Reusable Low-Rank Subspaces Explain Why Cross-Modal Transfer Adapts with Tiny Updates

Anonymous Authors[1]

Abstract

Parameter-efficient finetuning methods such as LoRA routinely adapt massive pretrained transformers to new tasks using only tiny low-rank updates, but the representational geometry that makes this possible remains unclear. We use cross-modal transfer from a language-pretrained transformer to time-series forecasting as a controlled probe of low-rank adaptation, asking why so few directions are sufficient. Across adaptation regimes, LoRA recovers most of the transfer benefit of full finetuning; effective-rank analyses show that pretrained representations already concentrate on a low-rank subspace that finetuning *redistributes* rather than rebuilds; and a single linear projection over frozen hidden states aligns with realistic time-series trajectories without paired supervision. Randomly initialized models, by contrast, first construct a compressed representation through a uniform layer-wise collapse before they can specialize. These results support a view of cross-modal adaptation as low-rank *direction selection* within reusable pretrained subspaces.

1. Introduction

Parameter-efficient finetuning methods such as LoRA (Hu et al., 2021) can adapt enormous pretrained transformers to new downstream tasks using only a small low-rank update, often matching the quality of full finetuning at a fraction of the trainable parameters. The empirical success of these methods raises a fundamental geometric question: *Why are such low-rank updates sufficient?*

A growing body of work suggests that the answer lies in *intrinsic dimensionality* (Aghajanyan et al., 2021): pretrained models already organize their representations into low-dimensional subspaces, and adaptation may only need to traverse a small number of directions rather than reconstruct the full computation. Under this view, transfer is largely about *selecting* and *specializing* existing structure.

We use *cross-modal* transfer as a controlled stress test of this hypothesis. We start from a language-pretrained transformer (Qwen3-0.6B (Yang et al., 2025)) and adapt it to probabilistic time-series forecasting (Ansari et al., 2024; Roger et al., 2025) on the GiftEval corpus (Aksu et al., 2024). The setting is stringent: numerical sequences and natural language share no surface vocabulary, so any reusable structure must be representational, not lexical. If a small low-rank update bridges this modality gap, it provides evidence that a compatible subspace was already present in the pretrained model.

Our findings. We provide a controlled empirical account of cross-modal adaptation as low-rank specialization:

- LoRA recovers most of the transfer benefit of full finetuning, despite touching only a small low-rank slice of attention parameters (Section 3).
- Pretrained representations are already low effective-rank; finetuning *selectively redistributes* rank in middle layers rather than collapsing the whole network. Random initialization, in contrast, undergoes a uniform layer-wise compression to construct a usable subspace from scratch (Section 4).
- Frozen pretrained hidden states already contain forecasting-compatible directions: a single linear map, trained without paired supervision, projects them onto realistic time-series trajectories (Section 5).
- The reused subspace lives mostly in middle layers and aligns with interpretable temporal primitives (periodicity, regime shifts, missingness), supporting a view of transfer as direction *reuse* rather than *reinvention* (Section 6).

Together, these results give a geometric account of parameter-efficient cross-modal finetuning: pretraining supplies much of the relevant representational structure, and adaptation specializes a small set of directions within it.

Related work. Our findings build on three intersecting lines of work. **Low-rank adaptation and intrinsic dimensionality.** Aghajanyan et al. (2021) showed that finetuning can be projected into a very low-dimensional subspace with-

[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.

out much loss. LoRA (Hu et al., 2021) operationalizes this as a rank-r update on attention projections, and a substantial follow-up literature on PEFT confirms that adaptation rarely needs the full parameter space (Kumar et al., 2022; Iurada et al., 2025). Our work asks where this low-rank structure *comes from* geometrically, by tracking effective rank across training. **Reusable cross-modal subspaces.** The Platonic Representation Hypothesis (Huh et al., 2024) and universal embedding geometry (Jha et al., 2025) argue that scaled models converge to a shared statistical structure across modalities. Frozen vision and language backbones can drive forecasting (Chen et al., 2024; Roschmann et al., 2025); we provide direct linear-probe evidence that a forecasting-compatible subspace exists inside a frozen language model. **Tokenized cross-modal forecasting.** Several systems adapt language models to time series via reprogramming, frozen-backbone alignment, or PEFT (Jin et al., 2023; Zhou et al., 2023; Gruver et al., 2023; Wolff et al., 2025), with mixed reports on when language pretraining helps (Ansari et al., 2024; Tan et al., 2024; Zheng et al., 2025; Zhang et al., 2025; Riachi et al., 2025; Qiu et al., 2026). We use this setup as a probe rather than a target, treating forecasting performance as a window into the geometry of low-rank cross-modal transfer.

2. Transfer Setup

We repurpose Qwen3-0.6B (Yang et al., 2025) as a probabilistic time-series forecaster by casting forecasting as next-token prediction over discretized values, following Roger et al. (2025); Ansari et al. (2024). Each series is normalized using z-scaling from the context window (C) and discretized into 1024 uniform bins over $[-5, 5]$, with bin indices mapped directly to the first vocabulary slots of the pretrained model. The standard Qwen3 special tokens preserved. Training uses sliding windows of length $C{+}L{=}512{+}64$. A weighted quantile loss over $\mathcal{Q} = \{0.1, \ldots, 0.9\}$ (Ansari et al., 2025) is used in training. The logits output is transformed into a CDF whose quantiles are extracted by inversion (full details in App. A).

We compare four adaptation strategies, in this setting:

- **Full FT:** every parameter is trainable.
- **IO Only:** the transformer trunk is frozen; only the input embedding and output head are trained.
- **LoRA Attn:** rank-8 LoRA adapters on $\{W_Q, W_K, W_V, W_O\}$; everything else frozen.
- **LoRA Attn+IO:** LoRA on attention plus trainable embedding and head.

We compare each regime starting either from the pretrained Qwen3 weights (**LANGINIT**) or from random reinitialization of the same architecture (**RANDINIT**). Models are trained on GiftEval (Aksu et al., 2024) with AdamW, on A100 GPUs, and evaluated on 1,000 held-out windows. Full hyperparameters are in App. A.

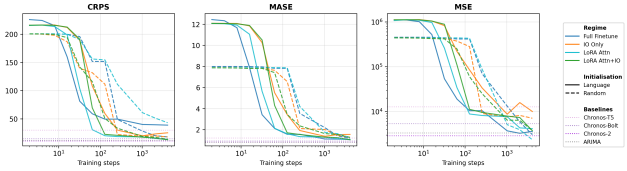

Figure 1. **Low-rank adaptation tracks full finetuning.** $h{=}1$ metrics for four regimes. Solid: LANGINIT; dashed: RANDINIT; dotted: baselines. LoRA Attn matches full finetuning on CRPS and MASE from a pretrained start.

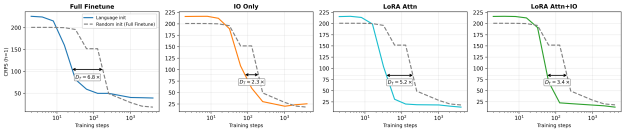

Figure 2. **Effective data transfer per regime.** D_T is the extra optimization needed by RANDINIT to match LANGINIT (Hernandez et al., 2021). LoRA Attn recovers most full-finetuning savings (5.2 vs. 6.8 when full), above IO-only (2.3).

3. Efficient Low-Rank Adaptation

We first ask: *how much of the pretrained transfer benefit can be captured by a strictly low-rank update?* If transfer is fundamentally low-dimensional, a small low-rank perturbation of the pretrained weights should recover most of the optimization advantage of full finetuning.

LoRA matches full finetuning. Figure 1 shows training progressions of $h{=}1$ CRPS, MASE, and MSE for the four training regimes. Starting from pretrained weights, LoRA Attn (rank 8, $\alpha{=}16$, attention only) closely tracks full finetuning across the full training trajectory and reaches the same final error level. IO-only adaptation, which freezes the entire transformer trunk, lags substantially: training only the input/output mappings cannot reach the same forecasting quality. The transfer benefit is therefore not concentrated in the embedding layers but in the attention pathway, which a small low-rank update is sufficient to retarget.

Effective-data view: how much do low-rank updates buy? To quantify the optimization benefit conferred by pretrained weights under each regime, we compute the *effective data transferred* of Hernandez et al. (2021): the multiplicative reduction in training steps needed to reach a given target loss when starting from pretrained vs. random initialization (App. A.4). Figure 2 shows that full finetuning achieves $D_T = 6.8\times$, attention-only LoRA achieves $D_T = 5.2\times$, and the constrained LoRA Attn+IO setting still reaches $D_T = 3.4\times$, all far above IO-only ($D_T = 2.3\times$).

A low-rank attention adapter therefore recovers $\sim 76\%$ of the data savings of full finetuning, despite touching a tiny fraction of the parameters and leaving the bulk of the transformer unchanged. Two implications follow: **(i)** the modification needed to translate a language-pretrained model into a competitive forecaster is intrinsically low-dimensional;

(ii) the bulk of the pretrained computation is preserved during transfer—adaptation operates as a small perturbation around the pretrained solution rather than a global rewrite. We make this perturbation precise in the next section through effective-rank dynamics.

4. Effective-Rank Dynamics

If adaptation is low-rank, then we should be able to see this directly in the geometry of the hidden states across training. We track effective rank at every layer and every checkpoint to characterize how representational capacity evolves under each initialization.

