Morphogenetic Harness: What LLM Agents Should and Should Not Borrow from Developmental Biology

Anonymous Authors[1]

Abstract

Developmental biology offers two kinds of inspiration for LLM agent design: *structural* principles (cell differentiation, morphogen gradients, gene regulatory networks) and *dynamic* processes (evolution, natural selection, population dynamics). We build a *morphogenetic harness*—a framework that applies both to LLM agents—and test which kind of inspiration actually helps. In Experiment 1 (static harness), we route queries to task-specific "cognitive skills" inspired by cell differentiation, testing on GSM8K, HumanEval, and MMLU across three LLMs. Skill routing significantly improves accuracy over unprompted baselines (Wilcoxon signed-rank $p < 0.05$ for Haiku and Grok), with gains concentrated on math and logic. The gain over a generic chain-of-thought prompt is not significant in aggregate; a cross-routing control (applying the *wrong* skill) instead pinpoints *correct task–skill matching* as the operative mechanism (40pp drop on math under mis-routing). Ablation reveals that *neither structure nor domain knowledge alone suffices*—their synergy is key, mirroring the dual role of morphogens in biological patterning. In Experiment 2 (dynamic harness), we run evolutionary LLM colonies with bio-inspired reproduction, lateral inhibition, and fitness tracking. We find that evolution provides *no benefit beyond the pass@k effect* of running multiple agents. Cross-routing experiments confirm that the value of biological metaphor lies in **the blueprint, not the process**: static differentiation helps, but runtime evolution does not. We formalize this distinction through Ashby's Law of Requisite Variety and discuss implications for the emerging agent harness paradigm.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. Introduction

Bio-inspired AI has a long history, from neural networks to evolutionary algorithms. The recent surge in LLM agents has revived interest in biological metaphors: multi-agent "colonies" (Hong et al., 2024), evolutionary prompt optimization (Fernando et al., 2024), and self-organizing agent teams. But which biological principles actually transfer to LLM systems?

This paper makes a simple but important distinction. Developmental biology offers two types of design knowledge:

1. **Blueprints**: How to structure a system—cell types, differentiation, spatial organization. In LLM terms: *what cognitive skills to have and how to assign them.*
2. **Processes**: How to run a system over time—evolution, reproduction, selection. In LLM terms: *whether to evolve agents rather than engineer them.*

We build a *morphogenetic harness* that implements both, and test each in controlled experiments. Our main finding: **borrow the blueprint, skip the process**. Static skill routing (Experiment 1) provides large, statistically significant improvements on standard benchmarks. Evolutionary dynamics (Experiment 2) provide no benefit beyond what comes from simply running more agents.

This is not a failure of our specific implementation—we show through careful ablation that the confound is fundamental: any multi-generation evolutionary system implicitly grants multiple attempts per problem, inflating apparent performance via a pass@k effect. Controlling for this, evolution's contribution vanishes.

Contributions.

1. A unified *morphogenetic harness* framework that translates developmental biology concepts (GRN, morphogen gradients, Notch signaling, MAPK) into LLM agent design patterns.
2. Controlled experiments on standard benchmarks (GSM8K, HumanEval, MMLU) across three LLMs,

with Wilcoxon signed-rank tests establishing that static skill routing significantly improves accuracy over vanilla prompting for Haiku ($p = 0.025$) and Grok ($p = 0.006$), with per-cell gains reaching $+13.3$pp on GSM8K.

3. A negative result on evolutionary LLM colonies with formal identification of the pass@k confound.

4. Evidence that the value of bio-inspiration lies in structural principles, not dynamic processes—"borrow the blueprint, skip the process."

2. Related Work

Prompt Engineering as Cognitive Scaffolding. Chain-of-thought (Wei et al., 2022), tree-of-thought (Yao et al., 2023), and self-consistency (Wang et al., 2023) demonstrate that external cognitive structure improves LLM reasoning. Our cognitive skills are task-specific instances of this general scaffolding principle, informed by developmental biology's concept of cell-type-specific gene expression.

Multi-Agent LLM Systems. MetaGPT (Hong et al., 2024), CAMEL (Li et al., 2023), and AutoGen (Wu et al., 2023) use multiple agents with assigned roles. Our framework differs in grounding role assignment in a formal biological model (GRN + morphogen gradients) rather than hand-designed personas.

Evolutionary LLM Methods. PromptBreeder (Fernando et al., 2024) evolves prompts; EvoPrompt uses evolutionary algorithms for prompt optimization. We take the metaphor further by implementing full population dynamics with reproduction, selection, and fitness landscapes—and find it does not help.

