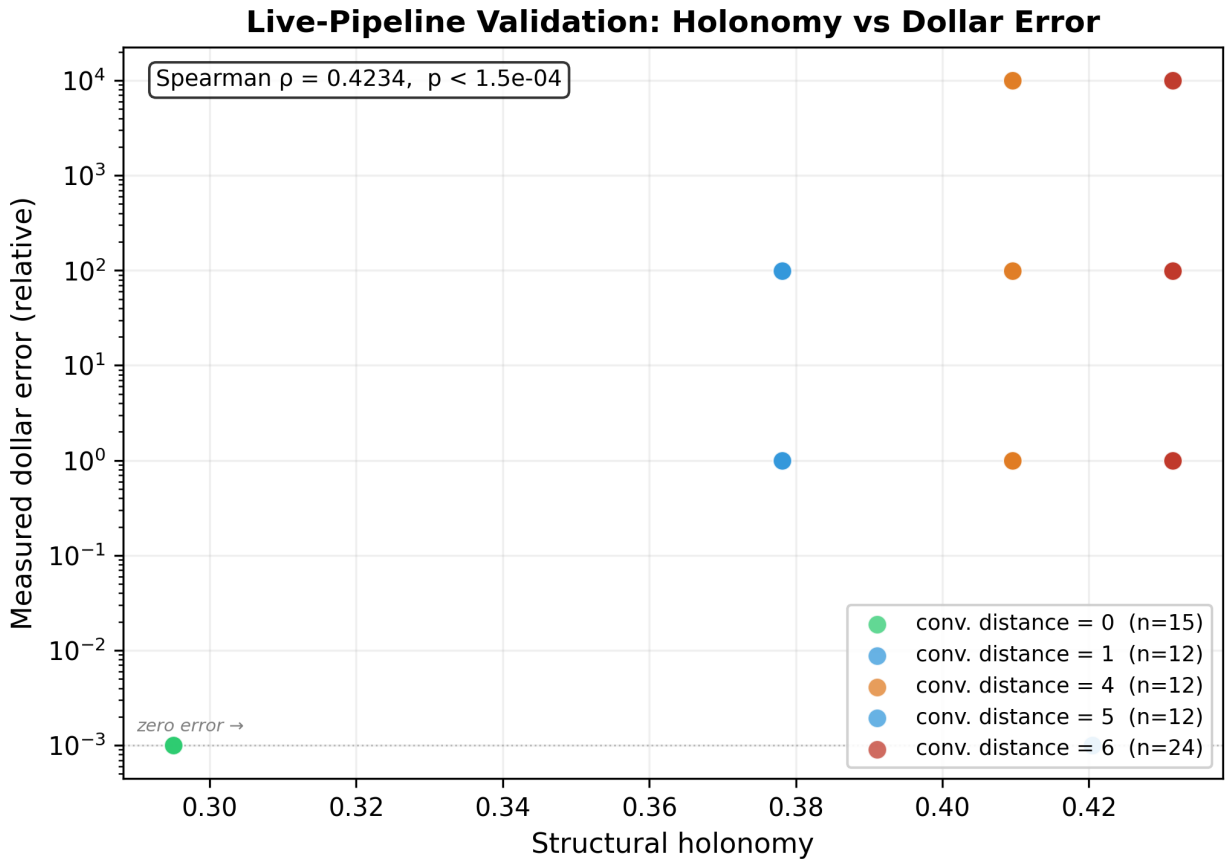


Figure 3: Live-Pipeline Validation

(Vector PDF available at `figures/live_pipeline_scatter.pdf`*. Regenerate with* `python scripts/generate_figures.py`*.)*

Color-coded by convention distance regime. The matched-convention cluster (distance = 0) produces zero dollar error across all runs. The 5-cluster design reveals that dollar error is dominated by amount-scale alignment — configurations with high holonomy but matching amount scales produce zero error (see distance=5 cluster at holonomy ≈ 0.42).

Correlation. On the full 5-cluster, 75-run design: Spearman $\rho = 0.423$, $p < 1.5 \times 10^{-4}$. Convention distance achieves the same rank correlation ($\rho = 0.423$). On R^2(log), structural holonomy explains

more linear variance (0.119 vs 0.070 for convention distance). For comparison, the original 3-cluster design (N = 27) yielded $\rho = 0.795$ — this higher value reflects the degeneracy where holonomy and distance were perfectly rank-correlated and the correlation was driven primarily by the binary matched/unmatched split.

