When to Start Communicating: Adaptive Stigmergy Gates Improve Multi-Agent RL Training Dynamics

Ashish Khandelwal
Independent Researcher

February 12, 2026

Abstract

Shared communication channels are often treated as an unconditional good in multi-agent reinforcement learning (MARL): giving agents access to messages, shared memory, or stigmergic traces should improve coordination. Yet communication can also destabilize learning when it is available before it is informative, inducing spurious correlations that interfere with credit assignment and representation learning. We present an *Adaptive Stigmergy Engine* that controls *when* agents may communicate through a shared pheromone field. The engine uses a simple gate evaluated mid-episode to activate the field only under population-level distress signals (population decline, energy crisis, or delivery inequality).

We evaluate the approach in a JAX multi-agent foraging simulation trained with PPO [17] and evolutionary pressure (birth, death, and reproduction). Across 10 seeds in a clean end-to-end run (10M training steps) and a 20-experiment development arc, adaptive deployment outperforms both ALWAYS_ON and ALWAYS_OFF baselines, improving hard-task deliveries by +34% and +19% respectively while yielding the lowest population variance. Analysis suggests a *channel-curriculum* mechanism: policies first acquire robust individual foraging competence with the channel disabled, then incorporate pheromone information as a residual correction once it becomes predictive. Timing ablations confirm that the gains arise from training dynamics rather than execution-time switching. Cross-domain validation with LLM agent collectives (481 runs across two models and three task domains) confirms the principle: on adversarial tasks with planted decoy bugs, communication exposure increases false consensus capture monotonically (0% at 0 shared rounds to 100% at 3 shared rounds), demonstrating that communication costs depend on information quality in both RL and LLM settings.

1 Introduction

Coordination is a central challenge in multi-agent reinforcement learning (MARL). Agents that learn independently must still solve joint problems: avoid redundant exploration, allocate roles, and exploit complementary information under partial observability. A common response is to provide a shared information channel—explicit communication, differentiable message passing, shared memory, or an external "blackboard"—so that coordination can be learned end-to-end [8, 14, 19]. Stigmergy offers a particularly lightweight variant: agents coordinate indirectly by writing to and reading from a persistent medium (e.g., pheromone trails), as in social insects [6, 9].

However, a coordination channel is not a free lunch. In most MARL systems, communication is treated as *static*: the channel is either always available (always-on) or fully removed (always-off). This design choice implicitly assumes that "more information" is monotonically helpful. In practice, an always-on channel can be actively harmful early in learning. When agents begin training, shared signals may be empty, unstructured, or noisy; policies that condition on them can

learn accidental correlations that later impede credit assignment and representation learning. This risk is exacerbated in on-policy deep RL, where changing data distributions and unstable value targets already create strong nonstationarities.

1.1 When communication hurts

Our foraging experiments make this failure mode concrete. We study a multi-agent foraging task in which agents must discover food, carry it back to a nest, and maintain viability under evolutionary pressure (birth, death, and reproduction). Agents can optionally interact through a shared pheromone field: a grid-valued state that agents may read from and write to, inspired by stigmergic coordination in ant colonies. Intuitively, pheromones should help with exploration, trail following, and division of labor.

Surprisingly, always-on pheromone access can degrade performance during training. In our setting, the ALWAYS_ON condition exhibits the worst performance at 5M training steps, even compared to a no-communication baseline. A plausible explanation is that at step 0 the pheromone field is effectively uninformative: deposits are random, spatial structure has not yet emerged, and the field provides a high-dimensional input stream that is easy for a function approximator to overfit. Policies can then entangle early pheromone features with action selection, creating representational interference that slows or prevents the later formation of robust foraging skills.

This observation reframes the communication design problem: the question is not only *what* agents should share, but *when* the channel should be available. The appropriate answer may change over the course of training, as agents transition from basic competence (individual skill acquisition) to coordinated competence (collective strategy formation).

1.2 Adaptive stigmergy as a channel-curriculum

We propose an *Adaptive Stigmergy Engine* that gates access to the pheromone field based on population-level distress signals. Rather than learning a continuous attention weight over communication, we start with a simple, interpretable mechanism: a rule-based gate evaluated at a fixed midpoint of each episode that decides whether the pheromone channel is active. The gate triggers when any of three conditions indicate that the population is struggling to sustain productive foraging: (i) population decline, (ii) an energy crisis, or (iii) delivery inequality across agents.

The key hypothesis is that adaptive deployment induces a *channel-curriculum* [3]: by default, agents learn to forage using only local observations, which encourages stable individual competence and reduces reliance on a noisy shared channel. When conditions warrant it, the gate activates and agents learn to use pheromone information as a residual correction layered on top of a competent base policy. This sequencing resembles curriculum learning in that it structures the informational difficulty of the learning problem over time, but here the "curriculum" is over communication availability rather than task instances.

1.3 Results preview

Across easy and hard variants of the task, the adaptive condition dominates both static baselines: it achieves the highest mean deliveries while also exhibiting the lowest population variance and reduced tail risk. The gains are nonlinear in training duration: at 5M training steps, all modes are comparable, but by 10M steps adaptive gating pulls ahead, consistent with a learning-dynamics explanation rather than an execution-time trick. A timing ablation further supports this interpretation: varying the gating time without altering training dynamics eliminates the benefit, indicating that the advantage arises from how the policy is shaped during training rather than

from mid-episode switching per se. We further validate the principle in a fundamentally different setting—LLM agent collectives performing collaborative bug-finding—where adversarial tasks with planted decoy bugs reveal dosage-dependent false consensus cascades that scale monotonically with communication exposure.

1.4 Paper structure

Section 2 reviews relevant work on stigmergy, MARL communication, and curriculum learning. Section 3 describes the environment, pheromone system, and adaptive gate. Section 4 presents experimental results and ablations. Section 5 validates the core finding in a different domain—LLM agent collectives performing collaborative bug-finding—and reveals dosage-dependent epistemic cascades under adversarial conditions. Section 6 analyzes the channel-curriculum mechanism and discusses limitations. Section 7 concludes.

2 Related Work

2.1 Stigmergy in Multi-Agent Systems

Classical stigmergy describes how collectives coordinate through persistent modifications to the environment. Grassé's studies of termite construction framed stigmergy as a mechanism by which local actions leave traces that bias future actions, enabling global structure without centralized control [9]. In artificial systems, ant colony optimization (ACO) operationalized this idea via pheromone trails that guide distributed search, demonstrating that indirect communication through a shared medium can yield robust collective problem solving [6].

In robotics and multi-agent coordination, "digital" stigmergy generalizes pheromone-like traces to programmable fields: agents write local markers (e.g., occupancy, gradients, task claims) that other agents can later sense and exploit. This approach is attractive because it is decentralized, asynchronous, and naturally compatible with partial observability. A key gap, however, is that most stigmergic systems assume a static design: the stigmergic channel is always available and its influence is governed by fixed rules. Comparatively little work asks the meta-question of *when* stigmergy should be active during learning, especially when the field is initially uninformative or noisy.

2.2 Communication in Multi-Agent Reinforcement Learning

A large body of MARL research learns explicit communication policies that determine *what* information to share. Differentiable communication mechanisms such as DIAL and CommNet enable agents to exchange learned messages end-to-end with gradient-based training [8, 19]. Attention-based variants (e.g., TarMAC) further structure message aggregation, helping agents focus on relevant teammates [5]. In parallel, the centralized-training / decentralized-execution (CTDE) paradigm has become a standard recipe for scalable MARL, using centralized critics or value factorization to stabilize learning while preserving decentralized policies at test time [16, 24].