Automated Agent Design. A growing body of work automates agent architecture design. ADAS (Hu et al., 2025) uses a meta agent to program novel agents in code space, progressively discovering designs that outperform hand-crafted systems. MaAS (Zhang et al., 2025) frames the problem as neural architecture search, sampling query-dependent agent configurations from a probabilistic supernet. ARM (Yao et al., 2025) evolves agentic reasoning modules that generalize across tasks and models—notably finding that optimized single-agent CoT often matches complex multi-agent systems. SCALE (Wang et al., 2026) shows that a small set of task-level workflows suffices, obviating per-query workflow generation. AWO (Abuzakuk et al., 2026) discovers recurring tool-call patterns and compresses them into deterministic meta-tools. Our work complements this line by grounding agent design in developmental biology rather than automated search, and by providing controlled evidence that static structural design outperforms dynamic evolutionary search.

Bio-Inspired Agent Design. Our morphogenetic harness draws on Turing's reaction-diffusion model (Turing, 1952) for morphogen-based routing, Wolpert's positional information (Wolpert, 1969) for cell differentiation, and the Price equation (Price, 1970) for tracking evolutionary change.

3. The Morphogenetic Harness

3.1. Genome: Gene Regulatory Network

Each agent i has a genome $\mathbf{g}_i \in \mathbb{R}^{25}$ encoding a gene regulatory network (GRN). The GRN controls cognitive parameters: temperature, verbosity, verification depth, and domain focus. Gene expression dynamics follow:

$$g_j^{(t+1)} = \sigma\left(\sum_k w_{jk}\, g_k^{(t)} + b_j\right) \tag{1}$$

where σ is a sigmoid activation and $\mathbf{W}$ is the regulatory interaction matrix.

3.2. Differentiation: Cognitive Skills via Morphogens

Inspired by morphogen gradients in embryonic development, we define **cognitive skills** as task-specific system prompts with two components:

- **Protocol**: A structured reasoning sequence (analogous to a signaling cascade).
- **Domain knowledge**: Task-specific heuristics (analogous to tissue-specific gene expression).

Static routing (Experiment 1) assigns skills by task type—like morphogen-determined cell fate. **Dynamic routing** (Experiment 2) lets skills emerge through evolution—like adaptive differentiation.

3.3. Lateral Inhibition: Notch Signaling

In biology, Notch signaling ensures neighboring cells adopt different fates. We implement this as a penalty on agent role similarity within a colony, encouraging cognitive diversity:

$$\mathcal{L}_{\text{Notch}}(\mathbf{g}_i) = -\lambda_2(\mathbf{L}) \cdot \text{diversity}(\{\mathbf{g}_i\}) \tag{2}$$

where $\lambda_2(\mathbf{L})$ is the algebraic connectivity of the agent interaction graph.

3.4. Reproduction: MAPK Signaling

Agents reproduce when their fitness exceeds a threshold, gated by MAPK signal strength:

$$P(\text{reproduce}_i) = \mathbb{1}[\text{MAPK}_i > \theta] \cdot f(\text{fitness}_i) \tag{3}$$

We find empirically that $\theta = 0.05$ allows only ∼40% of runs to produce viable offspring.

3.5. Fitness and Evolution: Price Equation

We track evolutionary dynamics using the Price equation:

$$\Delta \bar{z} = \underbrace{\mathrm{Cov}(w_i, z_i)}_{\text{selection}} + \underbrace{E(w_i \Delta z_i)}_{\text{transmission}} \tag{4}$$

where z_i is a phenotypic trait and w_i is relative fitness. This decomposes performance change into selection pressure and heritable variation.

3.6. Information-Theoretic Framing: Ashby's Law

We connect our framework to Ashby's Law of Requisite Variety: a controller must have variety $\geq$ the system it regulates. A single-skill agent has variety $V = 1$; a harness with k skills has $V = k$. The *cognitive level* of an agent system is:

$$\mathcal{C} = \frac{H_{\text{agent}}}{H_{\text{env}}} = \frac{\log k + H(\text{routing})}{H(\text{tasks})} \tag{5}$$

Our experiments test whether increasing k (more skills) or increasing $H(\text{routing})$ (evolutionary adaptation) better improves $\mathcal{C}$.

4. Experiment 1: Static Harness (Skill Routing)

4.1. Setup

Benchmarks. GSM8K (50 math problems) (Cobbe et al., 2021), HumanEval (40 code problems) (Chen et al., 2021), MMLU formal logic + logical fallacies (50 problems) (Hendrycks et al., 2021).