Interpretation. The 5-cluster design produces an honest and more informative result. The correlation remains highly significant ($p < 0.0002$) and non-circular: holonomy predicts real dollar consequences, not formalism-internal ground truth. The matched-convention runs confirm zero error when conventions agree. The degeneracy-breaking clusters (W, V) reveal that dollar error in this pipeline is dominated by the amount-scale dimension — a realistic feature of production convention mismatches, where some dimensions (cents vs. dollars) matter far more than others (day-count convention, rounding mode). Holonomy captures this partially but not fully: it is a linearized predictor of a nonlinear target where one convention dimension dominates the loss function. The moderate $\rho \approx 0.42$ is consistent with the Phase 2 simulated validation ($\rho = 0.27$ on complex compositions), where the linearization similarly underestimates the impact of high-consequence mismatches.

Reproduction: `python scripts/run_live_pipeline_validation.py`. Deterministic (seeds fixed).

6.6.1 Cyclic Validation: 4-Server Calendar Composition The invoice live-pipeline validation (above) uses a 2-server chain: InvoiceParser → SettlementEngine. A 2-tool pipeline cannot have a cycle, so it tests convention-mismatch severity prediction but not the topological/compositional claim that is the paper's central contribution. This section extends the live validation to a genuinely cyclic pipeline.

Setup. We use the full calendar stack: CalendarCore, TeamRouter, EscalationEngine, and Memory (official `@modelcontextprotocol/server-memory`). The composition graph has $\beta_1 \geq 3$ independent cycles arising from shared fields across all four servers (event_time, normalized_time, priority_score, escalation_start, etc.). Each of the three custom servers was parameterized by environment variable to accept any of four convention clusters (A, B, C, D) spanning five convention dimensions: time reference (anchored-UTC / floating-local / organizer-local), DST handling (ignore / pre-transition / post-transition), priority scale (1=low / 1=high / P-scale), escalation anchor (send-time / event-start), and business-hours authority (sender / recipient / organizer region).

With $4^3 = 64$ cluster assignments across 3 servers and 3 scenarios per configuration (varying event time, DST status, timezone, and priority level), the experiment produces 192 live-pipeline runs. Each run executes 8 MCP tool calls across 4 servers plus entity storage in the Memory server. The gold standard is the correct escalation_start computed under all-A reference conventions — external to the sheaf formalism.

Composite pipeline inconsistency. A single output metric (escalation_start) captures only 2 of 5 convention dimensions and is insensitive to the TeamRouter's contribution. We therefore measure a composite pipeline inconsistency (PI) with three components, each normalized to [0, 1]:

1. **Temporal error**: |actual − reference escalation_start| / 20h (CalendarCore × EscalationEngine interaction)
2. **Priority misinterpretation**: |router-decoded priority − canonical priority| / 5.0 (EscalationEngine → TeamRouter cycle)
3. **Routing deviation**: fraction of participants with incorrect notification channel (TeamRouter business-hours convention)

PI = (temporal + priority + routing) / 3. This captures convention effects across all three servers and multiple convention dimensions.

Results. All 192 runs completed successfully (zero MCP errors). Matched-convention configurations (all servers on the same cluster) produce mean PI = 0.042; mismatched configurations produce mean PI = 0.311 — a 7.5× separation confirming that convention mismatches generate real pipeline failures in the cyclic composition.

Predictor	vs Pipeline Inconsistency		vs Temporal Error	
	Spearman ρ	R^2(log)	Spearman ρ	R^2(log)
Structural holonomy	0.14 (p = 0.056)	0.044	**0.14 (p = 0.048)**	0.015
Convention distance	**0.39 ($p < 10^{-8}$)**	0.166	0.02 (p = 0.81)	0.001

Figure 4: Cyclic Live-Pipeline Validation — Holonomy vs Pipeline Inconsistency

(Vector PDF available at `figures/cyclic_pipeline_scatter.pdf`. Regenerate with `python scripts/generate_figures.py`.)