Method. At each checkpoint and each of the 28 transformer layers, we pass 10,000 time-series windows of length 512 from the validation split through the model and accumulate the centred covariance Σ of the hidden states (excluding position 0 to avoid attention-sink artifacts). The effective rank is $\mathrm{erank}(\Sigma) = \exp\big(-\sum_i p_i \log p_i\big)$, where $p_i = \lambda_i / \sum_j \lambda_j$ are normalized eigenvalues (Roy & Vetterli, 2007). This soft rank equals 1 when variance concentrates on a single direction and equals the ambient dimension when isotropic, providing a continuous measure of intrinsic dimensionality.

Pretrained models start compressed and stay compressed. Figure 3 (left) shows that RANDINIT begins with near-isotropic hidden states ($\mathrm{erank} \approx 440$ in a 1024-dim space) and rapidly collapses to $\mathrm{erank} \approx 10$ within 500 steps—a 40× compression. LANGINIT, in contrast, starts from an already low effective rank ($\approx$50) and only mildly contracts to ≈ 27. The pretrained model never visits a high-rank intermediate state: the compressed representation that random initialization must *construct* is essentially what language pretraining *provides*.

Selective redistribution vs. uniform collapse. The layer-resolved view (Figure 3, right) shows the distinction more clearly. Random initialization compresses every layer in lockstep, all 28 layers collapse together as a tight bundle, with uniformly negative $\log_2$ rank change (-5 to -7). Pretrained initialization redistributes capacity instead: middle layers ($\sim$8–17) *increase* their effective rank during finetuning (positive $\log_2$ change up to $+1$), while early and late layers contract. Adaptation expands the representational capacity of the layers that already host the most reusable pretrained structure, and trims the rest.

This pattern is consistent with the representation-geometry phases described by Li et al. (2025): random initialization undergoes a warmup-collapse phase before any selective entropy redistribution can begin, while pretrained initialization skips the warmup and proceeds directly to selective redistribution. From a low-rank-adaptation standpoint, the key point is that pretrained models begin close to the compressed representation that finetuning would otherwise have to manufacture.

5. Reusable Forecasting Subspaces

The effective-rank evidence shows that pretrained models occupy a low-dimensional subspace before any forecasting supervision. We now ask whether this subspace is *forecasting-compatible*: does it already contain directions whose dynamics resemble real time series? If so, finetuning can emphasize those directions rather than inventing them.

Linear probe over frozen hidden states. We freeze Qwen3-0.6B, pass N=1,920 WikiText-103 sequences (T=512 tokens each) through it, and concatenate hidden states across all 28 layers to obtain $\mathbf{h}_{i,t} \in \mathbb{R}^D$ with $D = 28{,}672$. A single linear map $\hat{y}_{i,t} = \mathbf{w}^\top \mathbf{h}_{i,t} + b$ produces a scalar prediction at each token. We train $\mathbf{w}$ via an EM-style nearest-neighbour loss to a bank of $M = 10{,}000$ real GiftEval windows (Aksu et al., 2024), with no paired text–time-series supervision (full setup in App. B).

The pretrained subspace maps to real time series. Across the 1,920 training inputs, the decoded sequences match 686 *distinct* real time series via nearest neighbour. This diversity depends on both pretrained weights and meaningful inputs: a 2×2 ablation crossing weights (pretrained vs. random) and inputs (WikiText vs. random tokens) shows a steep collapse in unique matches when either factor is removed (random init: 54 matches; random inputs: 4). Only the combination of pretrained weights and meaningful text yields diverse, structured trajectories. On 1,000 held-out WikiText sequences, mean nearest-neighbour MSE is 0.72 vs. 0.67 at training, suggesting that the property is general to the pretrained representation space rather than memorized. App. B additionally shows that retrieving a WikiText projection from the observed half of a time series and reusing the projection's continuation outperforms last-value carry-forward in MSE, evidence that the projected dynamics carry predictive structure.

A single low-dimensional direction over frozen pretrained hidden states is already aligned with the manifold of real time series. The pretrained subspace is therefore not just compressed (Section 4) but compressed *toward forecasting-compatible directions*, before any time-series gradient ever passes through the model. This helps explain why a small low-rank update can be effective: adaptation refines an existing alignment rather than constructing one from scratch.

6. Representation Reuse vs. Reinvention

Sections 4 and 5 together imply that pretrained and randomly initialized models reach time-series competence through fundamentally different trajectories. We now exam-

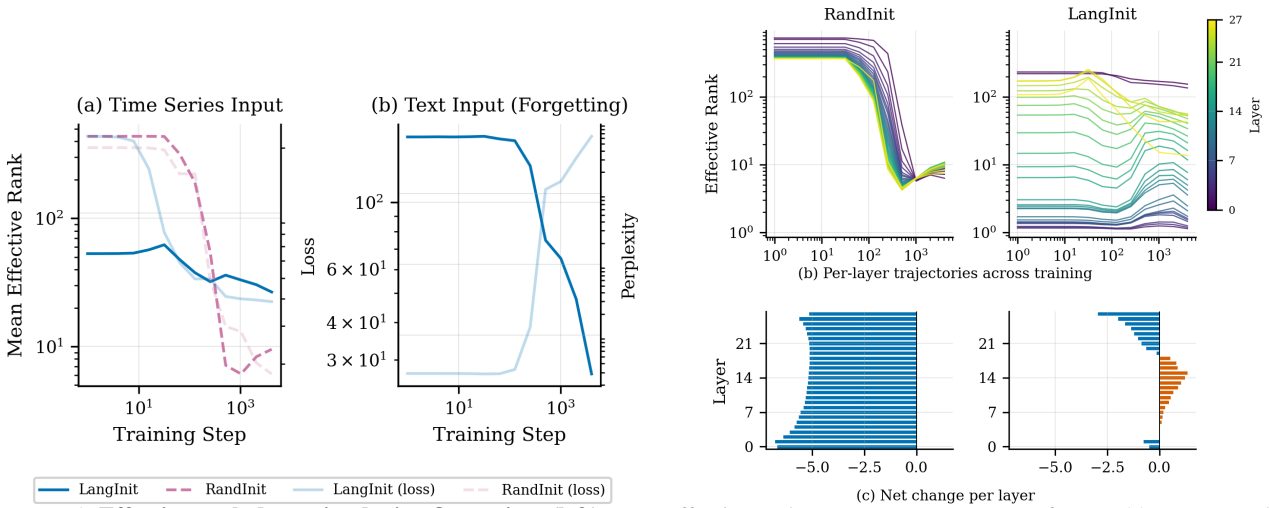

Figure 3. **Effective rank dynamics during finetuning. (left)** Mean effective rank: RANDINIT compresses from $\sim$440 to $\sim$10, while LANGINIT starts compressed ($\sim$50) and changes modestly. **(right)** RANDINIT compresses uniformly; LANGINIT redistributes capacity, increasing rank in middle layers ($\sim$8–17).

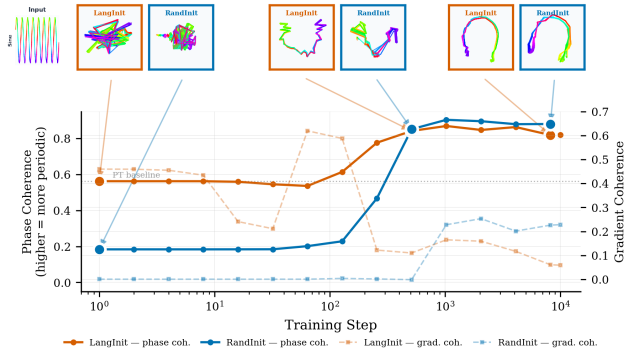

Figure 4. **Pretrained models inherit usable structure from step one.** Per-sample gradient coherence and CRPS for IO-only training. LANGINIT aligns from step 1; RANDINIT aligns only after a warmup around step 300.

ine the mechanistic consequences of this distinction directly.

Coherent gradients as a marker of subspace reuse. For a fixed evaluation batch of N=32 time series, we compute per-sample gradients $\mathbf{g}_i = \nabla_\theta \mathcal{L}(x_i; \theta)$ at every checkpoint and measure the mean off-diagonal pairwise cosine similarity (App. C). If diverse forecasting examples push θ in mutually reinforcing directions, gradient coherence is high; if they conflict, it is near zero (Mircea et al., 2025). Figure 4 shows that LANGINIT exhibits high coherence from step 1 and loss descends immediately, whereas RANDINIT sits at near-zero coherence for over a hundred steps and only starts to learn once a coherent direction emerges. The pretrained subspace therefore creates a low-rank optimization channel: examples tend to pull in the same few directions, matching the restriction imposed by a rank-8 LoRA adapter.

Reuse vs. reinvention in the learned geometry. Because both initializations eventually reach competitive forecasting performance, a natural question is whether they end up at the same solution. They do not. On simple synthetic inputs (sine, square, sawtooth), RANDINIT converges to nearly uniform clean arcs at every layer with 72–73% of variance captured by two PCs; while LANGINIT produces layer-specific loops with only 25–41% PCA variance, distributing the same periodic structure across a richer inherited representation (App. D, Figs. 13, 14). Quantitatively, the two solutions are highly aligned at the relational level (CKA 0.86–0.99 at layer 13), but RANDINIT achieves alignment by *constructing* a clean low-rank manifold whereas LANGINIT achieves it by *reusing* the pretrained one. Sparse-crosscoder analysis and a preliminary causal circuit (App. E) further show that the reused middle-layer subspace tracks interpretable temporal primitives—periodicity, regime shifts, magnitude jumps, missingness—rather than abstract numerical features.

