Emergent Latent-State Computation under Stochastic Volatility

Xiaoyu Huang [1] **Lulu Wang** [2]

Abstract

Mechanistic interpretability has largely focused on language models and deterministic toy tasks. Much less is known about how sequence models internally represent latent stochastic dynamics under noisy, partially observed observations. We study this question in a controlled multivariate stochastic volatility setting, where models observe only returns while the ground-truth latent volatility state is known to the researcher. This setting provides a useful benchmark for mechanistic interpretability under partial observability: the latent state is hidden from the model but directly available for evaluation. Across architectures, losses, and output heads, we find evidence for a two-stage computation. Hidden representations encode substantial information about the next latent volatility state, and the output head maps this representation to squared return forecasts. Furthermore, in Transformers, latent-state decodability emerges at identifiable architectural stages whose location depends on the volatility period. In long-cycle regimes, this computation simplifies into an explicit latent-state filter consisting of a learned linear projection followed by ℓ^2 normalization. Output-head replacement further shows that part of the degradation under noisy MSE training arises from readout misalignment rather than representation failure. These results suggest that stochastic volatility models provide a useful benchmark for mechanistic interpretability under noisy latent dynamics and partial observability.

[1]Department of Mathematics, Temple University, Philadelphia, PA, USA [2]Department of Data Analytics, Dickinson College, Carlisle, PA, USA. Correspondence to: Xiaoyu Huang <xiaoyu.huang@temple.edu>, Lulu Wang <wanglu@dickinson.edu>.

Mechanistic Interpretability Workshop at the 43rd International Conference on Machine Learning, Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

1. Introduction

Mechanistic interpretability has largely focused on language models and synthetic algorithmic tasks (Wang et al., 2022; Bricken et al., 2023; Nanda et al., 2023; Chughtai et al., 2023), where the target computation is often deterministic and fully observable. Much less is known about how neural sequence models represent latent stochastic dynamics in noisy, partially observed settings. Stochastic volatility models (Taylor, 1986; Kim et al., 1998; Shephard, 1996) provide a natural testbed: a central class of financial econometric models in which volatility evolves as an unobserved stochastic state rather than as a deterministic function of past returns, as in ARCH/GARCH models (Engle, 1982; Bollerslev, 1986; Shephard, 1996). In this setting, returns are noisy observations of the latent process, and accurate forecasting often requires an implicit filtering computation that infers hidden states from past observations.

We study whether small Transformers trained only to forecast future squared returns recover internal representations of latent volatility states, and how these representations emerge across the architecture. Using a controlled multivariate stochastic volatility process with cross-asset spillovers and varying volatility periods, we compare Transformers with MLP controls, vary training objectives and output heads, and analyze the learned computation using linear probes (Alain & Bengio, 2016), stage-wise representation analysis, and targeted interventions (Meng et al., 2022; Zhang & Nanda, 2023; Belrose et al., 2023).

Our results reveal three main findings. First, across architectures and losses, both Transformers and MLPs exhibit a robust two-stage decomposition: hidden states encode substantial information about the next latent volatility state, while the output head maps that representation to squared return forecasts. Second, in Transformers, latent-state decodability emerges at identifiable architectural stages, and the stage at which it first appears depends on the volatility period. Long-cycle regimes become largely decodable immediately after ℓ^2 normalization, whereas intermediate regimes require additional feed-forward refinement, and a short-period control regime depends strongly on positional embeddings. Third, in long-cycle regimes, the Transformer's latent-state computation simplifies into an explicit filter of the form in Equation (3), showing that most latent-state information is

extracted by a learned linear projection followed by nonlinear normalization. We further show that replacing the learned output head with a probe-aligned readout partially stabilizes performance under noisy MSE training, indicating that some forecast degradation arises from readout misalignment rather than failure to encode the latent state itself.

More broadly, we argue that stochastic volatility models offer a useful benchmark for mechanistic interpretability under noisy latent dynamics. Unlike many deterministic toy settings, they require implicit state inference under partial observability, but still permit controlled evaluation because the ground-truth latent state is known in simulation. This makes them a promising setting for studying how learned sequence models represent, transform, and use latent stochastic structure internally.

2. Background and Setup

Multivariate stochastic volatility process. We use a controlled multivariate stochastic volatility (MSV) process (Harvey et al., 1994) with $N = 6$ assets, where returns $\boldsymbol{r}_t = [r_{1,t}, \ldots, r_{N,t}]$ are noisy observations of a latent stochastic log-volatility state $\boldsymbol{h}_t = [h_{1,t}, \ldots, h_{N,t}]^\top$:

$$\boldsymbol{r}_t = \exp(\boldsymbol{h}_t/2) \odot \boldsymbol{\varepsilon}_t, \qquad \boldsymbol{\varepsilon}_t \sim \mathcal{N}(\mathbf{0}, \Sigma_\varepsilon), \tag{1}$$

where $\odot$ denotes elementwise multiplication. Thus,

$$\mathbb{E}(\boldsymbol{r}_t^2 \mid \boldsymbol{h}_t) = \mathrm{Var}(\boldsymbol{r}_t \mid \boldsymbol{h}_t) = \exp(\boldsymbol{h}_t).$$