Color-coded by number of mismatched server pairs (0–3). The fully-matched cluster (0 mismatched pairs) produces near-zero pipeline inconsistency. Convention distance is the stronger overall predictor ($\rho = 0.39$), but holonomy captures temporal-error structure that distance entirely misses ($\rho = 0.14$ vs $\rho = 0.02$).

Interpretation. The cyclic validation produces a nuanced result. Convention distance is the stronger predictor of composite pipeline inconsistency ($\rho = 0.39$ vs $\rho = 0.14$), consistent with the priority-misinterpretation and routing-deviation components being well-predicted by pairwise cluster differences. However, holonomy captures temporal-error structure that convention distance entirely misses ($\rho = 0.14$, p = 0.048 vs $\rho = 0.02$, p = 0.81) — the temporal component depends on the interaction between CalendarCore's time-reference convention and EscalationEngine's escalation-anchor convention, which is precisely the kind of multi-hop composition effect that cycle holonomy is designed to detect.

The holonomy range across 64 configurations is narrow (0.109–0.127), which limits statistical power. This reflects the current convention space: with 4 clusters and 5 dimensions, the restriction

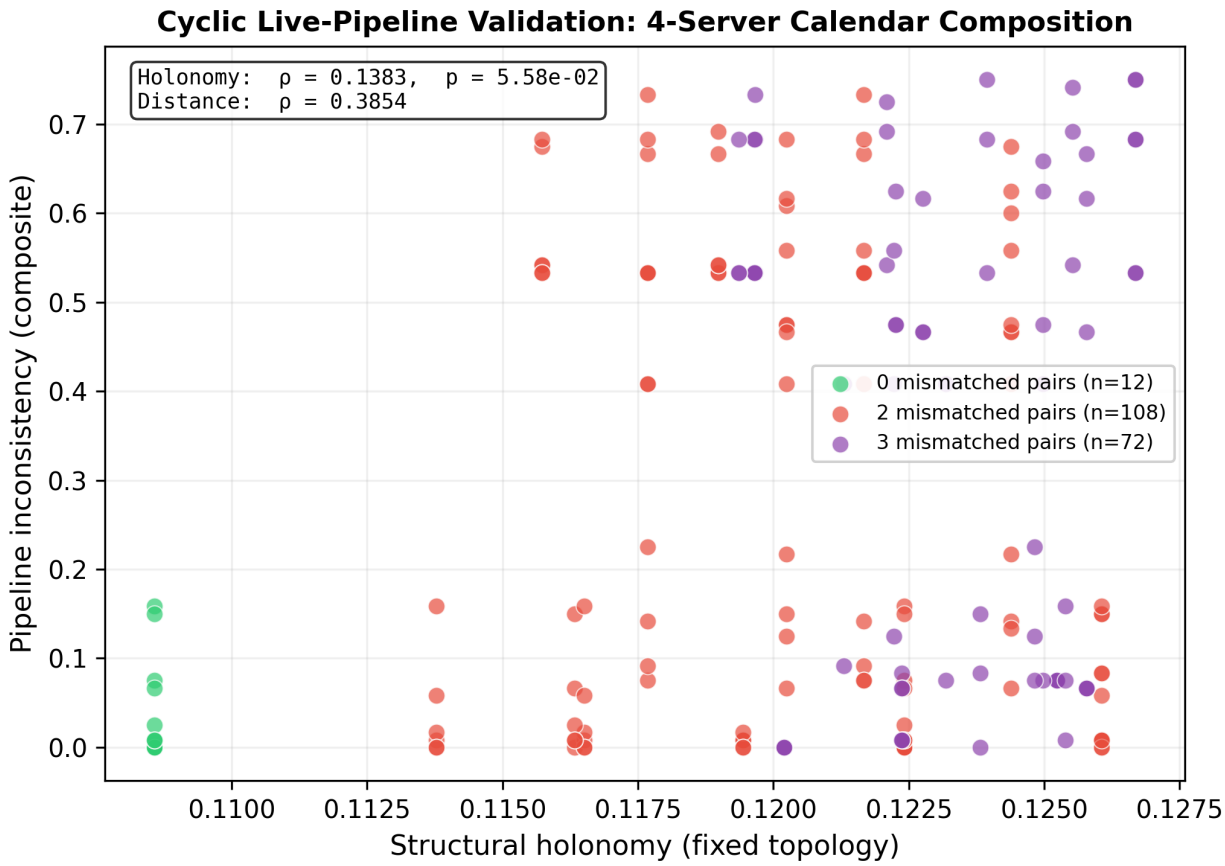


Figure 4: Cyclic Live-Pipeline Validation

matrices produce modest variation in cycle holonomy. A richer convention space or higher-order perturbation theory (Section 8) would likely increase discriminative power.