7. Discussion and Implications

Across effective-rank, probe, and gradient analyses, LoRA's low-dimensional assumption becomes representational: pretraining supplies a compressed, forecasting-compatible subspace, coherent gradients, and reusable middle-layer structure, while random initialization must construct these ingredients from scratch. This predicts larger pretraining gains in low-data or shifted regimes (Qiu et al., 2026), and suggests screening transfers with effective-rank and frozen-state alignment diagnostics. Cross-modal forecasting transfer is low-rank: finetuning selects middle-layer directions, and LoRA captures most full-finetuning data savings.

Limitations. We study one backbone, corpus, and tokenization scheme, so larger scales and alternative architectures remain open. The linear probe shows compatibility, not optimality. Our claim is existence of a low-rank transfer solution, not optimality of a specific rank.

References

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7319–7328, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Aksu, T., Woo, G., Liu, J., Liu, X., Liu, C., Savarese, S., Xiong, C., and Sahoo, D. Gift-eval: A benchmark for general time series forecasting model evaluation, 2024. URL https://arxiv.org/abs/2410.10393.

Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D. C., Rangapuram, S., Salinas, D., Schulz, J., et al. Gluonts: Probabilistic and neural time series modeling in python. *Journal of Machine Learning Research*, 21(116): 1–6, 2020.

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., Zschiegner, J., Maddix, D. C., Wang, H., Mahoney, M. W., Torkkola, K., Wilson, A. G., Bohlke-Schneider, M., and Wang, Y. Chronos: Learning the language of time series, 2024. URL https://arxiv.org/abs/2403.07815.

Ansari, A. F., Shchur, O., Küken, J., Auer, A., Han, B., Mercado, P., Rangapuram, S. S., Shen, H., Stella, L., Zhang, X., Goswami, M., Kapoor, S., Maddix, D. C., Guerron, P., Hu, T., Yin, J., Erickson, N., Desai, P. M., Wang, H., Rangwala, H., Karypis, G., Wang, Y., and Bohlke-Schneider, M. Chronos-2: From univariate to universal forecasting, 2025. URL https://arxiv.org/abs/2510.15821.

Chen, M., Shen, L., Li, Z., Wang, X. J., Sun, J., and Liu, C. Visionts: Visual masked autoencoders are free-lunch zero-shot time series forecasters, 2024. URL https://arxiv.org/abs/2408.17253.

Gruver, N., Finzi, M., Qiu, S., and Wilson, A. G. Large language models are zero-shot time series forecasters, 2023. URL https://arxiv.org/abs/2310.07820.

Hernandez, D., Kaplan, J., Henighan, T., and McCandlish, S. Scaling laws for transfer, 2021. URL https://arxiv.org/abs/2102.01293.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Huh, M., Cheung, B., Wang, T., and Isola, P. The platonic representation hypothesis, 2024. URL https://arxiv.org/abs/2405.07987.

Iurada, L., Ciccone, M., and Tommasi, T. Efficient model editing with task-localized sparse fine-tuning, 2025.

Jha, R., Zhang, C., Shmatikov, V., and Morris, J. X. Harnessing the universal geometry of embeddings, 2025. URL https://arxiv.org/abs/2505.12540.

Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S., and Wen, Q. Time-llm: Time series forecasting by reprogramming large language models, 2023. URL https://arxiv.org/abs/2310.01728.

Kumar, A., Raghunathan, A., Jones, R., Ma, T., and Liang, P. Fine-tuning can distort pretrained features and underperform out-of-distribution, 2022. URL https://arxiv.org/abs/2202.10054.

Li, M. Z., Agrawal, K. K., Ghosh, A., Teru, K. K., Santoro, A., Lajoie, G., and Richards, B. A. Tracing the representation geometry of language models from pretraining to post-training, 2025. URL https://arxiv.org/abs/2509.23024.

Mircea, A., Chakraborty, S., Chitsazan, N., Rish, I., and Lobacheva, E. Training dynamics underlying language model scaling laws: Loss deceleration and zero-sum learning. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28154–28188, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1366. URL https://aclanthology.org/2025.acl-long.1366/.

Qiu, X., Tong, J., Sun, Y., Ma, Y., Zhang, W., and Shen, X. Rethinking the role of llms in time series forecasting, 2026. URL https://arxiv.org/abs/2602.14744.

Riachi, R., Rasul, K., Ashok, A., Humane, P., Roger, A., Williams, A. R., Nevmyvaka, Y., and Rish, I. Random initialization can't catch up: The advantage of language model transfer for time series forecasting, 2025. URL https://arxiv.org/abs/2506.21570.

Roger, A., Legate, G., Rasul, K., Nevmyvaka, Y., and Rish, I. Small vocabularies, big gains: Pretraining and tokenization in time series models, 2025. URL https://arxiv.org/abs/2511.11622.

Roschmann, S., Bouniot, Q., Feofanov, V., Redko, I., and Akata, Z. Time series representations for classification lie hidden in pretrained vision transformers. *arXiv preprint arXiv:2506.08641*, 2025.

Roy, O. and Vetterli, M. The effective rank: A measure of effective dimensionality. In *2007 15th European Signal Processing Conference*, pp. 606–610, Sep. 2007.

Tan, M., Merrill, M. A., Gupta, V., Althoff, T., and Hartvigsen, T. Are language models actually useful for time series forecasting? In *Advances in Neural Information Processing Systems*, volume 37, pp. 60162–60191, 2024.

Wang, P., Zhang, N., Tian, B., Xi, Z., Yao, Y., Xu, Z., Wang, M., Mao, S., Wang, X., Cheng, S., Liu, K., Ni, Y., Zheng, G., and Chen, H. Easyedit: An easy-to-use knowledge editing framework for large language models, 2024. URL https://arxiv.org/abs/2308.07269.

Wolff, M. L., Yang, S., Torkkola, K., and Mahoney, M. W. Using pre-trained llms for multivariate time series forecasting, 2025. URL https://arxiv.org/abs/2501.06386.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Zhang, X., Feng, S., and Li, X. From text to time? rethinking the effectiveness of the large language model for time series forecasting. *arXiv preprint arXiv:2504.08818*, 2025.

Zheng, L. N., Dong, C., Zhang, W. E., Yue, L., Xu, M., Maennel, O., and Chen, W. Understanding why large language models can be ineffective in time series analysis: The impact of modality alignment. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 4026–4037. Association for Computing Machinery, 2025.

Zhou, T., Niu, P., Wang, X., Sun, L., and Jin, R. One fits all: Power general time series analysis by pretrained lm, 2023. URL https://arxiv.org/abs/2302.11939.

A. Experimental Setup

A.1. Hyperparameters and Reproducibility

Table 1. Complete hyperparameter configuration for all experiments. All runs share the same configuration except where noted.

Category	Parameter	Value
Model	Base model	Qwen3-0.6B
	Architecture	Qwen3ForCausalLM and Qwen defaults
Tokenizer	Scaling	Z-score (mean/std from context)
	Binning	Uniform
	Vocab size (V)	1024
	Bin range	$[-5, 5]$
	Use EOS token	No
Training	Optimizer	AdamW
	Learning rate	1×10^{-4}, 3×10^{-4}, 1×10^{-3}, or 3×10^{-3}
	LR schedule	Linear warmup + cosine decay
	Warmup ratio	3%
	LR end factor	1×10^{-4}
	Precision	bf16 (mixed)
Batching	Effective batch size	128
	Epoch length	5,000 steps
Data	Dataset	GiftEval Pretrain (152 sub-datasets)
	Context length	512
	Target length	64
	Window stride	600
Loss	Loss function	Quantile (pinball) loss
	Quantiles	$\{0.1, 0.2, \ldots, 0.9\}$
	Softmax temperature	10^{-2}
Evaluation	Eval samples	1,000 extracted from the dataset (see script)
	Eval horizons	$h \in \{1, 64\}$
	Eval strategy	Autoregressive
LoRA	Rank (r)	4, 8
	Alpha (α)	16, 32
	Dropout	0.05
	Bias	None
	Targets (Attn)	`q_proj, k_proj, v_proj, o_proj`
Reproducibility	Random seed	420
	Seeded modules	Python, NumPy, PyTorch (CPU + CUDA)
	Compute Usage	About 12 hours on 8 Nvidia A100 per model

A.2. Evaluation Metrics

This section details the evaluation metrics implemented for assessing the forecasting performance of the models. For all fixed-point metrics, the 0.5 quantile (median) is extracted from the probabilistic forecasts and used as the deterministic prediction.

Aggregation Strategy and Implementation: In our evaluation methodology, we follow closely the GluonTS library (Alexandrov et al., 2020), particularly, the aggregation approach. All metrics described below are computed *per-sequence first*, and then averaged across all sequences in the dataset. Specifically, the error or score is aggregated over the forecast horizon H for each individual time series, and the final reported metric is the unweighted mean of these per-sequence scores.

This macro-averaging ensures that every sequence contributes equally to the final evaluation, regardless of its absolute magnitude or scale.

