On the Role of Learned Alignment Matrices in LatentMAS

Spursh Deshpande [1] **Wenhao Lu** [2]

Abstract

Latent multi-agent systems propose that agents collaborate more effectively by sharing hidden states via a learned alignment matrix than through natural language. We test this premise through systematic ablations across three reasoning benchmarks. Replacing the trained alignment matrix with an identity transformation causes no statistically significant accuracy drop on any task. Removing cross-agent transfer entirely performs comparably or better than the full system on mathematical and scientific reasoning. On code generation, latent collaboration provides a genuine benefit over single-agent baselines, but text-based multi-agent communication substantially outperforms latent communication. A spectrum-matched random matrix also beats the trained one on code generation. We conclude that the trained alignment matrix may be functionally trivial, that genuine latent collaboration is task-specific, and that text communication remains superior for code despite being substantially slower.

1. Introduction

The case for latent multi-agent communication rests on two claims: that agents can transfer richer information through hidden states than through text, and that a learned alignment matrix $\mathbf{W}_a$ optimally transforms one agent's representations for use by another. Frameworks such as AutoGen and MetaGPT have demonstrated substantial benefits from structured inter-agent communication, motivating interest in whether hidden-state channels can deliver those benefits more efficiently. We subject both underlying claims to direct empirical test.

We ablate $\mathbf{W}_a$ by replacing it with identity and random matrices, remove cross-agent transfer entirely, and compare against text-based multi-agent communication. We also introduce a bucket analysis that cleanly separates true collaboration wins from interference cases, providing a more informative measure of multi-agent benefit than aggregate accuracy alone.

Our findings are largely negative: the learned $\mathbf{W}_a$ contributes no statistically significant benefit on any task, and text communication outperforms latent communication on the one task where multi-agent collaboration is genuinely beneficial. These results directly challenge the premise that alignment training is necessary for effective latent coordination, and raise questions about what, if anything, the latent channel is communicating.

We discuss several possible explanations – representational compatibility of the shared base model, near-trivial alignment solutions, and task-specific differences in the value of expressive communication – and their implications for the design of compositionally structured multi-agent systems.

2. Related Work

Latent Communication. Hidden-state transfer between LLM agents proposes training-free learned communication, arguing continuous representation channels outperform discrete token-level messages. Related work in multi-agent reinforcement learning shows that differentiable communication channels can emerge under joint training. We test the benefit claim of learned alignment directly against ablations on language-model agents, a setting that differs substantially from the RL and emergent communication literature.

Multi-Agent Collaboration. Prior work has shown task-dependent benefits from multi-agent debate, with improvements on factual and reasoning tasks but diminishing returns when agents converge prematurely. Decomposition benefits have been found specifically for code, with specialised agents outperforming single models on complex programming tasks. Mixture-of-agents approaches demonstrate improvements across broad benchmark suites. Our bucket analysis provides finer-grained measurement than the aggregate accuracy metrics used in this prior work, separating genuine collaboration wins from interference effects that may be masked at the level of overall accuracy.

[1]Crescenta Valley High School, La Crescenta, CA, USA [2]Millburn High School, Millburn, NJ, USA. Correspondence to: Wenhao Lu <wenhao.lu.2020@gmail.com>.

Accepted to the 2nd Workshop on Compositional Learning at ICML 2026, Seoul, South Korea. *Copyright 2026 by the author(s).*

Compositional Agents. Compositional generalisation requires sub-components contributing distinct, reusable capabilities. Neural module networks instantiate this principle by composing specialised modules at inference time to solve novel problems. We test whether latent multi-agent systems achieve analogous complementarity through task-conditioned performance breakdowns, asking whether agents contribute distinct knowledge or largely replicate one another.

3. Methods

System. Four-agent LatentMAS (Planner A0, Critic A1, Refiner A2, Judger A3) on Qwen3-4B. Let $h_t^{(i)} \in \mathbb{R}^D$ denote the final-layer hidden state of agent i at round t, $D = 2560$. Communication from agent i to agent j proceeds via a learned projection:

$$\tilde{h}_t^{(j)} = \mathbf{W}_a \, h_t^{(i)}, \quad \mathbf{W}_a \in \mathbb{R}^{D \times D}. \tag{1}$$

The transformed vector $\tilde{h}_t^{(j)}$ is injected into the KV cache of agent j before its next generation step. Evaluated on GSM8K (n=500), ARC-Challenge (n=500), and MBPP+ (n=378). All conditions use the same random seed and question ordering to ensure paired comparability.