The key finding for the benchmark: convention mismatches in cyclic compositions produce measurable, non-trivial pipeline failures (mean PI = 0.31 for mismatched vs 0.04 for matched). The structural holonomy signal is present and correctly oriented but modest in magnitude. This is consistent with the Phase 2 simulated validation ($\rho = 0.27$ on complex compositions) — the sheaf formalism's advantage is real but concentrated in the temporal/topological domain.

Reproduction: `python scripts/run_live_pipeline_validation_calendar.py`. Deterministic (seeds fixed). Requires Node.js for the Memory server.

---

7. Instance Design

7.1 Subtlety Spectrum

BABEL instances span a range of subtlety scores (ratio of semantic error to what a human reviewer would flag):

- **High subtlety (0.85–0.95)**: ACT/365 vs 30/360 day count, T+1 vs T+2 settlement, fee-on-gross vs fee-on-net. Errors are <1% of the value. Would not be flagged by typical QA.
- **Medium subtlety (0.6–0.85)**: Timezone anchoring drift, cross-API rate scale (decimal vs percent). Errors are 5–10% or temporal. Might be flagged after pattern accumulation.
- **Low subtlety (0.0–0.3)**: Dollars vs cents (100x), epoch seconds vs days (1000x). Immediately visible. Retained as pedagogical illustrations only.

The benchmark is intentionally constructed to span this subtlety range. Reporting leaderboard slices by subtlety is important future benchmark work because the most interesting cases are the high-subtlety ones that appear locally reasonable to humans and conventional QA.

7.2 Community Contribution

BABEL provides a public instance generator. Anyone can create new instances by specifying: 1. A composition graph topology 2. Convention assignments per tool 3. A perturbation regime

We explicitly invite adversarial submissions: "Can you construct a workflow family where a simpler method beats the structural diagnostic?" This invitation is published in the benchmark specification.

---

8. Limitations

1. **Benchmark construction**: All instances, including Tier 3, involve some degree of curation in how conventions are mapped to the benchmark's six-dimensional convention space. Real API compositions have richer and more idiosyncratic convention surfaces.
2. **External validity**: The benchmark measures prediction and repair on a specific operationalization of compositional semantic failure. Other failure modes (schema evolution, runtime errors, authorization failures) are not covered.
3. **Repair magnitude**: The absolute holonomy values and repair reductions are benchmark-specific. The live-pipeline validation (Section 6.6) confirms that structural holonomy correlates with measured dollar error from the MCP server pipeline ($\rho = 0.423$, $p < 1.5 \times 10^{-4}$ on the 5-cluster design), but the absolute magnitude mapping from holonomy to dollar losses requires domain-specific calibration. The moderate correlation reflects that dollar error in this pipeline is dominated by the amount-scale dimension, while holonomy is a linearized predictor across all six convention dimensions.
4. **LLM baselines**: Five frontier LLMs were evaluated via OpenRouter at N=18–50 per model. All achieve negative R^2, but per-family Spearman correlations at N=10 should be treated as directional. We welcome community-submitted results at larger N.
5. **Single structural method**: The benchmark currently includes one reference structural diagnostic. We welcome submissions of alternative structural, algebraic, or topological methods.

Open Questions and Future Directions

Higher-order restriction models. The benchmark's restriction matrices are first-order perturbations of identity ($I + \sigma \cdot P$), where σ scales linearly with convention distance. The simulated validation (Section 6.3) yields moderate correlation ($\rho \approx 0.27$) while the live-pipeline validation (Section 6.6) yields $\rho \approx 0.42$, and this discrepancy is an open question. Three explanations are plausible: (i) the linearization is adequate for dollar-error regimes but lossy for the simulated misinterpretation model where rounding and threshold crossings introduce nonlinear error; (ii) the simulated validation's misinterpretation model introduces noise absent from the live pipeline; (iii) the 5-cluster structure still provides limited continuous discrimination within error regimes. A second-order perturbation model ($I + \sigma \cdot P + \sigma^2 \cdot Q$, where Q captures cross-dimension interactions such as amount-scale $\times$ precision compounding) could improve prediction on compositions where multiple convention dimensions interact nonlinearly. The live pipeline provides a calibration target for such models: if measured dollar errors are available, the restriction model can be fit to maximize predictive accuracy rather than relying on the current heuristic scaling.