These approaches primarily address message content, routing, and credit assignment under a fixed communication interface. Our setting is deliberately different: the shared channel content is not a learned message but a stigmergic pheromone field with a fixed read/write semantics. The main design choice is temporal: rather than asking agents to learn what to communicate, we ask when the channel should be made available so that it improves learning rather than corrupting it.

2.3 Curriculum Learning in RL

Curriculum learning formalizes the idea that learning can be accelerated by presenting training experiences in a meaningful progression, often from easier to harder tasks [3]. In reinforcement learning, related ideas appear as staged training, auxiliary prediction objectives, and representation-learning scaffolds that improve sample efficiency and stability; UNREAL is a canonical example where auxiliary tasks help shape useful features for control [10].

Our results connect curriculum ideas to communication design. We introduce a *channel-curriculum*: instead of progressing task difficulty, we progressively expose an additional information channel (stigmergic state) only after agents have acquired a competent base policy. This perspective is consistent with the broader observation that adding auxiliary information too early can induce representational interference and hinder credit assignment, particularly in nonstationary multi-agent settings.

2.4 Evolutionary Multi-Agent Systems

Evolutionary methods have long been used to induce diversity, robustness, and open-ended complexity in multi-agent populations. NEAT introduced principled mechanisms for speciation and incremental complexification of neural policies [18]. Quality-Diversity algorithms such as MAP-Elites explicitly search for a diverse set of high-performing behaviors across a behavior space [15]. At larger scale, multi-agent self-play with population dynamics has produced emergent tool use and strategy escalation, as in the OpenAI hide-and-seek study [1].

Our system combines gradient-based reinforcement learning with population-level evolutionary pressure: agents are trained with PPO while birth, death, and reproduction modulate the population, enabling weight inheritance and mutation. This combination provides a controlled setting in which coordination failures (e.g., population collapse, unequal outcomes) are both measurable and consequential, making it a natural substrate for gating communication based on distress signals.

Unlike prior approaches that focus on learning what to communicate, we investigate *when* to communicate—and show that adaptive channel activation acts as an implicit curriculum that improves both learning dynamics and asymptotic performance.

2.5 Multi-Agent LLM Reasoning

Recent work has shown that multiple LLM agents can improve factuality and reasoning through structured debate: agents generate independent responses, share them, and iteratively refine toward consensus [7, 13]. These systems implicitly assume that sharing is always beneficial and typically enable it from the first round. Chain-of-thought prompting [22] and self-consistency [21] provide the single-agent foundation for such multi-agent deliberation. Our LLM experiments (Section 5) test whether the timing of inter-agent sharing matters in this paradigm, connecting multi-agent LLM debate to the classical literature on information cascades [2, 4] and anchoring effects [20].

3 Method

3.1 Environment

We study a central-place foraging task implemented as a discrete-time grid-world. The default world is a 20×20 grid containing (i) a fixed nest at the center, (ii) F food sources placed in the environment, and (iii) a population of up to $N_{\max} = 32$ agents. Each agent occupies a single cell and acts synchronously at each environment step.

Actions and dynamics. Agents have a discrete action space of size 5: {`stay`, `up`, `down`, `left`, `right`}. Movement is clipped at the grid boundary. The core loop is central-place foraging: an agent searches for a food source, picks up food upon contact (`has_food`=`True`), carries it back to the nest, and delivers it for reward. After delivery, `has_food` resets to `False` and the agent resumes searching.

Energy, death, and population dynamics. Each agent maintains a scalar energy variable that decreases by a fixed amount each step (energy drain). If an agent's energy reaches zero, it dies and is removed from the active population. The population is therefore dynamic and may range from 0 to $N_{\max}$ during an episode.

Reproduction. Reproduction is triggered when an agent returns to the nest with energy exceeding a threshold. The agent "splits" into parent and child: the child is spawned into an available inactive slot, inherits the parent's neural network weights, and then receives an additive Gaussian mutation with standard deviation $\sigma = 0.01$ applied to the parameters. Reproduction therefore couples individual performance to population-level compute and exploration.

JAX implementation details. To preserve JIT compatibility in JAX, we represent the agent population using fixed-size arrays of length $N_{\max}$ together with an `alive` mask. Dead agents remain as inactive entries (`alive`=0), receive all-zero observations, and are masked out of learning updates (Section 3.4).

Task difficulty. We evaluate two resource regimes that share the same grid size (20×20) and initial agent count (8): *Easy* uses $F = 20$ food sources, while *Hard* uses $F = 10$ food sources. The hard regime induces scarcity and makes coordination more valuable by increasing competition and the risk of population collapse.

3.2 Pheromone Field (Stigmergic Communication)

Agents may coordinate through a shared pheromone field that acts as an external, persistent communication medium. The field is a dense tensor $\mathbf{P} \in \mathbb{R}^{H \times W \times C}$ aligned with the grid world, with $H = W = 20$ and $C = 4$ channels. At each step, each agent can (i) *read* pheromone values from a local neighborhood around its position, and (ii) *write* by incrementing the pheromone value at its current cell according to channel-specific rules.

Channel semantics. We use four channels with the following intended roles.

- **Channel 0 (Recruitment).** A pheromone trail used to recruit other agents toward recently successful foraging locations. Deposits are *success-gated*: only agents currently carrying food (`has_food`=`True`) are permitted to deposit on this channel. While carrying food, an agent adds a fixed deposit of 0.08 per step to the current cell. In addition, upon delivery at the nest, we add a bonus deposit of 0.4 at the delivered food source location (i.e., the location that generated the carried item). The design rationale is to ensure that only agents with real information contribute to trails, reducing early-training noise from random exploration.
- **Channel 1 (Territory).** Reserved for future use; in the current experiments its write strength is set to 0.0 (disabled).
- **Channels 2–3.** Reserved for future learned stigmergic signals.

Table 1: Observation vector components (45 dimensions).

Component	Description	Dims
Position	Normalized $(x, y) \in [-1, 1]^2$	2
Energy	Normalized energy in $[0, 1]$	1
Has food	Binary flag (0/1)	1
Nest compass	Noisy direction-to-nest unit vector	2
Field spatial	Local 3×3 patch summary (center+N/S/E/W) for each channel	20
Field temporal	Per-channel Δpheromone at current cell	4
Nearest food	For $K = 5$: relative $(\Delta x, \Delta y)$ and availability flag	15
Total		**45**

Field dynamics and constraints. All channels decay independently at rate 0.03 per step (approximately a 35-step half-life), causing trails to fade unless actively refreshed. We set diffusion to zero (diffusion rate 0.0), so deposits remain spatially localized; this preserves precise spatial information and makes the trail-following problem well-defined. Finally, values are capped per cell at 5.0 to prevent runaway accumulation.

3.3 Observation Space

Each alive agent receives a 45-dimensional observation vector that combines local proprioception, partial global cues, and stigmergic context. Dead agents receive an all-zero vector.

Noisy nest compass. To prevent the task from collapsing into trivial homing, agents are given a "compass" pointing toward the nest center, but with distance-dependent angular noise: the direction is perturbed by up to $\pm 40°$, with noise magnitude increasing with distance to the nest. This encourages reliance on pheromone trails for long-range navigation while still providing enough signal to support learning near the nest.