To ensure robust evaluation, any time steps containing NaN values in the ground truth are dynamically excluded from computation by masking. Additionally, a small utility constant (ϵ = 1e-8) is added to denominators in percentage and scale-normalized metrics to prevent division by zero.

A.2.1. Point Forecast Metrics

Basic Error Metrics Mean Squared Error (MSE) quantifies prediction accuracy by heavily penalizing larger errors, making it sensitive to outliers. Root Mean Squared Error (RMSE) provides the error in the original data units while retaining MSE's sensitivity.

$$\text{MSE} = \frac{1}{H}\sum_{t=1}^{H}(y_t - \hat{y}_t)^2$$

$$\text{RMSE} = \sqrt{\text{MSE}}$$

where H is the forecast horizon, y_t is the true value, and $\hat{y}_t$ is the predicted median.

Mean Absolute Error (MAE) applies a linear penalty to measure the average absolute difference:

$$\text{MAE} = \frac{1}{H}\sum_{t=1}^{H}|y_t - \hat{y}_t|$$

Normalized Metrics Mean Absolute Scaled Error (MASE) measures forecast accuracy relative to a naive baseline, allowing for cross-series comparison across data with vastly different scales.

$$\text{MASE} = \frac{\text{MAE}}{\frac{1}{H-m}\sum_{t=m+1}^{H}|y_t - y_{t-m}|}$$

The seasonality parameter m is selected based on the expected periodic patterns of the timeframe ($m = 1$ for a general naive baseline, $m = 60$ for 1-minute data, and $m = 24$ for 1-hour data).

To further achieve scale-invariance, RMSE and absolute deviations are normalized by the scale of the data (mean or sum of absolute values) to compute the Normalized Root Mean Squared Error (NRMSE) and Normalized Deviation (ND):

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{H}\sum_{t=1}^{H}(y_t - \hat{y}_t)^2}}{\frac{1}{H}\sum_{t=1}^{H}|y_t|}$$

$$\text{ND} = \frac{\sum_{t=1}^{H}|y_t - \hat{y}_t|}{\sum_{t=1}^{H}|y_t|}$$

Percentage-Based Metrics Mean Absolute Percentage Error (MAPE) provides a scale-independent metric as a percentage. To address MAPE's asymmetry problem and instability near zero, the Symmetric Mean Absolute Percentage Error (sMAPE) equally penalizes over-predictions and under-predictions:

$$\text{MAPE} = \frac{100}{H}\sum_{t=1}^{H}\frac{|y_t - \hat{y}_t|}{|y_t| + \epsilon}$$

$$\text{sMAPE} = \frac{200}{H}\sum_{t=1}^{H}\frac{|y_t - \hat{y}_t|}{|y_t| + |\hat{y}_t| + \epsilon}$$

A.2.2. PROBABILISTIC METRICS

Continuous Ranked Probability Score (CRPS) The approximated CRPS measures the accuracy of probabilistic forecasts by evaluating the entire predictive distribution using quantile loss (QL).

$$\text{CRPS}_{\text{approx}} = 2 \sum_{q} w_q \cdot \text{QL}_q(y_{\text{true}}, y_{\text{pred}})$$

where $\text{QL}_q(y, \hat{y}) = (q - \mathbf{1}_{y \leq \hat{y}}) \cdot (y - \hat{y})$ and w_q represents the discrete weight calculated based on the distance between quantile levels.

Weighted Mean Absolute Percentage Quantile Loss (wMAPE) To compare probabilistic performance across heterogeneous time series, we implement a scale-invariant version of quantile loss weighted by the sum of absolute true values:

$$\text{wMAPE} = \frac{1}{Q} \sum_{q=1}^{Q} \frac{\sum_{t=1}^{H} \text{QL}_q(y_t, \hat{y}_{q,t})}{\sum_{t=1}^{H} |y_t|}$$

A.2.3. DIRECTIONAL AND CORRELATION METRICS

Directional Accuracy (DA) DA evaluates how often the model correctly predicts the direction of movement relative to a historical anchor point y_0 (the last observed value).

$$\text{DA} = \frac{1}{H} \sum_{t=1}^{H} \mathbf{1}_{\text{sign}_\tau(\hat{y}_t - y_0) = \text{sign}_\tau(y_t - y_0)}$$

Correlation Metrics To evaluate the model's ability to capture trend dynamics independently of absolute magnitude errors, we utilize Pearson correlation coefficients. Pearson evaluates the linear relationship between predictions and ground truth.

A.3. Experimental Results

Here we provide the complete set of evaluation results across all training regimes and metrics. Figure 5 shows the training progression of all eleven h=1 metrics over 4096 training steps, revealing a consistent pattern across metrics: language-pretrained models begin converging within the first 10–100 steps, while randomly initialized models require half an order of magnitude more training to reach comparable performance. By step 4096, both initialization strategies converge to similar error levels across all regimes, confirming that the pretrained weights accelerate optimization rather than alter the final solution. Tables 2 and 3 report the full numerical results at step 128 for single-step (h=1) and multi-step (h=64) forecasting respectively, alongside four established baselines and a zero-shot Qwen3 text baseline. At this early checkpoint, the transfer advantage of language pretraining is clearly visible: pretrained LoRA Attn achieves a CRPS of 20.10 compared to 154.5 for its randomly initialized counterpart, while all pretrained regimes outperform the random initializations that have not yet begun to converge. The Qwen3 text baseline performs orders of magnitude worse than all trained models, confirming that the language model's pretrained representations require domain-specific finetuning to be useful for time series forecasting.

A.4. Effective Transfer

Let $\mathcal{L}_R(d)$ and $\mathcal{L}_P(d)$ denote the validation losses achieved after training for d time series tokens using random and pre-trained initializations, respectively. For a given validation loss ℓ, $\mathcal{L}_\cdot^{-1}(\ell)$ therefore denotes the amount of data required to train a model to reach the loss level ℓ begining from a given initialization. We define transfer as "effective" or "positive" when starting the timeseries model training with the language model's weights allows us to achieve the same validation loss with less data than a model initialised randomly. This data quantity difference is what we refer to as the amount of data we "saved". Concretely, the *effective data transferred* $D_T(\ell)$ for a target loss ℓ is defined as the difference of these data amounts:

$$D_T(\ell) := \mathcal{L}_R^{-1}(\ell) - L_P^{-1}(\ell).$$

A positive $D_T(\ell)$ indicates that initializing the model from pre-trained language weights requires fewer training examples to reach the loss ℓ compared to random initialization, signifying an effective transfer from the upstream task to time series forecasting.

Table 2. Single-step (h=1) forecasting performance on the held-out evaluation set. All NanoTS variants use Qwen3-0.6B at training step 128. Best value per metric within each section is **bolded**. ↑: higher is better; ↓: lower is better.

	Model	CRPS↓	MSE↓	MAE↓	MASE↓	RMSE↓	NRMSE↓	ND↓	MAPE↓	sMAPE↓	wMAPE↓	DA↑
Pretrained	Full Finetune	49.51	10461	27.60	**1.561**	27.60	0.286	0.286	0.286	0.175	0.199	0.453
	IO Only	59.56	84846	66.53	3.430	66.53	0.614	0.614	0.614	0.300	0.250	0.469
	LoRA Attn	**20.10**	**8818**	**24.68**	1.607	**24.68**	**0.243**	**0.243**	**0.243**	**0.168**	**0.107**	**0.478**
	LoRA Attn+IO	22.54	11220	26.11	1.702	26.11	0.270	0.270	0.270	0.174	0.124	0.450
Random Init	Full Finetune	151.8	415339	180.4	7.819	180.4	0.914	0.914	0.914	0.532	0.463	**0.542**
	IO Only	111.6	280932	142.3	6.565	142.3	0.706	0.706	0.706	0.462	0.313	0.540
	LoRA Attn	154.5	430387	182.2	7.872	182.2	0.917	0.917	0.917	0.532	0.481	0.535
	LoRA Attn+IO	**50.68**	**59864**	**62.90**	**3.438**	**62.90**	**0.599**	**0.599**	**0.599**	**0.281**	**0.220**	0.476
Baselines	Chronos-Bolt	14.94	5409	18.91	0.934	18.91	0.240	0.240	0.240	0.146	0.096	0.678
	Chronos-T5	29.88	12764	32.99	1.212	32.99	**0.173**	**0.173**	**0.173**	**0.124**	**0.073**	0.646
	Chronos-2	**10.89**	**2886**	**13.38**	0.830	**13.38**	0.192	0.192	0.192	0.133	0.077	**0.724**
	ARIMA	11.94	3327	14.86	**0.805**	14.86	0.201	0.201	0.201	0.138	0.083	0.708
	Qwen3 Text	213.5	2.6M	213.5	376667	213.5	39508	39508	39508	0.563	19754	0.503