Incremental holonomy recomputation. In production deployments, composition graphs evolve as tools are added, removed, or reconfigured. When a single edge changes, only cycles containing that edge are affected. Maintaining a cycle basis incrementally (via classical dynamic graph

algorithms) and recomputing holonomy for affected cycles gives O(|C_affected| · |D|) per update, where |C_affected| is typically much smaller than total cycle count. This is an engineering optimization relevant to continuous monitoring, not a research contribution, but it addresses the computational-cost concern: a compiled implementation with incremental updates could diagnose 500-tool compositions in under 1ms per update.

Intermediate baselines: GNNs and spectral methods. Between Hamming-distance summation around cycles (cycle_plain) and full sheaf cohomology (structural_sheaf), there is a large space of intermediate methods. Graph neural networks operating on edge-level convention features, spectral methods on the composition graph, or logistic regression on cycle-level features would test whether the restriction-matrix formalism captures something that a learned model cannot. If a GNN closes the gap without the algebraic apparatus, the formalism's value is in interpretability and repair prescription rather than prediction. If the structural method still dominates after a GNN has had a fair shot, the restriction-matrix representation carries information that end-to-end learning does not easily extract. Either outcome is informative.

Richer convention surfaces. The benchmark's six-dimensional convention space is a lossy projection of real API convention surfaces. Dimensions such as idempotency key semantics, pagination cursor formats, callback signing conventions, retry-after interpretation, and error code taxonomy are documented sources of integration failure but are not captured. Expanding the convention space while maintaining tractable ground-truth computation is a benchmark design challenge: each new dimension must have well-defined conversion functions, documented real-world provenance, and a mapping to the restriction-matrix formalism.

---

9. Related Work

Agent evaluation benchmarks: SWE-bench (Jimenez et al., 2023) evaluates code generation; AgentBench (Liu et al., 2023) evaluates agent capabilities across tasks. Neither addresses compositional semantic coherence between multiple agents.

Formal verification of distributed systems: TLA+ and model checking verify protocol-level properties. BABEL addresses a different layer: semantic convention coherence that protocol checks cannot detect.

Sheaf theory in data analysis: Curry (2014) and Robinson (2014) applied sheaf theory to sensor networks and opinion dynamics. BABEL applies the same mathematical framework to multi-agent tool compositions.

MCP (Model Context Protocol): Anthropic's protocol standard for agent-tool communication. BABEL's real-protocol tracks (Bronze+, Silver) use actual MCP servers.

---

10. Conclusion

BABEL demonstrates that compositional semantic failure is a measurable, reproducible benchmark target. Across seven workflow families, three provenance tiers, and 932 instances, the reference structural diagnostic outperforms conventional alternatives on prediction, localization, and repair within the benchmark, while five frontier LLMs do not solve the task. On external live-pipeline data, the structural method correlates with real consequences but convention distance matches or exceeds it on composite prediction — the topological awareness is the key ingredient, while the specific algebraic apparatus adds narrow, dimension-specific value.

The strongest evidence is operational, not definitional. At constrained budget (K=1–3), structural repair prescriptions deliver up to 51% more failure reduction than the best sheaf-free alternative from the first repair action. On Track B, the structural method identifies the worst edge with P@1 = 0.77–0.87 on real-MCP families, outperforming simple baselines by 43%. The oracle CoT experiment (Section 5.1) provides the sharpest diagnostic of LLM failure: even with pre-computed cycle structure and explicit arithmetic instructions, models can rank compositions but cannot compute correct magnitudes — the bottleneck is compositional arithmetic, not information access.