Conditions. *LatentMAS*: trained $\mathbf{W}_a$. *Identity*-$\mathbf{W}_a$: $\mathbf{W}_a = \mathbf{I}$, isolating the effect of KV-cache injection without learned transformation. *No-transfer*: agents run sequentially without hidden-state sharing, providing a lower bound on inter-agent communication benefit. *Random*-$\mathbf{W}_a$: random orthogonal matrix with singular value spectrum matched to the trained $\mathbf{W}_a$, controlling for the spectral properties of the transformation. *TextMAS*: agents communicate via full text sequences, providing an upper bound on natural language coordination. *Single-agent*: one agent, sampled or greedy decoding.

Bucket analysis. Each example is classified by comparing LatentMAS against the greedy single-agent baseline: B1 (collaboration win): single-agent wrong, LatentMAS right. B2 (interference): single-agent right, LatentMAS wrong. B3: both right. B4: both wrong. Aggregate accuracy α relates to bucket counts as:

$$\alpha_{\text{MAS}} = \frac{|B1| + |B3|}{N}, \quad \alpha_{\text{single}} = \frac{|B2| + |B3|}{N}. \tag{2}$$

The accuracy difference $\Delta\alpha = \alpha_{\text{MAS}} - \alpha_{\text{single}} = (|B1| - |B2|)/N$ conflates benefit and harm. We therefore report the *collaboration gain* γ and *interference rate* ι separately:

$$\gamma = \frac{|B1|}{|B1| + |B4|}, \quad \iota = \frac{|B2|}{|B2| + |B3|}, \tag{3}$$

where γ is the fraction of recoverable failures that the system actually recovers, and ι is the fraction of solvable examples that the system corrupts. A system with high $\Delta\alpha$ but also high ι is unreliable despite good average accuracy.

Spectral analysis of $\mathbf{W}_a$. To characterise what the alignment matrix has learned, we compute its singular value decomposition $\mathbf{W}_a = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ and compare to the identity. Let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_D$ be the singular values. For a pure identity, all $\sigma_k = 1$. We measure the *spectral deviation*:

$$\delta_{\text{spec}} = \frac{1}{D} \sum_{k=1}^{D} (\sigma_k - 1)^2, \tag{4}$$

and the alignment of the learned singular directions with the PCA principal components of the hidden states, quantified by the mean squared cosine between the top-K right singular vectors $\{v_k\}$ and the top-K PCA directions $\{u_k\}$:

$$\rho_K = \frac{1}{K} \sum_{k=1}^{K} \cos^2(v_k, u_k). \tag{5}$$

A random orthogonal matrix gives $\rho_K \approx K/D$ in expectation; a matrix that truly learns to align representations would give $\rho_K \gg K/D$. Results are reported alongside the ablation in Table 1 and discussed in Appendix C.

Statistical tests. Pairwise accuracy differences use McNemar's test on matched examples. Results marked NS have $p > 0.05$; all other differences are reported with exact p-values.

4. Results

4.1. The Alignment Matrix Contributes Nothing Significant

Table 1 shows accuracy across all conditions. Replacing $\mathbf{W}_a$ with identity causes no statistically significant change on any task: GSM8K (-1.0pp, p=0.125), ARC ($+0.0$pp, p=1.0), MBPP+ ($+0.7$pp, p=0.72). No-transfer also produces no significant accuracy drop on GSM8K or ARC.

Notably, on ARC the no-transfer baseline outperforms full LatentMAS (93.2% vs. 92.6%), and spectrum-matched random $\mathbf{W}_a$ beats the trained matrix on MBPP+ (77.5% vs. 73.3%). Both results are consistent with the interpretation that $\mathbf{W}_a$ is not learning a task-useful transformation.