The latent log-volatility state follows

$$\boldsymbol{h}_{t+1} = \boldsymbol{\mu} + A(\boldsymbol{h}_t - \boldsymbol{\mu}) + \boldsymbol{\eta}_t, \qquad \boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, \Sigma_\eta),$$

where the matrix A governs cross-asset volatility spillovers. In the baseline calibration, $\boldsymbol{\mu} = -1$, $\Sigma_\varepsilon = I_N$ and $\Sigma_\eta = \sigma_\eta^2 I_N$ with $\sigma_\eta = 0.02$. To generate persistent cyclic spillovers between assets, we construct

$$D = \mathrm{blkdiag}\left(D_1(\omega), D_2(\omega), \ldots\right),$$
$$D_j(\omega) = \begin{pmatrix} \cos(2\pi/\omega) & -\sin(2\pi/\omega) \\ \sin(2\pi/\omega) & \cos(2\pi/\omega) \end{pmatrix}, \tag{2}$$

and set $A = QDQ^\top$ for an orthogonal matrix Q. We scale A so that its largest absolute eigenvalue $\rho = 0.999999$ in the main specifications. The period parameter ω controls the cyclicity of the latent volatility dynamics and affects cross-asset volatility correlations.

We split the simulated trajectory into $n = 6000$ windows of length $L = 22$, giving a sample size comparable to a realistic forecasting setting, using the first $T = 21$ returns to predict the next-step squared returns:

$$y^{(k)} = \left(r_{1,T+1}^2, \ldots, r_{N,T+1}^2\right)^\top.$$

Thus, each model 21 past returns for all assets and predicts the next squared return for each asset.

Architectures. Our main model is a compact Transformer encoder with one layer and one attention head, chosen to facilitate interpretation. Each asset is treated as a token whose features are its length-T return history. For asset i, the input $x_i \in \mathbb{R}^T$ is linearly embedded to $\mathbb{R}^{64}$, ℓ^2-normalized, and combined with a learnable positional embedding:

$$\frac{x_i W_{\text{in}}^\top}{\|x_i W_{\text{in}}^\top\|_2} + p_i,$$

The resulting token representations are passed through a single Transformer encoder layer with self-attention, a feed-forward network (FFN) with GELU activation and residual connections, and dropout. Let z_i denote the final hidden representation for asset i. The output head maps z_i to a positive variance forecast,

$$\widehat{r}_{i,T+1}^2 = f\left(z_i W_{\text{head}}^\top + b_{\text{head}}\right),$$

where f is either the exponential or softplus function.

We compare the Transformer with an MLP baseline that processes each asset's return history independently, isolating the effect of Transformer-specific components such as attention-based cross-asset interaction.. The MLP uses three hidden layers of width 64 with GELU activations, together with the same output-head form as the Transformer. Full architectural details for both models are provided in Appendix A. Full architectural details for both the Transformer and MLP are provided in Appendix A.

Loss functions. We train models with two objectives. The first is MSE on next-step squared returns, as $\boldsymbol{r}_{T+1}^2$ is noisy proxy for conditional variance $\mathbb{E}(\boldsymbol{r}_t^2 \mid \boldsymbol{h}_t) = \mathrm{Var}(\boldsymbol{r}_t \mid \boldsymbol{h}_t)$. The second is the Gaussian negative log-likelihood (NLL), which uses the fact that returns are mean-zero Gaussian:

$$\mathcal{L}_{\text{NLL}} = \frac{1}{N} \sum_{i=1}^{N} \left[\log(\widehat{r}_{i,T+1}^2 + \delta) + \frac{r_{i,T+1}^2}{\widehat{r}_{i,T+1}^2 + \delta} \right],$$

with $\delta = 10^{-6}$ for numerical stability. Unlike MSE, NLL (Nix & Weigend, 1994) is directly aligned with conditional variance estimation and is uniquely minimized when the predicted $\widehat{\boldsymbol{r}}_t^2$ matches the true conditional variance $\mathbb{E}(\boldsymbol{r}_t^2 \mid \boldsymbol{h}_t) = \exp(\boldsymbol{h}_t)$ shown in Appendix C.

3. Emergent Mechanisms in Latent Volatility Forecasting

This section evaluates three hypotheses about the learned computation.

H1: Two-stage latent-state computation. The model first forms a hidden representation that encodes substantial information about the next latent log-volatility state $\boldsymbol{h}_{T+1}$, and the output head then maps this representation to squared-return forecasts.

H2: Stage-wise emergence in Transformers. In Transformers, latent-state decodability first appears at identifiable architectural stages, and the stage at which it emerges depends on the volatility period ω.