Models. Claude Haiku (claude-haiku-4-5), GLM-4-Flash, Grok-3-Mini-Fast.

Conditions. (1) **Vanilla**: no system prompt. (2) **Single prompt**: one general-purpose CoT prompt for all tasks. (3) **Skill-routed**: task-specific cognitive skill (math/code/logic) selected by oracle routing.

Repetitions. 3 per cell. Total: $3 \times 3 \times (50+40+50) \times 3 =$ 3,780 API calls.

4.2. Results

Skill routing consistently improves GSM8K and MMLU accuracy across all three models. HumanEval gains are modest, likely because code generation benefits less from prompt scaffolding than math and logic. For GLM the single-prompt condition outperforms skill-routed, suggesting that weaker models may benefit more from general CoT than from domain-specific structure—consistent with our Ashby framing: requisite variety must be calibrated to model capacity; exceeding a model's ability to follow structured instructions converts directed variety into noise.

Table 1. Experiment 1: Accuracy (%) on standard benchmarks. Bold marks best per model per column. Superscripts on *Skill-Routed* cells denote Wilcoxon signed-rank significance vs *Vanilla*, paired at the per-problem level ($n = 50/40/50$ for GSM8K/HE/MMLU; $^{*}p < 0.05$, $^{**}p < 0.01$). The $All^{\dagger}$ column reports the pooled per-problem mean ($n = 140$) and its pooled test. Each cell averages 3 repetitions.

Model	Condition	GSM8K	HE	MMLU	All†
Haiku	Vanilla	86.0	43.3	79.3	71.4
Haiku	Single Prompt	95.3	45.0	80.0	75.5
Haiku	Skill-Routed	**97.3****	**45.0**	**81.3**	**76.7***
GLM	Vanilla	78.7	40.0	56.7	59.8
GLM	Single Prompt	**88.0**	**47.5**	**64.0**	**67.9**
GLM	Skill-Routed	85.3	43.3	59.3	64.0
Grok	Vanilla	84.7	45.0	88.0	74.5
Grok	Single Prompt	98.0	42.5	90.0	79.3
Grok	Skill-Routed	**98.0****	41.7	**94.7***	**80.7****

Statistical testing. We assess reliability with Wilcoxon signed-rank tests paired at the per-problem level (each condition's mean over 3 reps per problem; $n = 50, 40, 50$ for GSM8K, HumanEval, MMLU). Pooling all three benchmarks, *Skill-Routed* > *Vanilla* is significant for Haiku ($W = 115$, $p = 0.025$) and Grok ($W = 48.5$, $p = 0.006$) but not GLM ($p = 0.31$). The effect is concentrated on math and logic: Haiku and Grok each show $p = 0.007$ on GSM8K, and Grok reaches $p = 0.023$ on MMLU; all HumanEval comparisons are non-significant. *Skill-Routed* > *Single-Prompt*, however, is **not** significant in aggregate for any model (Haiku $p = 0.41$, GLM $p = 0.12$, Grok $p = 0.28$). The measurable improvement over no-prompt baselines is substantially explained by the presence of structured CoT, not by task-specific specialization per se—motivating the cross-routing ablation in §4.4, which isolates the contribution of *correct* task–skill matching.

4.3. Ablation: Structure vs. Knowledge

Our cognitive skills combine structured protocol ("parse → model → solve → verify") with domain knowledge (common pitfalls, output format). We ablate each component on a subset of problems.

The synergy between structure and knowledge mirrors the biological principle that morphogens provide both positional information (where am I?) and differentiation signals (what should I become?). Neither alone determines cell fate.

Table 2. Ablation on pilot data (Claude Haiku, 5 representative problems per task).

Component	Math	Logic
Vanilla	12.0	44.6
Structure only	80.0	65.8
Knowledge only	84.0	57.3
Full skill	**100.0**	**66.4**

Table 3. Cross-routing results (pilot data, Claude Haiku).

Routing	Math	Logic
Correct skill	100.0	69.2
Wrong skill	60.0	64.7
Δ	-40.0	-4.6

4.4. Cross-Routing: Does Correct Matching Matter?

Applying the wrong cognitive skill (e.g., math skill on logic problems) tests whether the benefit comes from *having* a skill or from *correct* routing.

The 40pp drop on math confirms that routing quality, not just prompt presence, drives the benefit. This is consistent with morphogenetic patterning: a cell in the wrong position adopts the wrong fate.