Spectral analysis provides a mechanistic account of why. The trained $\mathbf{W}_a$ has spectral deviation $\delta_{\text{spec}} \approx 0.03$ (Eq. 4), extremely close to zero, indicating singular values that hover near 1.0 – barely distinguishable from the identity. The principal direction alignment ρ_{64} is comparable to what a random orthogonal matrix would produce ($\approx 64/2560 = 0.025$), meaning the learned singular directions do not preferentially align with the dominant axes

Table 1. Accuracy across conditions. Identity and no-transfer match trained LatentMAS on all tasks. TextMAS strongly outperforms on MBPP+. NS = not significant vs. LatentMAS (McNemar's test).

Condition	GSM8K	ARC	MBPP+
LatentMAS (trained)	92.0	92.6	73.3
Identity-$\mathbf{W}_a$	91.0†	91.0†	74.0†
No-transfer	90.8†	**93.2**†	67.5
Random-$\mathbf{W}_a$	91.5†	92.0†	**77.5**†
TextMAS	91.0†	90.5†	**89.0***
Single-agent (sampled)	90.8	92.8	66.9

†NS vs. LatentMAS. $^*p<0.01$. Bold = best per task.

of the hidden-state distribution. The trained $\mathbf{W}_a$ is therefore not only spectrally near-identity but also geometrically uninformative: it does not rotate representations toward task-relevant subspaces. Appendix C provides a full decomposition of the channel.

4.2. Collaboration Wins Are Task-Specific and Rare

B1 rates on GSM8K (3.0%) and ARC (1.0%) are well below a meaningful threshold – the multi-agent protocol changes almost nothing for examples that the single agent gets wrong. MBPP+ shows a genuine 13.0% B1 rate with only 5.5% interference (B2), giving a net positive collaboration effect of 7.5pp.

Using the collaboration gain and interference rate metrics from Eq. 3: on GSM8K, $\gamma = 6/19 = 0.316$ and $\iota = 1/181 = 0.006$; on ARC, $\gamma = 2/15 = 0.133$ and $\iota = 4/185 = 0.022$; on MBPP+, $\gamma = 26/69 = 0.377$ and $\iota = 11/131 = 0.084$. The MBPP+ γ of 0.377 means the system recovers more than one third of the failures that are in principle recoverable – a genuine signal of complementary agent contributions. The GSM8K and ARC γ values indicate that roughly two thirds of recoverable failures remain unrecovered even with four agents, a poor return for the added complexity.

The asymmetry between tasks is striking. On GSM8K and ARC, the vast majority of examples fall into B3 (both right) or B4 (both wrong): the multi-agent protocol changes almost nothing. This suggests that for mathematical and scientific reasoning, the base model capability is the dominant factor and multi-agent coordination adds negligible signal. On MBPP+, genuine collaboration wins are present and the protocol does add value over single-agent, consistent with prior evidence that code generation benefits from role decomposition.

4.3. Text Communication Dominates on Code

On MBPP+, TextMAS outperforms LatentMAS by 15.7pp: 89.0% vs. 73.3% ($p<0.01$). This large gap suggests that structured text communication is critical for code generation, where agents must coordinate on syntax, logic, and test case constraints in ways that benefit from the expressive power of natural language. The mixture-of-agents literature has similarly found that richer inter-agent communication improves performance on complex tasks; our results extend this finding to suggest that the modality of communication matters, not just its presence.

TextMAS requires 112.8s wall clock per example vs. 23 to 44s for LatentMAS (4 to 5$\times$ slower). Despite this substantial compute disadvantage, LatentMAS is 15.7pp less accurate on MBPP+. This yields a poor accuracy-per-second tradeoff: LatentMAS sacrifices significant performance for its speed advantage on the task where it is most needed. Full compute details are in Appendix D.

5. Discussion

What does $\mathbf{W}_a$ learn? Identity and random matrices perform equivalently to the trained $\mathbf{W}_a$, suggesting alignment training converges to a near-trivial solution. A natural explanation is that agents instantiated from the same base model already produce representationally compatible hidden states, rendering explicit learned alignment redundant. The no-transfer condition performance further supports this: KV-cache injection itself carries minimal information when $\mathbf{W}_a$ is trivial, yet performance is nearly unchanged on GSM8K and ARC. The ARC anomaly – where no-transfer slightly outperforms the trained system – hints that cross-agent injection may occasionally introduce noise, a possibility explored further in Appendix C.