Field features. The spatial field features comprise 5 samples (center plus the 4 von Neumann neighbors) for each of the 4 channels, yielding $5 \times 4 = 20$ dimensions. Temporal features are computed as the per-channel difference between the current pheromone value at the agent's cell and the value at the previous step.

3.4 Training

All agents share a single actor–critic policy network. The policy is an MLP with LayerNorm and tanh activations, producing (i) a categorical distribution over the 5 actions and (ii) a scalar value estimate.

Optimization. We train using PPO [17] with an `alive` mask: transitions from dead agents contribute zero loss, and gradients are blocked for inactive slots. Rollouts are generated for 32 environments in parallel using `vmap`, and time unrolling uses `lax.scan` to keep the simulation and learning loop JIT-compilable.

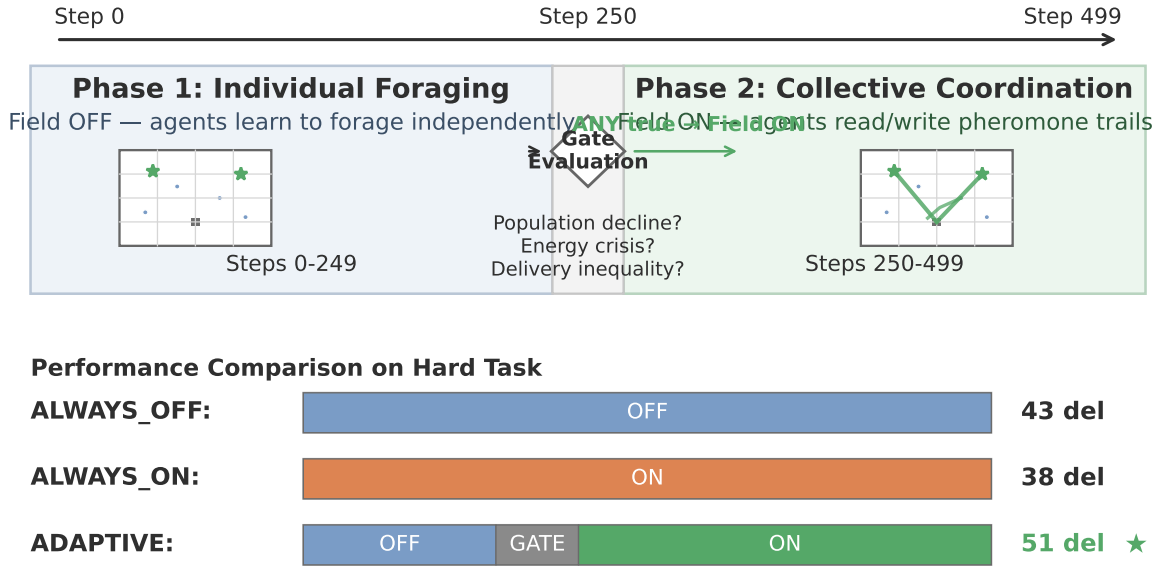

Figure 1: Schematic of the adaptive gate mechanism showing Phase 1 (individual foraging, field OFF), gate evaluation at step 250, and Phase 2 (collective coordination, field ON).

Coupling PPO and evolution. Reproduction creates per-agent parameter copies with mutation (Section 3.1), while PPO provides a population-level gradient update. Concretely, PPO updates are applied to a shared parameter vector, and these shared weights are periodically synchronized to currently alive agents; reproduction then perturbs the inherited weights to maintain diversity. We train for 10M environment steps.

3.5 Adaptive Stigmergy Gate

3.5.1 Motivation

A stigmergic pheromone field is only useful after it encodes information about successful interaction histories. At the start of an episode (and, more importantly, at the start of training), the field is typically empty or reflects random exploration, so reading it provides little signal and substantial opportunity for overfitting. Under an ALWAYS_ON design, a policy must simultaneously (i) learn to ignore the empty field, (ii) learn when the field becomes informative as trails form, and (iii) avoid encoding spurious early correlations between field noise and actions. This makes the learning problem strictly harder than a staged alternative in which agents first acquire robust individual foraging competence with the field disabled and only later learn to exploit trails once meaningful structure exists.

The adaptive gate operationalizes this two-phase structure directly: it enforces an initial individual-learning phase (field OFF), then conditionally enables stigmergic coordination (field ON) only when the population appears to need additional coordination capacity (Figure 1).

3.5.2 Gate design

The gate is a rule-based, evaluation-time mechanism that modulates access to the pheromone field within each episode. Episodes have fixed horizon $T = 500$ steps. We define a boolean scalar `field_enabled` that controls both pheromone reads and writes through a multiplicative mask

$$f_e = \texttt{field\_enabled.astype(float32)} \in \{0.0, 1.0\},$$

so that (i) observation features derived from the field are multiplied by f_e, and (ii) pheromone deposits are multiplied by f_e before being applied to the grid.

Phase 1 (steps 0–249). We set `field_enabled = False`. All pheromone reads and writes are therefore masked to zero, ensuring that agents learn a baseline foraging policy without access to stigmergic state.

Gate evaluation (step 250). At the midpoint of the episode, we evaluate three OR-conditions using population-level statistics:

1. **Population decline:** current alive population N_{alive} is less than the starting population N_0.
2. **Energy crisis:** mean energy among alive agents is less than 0.5 times the starting energy.
3. **Delivery inequality:** the Gini coefficient of per-agent delivery counts exceeds 0.6.

If any condition is true, we set `field_enabled = True` for the remainder of the episode; otherwise it remains false.

Phase 2 (steps 250–499). If activated, the pheromone field becomes fully available: field observations are unmasked and pheromone writes are applied normally. If not activated, the episode remains in the individual-foraging regime.

JAX/JIT implementation detail. For JIT stability, `field_enabled` is always represented as a boolean JAX scalar (never `None`), and gating is implemented via multiplication by f_e rather than Python conditionals. This ensures a single compiled computation graph for all modes. In early versions, we used `None` versus `True` to represent "off" versus "on," which implicitly changed control flow and led to divergent compilation artifacts.

3.5.3 Evaluation modes

We evaluate three modes that differ only in the definition of `field_enabled`:

- **ALWAYS_OFF:** `field_enabled = False` for all $T = 500$ steps.
- **ALWAYS_ON:** `field_enabled = True` for all $T = 500$ steps.
- **ADAPTIVE:** `field_enabled = False` for steps 0–249; evaluate at step 250 and set `field_enabled` for steps 250–499 according to the OR-conditions above.

Algorithm 1 Adaptive Stigmergy Gate

Input: episode length $T = 500$, `gate_step`=250, trigger conditions C
`field_enabled` ← `False`
`state` ← `reset(environment)`
for $t = 0$ to $T - 1$ **do**
  `obs` ← `get_observations(state, field_enabled)` *// field obs masked by field_enabled*
  `actions` ← `policy(obs)`
  `state` ← `step(state, actions, field_enabled)` *// field writes masked by field_enabled*
  if $t ==$ `gate_step` **then**
    `pop_decline` ← (`alive_count` $<$ `starting_count`)
    `energy_crisis` ← (`mean_energy` $< 0.5\times$ `starting_energy`)
    `delivery_ineq` ← (`gini(delivery_counts)` > 0.6)
    if `pop_decline OR energy_crisis OR delivery_ineq` **then**
      `field_enabled` ← `True`
    end if
  end if
end for
return `total_deliveries`, `final_population`

3.5.4 Why rule-based (not learned)

We intentionally implement the gate with handcrafted, interpretable conditions rather than a learned module. First, this isolates the scientific question of whether adaptive timing helps from the separate question of whether timing can be learned reliably. Second, the conditions are debuggable and make failure cases legible (e.g., distinguishing population collapse from inequality-driven underutilization). Finally, the rule-based gate provides a clean baseline for future work that replaces the conditions with a learned gating policy trained by gradients or evolutionary search.