5. Experiment 2: Dynamic Harness (Evolutionary Colony)

5.1. Setup

We run an evolutionary colony of LLM agents on 164 HumanEval-style coding problems using GLM-4-Flash:

- Initial population: 2 agents with random GRN genomes
- Reproduction gated by MAPK signaling ($\theta = 0.05$)
- 20 generations of evaluation and selection
- Notch lateral inhibition for role diversity
- 5 independent runs (seeds 42–46)

Baselines. (1) Single agent, one attempt per problem. (2) 8 independent agents, same total compute (pass@8 baseline).

5.2. Results

5.3. The pass@k Confound

The colony's apparent improvement (67.8% $\rightarrow$ 77.4%) is confounded. The benchmark scores a problem as "passed" if *any* agent in *any* generation solves it. With 8 agents $\times$ 20 generations, each problem gets $\sim$160 attempts—this is pass@k, not pass@1.

Table 4. Experiment 2: Evolutionary colony vs baselines on HumanEval-style problems (GLM-4-Flash).

Condition	Pass@1	Notes
Single Agent	67.8%	Strict pass@1
8 Indep. Agents	–	pass@8, fair comp.
Colony+Evol.	77.4%	*Confounded*
Run 2 (best)	90.7%	Pop. grew to 8
Run 4 (worst)	72.0%	Fitness spikes

The 8-agent baseline provides a fair comparison: same number of agents, each attempting every problem once. If colony+evolution $\approx$ 8-agent baseline, then evolution contributed nothing beyond the extra attempts.

5.4. Dynamics: Phase Transitions and Bifurcation

Despite the negative result on aggregate performance, the evolutionary dynamics reveal interesting structure:

- **MAPK phase transition**: With $\theta = 0.05$, only 2/5 runs achieve reproduction. Lowering to $\theta = 0.001$ allows reproduction in all runs.
- **Stability bifurcation**: Run 2 (pass@1 = 0.907) shows stable fitness evolution with low variance ($\sigma = 1.23$), while Run 4 (pass@1 = 0.720) shows catastrophic fitness spikes.
- **Connectivity predicts success**: Successful runs have higher algebraic connectivity ($\lambda_2 = 54.0$ vs 47.4), suggesting agent communication topology matters.

These dynamics are scientifically interesting but do not translate to practical performance improvements over the simpler static harness.

6. Discussion

6.1. Borrow the Blueprint, Skip the Process

Our central finding is a design principle for bio-inspired LLM agents: structural concepts from developmental biology (cell types, morphogen-based routing, GRN-structured genomes) transfer productively to LLM systems, but dynamic processes (evolution, natural selection, population dynamics) do not—at least not naively.

Why? We hypothesize two reasons:

1. **LLMs are not cells.** Biological evolution operates on organisms with continuous, heritable variation. LLM "genomes" (prompts, temperatures) are discrete and

shallow—there is not enough genetic material for selection to act on meaningfully.

2. **The evaluation bottleneck.** Each evolutionary generation requires expensive API calls. With a fixed compute budget, evolution spends resources on exploration that could be spent on exploitation (more attempts with a good prompt).

6.2. Ashby's Law and the Routing Principle

Our results support a cybernetic interpretation through Ashby's Law, though with an important nuance. A harness with k skills does not *guarantee* k-fold cognitive variety—if skills are redundant, effective variety is less than k. Conversely, evolutionary search could in principle discover k orthogonal strategies. The key empirical finding is probabilistic: *directed* design (skill routing with correct matching) is far more likely to achieve full-rank cognitive variety than *undirected* search (evolution), because the latter must discover task-relevant diversity from a high-dimensional search space where most directions are redundant. Our cross-routing experiment (Table 3) confirms this—wrong routing destroys the benefit, showing that variety must be aligned with task structure. In Ashby's terms, the harness should be a **designed regulator**, not an evolved one.

6.3. Limitations

(1) Our static harness uses oracle routing; automatic routing remains future work. (2) The evolutionary experiment uses GLM-4-Flash only; stronger models may benefit differently from evolution. (3) Our skill library is small (3 skills); scaling to 20+ skills may change the routing calculus. (4) The pass@k confound may be addressable with per-generation-only scoring, which we leave to future work. (5) In aggregate Wilcoxon tests, *Skill-Routed* does not significantly exceed a single generic CoT prompt ($p > 0.12$ for all three models). The evidence that skill specialization matters rests on the cross-routing control (Table 3)—applying the wrong skill costs 40pp on math—rather than on a universal accuracy delta over CoT. Our static-harness claim is thus that *correct routing* is the operative mechanism.