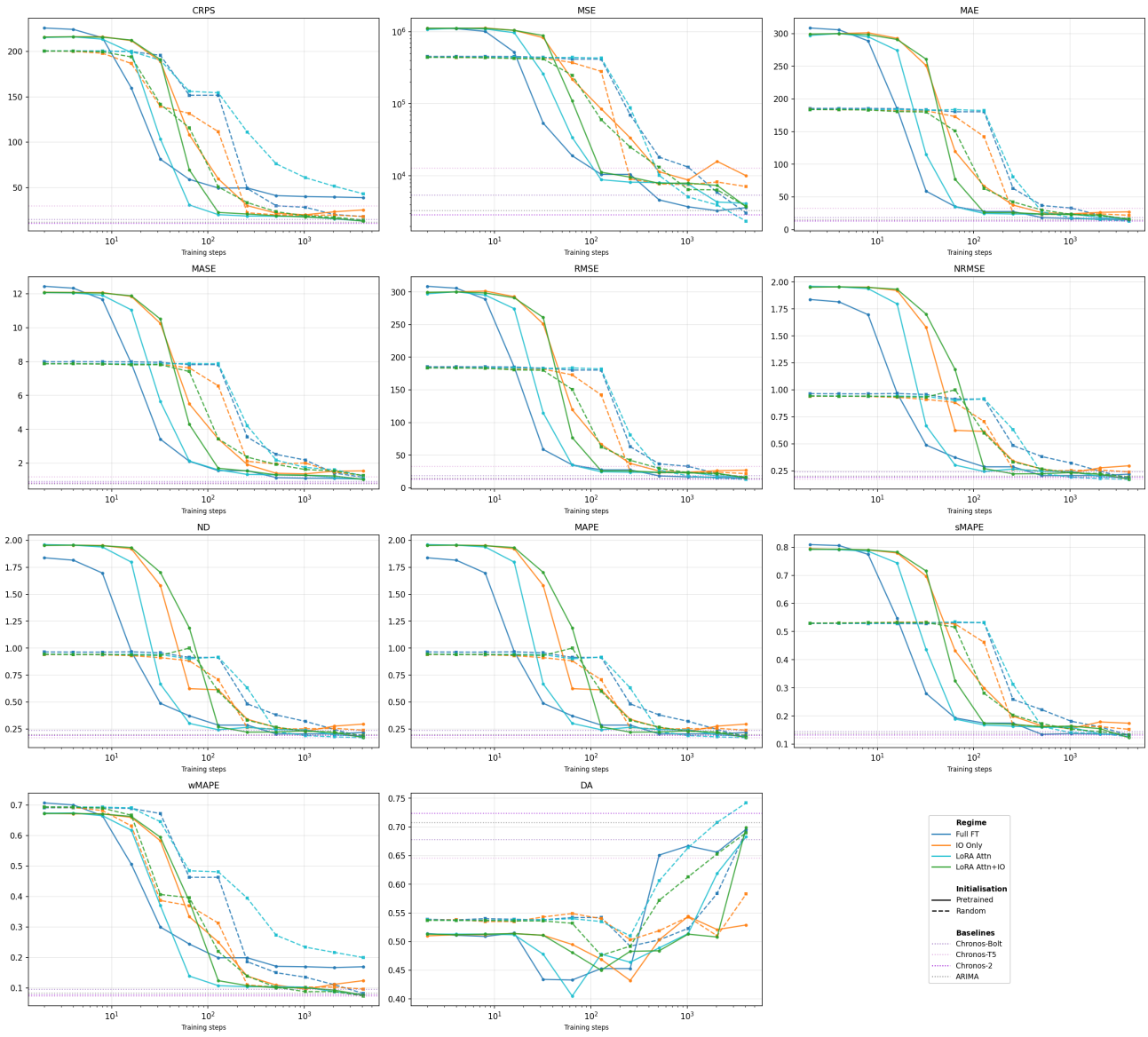

Figure 5. Training progression of all $h = 1$ forecasting metrics across training steps for four training regimes. Solid lines denote language-pretrained initialization (Qwen3-0.6B); dashed lines denote random initialization. Horizontal dotted lines indicate baseline performance (Chronos-T5, Chronos-Bolt, Chronos-2, ARIMA). Language-initialized models consistently converge earlier than their randomly initialized counterparts across all metrics, with the gap most pronounced in the first 100 steps. Both initializations converge in performance by 4096 steps, reaching levels competitive with established forecasting baselines.

Figure 6. Training progression of all $h = 64$ forecasting metrics across training steps for four training regimes. Solid lines denote language-pretrained initialization (Qwen3-0.6B); dashed lines denote random initialization. Horizontal dotted lines indicate baseline performance (Chronos-T5, Chronos-Bolt, Chronos-2, ARIMA). Language-initialized models consistently converge earlier than their randomly initialized counterparts across all metrics, with the gap most pronounced in the first 100 steps. Both initializations converge in performance by 4096 steps, reaching levels competitive with established forecasting baselines.

Table 3. Multi-step (h=64) forecasting performance on the held-out evaluation set. All NanoTS variants use Qwen3-0.6B at training step 128. Best value per metric within each section is **bolded**. ↑: higher is better; ↓: lower is better.

	Model	CRPS↓	MSE↓	MAE↓	MASE↓	RMSE↓	NRMSE↓	ND↓	MAPE↓	sMAPE↓	wMAPE↓	DA↑	Pearson↑
Pretrained	Full Finetune	**119.0**	271341	143.7	**6.424**	168.2	0.586	0.495	**2.85M**	**0.483**	**696208**	0.540	0.026
	IO Only	141.5	319226	147.9	6.502	174.9	**0.557**	**0.462**	2.68M	0.503	1.32M	**0.612**	0.015
	LoRA Attn	137.2	**259652**	**140.8**	6.602	**164.0**	0.581	0.487	5.39M	0.505	2.27M	0.555	0.029
	LoRA Attn+IO	138.8	265546	144.0	6.628	169.0	0.566	0.473	2.82M	0.485	1.36M	0.571	**0.052**
Random Init	Full Finetune	168.9	583696	210.5	7.700	232.7	0.575	0.494	693452	0.534	377358	0.642	0.009
	IO Only	141.3	445540	183.5	**7.341**	205.0	**0.563**	**0.478**	1.17M	**0.517**	394850	**0.653**	**0.036**
	LoRA Attn	171.4	592435	212.3	7.770	234.7	0.579	0.499	583786	0.536	339884	0.639	0.001
	LoRA Attn+IO	**137.9**	**346961**	**166.8**	7.812	**190.5**	0.719	0.628	**55012**	0.569	**104026**	0.502	−0.030
Baselines	Chronos-Bolt	**101.9**	**251450**	**131.7**	6.068	156.8	0.504	0.416	**1.55M**	0.452	**587680**	0.672	0.227
	Chronos-T5	115.2	292023	147.4	6.291	173.7	0.484	0.393	5.84M	0.462	1.98M	0.650	0.191
	Chronos-2	102.4	255127	132.0	**5.709**	**156.2**	**0.460**	**0.376**	2.48M	**0.441**	888025	**0.690**	**0.255**
	ARIMA	116.6	345041	145.4	6.609	171.8	0.534	0.446	5.80M	0.478	2.27M	0.608	0.105
	Qwen3 Text	489.1	213.1M	489.1	9911.0	720.8	5672.0	824.6	3.23M	0.568	1.62M	0.589	−0.048

B. Linear Probe over Frozen Pretrained States (Section 5 extended)

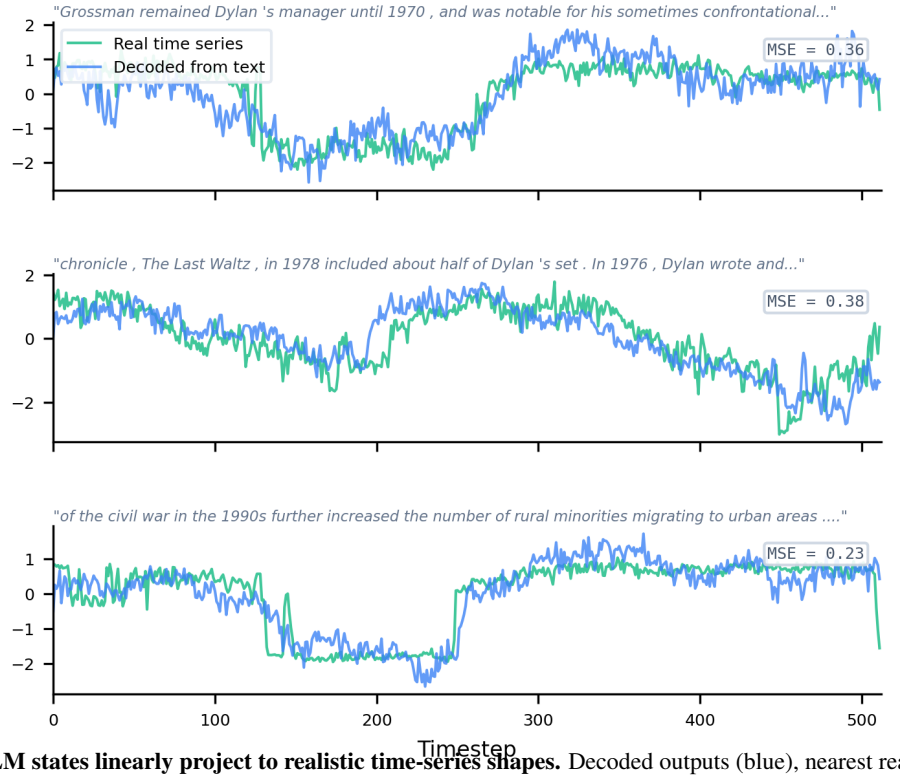

Figure 7. **Frozen LLM states linearly project to realistic time-series shapes.** Decoded outputs (blue), nearest real series (green), and WikiText input (gray). No paired supervision is used.