4 Experiments & Results

4.1 Experimental setup

All experiments use 10 random seeds per condition. Episodes have fixed horizon $T = 500$ steps. We report two primary evaluation metrics: (i) total food deliveries per episode (higher is better), and (ii) final population count at episode end (higher indicates greater viability). We evaluate both the *Easy* regime ($F = 20$ food sources) and the *Hard* regime ($F = 10$ food sources) described in Section 3.1.

We compare three communication modes (Section 3.5): **ALWAYS_OFF**, **ALWAYS_ON**, and **ADAPTIVE**. Training uses PPO [17] for 10M environment steps with 32 parallel environments (JAX `vmap`) and rollout unrolling via `lax.scan`.

Evaluation protocol. We evaluate using a deterministic policy (argmax actions) and run rollouts with `lax.scan`. For each reported point, we perform 60 evaluations total (3 modes $\times$ 2 difficulties $\times$ 10 seeds). A full evaluation sweep takes approximately 9 minutes on our setup.

Success criteria. We define four pass/fail criteria evaluated on the hard task unless stated otherwise:

Table 2: Main evaluation results (mean ± std over 10 seeds). "Pop" denotes final population count.

System	Easy Del	Easy Pop	Hard Del	Hard Pop
ALWAYS_OFF	36 ± 18	9.0 ± 9.6	43 ± 21	22.0 ± 8.2
ALWAYS_ON	38 ± 19	1.6 ± 2.7	38 ± 23	19.5 ± 10.1
ADAPTIVE	44 ± 21	12.5 ± 10.2	51 ± 20	24.8 ± 7.9

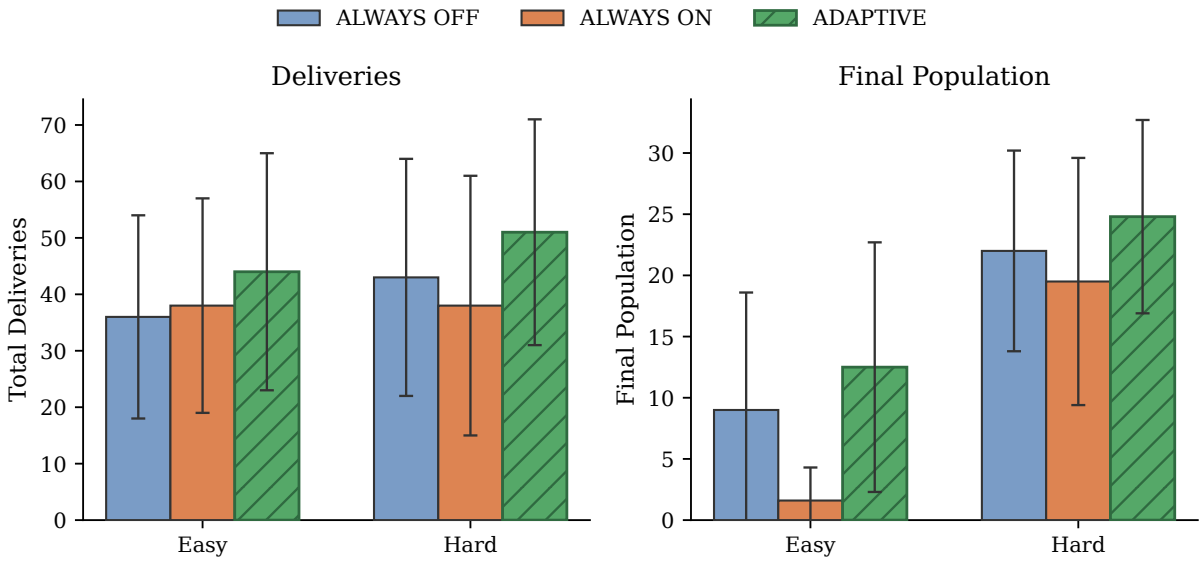

Figure 2: Main evaluation results (bar chart with error bars).

1. **Variance reduction:** std(ADAPTIVE) < std(ALWAYS_OFF) for hard deliveries.
2. **Tail risk:** min(ADAPTIVE) ≥ min(ALWAYS_OFF) for hard deliveries.
3. **No harm (easy task):** $\mathbb{E}[\text{ADAPTIVE}] \geq 0.9\,\mathbb{E}[\text{ALWAYS\_OFF}]$ for easy deliveries.
4. **Population stability:** std(ADAPTIVE population) < std(ALWAYS_OFF population) on the hard task.

4.2 Main results

Table 2 reports the primary evaluation results (mean ± standard deviation across seeds). A visual summary is provided in Figure 2.

Observations. ADAPTIVE achieves the highest mean deliveries on both tasks (44 on Easy and 51 on Hard) and the highest final population on both tasks (12.5 on Easy and 24.8 on Hard). On the hard task, ADAPTIVE also has the lowest population variance (7.9 vs 8.2 for ALWAYS_OFF and 10.1 for ALWAYS_ON). The hard-task delivery advantage is substantial: ADAPTIVE improves over ALWAYS_OFF by 19% and over ALWAYS_ON by 34% (51 vs 43 and 38). In this run, the adaptive gate fired in 10/10 evaluation episodes on both difficulties (i.e., at least one trigger condition was satisfied).

All four success criteria were satisfied.

Table 3: Timing ablation on the hard task (mean ± std over 10 seeds). The "ON" time denotes when the field becomes available within the 500-step episode.

Condition	**Deliveries**	**Population**
ALWAYS_OFF	31.8 ± 17.7	18.3 ± 7.6
ALWAYS_ON	32.3 ± 12.9	20.5 ± 7.6
EARLY_ON (100)	35.1 ± 24.7	19.3 ± 11.1
ADAPTIVE (250)	30.6 ± 13.4	19.0 ± 7.5
LATE_ON (350)	35.4 ± 18.9	21.1 ± 8.1

Table 4: Hard-task deliveries at two training durations (mean ± std over 10 seeds).

Training steps	**ALWAYS_OFF**	**ALWAYS_ON**	**ADAPTIVE**
5M	26 ± 14	22 ± 7	26 ± 14
10M	43 ± 21	38 ± 23	51 ± 20

4.3 Timing ablation

To test whether the *evaluation-time* timing of field activation explains performance, we ran a timing ablation on the hard task in which the pheromone field becomes available at a fixed step t_{on} (or never), keeping all other components unchanged. Results are shown in Table 3 and Figure 4.

All timing conditions fall within 15% of the overall mean (threshold 20%), and we do not observe a systematic advantage attributable to turning the field on earlier or later at evaluation time. This is critical: the benefit of ADAPTIVE in Table 2 cannot be explained as an execution-time switching trick, and instead points to differences in training dynamics.