7. Conclusion

We introduced the morphogenetic harness, a framework that translates developmental biology principles into LLM agent design. Through controlled experiments on standard benchmarks with three LLMs, we found that *static* biological inspiration (structured skill routing) consistently improves performance, while *dynamic* biological inspiration (evolutionary colony dynamics) does not survive controlling for the pass@k confound. Our key message: developmental biology offers excellent blueprints for cognitive architecture, but LLM agents should be engineered, not evolved. The value of the morphogenetic metaphor lies in telling us *what to build*, not *how to search for it*.

LLM and Agent Usage Disclosure

Following the workshop's call for honest disclosure, we describe our LLM usage at two levels.

As experimental subjects. Claude Haiku (claude-haiku-4-5), GLM-4-Flash, and Grok-3-Mini-Fast were the *targets* of our benchmarks (Experiment 1); GLM-4-Flash additionally served as the agent substrate for the evolutionary colony (Experiment 2). All API calls, prompts, and responses are determined by the scripts described in the main text.

As a collaborator in research and writing. We used an LLM coding agent (Claude Code, an interactive CLI tool running Claude models) as an active collaborator across the project: (i) *ideation*—refining the blueprint-vs-process framing, pressure-testing the pass@k confound argument, and suggesting the cross-routing ablation; (ii) *implementation*—drafting portions of the benchmark harness, data-analysis scripts, and the Wilcoxon signed-rank post-hoc analysis; (iii) *writing*—drafting and revising sections of this manuscript, including tables and figures. All scientific claims, experimental results, and final editorial decisions are the authors' responsibility, and every LLM-generated artifact was reviewed and modified before inclusion. The statistical significance testing reported in §4.4 and Table 1 was added in response to collaborative review that identified the need to distinguish *Skill-Routed vs. Vanilla* from *Skill-Routed vs. Single-Prompt*.

References

W. R. Ashby. *An Introduction to Cybernetics*. Chapman & Hall, 1956.

M. Chen, J. Tworek, H. Jun, et al. Evaluating large language models trained on code. *arXiv:2107.03374*, 2021.

K. Cobbe, V. Kosaraju, M. Bavarian, et al. Training verifiers to solve math word problems. *arXiv:2110.14168*, 2021.

C. Fernando, D. Banarse, H. Michalewski, et al. PromptBreeder: Self-referential self-improvement via prompt evolution. *arXiv:2309.16797*, 2024.

D. Hendrycks, C. Burns, S. Basart, et al. Measuring massive multitask language understanding. *ICLR*, 2021.

S. Hong, M. Zhuge, J. Chen, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. *ICLR*, 2024.

G. Li, H. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem.

CAMEL: Communicative agents for "mind" exploration of large language model society. *NeurIPS*, 2023.

G. R. Price. Selection and covariance. *Nature*, 227:520–521, 1970.

A. M. Turing. The chemical basis of morphogenesis. *Phil. Trans. R. Soc. B*, 237(641):37–72, 1952.

X. Wang, J. Wei, D. Schuurmans, et al. Self-consistency improves chain of thought reasoning in language models. *ICLR*, 2023.

J. Wei, X. Wang, D. Schuurmans, et al. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 2022.

L. Wolpert. Positional information and the spatial pattern of cellular differentiation. *J. Theor. Biol.*, 25(1):1–47, 1969.

Q. Wu, G. Banber, K. Somani, et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv:2308.08155*, 2023.

S. Yao, D. Yu, J. Zhao, et al. Tree of thoughts: Deliberate problem solving with large language models. *NeurIPS*, 2023.

S. Hu, C. Lu, and J. Clune. Automated design of agentic systems. *ICLR*, 2025.

G. Zhang, L. Niu, J. Fang, K. Wang, L. Bai, and X. Wang. Multi-agent architecture search via agentic supernet. *ICML*, 2025.

B. Yao, S. K. R. Malay, and V. Yadav. ARM: Discovering agentic reasoning modules for generalizable multi-agent systems. *arXiv:2510.05746*, 2025.

Z. Wang, B. Xu, Y. Yuan, H. Shen, and X. Cheng. Do we always need query-level workflows? Rethinking agentic workflow generation for multi-agent systems. *arXiv:2601.11147*, 2026.

S. Abuzakuk, A.-M. Kermarrec, R. Sharma, R. M. Veski, and M. de Vos. Optimizing agentic workflows using meta-tools. *arXiv:2601.22037*, 2026.