Task-specific compositionality. Code generation is the one domain where latent collaboration provides genuine benefit (B1=13%). This may reflect the natural decomposability of code tasks into planning, specification, implementation, and verification, which maps onto the Planner/Critic/Refiner/Judger role structure. Mathematical reasoning on GSM8K, by contrast, appears to be predominantly a single-agent skill: the base model either solves the problem or it does not, and multi-agent coordination cannot recover examples the single agent fails on. This is consistent with findings in the compositional learning literature that the benefit of modular decomposition depends heavily on whether the task is genuinely decomposable.

Text vs. latent for compositional tasks. The 15.7pp TextMAS advantage on MBPP+ suggests that for compositionally complex tasks, expressive text communication matters more than the efficiency of hidden-state transfer. This challenges a core assumption of the LatentMAS design. One interpretation is that text forces agents to externalise intermediate reasoning in ways that can be checked and cri-

tiqued, while latent channels may carry representations that are structurally compatible but semantically underspecified for the purposes of code coordination. Emergent communication research has shown that learned protocols can be highly task-specific; the failure of latent alignment on code may reflect that the trained protocol is optimised for the training distribution and does not generalise to the complex coordination demands of code generation.

Implications for system design. These results motivate two practical recommendations. First, practitioners deploying latent multi-agent systems for code generation should default to text communication unless compute constraints are severe. Second, the equivalence of identity and trained $\mathbf{W}_a$ suggests that alignment training is an unnecessary source of complexity: simpler architectures that use direct KV injection without a learned projection may be preferable for deployment. Hybrid approaches that use latent communication for simple sub-tasks and text for complex coordination remain an open direction.

Limitations and future work. Several limitations constrain the generality of our conclusions. First, all experiments use a single model family (Qwen3-4B); it is possible that larger models or models with different pre-training distributions produce more informative $\mathbf{W}_a$ solutions. Second, our evaluation covers three benchmark types; domains such as multi-step planning, tool use, or long-horizon reasoning may exhibit qualitatively different collaboration dynamics. Third, the alignment matrix is trained with a fixed objective; alternative training regimes that explicitly penalise representational redundancy or maximise mutual information between agent states could in principle produce more task-useful transformations. Finally, we evaluate only four-agent configurations; the scaling behaviour of latent collaboration with agent count is unknown.

Future work should examine whether explicit diversity objectives during $\mathbf{W}_a$ training produce non-trivial transformations, whether mid-layer rather than final-layer hidden-state transfer changes the results, and whether the TextMAS advantage on code persists under tighter token budgets that limit the expressiveness of text channels.

6. Conclusion

The trained alignment matrix in LatentMAS is functionally equivalent to an identity transformation across all tasks tested. True collaboration wins are rare on mathematical and scientific reasoning ($\leq 3\%$ B1 rate), but present and meaningful on code generation (13% B1 rate). Yet on code – the one domain where multi-agent coordination genuinely helps – text-based communication outperforms latent by 15.7pp, at the cost of being 4 to 5$\times$ slower. A spectrum-matched random matrix also outperforms the trained one on MBPP+, reinforcing the conclusion that alignment training is not the source of whatever benefit exists.

These results challenge the premise that learned latent alignment enables superior compositional coordination, and motivate task-aware communication protocol design: latent channels for speed-sensitive low-complexity tasks, text channels when expressive coordination is needed. Future work should examine whether alignment training becomes useful with larger model families, longer training, or explicit diversity objectives.

Impact Statement

This paper presents work whose goal is to advance understanding of multi-agent AI system design. Our negative findings on alignment training may help practitioners avoid unnecessary complexity when deploying such systems. There are no immediate negative societal impacts.

References

[1] J. Zou et al. Latent collaboration in multi-agent systems. *arXiv preprint arXiv:2511.20639*, 2025.

[2] S. Sukhbaatar, R. Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[3] A. Lazaridou and M. Baroni. Emergent multi-agent communication in the deep learning era. *arXiv preprint arXiv:2006.02419*, 2020.