4.4 Training duration effect

To probe how the advantage emerges over training, we compare evaluation at two checkpoints (5M and 10M training steps) on the hard task. Table 4 shows that at 5M steps the three modes are statistically indistinguishable, whereas by 10M steps ADAPTIVE pulls ahead nonlinearly.

At 10M steps, ADAPTIVE improves by +25 deliveries relative to its 5M checkpoint, compared to +17 for ALWAYS_OFF and +16 for ALWAYS_ON. This pattern supports the channel-curriculum hypothesis: the policy appears to require sufficient optimization time to cross a "field legibility" threshold at which pheromone information becomes reliably exploitable, and adaptive gating facilitates this transition.

4.5 Bug discovery and correction

During development, we ran 20 experiments across multiple phases and repeatedly audited surprising results. In an early run (Experiment 17), ALWAYS_ON achieved only 12 deliveries, which we initially attributed to a "warm start" advantage from beginning training without communication.

Further investigation revealed a confound: a territory pre-seeding bug initialized Channel 1 (Territory) with artificial values around the nest in the ALWAYS_ON condition, while ALWAYS_OFF and ADAPTIVE began with an empty field. This asymmetry, not communication timing, explained the depressed ALWAYS_ON performance. After fixing the initialization and retraining,

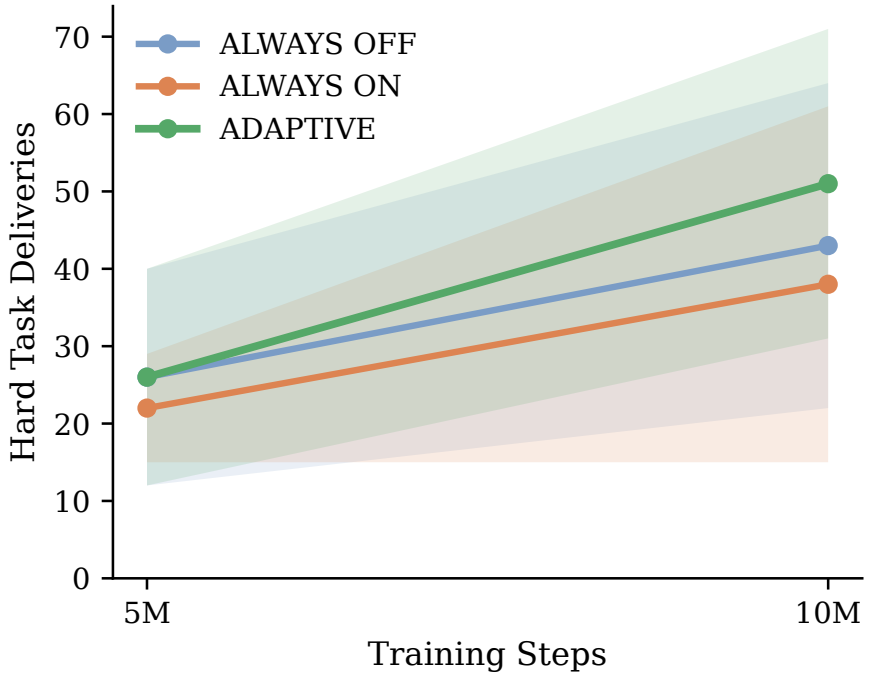

Figure 3: Training duration effect showing nonlinear ADAPTIVE advantage.

ALWAYS_ON recovered to 38 deliveries and the stable ordering emerged: ADAPTIVE > ALWAYS_OFF > ALWAYS_ON.

5 Cross-Domain Validation: LLM Agent Collectives

The RL results in Section 4 show that communication timing affects training dynamics in gradient-based multi-agent learning. A natural question is whether the same principle—that premature communication can be harmful—transfers to a fundamentally different multi-agent paradigm: large language model (LLM) agents performing collaborative reasoning [7, 13]. We conducted a series of experiments totaling 481 runs across two models, three task domains, and six communication conditions to test this.

5.1 Experimental design

We instantiate three LLM agents that independently investigate Python bug-finding tasks over five deliberation rounds. Each agent maintains a structured scratchpad containing its current hypothesis (as a `CATEGORY::TARGET` key), a confidence score in $[0, 1]$, supporting findings, and free-form reasoning—a structured variant of chain-of-thought prompting [22]. At the end of each round, a sharing policy determines whether agents see each other's scratchpads.

Communication conditions. We test six conditions that mirror the RL experiments:

- **NEVER_SHARE**: Agents never see each other's scratchpads.
- **ALWAYS_SHARE**: Scratchpads shared every round from round 1.

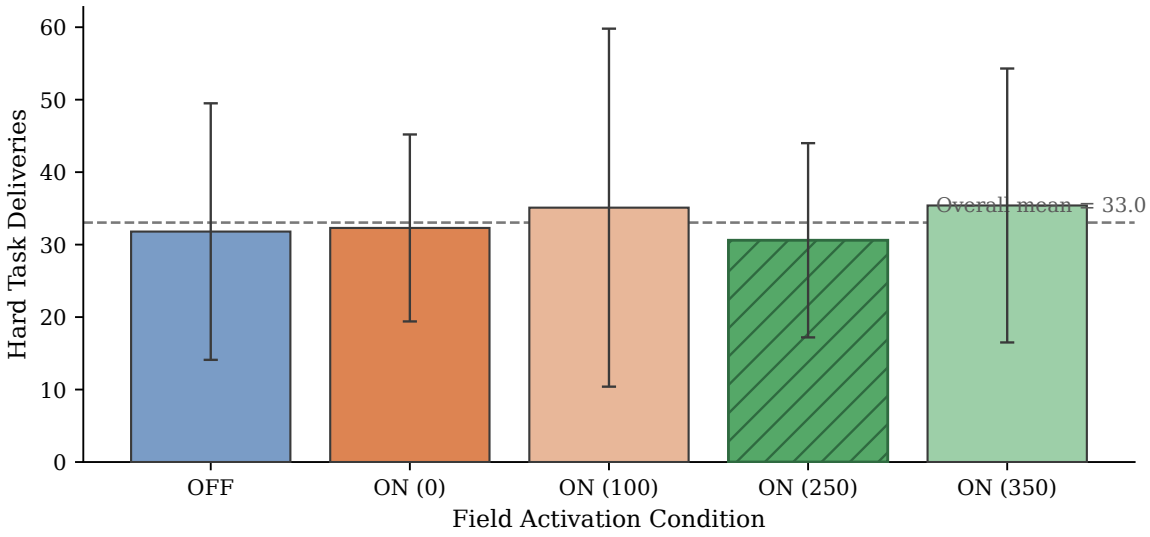

Figure 4: Timing ablation on hard-task deliveries. Similar means across activation times indicate no systematic execution-time timing effect. Dashed line denotes the overall mean.

- **FIXED_DELAY**_k ($k \in \{1, 2, 3\}$): Sharing begins after round k.
- **ADAPTIVE**: A gate decides when to share (see below).

Adaptive gate (v2). The LLM gate fires (enables sharing) when any of three conditions is met: (i) *Convergence*: at least two agents propose the same `CATEGORY::TARGET` hypothesis; (ii) *Saturation*: at most one agent changed its hypothesis from the previous round; (iii) *Force open*: the maximum number of rounds is reached. This mirrors the RL gate's distress-signal logic (Section 3.5), adapted from population-level metrics to epistemic-state metrics.