Setup. We pass $N = 1{,}920$ WikiText-103 sequences of $T = 512$ tokens through frozen Qwen3-0.6B and concatenate hidden states from all 28 layers:

$$\mathbf{h}_{i,t} = \left[\mathbf{h}_{i,t}^{(0)};\ \mathbf{h}_{i,t}^{(1)};\ \ldots\ ;\ \mathbf{h}_{i,t}^{(27)}\right] \in \mathbb{R}^D, \quad D = 28 \times 1024 = 28{,}672. \tag{1}$$

A single linear layer is applied independently at each timestep:

$$\hat{y}_{i,t} = \mathbf{w}^\top \mathbf{h}_{i,t} + b, \quad \mathbf{w} \in \mathbb{R}^D,\ b \in \mathbb{R}. \tag{2}$$

The output is z-score normalized to zero mean and unit variance.

Training objective. We use an EM-style algorithm matching each prediction to its nearest z-scored window from a bank of $M = 10{,}000$ GiftEval time series (Aksu et al., 2024). At each step we sample $K = 128$ candidate windows per prediction, choose $j_i^* = \arg\min_j \frac{1}{T}\|\hat{\mathbf{y}}_i - \mathbf{Y}_j\|^2$, and optimize:

$$\mathcal{L} = \underbrace{\frac{1}{B}\sum_{i=1}^{B}\frac{1}{T}\|\hat{\mathbf{y}}_i - \mathbf{Y}_{j_i^*}\|^2}_{\text{MSE to nearest real TS}} + \lambda \underbrace{\frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} \cos\left(|\text{FFT}(\hat{\mathbf{y}}_i)|^2,\ |\text{FFT}(\hat{\mathbf{y}}_j)|^2\right)}_{\text{PSD diversity penalty}}. \tag{3}$$

The PSD diversity penalty discourages mode collapse; we set $\lambda = 0.5$ and train with Adam ($\eta = 10^{-3}$) for 100 epochs with batch size $B = 32$.

2×2 ablation. Figure 8 shows the full ablation. Only pretrained weights and meaningful text together produce diverse, realistic temporal signals: with random initialization, unique coverage drops to 2.8% (54 matches); with random tokens through pretrained weights, it collapses to 0.2% (4 matches). Table 4 reports a fair top-K comparison controlling for diversity at each K value supported.

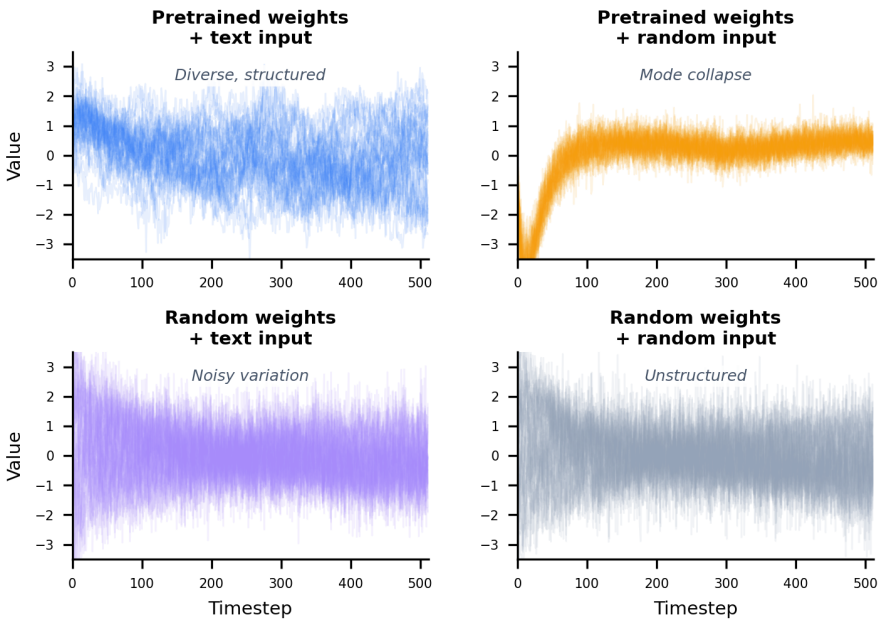

Figure 8. 2×2 **ablation: what creates the diverse temporal structure?** Crossing model weights (pretrained vs. randomly initialized, same architecture) with input (WikiText vs. random tokens). Only pretrained weights *and* meaningful text together produce diverse temporal signals; removing either factor causes mode collapse.

Table 4. Fair top-K comparison. For each condition we take the best-K unique matches and report mean nearest-neighbour MSE; "—" indicates fewer than K unique matches. Text + PT achieves the lowest MSE at every K.

K	text + PT	text + RandInit	rand + PT	rand + RandInit
4	**0.254**	0.383	0.330	0.412
54	**0.350**	0.721	—	0.755
99	**0.383**	0.825	—	0.862
200	**0.441**	0.971	—	1.004
439	**0.535**	—	—	—
686	**0.639**	—	—	—

Retrieval forecasting from projected dynamics. Beyond shape similarity, the projection carries *predictive* structure. Given the first 256 of 512 timesteps of a real time series, we retrieve the WikiText projection minimizing MSE on the observed half and use the projection's continuation as the forecast (Figure 9). Across 500 queries, retrieval MSE is 1.91 vs. 2.27 for last-value carry-forward (a 16% improvement); the median win, when retrieval succeeds, is ∼4× lower MSE because the retrieved projection captures level shifts and trends that last-value misses.

Implication. Define the pretrained model as a deterministic update $\mathbf{h}_{t+1} = F(\mathbf{h}_t, x_{t+1})$ over its prefix representations. Our linear probe identifies a direction $\mathbf{w}$ whose projected trajectory $\{y_t = \mathbf{w}^\top \mathbf{h}_t\}$ resembles real time series, and these projections carry useful continuation signal. This makes the pretrained representation space a plausible target for low-rank updates, rather than an obstacle to overcome.

C. Gradient Coherence (Section 6 extended)

For a fixed evaluation batch of N=32 individual time-series sequences, we compute per-sample gradients $\mathbf{g}_i = \nabla_\theta \mathcal{L}(x_i; \theta)$ over all model parameters at every checkpoint and measure mean off-diagonal pairwise cosine similarity: $\frac{1}{N(N-1)} \sum_{i \neq j} \cos(\mathbf{g}_i, \mathbf{g}_j)$. Figure 10 extends Figure 4 to all four adaptation regimes: across every regime, LANGINIT starts at high gradient coherence and RANDINIT starts near zero, with the loss only beginning to descend after coherence has emerged. The pattern is consistent with low-rank adaptation operating along a small set of pre-aligned directions: when

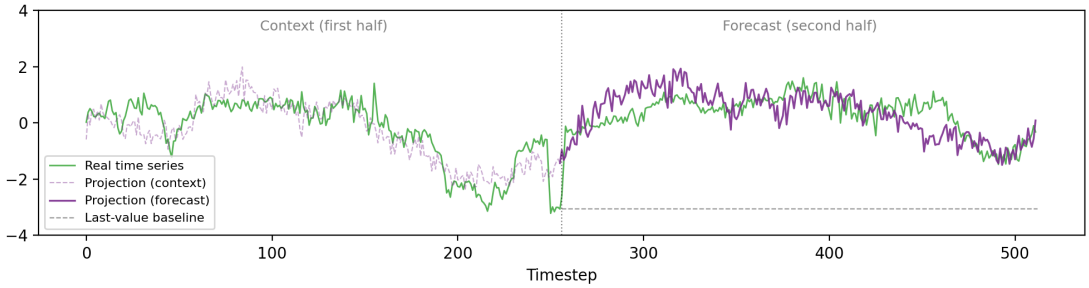

Figure 9. **Retrieval forecasting from projected pretrained states.** The first 256 timesteps are used to retrieve the closest WikiText projection; its continuation forecasts the rest. The retrieved projection captures the level shift and subsequent trend (MSE $= 0.42$), while last-value carry-forward (gray dashed) is stranded (MSE $= 11.7$).

those directions exist (LANGINIT), every example agrees on which way to update; when they do not (RANDINIT), the optimizer must first build them.

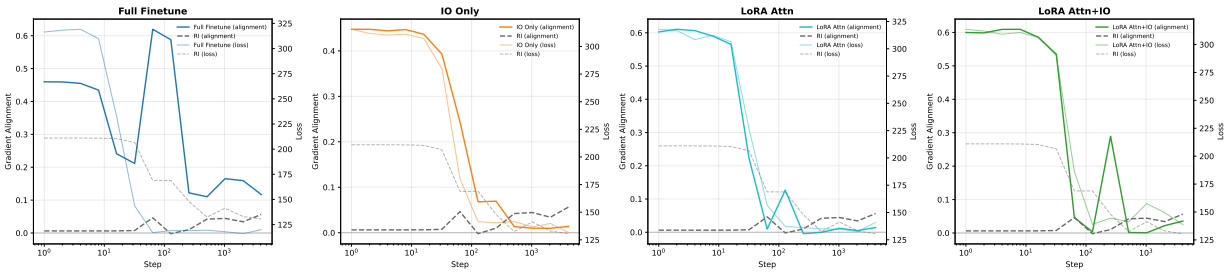

Figure 10. **Gradient alignment and evaluation loss across training, all four regimes.** Per-sample gradient alignment (left axis) and CRPS (right axis). Solid: LANGINIT; dashed: RANDINIT. LANGINIT models exhibit high gradient coherence from step 1 and loss descends immediately; RANDINIT models remain near zero coherence until step ~ 64–256, at which point loss begins to fall. The coupling between coherence onset and loss descent is the optimization-side counterpart of the geometric reuse described in the main text.