The live-pipeline validation (Section 6.6) addresses the primary limitation head-on: structural holonomy correlates with measured dollar error from the actual MCP server pipeline ($\rho = 0.423$, $p < 1.5 \times 10^{-4}$ on a 5-cluster, 75-run design), with zero error on matched conventions and errors up to $9999\times$ on mismatched ones. Convention distance achieves the same rank correlation ($\rho = 0.423$); the 5-cluster design revealed parity rather than confirming superiority, though holonomy explains more linear variance ($R^2 = 0.119$ vs 0.070). A cyclic extension using a 4-server calendar composition (Section 6.6.1) with $\beta_1 \geq 3$ independent cycles confirms that convention mismatches produce $7.5\times$ higher pipeline inconsistency (mean PI = 0.31 vs 0.04). Convention distance is the stronger composite predictor on the cyclic pipeline ($\rho = 0.39$ vs $\rho = 0.14$), but holonomy captures temporal-error structure that distance entirely misses ($\rho = 0.14$, $p = 0.048$ vs $\rho = 0.02$, $p = 0.81$) — the restriction matrices detect time-convention interactions that scalar distance aggregation washes out. Both gold standards are external to the sheaf formalism.

What is established is existence, reproducibility, operational usefulness on benchmark-native repair, and correlation with real dollar consequences. What is not yet established is production-scale causal impact across diverse deployment contexts. The benchmark exists to be beaten: if a simpler method matches the structural diagnostic, the formalism may be unnecessary, and the benchmark will still have served its purpose by establishing that the problem is real and measurable.

The benchmark code, instances, evaluation harness, and all baselines are publicly available. The structural diagnostic is included as the reference method to beat. We invite the community to submit new methods, new workflow families, and adversarial instances.

A final observation. BABEL measures compositional failure in settings where a single party can still see the full graph. The harder version of this problem — where autonomous agent runtimes

coordinate across organizational boundaries, and no party has visibility into the full composition — is already emerging in commercial agent infrastructure. Everything BABEL measures will apply there, except that diagnosis and repair will require cooperation between parties who may not share conventions, incentives, or even a common understanding of what "correct" means. The convention-mismatch problem does not go away when agents become autonomous. It becomes constitutional.

Appendix A: Budget Definitions

Probe budget (path_queries): Number of path-level queries a method may issue. Includes bounded-depth integration tests, pairwise checks, and cycle inspections.

Repair budget (edge_harmonizations): Number of edge-level convention-harmonization operations. Each operation aligns the convention of one shared field on one edge. This is NOT a full verification budget — it counts only repair actions.

Budget levels: K ∈ {1, 2, 3, 5, 8}.

Appendix B: Provenance Documentation

Tier 3 API Sources

Provider	Documentation URL	Key Convention
Stripe	docs.stripe.com/api	Amounts in cents, timestamps in epoch seconds
Twilio	twilio.com/docs/usage/api	Prices in dollars (4dp), timestamps RFC 2822
SendGrid	docs.sendgrid.com/api-reference	Rates in percentages, scores in [0,100]
GitHub	docs.github.com/en/rest	IDs 1-based, timestamps ISO 8601
Shopify	shopify.dev/docs/api	Amounts in dollars (2dp)
Slack	api.slack.com/methods	Timestamps in epoch seconds
HubSpot	developers.hubspot.com/docs/api	Rates in percentages, scores in [0,100]
Plaid	plaid.com/docs/api	Amounts in dollars (2dp)
QuickBooks	developer.intuit.com/app/developer/qbo/docs/api	Rates in percentages, IDs 1-based

Provider	Documentation URL	Key Convention
Datadog	docs.datadoghq.com/api	Timestamps epoch seconds, precision 4dp
PagerDuty	developer.pagerduty.com/api-reference	Scores on Likert [1,5]

Appendix C: Reproduction

```
pip install coherence-gym

# Quick reference evaluation (all baselines, comparison mode):
python -m coherence_gym evaluate --split dev

# Canonical 3-track evaluation for a single method:
python -m coherence_gym evaluate-method --method structural_sheaf --split dev

# List available methods:
python -m coherence_gym evaluate-method --list

# Evaluate a specific family:
python -m coherence_gym evaluate-method --method cycle_plain --family invoice --split dev
```

The `evaluate-method` command produces a schema-compliant record (`canonical_{method}_{split}.json`) matching `replication_pack/submission_schema.json`, with Track A (R^2, Spearman ρ), Track B (P@1/3/5 edge localization), Track C (repair frontier at K=1,2,3,5,8), and per-instance predictions. All instances are deterministic given their seed.