Models and tasks. We test two models: Claude Sonnet 4.5 (stronger reasoning, well-calibrated confidence) and Claude Haiku 4.5 (smaller, faster, overconfident)—consistent with prior findings that LLM confidence calibration varies substantially across model scale [12]. Tasks span two standard regimes—*debug* (20 Python microservice bugs, ∼1000 LOC each) and *incident* (20 multi-modal tasks with logs, metrics, and traces containing false leads)—plus an *adversarial* regime described in Section 5.3. In total, 361 runs cover the standard tasks (6 conditions × 20 tasks × 3 model/domain combinations, plus one free-tier pilot) and 120 runs cover the adversarial task (6 conditions × 2 round regimes × 5 replications × 2 models), for 481 experiments total at an estimated cost of $125.

5.2 Standard tasks: ceiling effect and communication efficiency

Table 5 reports accuracy across the standard task regimes. All sharing conditions achieve perfect accuracy regardless of timing. The only differentiation is NEVER_SHARE, where accuracy degrades to 50–85% depending on model and domain.

Communication efficiency. Although accuracy is saturated, communication cost is not. On the incident domain (Haiku), ADAPTIVE uses 188K tokens per task versus 217K for ALWAYS_SHARE—

Table 5: Accuracy on standard tasks by condition, model, and domain (20 tasks per cell).

Condition	Sonnet / Debug	Haiku / Debug	Haiku / Incident
NEVER_SHARE	50%	70%	85%
ALWAYS_SHARE	100%	100%	100%
FIXED_DELAY_1	100%	100%	100%
FIXED_DELAY_2	100%	100%	100%
FIXED_DELAY_3	100%	100%	100%
ADAPTIVE	100%	100%	100%

a 13.3% reduction—while achieving the same 100% accuracy. ADAPTIVE shares scratchpads for approximately 1 of 5 rounds (80% less communication overhead) because the gate delays sharing until convergence or saturation is detected. At the category level, ADAPTIVE achieves 95% category-match accuracy versus 85% for ALWAYS_SHARE, suggesting that independent deliberation before sharing promotes more precise diagnoses.

Confidence dynamics and epistemic erosion. NEVER_SHARE agents exhibit a striking pattern: mean confidence decays from 0.88 to 0.56 over five rounds on the incident domain. Without external validation, agents re-examine evidence, progressively doubt initially correct hypotheses, and sometimes reverse correct answers by round 5. This contrasts with self-consistency [21], where sampling multiple *independent* reasoning paths improves accuracy; here, *sequential* re-examination of the same evidence by a single agent degrades it. We term this *isolation-induced epistemic erosion*—the LLM analogue of the variance increase observed under ALWAYS_OFF in the RL experiments (Section 4). By contrast, ALWAYS_SHARE agents see confidence rise from 0.88 to 0.91 as peer validation reinforces correct hypotheses, and ADAPTIVE agents show a characteristic dip-then-recovery trajectory ($0.88 \rightarrow 0.81 \rightarrow 0.87$) as sharing activates mid-deliberation.

Interpretation. On standard tasks, communication is redundant but harmless: any sharing strategy reaches 100% accuracy. ADAPTIVE functions as an efficiency optimizer—same outcome, less overhead. This mirrors the RL finding that the pheromone field's primary value under favorable conditions is variance reduction rather than mean improvement.

5.3 Adversarial task: dosage-dependent epistemic cascades

To test whether premature sharing can actively harm accuracy—as ALWAYS_ON harmed RL training—we designed an adversarial task with a planted epistemic trap.

Task design. The task contains a *real bug* (a subtle async time-of-check-to-time-of-use race in `inventory.py`, where an `await` between check and decrement creates a window for concurrent corruption) and a *decoy bug* (an obvious non-atomic read-modify-write with `time.sleep` between read and write in `counter.py`). Both are in the same category (`race_condition`), making them difficult to distinguish by label alone. Three agents each receive different code files: Agent 0 sees the decoy (compelling, textbook race condition), Agent 1 sees the real bug (subtle, requires async reasoning), and Agent 2 sees falsifying evidence (`server.py` documents single-threaded asyncio with no threading, which debunks the decoy). The hypothesis is that early sharing causes Agent 0's high-confidence decoy finding to anchor the group via a false consensus cascade—an informational cascade [2, 4] driven by anchoring bias [20].

Table 6: Adversarial task, 5-round regime (Haiku 4.5): Agent 2 (swing vote) final hypothesis target across 5 replications per condition.

Condition	Ends on decoy	Ends on real	Self-corrected?
NEVER_SHARE	2/5	0/5	0/5
ALWAYS_SHARE	4/5	1/5	1/5 (at R5)
FIXED_DELAY_1	4/5	1/5	1/5 (at R5)
FIXED_DELAY_2	4/5	1/5	1/5 (at R5)
FIXED_DELAY_3	4/5	1/5	1/5 (at R4)
ADAPTIVE	5/5	0/5	0/5

Table 7: Dosage-response: decoy capture rate (Agent 2 ends on decoy, Haiku 4.5) as a function of shared rounds in the 3-round regime.

Shared rounds	Decoy capture rate
0	0–20%
1	60%
2	80%
3	100%

Haiku results. We ran 60 experiments with Haiku 4.5: 5 replications × 6 conditions × 2 round regimes (5-round and 3-round). Table 6 reports Agent 2's (the swing vote's) final hypothesis in the 5-round regime.

Haiku exhibits a full informational cascade: sharing causes Agent 2 to abandon its own analysis and converge on Agent 0's decoy category. Agent 1 (who sees the real bug) maintains the correct answer in 58 of 60 runs—truth is always present in the collective but is outvoted by the more compelling decoy, a dynamic reminiscent of groupthink [11]. Self-correction is stochastic (∼20% when it occurs) and always happens at the final round. ADAPTIVE produces the worst self-correction rate (0/5) because the gate fires at round 3, providing only two shared rounds—enough to anchor on the decoy but not enough to recover.

Dosage-response curve. The 3-round regime provides a clean test of communication dosage. Table 7 shows Haiku's decoy capture rate as a function of the number of rounds in which scratchpads are shared.

Decoy capture increases monotonically with communication exposure. The 3-round regime eliminates self-correction entirely for ALWAYS_SHARE (0/5 versus 1/5 at 5 rounds), confirming that correction requires sustained deliberation time. Error propagation is fast (one shared round is sufficient to anchor), while error correction is slow (requiring 4–5 rounds of exposure to a dissenting hypothesis).

Sonnet replication: confidence inflation without convergence. To test whether the dosage-response curve is model-dependent, we replicated the full adversarial experiment with Sonnet 4.5 (60 runs: same 6 conditions × 5 replications × 2 round regimes). The confidence dosage-response curve is confirmed: mean confidence across all agents scales monotonically with shared rounds for both models (Table 8). However, the failure mode differs qualitatively.

Haiku exhibits *cascade capture*: Agent 2 abandons its own category and adopts Agent 0's decoy

Table 8: Mean final confidence across all agents by condition and model (5-round adversarial regime). Both models show a monotonic dosage-response curve, but their baselines and convergence behaviors differ.