D. Reuse vs. Reinvention: Synthetic Waveform Geometry

Method details. We generate five length-512 synthetic inputs: a sine wave (period 64), a square wave (period 64), a sawtooth wave (period 64), a two-frequency signal $\sin(2\pi t/64) + 0.5\sin(2\pi t/17)$, and a linear trend with oscillation $t/T + 0.3\sin(2\pi t/80)$. Each is independently z-score normalized, clipped to $[-5, 5]$, and uniformly binned into 1024 tokens (matching the forecasting tokenizer). We extract hidden states and fit PCA independently per model–input pair; plots compare within-trajectory structure and PCA variance rather than absolute axes.

For phase coherence, we compute Euclidean distances d_{ij} between hidden states at positions i and j in the full hidden-state space (excluding the first five positions). For period P, positions are same-phase when $i \bmod P = j \bmod P$. We define

$$\text{coherence} = \frac{\text{mean}(d_{ij} \mid i \bmod P = j \bmod P)}{\text{mean}(d_{ij} \mid \text{all pairs})}. \tag{4}$$

Lower values indicate stronger phase structure; we plot $1 - \text{coherence}$ in training-dynamics figures so higher values correspond to stronger periodic encoding.

Layer-13 trajectories across waveforms. Figure 11 shows hidden-state trajectories for all five inputs at layer 13 across Random, Base (frozen pretrained), LANGINIT (finetuned pretrained), and RANDINIT. RANDINIT discovers clean, low-dimensional arcs (62–92% PCA variance); LANGINIT retains heterogeneous, layer-specific loops (34–98%) inherited from pretraining. The Base model already shows nontrivial structure for some inputs (e.g., square wave), evidence that finetuning is partly reshaping pre-existing sequential features rather than inventing temporal primitives. Despite different global geometries, t-SNE reveals near-identical local neighborhood structure between LANGINIT and RANDINIT (Figure 12), indicating that both models recover functionally equivalent representations through different geometric paths.

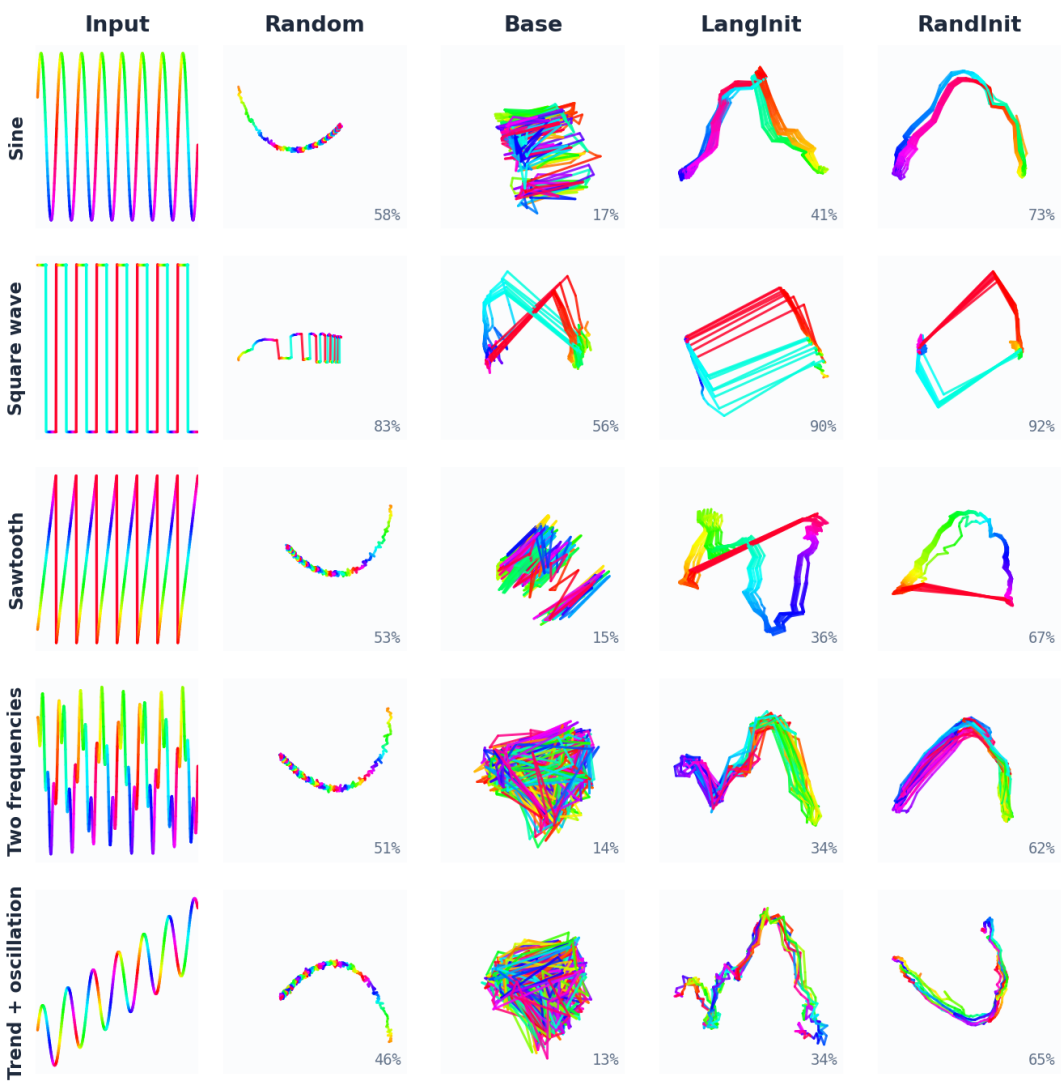

Figure 11. **Hidden-state trajectories for synthetic inputs at Layer 13 (PCA).** Trajectories are 2D PCA projections colored by input phase. Random and Base produce unstructured trajectories. RANDINIT discovers clean low-dimensional representations (62–92% PCA variance); LANGINIT creates geometrically complex but structured trajectories (34–98%) varying by input type.

All-layer sine wave PCA. Figures 13 and 14 show PCA projections of a sine wave (period 64) across all 28 layers for both initializations. LANGINIT exhibits layer-specific geometry with varying complexity and moderate PCA variance (17–49%), reflecting the rich heterogeneous representations inherited from language pretraining. RANDINIT produces nearly identical clean arcs at every layer with uniformly high PCA variance (67–79%), confirming geometric homogeneity across the network.

E. Sparse Crosscoder: Shared Temporal Primitives

To test whether the reused subspace corresponds to interpretable features, we train one sparse crosscoder per layer with a shared encoder $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{1024\times4096}$ followed by Top-K sparsity ($K = 64$) and two per-domain decoders. The encoder maps pretrained activations on decimal-string time series and finetuned activations on binned time series into a common 4096-dim latent space; features are ranked by cross-domain firing between time series and WikiText.