[4] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

[5] T. Liang et al. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

[6] W. Chen et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. *arXiv preprint arXiv:2308.10848*, 2023.

[7] J. Wang et al. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*, 2024.

[8] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2018.

[9] D. Hupkes, V. Dankers, M. Mul, and E. Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

[10] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 39–48, 2016.

[11] Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[12] K. Cobbe et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[13] P. Clark et al. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[14] J. Austin et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[15] Q. Wu et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

[16] S. Hong et al. MetaGPT: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023.

A. Full Accuracy and Ablation Results

Table 2. Full accuracy table with sample sizes and paired differences vs. trained LatentMAS (McNemar's test).

Condition	GSM8K	ARC	MBPP+	Notes
LatentMAS (trained)	92.0%	92.6%	73.3%	n=500/500/378
Identity-$\mathbf{W}_a$	91.0%	91.0%	74.0%	n=100, all NS
No-transfer	90.8%	93.2%	67.5%	ARC beats LMAS
Random-$\mathbf{W}_a$	91.5%	92.0%	77.5%	n=200
TextMAS	91.0%	90.5%	89.0%	n=200, MBPP+ $p < 0.01$
Single-agent (sampled)	90.8%	92.8%	66.9%	full n
Single-agent (greedy)	90.5%	92.5%	65.5%	n=200
KV-blocked	89.5%	89.0%	69.0%	n=200

B. Full Bucket Analysis

Table 3. Full bucket analysis: LatentMAS vs. greedy single-agent per example. B1 = collaboration win; B2 = interference; B3 = both right; B4 = both wrong.

Task	n	B1 (collab win)	B2 (interference)	B3 (both right)	B4 (both wrong)
GSM8K	200	6 (3.0%)	1 (0.5%)	180 (90.0%)	13 (6.5%)
ARC	200	2 (1.0%)	4 (2.0%)	181 (90.5%)	13 (6.5%)
MBPP+	200	26 (13.0%)	11 (5.5%)	120 (60.0%)	43 (21.5%)
Total	600	34 (5.7%)	16 (2.7%)	481 (80.2%)	69 (11.5%)

C. $\mathbf{W}_a$ Decomposition Analysis

The trained vs. identity vs. no-transfer comparison decomposes the communication channel into components:

No-transfer vs. Identity-$\mathbf{W}_a$ isolates the effect of KV-cache injection without any transformation. On GSM8K, no-transfer $\approx$ identity (90.8% vs. 91.0%), suggesting KV injection carries minimal information when $\mathbf{W}_a$ is trivial.

Identity-$\mathbf{W}_a$ vs. trained $\mathbf{W}_a$ isolates the effect of the learned transformation. Differences are non-significant on all tasks ($p > 0.05$), suggesting $\mathbf{W}_a$ learns a near-identity mapping.

ARC anomaly: No-transfer (93.2%) exceeds both identity (91.0%) and trained (92.6%) on ARC, suggesting cross-agent hidden-state transfer may slightly hurt ARC performance. Complex interaction effects where the injected hidden state introduces irrelevant signal cannot be ruled out.

Flocking patterns (mean pairwise cosine across agents) are nearly identical between trained and identity $\mathbf{W}_a$ conditions, consistent with $\mathbf{W}_a$ applying a near-trivial transformation that preserves representation geometry.

D. Compute Accounting

TextMAS is 4 to 5$\times$ slower in wall clock time because it generates full text sequences between agents. Despite this cost, TextMAS outperforms LatentMAS on MBPP+ by 15.7pp, yielding a poor accuracy-per-second tradeoff for LatentMAS on code generation tasks.

Table 4. Compute accounting per condition and task (means per example). TextMAS forward passes not tracked.

Condition	Task	Fwd. Passes	Wall (s)	Gen. Tokens	GPU MB
latent_mas	gsm8k	623	23.0	379	7960
latent_mas	mbpp+	1222	44.0	776	8102
single_agent_sampled	gsm8k	608	20.4	373	7906
kv_blocked	gsm8k	888	38.2	546	8012
no_transfer	gsm8k	722	26.0	434	7929
text_mas	gsm8k	–	87.5	–	8448
text_mas	mbpp+	–	112.8	–	8772