Condition	Shared rounds	Haiku 4.5	Sonnet 4.5
NEVER_SHARE	0	0.77	0.46
FIXED_DELAY_3	2	0.91	0.70
FIXED_DELAY_2	3	0.90	0.78
FIXED_DELAY_1	4	0.93	0.87
ALWAYS_SHARE	5	0.93	0.93
ADAPTIVE	∼2	0.89	0.48

target in 4/5 ALWAYS_SHARE runs, producing high confidence on a shared wrong answer. Sonnet exhibits a different failure mode—*confidence inflation without convergence*: Agent 2 maintains its own diagnostic category across all 5 ALWAYS_SHARE runs (0/5 converge to the decoy), but its confidence inflates from ∼0.08 (NEVER_SHARE) to ∼0.90 (ALWAYS_SHARE) without any change in underlying analysis. The sharing pressure amplifies confidence without altering conclusions.

This distinction maps onto the models' epistemic styles observed in Section 5.2: Haiku is socially conformist (changes conclusions to match peers), while Sonnet is individually anchored (resists category change but absorbs peer confidence levels). Both failure modes are driven by communication dosage, but they affect different dimensions of collective judgment—opinion convergence for Haiku, calibration distortion for Sonnet.

Critically, ADAPTIVE keeps confidence healthy for both models. Sonnet under ADAPTIVE (0.48) closely matches NEVER_SHARE (0.46), indicating that the gate successfully prevents confidence inflation. For Haiku, ADAPTIVE (0.89) is modestly lower than ALWAYS_SHARE (0.93), though the gate still permits cascade capture because even two shared rounds suffice for the smaller model to converge on the decoy.

5.4 Interpretation: communication is not monotonically helpful

The LLM experiments confirm and extend the RL finding from Section 4 that communication has information-quality-dependent costs. The results also connect to a broader principle from collective intelligence research: groups outperform individuals when members are diverse and independent, but correlated errors degrade collective accuracy [23].

Parallel failure modes. In the RL domain, an empty pheromone field acts as noise: agents that read it from step 0 overfit to uninformative gradients, and ALWAYS_ON produces the worst performance. In the LLM domain, a compelling decoy acts as a loud wrong signal: agents that share early anchor on it, and more sharing rounds monotonically increase decoy capture. Both domains demonstrate the same principle—premature access to a shared information channel degrades collective performance when the channel content is misleading.

From onset to duration. The RL gate optimizes communication *onset*: when to activate the pheromone field during training. The LLM results reveal that communication *duration* also matters. ADAPTIVE's gate fires at round 3, giving agents two shared rounds—enough to propagate the decoy but not enough for Agent 1's correct-but-quiet hypothesis to overcome the anchoring effect. Self-correction, when it occurs, requires sustained exposure to a dissenting view over 4–5 rounds.

This extends the channel-curriculum concept from Section 6: the curriculum must regulate not only when to introduce the channel, but how long agents are exposed to shared information before conclusions are drawn.

Dosage as a mechanistic finding. The dosage-response curve (Table 7) provides a clean, quantitative measure of communication cost: each additional shared round increases decoy capture by approximately 20–30 percentage points. This is the LLM analogue of the RL training-duration effect (Table 4), where ADAPTIVE's advantage grows nonlinearly with optimization time. In both cases, the relationship between communication exposure and outcome is not monotonic—there is an optimal dosage that balances information gain against anchoring risk.

6 Analysis & Discussion

6.1 The channel-curriculum mechanism

The central empirical pattern is that adaptive activation improves mean performance and stability, while timing ablations show that *execution-time* switching alone does not explain the gains (Section 4). We therefore interpret ADAPTIVE as a training-dynamics intervention: it changes what the policy must represent at different stages of learning.

ALWAYS_ON: learning with an initially weak channel. Under ALWAYS_ON training, the policy receives pheromone features from step 0, when the field is mostly uninformative. PPO updates can then fit transient correlations between noisy field values and actions. As trails become meaningful later, the policy must reconfigure those representations, which increases interference and variance.

ADAPTIVE: staged learning via delayed channel exposure. Under ADAPTIVE training, Phase 1 (field OFF) first builds robust individual control (search, homing, and energy management). Phase 2 (field ON) introduces pheromone features only after useful trail structure emerges. This encourages communication to act as a residual correction on top of a competent base policy, rather than as a brittle early dependency.

A curriculum over information channels. These dynamics match curriculum learning [3], but over channel availability rather than task difficulty. We therefore frame the method as *channel-curriculum learning*: delay access to coordination features until they are informative and exploitable. Table 4 is consistent with this view.

6.2 Stigmergy as both stabilizer and amplifier

Across our experiments, stigmergic communication has a horizon-dependent role. Early in training, it acts mainly as a *risk controller* (variance and tail-risk effects); later, once trails are legible, it acts as a *performance amplifier* (mean-delivery gains with stability retained).

This helps reconcile mixed prior results where shared channels alternately help or hurt. In our setting, the key is not communication per se, but whether learning has crossed the legibility horizon. Adaptive gating is effective because it suppresses early interference while preserving late-stage amplification.

6.3 Why the gate always fires

In this environment, the adaptive gate functions primarily as a delayed-activation mechanism rather than a selective on/off switch. The gate fired in 10/10 episodes on both tasks, meaning population-level distress conditions are consistently met by step 250. The contribution of the gate is therefore not in deciding WHETHER to communicate, but in ensuring that communication begins only after agents have developed individual competence and the field contains meaningful structure. Testing on easier environments where the gate might not fire—demonstrating true selectivity—is an important direction for future work.

6.4 Limitations

We view these results as evidence for a general phenomenon, but several limitations remain.

1. **Rule-based gate.** The gate conditions and thresholds are hand-designed in both the RL and LLM settings. A learned gate may discover better criteria or timing.

2. **Statistical power.** The RL experiments use 10 seeds per condition; the adversarial LLM experiment uses 5 replications of a single task. While effects are large, larger studies would strengthen confidence intervals.

3. **Fixed gate time.** The RL gate evaluates at the episode midpoint (step 250); optimal timing likely depends on resource regime and population dynamics.

4. **No learned-communication baselines.** We do not compare directly against learned communication methods (e.g., DIAL, CommNet, MAPPO). Our contribution is orthogonal: adaptive timing could be applied to learned channels as well.

5. **Adversarial task breadth.** The dosage-response finding (Section 5.3) is demonstrated on a single adversarial task design. While the effect is strong and mechanistically clear, replication across diverse adversarial structures would strengthen the claim.

6. **LLM ceiling effect.** On standard (non-adversarial) tasks, all sharing conditions reach 100% accuracy, preventing measurement of timing effects on accuracy. Harder tasks or weaker models may be needed to reveal timing-dependent accuracy differences in non-adversarial settings.

6.5 Future work

We see four immediate directions.

1. **Learned gating.** Replace the handcrafted conditions with a learned gating policy (gradient-trained, evolutionary, or hybrid) that optimizes when to enable communication.

2. **Environment generalization.** Evaluate adaptive channel activation on additional MARL benchmarks (e.g., predator–prey, cooperative navigation, resource management) to test domain transfer.

3. **Scaling analysis.** Increase coordination load by scaling population size (e.g., 16/32/64 agents) and world size to test whether adaptive timing becomes more beneficial as coordination demands grow.

4. **Duration-aware gating.** The LLM results suggest that gate design should optimize not only communication onset but also duration. A gate that modulates both when sharing begins and how many rounds of deliberation occur before consensus could outperform fixed-duration sharing policies.