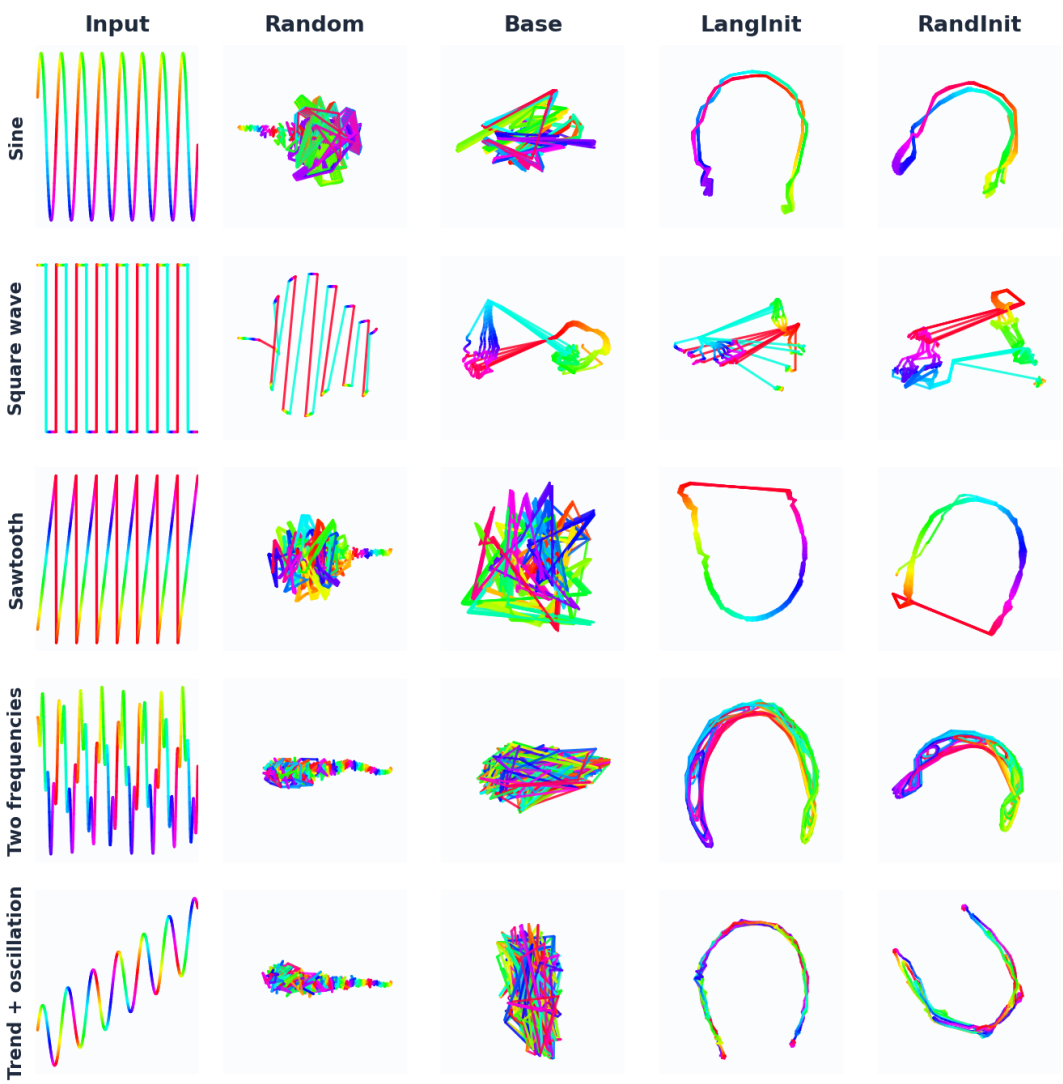

Figure 12. **Same trajectories, t-SNE projection.** Despite different global geometries from PCA, t-SNE shows that LANGINIT and RANDINIT develop similar local neighborhood structure: periodic inputs form smooth phase-ordered curves in both models, confirming functional equivalence reached via different geometric paths.

Examples of shared cross-modal features. Shared features concentrate in middle layers (7–10), the same layers identified by effective-rank dynamics as expanding their representational capacity during finetuning. The features link concrete time-series motifs to coherent WikiText genres (Table 5). The reused subspace therefore appears to be populated by reusable temporal primitives—periodicity, regime shifts, magnitude jumps, missingness—rather than abstract low-rank directions.

Causal evidence. A preliminary causal circuit analysis using zero-ablation (Wang et al., 2024) on the IO-only finetuned model identifies a superadditive Layer 1 circuit (head L1H4 $\leftrightarrow$ MLP$_{L1}$, $\rho = 1.51$) critical for periodic time-series prediction. Ablating the same circuit on WikiText-103 most degrades passages with sequential repetitive structure (biographical sequences, parallel-clause game mechanics, structured Wikipedia sections; mean $\Delta\mathcal{L} = 4.53$), suggesting the circuit tracks abstract sequential repetitive structure shared between modalities rather than periodicity-specific computation.

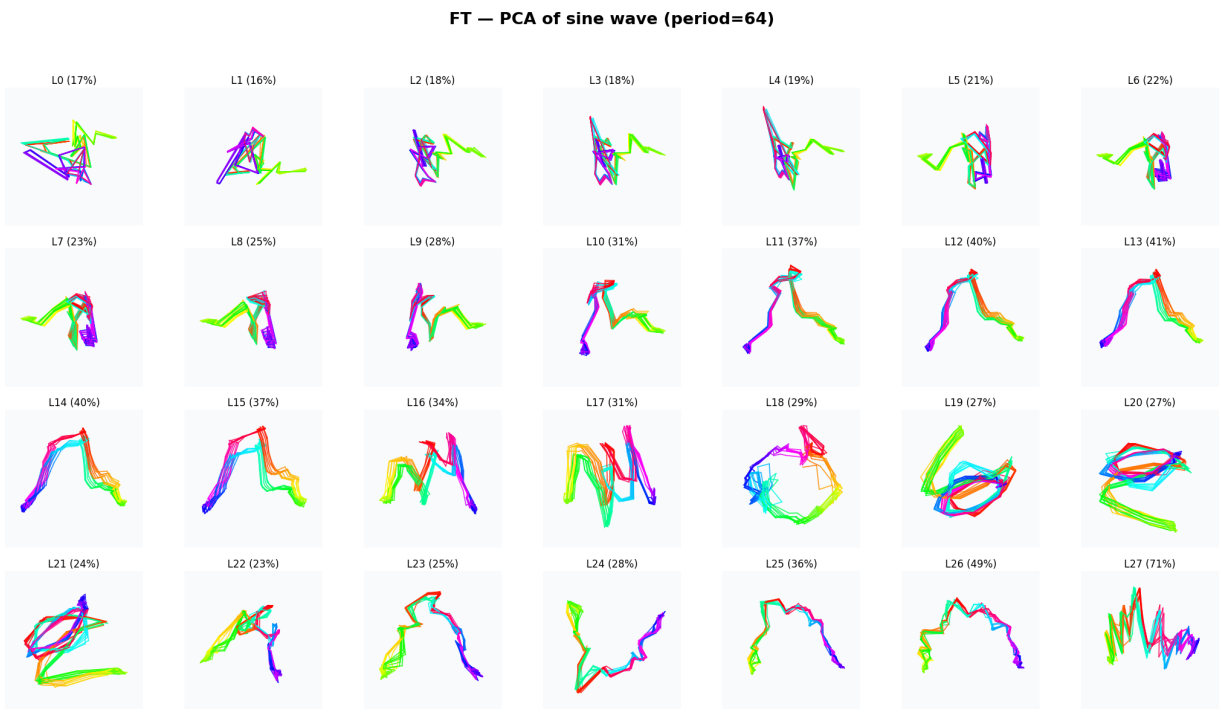

Figure 13. **LANGINIT: sine wave PCA across all 28 layers.** Layer-specific geometry with moderate PCA variance (17–49%); finetuning preserves much of the inherited structure.

Table 5. Examples of shared PT–FT crosscoder features. Shared features concentrate in middle layers and connect time-series motifs to coherent WikiText genres.

Layer	Feature	Time-series motif	WikiText motif
10	1712	magnitude jumps / plateaus	measurements and unit conversions
9	2469	volatile regime changes	tropical cyclone narratives
7	3888	isolated spikes	timestamped naval battles
8	2567	missing / NaN windows	<unk> and incomplete references

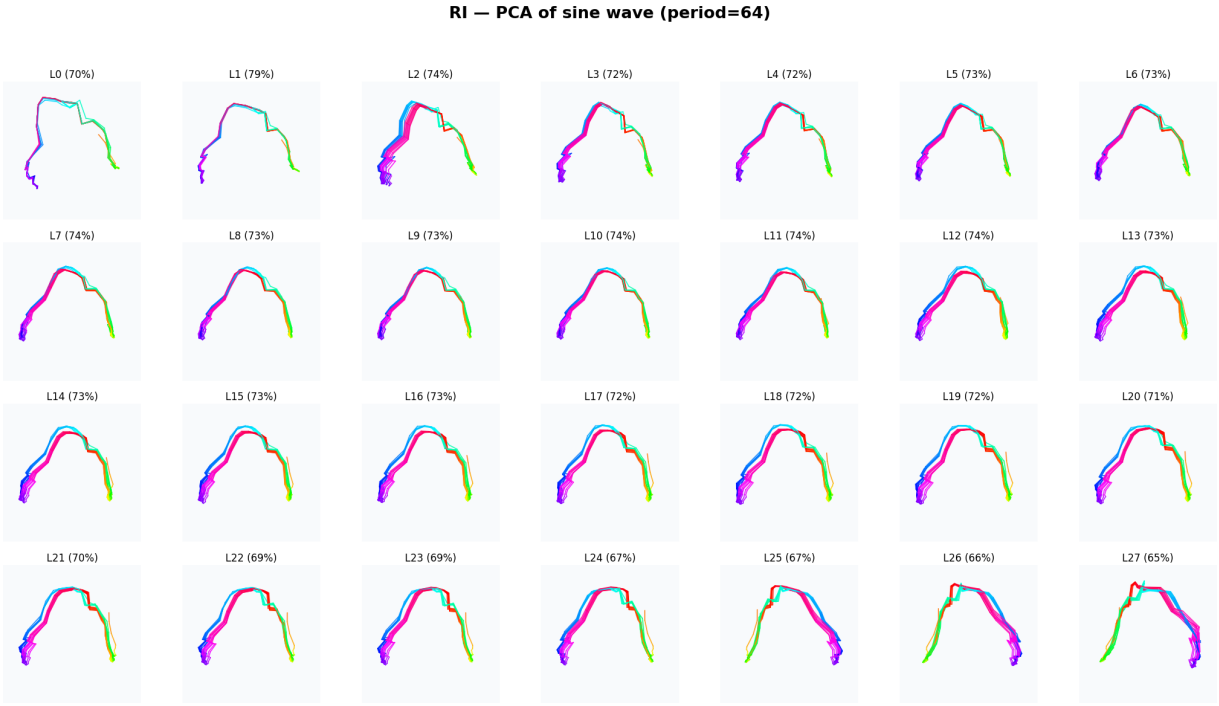

Figure 14. **RANDINIT: sine wave PCA across all 28 layers.** Nearly identical clean arcs at every layer with uniformly high PCA variance (67–79%): a homogeneous low-dimensional manifold built from scratch.