7 Conclusion

Adaptive stigmergic communication can outperform static communication strategies in multi-agent learning when the channel is informative only after agents have acquired basic competence. In a JAX multi-agent foraging simulation trained with PPO and evolutionary pressure, ADAPTIVE improves mean deliveries on the hard task by 19% over ALWAYS_OFF and 34% over ALWAYS_ON, while also reducing population variance across 10-seed evaluations.

We argue that the mechanism is a *channel-curriculum*: delaying access to the pheromone field allows policies to first learn robust individual foraging skills, then incorporate pheromone information as residual corrections once meaningful trails exist. This avoids representational interference induced by training with an initially empty or noisy communication channel. The timing ablation further clarifies that the benefit is not due to execution-time switching; it arises from how delayed channel exposure shapes learning dynamics.

Cross-domain validation with LLM agent collectives extends these findings beyond RL. On standard tasks, adaptive gating achieves the same accuracy as unconditional sharing while reducing communication overhead by 80%. On an adversarial task with a planted decoy bug, we observe dosage-dependent epistemic cascades: each additional round of shared deliberation increases false consensus capture by 20–30 percentage points, reaching 100% at 3 shared rounds. The failure mode is structurally analogous to the RL case—premature exposure to a shared channel whose content is misleading degrades collective performance—but reveals an additional dimension: communication *duration*, not only onset, determines whether error propagation or error correction dominates.

More broadly, our results suggest that shared channels in multi-agent systems should not be assumed always-on. *When* a channel is activated can matter as much as *what* it encodes, and this holds across both gradient-trained RL agents and pretrained LLM agents. The practical takeaway is simple: *do not coordinate until there is something worth coordinating about—and regulate how long the channel stays open, because errors propagate faster than corrections.*

References

[1] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. *International Conference on Learning Representations (ICLR)*, 2019.

[2] Abhijit V. Banerjee. A simple model of herd behavior. *The Quarterly Journal of Economics*, 107(3):797–817, 1992.

[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009.

[4] Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. A theory of fads, fashion, custom, and cultural change as informational cascades. *Journal of Political Economy*, 100(5):992–1026, 1992.

[5] Abhishek Das, Théophile Gervet, Joshua Romoff, Olivier Bousquet, Joelle Pineau, and Doina Precup. TarMAC: Targeted multi-agent communication. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

[6] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system: Optimization by a colony of cooperating agents. In *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 1996.

[7] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate, 2023. arXiv preprint arXiv:2305.14325.

[8] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[9] Pierre-Paul Grassé. La reconstruction du nid et les coordinations inter-individuelles chez Bellicositermes natalensis et Cubitermes sp. La théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux*, 1959.

[10] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations (ICLR)*, 2017.

[11] Irving L. Janis. *Victims of Groupthink: A Psychological Study of Foreign-Policy Decisions and Fiascoes*. Houghton Mifflin, 1972.

[12] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know, 2022. arXiv preprint arXiv:2207.05221.

[13] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate, 2023. arXiv preprint arXiv:2305.19118.

[14] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[15] Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. In *arXiv preprint arXiv:1504.04909*, 2015.

[16] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[18] Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. In *Evolutionary Computation*, 2002.

[19] Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[20] Amos Tversky and Daniel Kahneman. Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131, 1974.

[21] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. arXiv preprint arXiv:2203.11171.

[22] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022. arXiv preprint arXiv:2201.11903.

[23] Anita Williams Woolley, Christopher F. Chabris, Alex Pentland, Nada Hashmi, and Thomas W. Malone. Evidence for a collective intelligence factor in the performance of human groups. *Science*, 330(6004):686–688, 2010.

[24] Chao Yu, Anand Velu, Eugene Vinitsky, Jianpeng Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. *arXiv preprint arXiv:2103.01955*, 2022.

A Experiment History

Table A1 summarizes the full experimental arc, showing how systematic debugging and hypothesis revision led to the final clean result. Experiments 1–5 established that stigmergy is redundant when agents have sufficient individual sensing. Experiments 9–16 identified the regime where stigmergy provides value (information asymmetry + fast decay). Experiments 17–18 contained a territory pre-seeding bug that created a false "warm start" narrative. Experiments 19–20 are the clean results reported in the main paper.

Table A1: Full 20-experiment development arc.

Exp	Description	Field OFF	Field ON	Key Finding
1	Full food obs, 20×20	132 del, pop 10	93 del, pop 5	Field redundant with full food sensing
2	No food obs, 20×20	0 del, dead	0 del, dead	Can't bootstrap without any food info
3	Food odor, 20×20	0 del, dead	1.3 del, dead	Odor too weak on large grid
4	Food odor, 10×10, 20 food	5875 del, pop 32	5212 del, pop 32	Too easy—field redundant
5	Food odor, 14×14, 10 food	2187 del, pop 32	1548 del, pop 32	Field still hurts (-41%)
9	Nuclear v1: no food obs, 4 food	Dead	Dead (2x longer)	First positive signal for field
10	Nuclear v2: 10 food, 500 energy	1514 del, pop 29	1864 del, pop 32	False positive (3 seeds, $d = 2.10$)
11	Nuclear v2 (5 seeds)	1792±390 del	1823±147 del	Replication: false positive debunked

Exp	Description	Field OFF	Field ON	Key Finding
12	Compass cutoff radius=4	1486±497 del	1633±394 del	Wrong signal—compass points home
13	Compass noise ±40° (5 seeds)	1063±661 del	1455±466 del	Promising but not significant
14	Compass noise ±40° (10 seeds)	1464±623 del	1368±627 del	Field ON lost—death spiral discovered
15	Fast decay 0.03 (10 seeds)	1464±623 del	1620±534 del	Death spiral dampened
16	Delivery reinforcement (10 seeds)	1464±623 del	1666±459 del	Population stabilized, not significant
17	Adaptive gate (3 seeds, buggy)	34±15 del	12±10 del (ON)	"Warm start"—later found to be bug
18	Timing ablation (buggy)	27±8 del	33±14 del (ON)	Contradiction revealed territory bug
19	Clean eval, v4 checkpoint (5M)	28±17 del	22±7 del (ON)	Bug fixed, all modes similar
20	Clean end-to-end, v6 (10M)	43±21 del	38±23 del (ON)	ADAPTIVE = 51±20, dominates all metrics

B Hyperparameters

Table B1: Key hyperparameters for Experiment 20 (clean end-to-end run).

Category	Parameter	Value
Environment	Grid size	20×20
	Starting agents	8
	Max agents	32
	Food sources (Easy / Hard)	20 / 10
	Episode length	500 steps
Evolution	Starting energy	200
	Max energy	300
	Food energy (delivery reward)	100
	Reproduce threshold	120
	Reproduce cost	80
	Mutation std	0.01
Pheromone Field	Channels	4 (1 active, 3 reserved)
	Ch0 write strength	0.08
	Ch0 delivery bonus	0.4
	Decay rate (all channels)	0.03
	Diffusion rate	0.0
	Field value cap	5.0
Observation	Obs dim	45
	Compass noise	±40°
	Nearest food K	5
	Field patch	3×3 (center + NSEW)
Training	Algorithm	PPO
	Total steps	10M
	Parallel envs	32
	Nest radius	2
Gate	Gate evaluation step	250
	Energy crisis threshold	50% of starting

Category	Parameter	Value
	Gini threshold	0.6