H3: Explicit latent filter in long-cycle regimes. For sufficiently long volatility cycles, most latent-state information is extracted by a simple computation consisting of a learned linear projection followed by ℓ^2-normalization, yielding an explicit latent-state filter.

3.1. Robust Two-Stage Latent-State Decomposition

Across output activations and training objectives, both the Transformer and MLP exhibit a two-stage computation that mirrors the classical SV pipeline of latent-state inference followed by variance prediction. Because the observation equation is nonlinear, latent-state filtering in stochastic-volatility models is itself nontrivial and is typically carried out with approximate or simulation-based methods rather than by simple closed-form filtering (Jacquier et al., 2002a; Kim et al., 1998).

1. **Latent volatility inference:** The model first maps past returns to hidden representations $\boldsymbol{z} = [z_1, \ldots, z_N] \in \mathbb{R}^{N \times d_{\text{model}}}$ that encode information about the next latent log-volatility state $\boldsymbol{h}_{T+1}$.

2. $\boldsymbol{r}_{T+1}^2$ **Prediction.** The output head then maps this representation to forecast $\hat{\boldsymbol{r}}_{T+1}^2$.

Linear probing. We fit a linear ridge-probe (Alain & Bengio, 2016) from each asset representation z_i to the corresponding MSV model's latent state $h_{i,T+1}$. If the first stage performs latent-volatility inference, then $h_{i,T+1}$ should be linearly decodable from z_i. We also compare this probe with the latent-state fit induced by the learned linear head readout $z_i W_{\text{head}}^\top$, which measures how well the output head uses the encoded latent information. Results for $\omega = 365$ corresponding to year-long volatility cycle are shown in Table 1.

The probe R^2 remains high across architectures, losses, and activations, showing that $\boldsymbol{z}$ linearly encodes substantial information about $\boldsymbol{h}_{T+1}$. In contrast, the learned readout R^2 is considerably more sensitive to both the training objective and output activation. Under the softplus activation, alignment is consistently weaker, likely because it is mismatched with the exponential observation model in Equation (1). Under MSE training, the learned output head often exhibits substantially lower R^2 than the probe despite strong latent-state decodability in $\boldsymbol{z}$. This suggests that part of the forecast degradation arises from readout misalignment rather than failure to encode the latent state itself.

Output-head intervention. We test whether the alignment gap is functionally relevant by replacing the learned output head with a ridge-probe readout trained on the final-layer representation $\boldsymbol{z}$. Under MSE training with exponential activation, the probe readout consistently improves MAE across volatility periods for both architectures, with larger gains for the Transformer in most regimes. MSE gains are more mixed, suggesting that the intervention primarily stabilizes typical forecast error rather than uniformly improving tail-sensitive squared error. The full alignment-gap and output-head intervention analysis are reported in Appendix D, and intervention results are summarized in Table 2.

3.2. Stage-wise Emergence of Latent-State Decodability

In this section, we identify where inside the Transformer the latent state $\boldsymbol{h}_{T+1}$ first becomes linearly decodable from the intermediate representation. We find that the earliest stage at which $\boldsymbol{h}_{T+1}$ is linearly recoverable depends on the volatility period ω of the AR component in Equation (2).

Section 3.2 reports the Ridge R^2 for predicting h_{t+1} at each component of the Transformer. For long-period dynamics ($\omega \in \{178, 365\}$), linear decodability already emerges after ℓ^2 normalization. For intermediate periods ($\omega \in \{30, 90\}$), the activation in the FFN also produces substantial additional gains, suggesting that the FFN refines representations that attention alone leaves only partially structured. We also include a control regime with $\omega = 21$, where the oscillation period matches the observation window length T and one can memorize h_{T+1} by periodicity. In this case, decodability remains near zero until positional embeddings are added, after which R^2 jumps sharply to approximately 1. Attention contributes little to all periods other than $\omega = 30$.

Less Structured Emergence in the MLP. By contrast, the MLP exhibits a less modular stage-wise pattern: latent-state decodability rises after the first activation/linear block and then improves gradually across layers, rather than emerging at distinct architecture-specific stages. Even with an inserted ℓ^2-normalization step, the MLP only partially mirrors the Transformer, with linear embedding and ℓ^2 normalization accounting for most of the decodability in long-cycle regimes (Appendix E).

Multi-Layer Transformer In the two-layer Transformer, most latent-state computation still occurs early in the first layer in a similar manner, and the second layer contributes only modest refinement. Detailed layer-wise results are shown in Appendix F.

Table 1. Latent-state decodability and output-head alignment across activation functions, model architectures, and training objectives. Entries report median [IQR] over $n = 10$ seeds.

Act.	Model	Loss	Ridge Probe R^2	Learned Readout R^2
exp	Transformer	NLL	0.9069 [0.8827, 0.9175]	0.8975 [0.8733, 0.9065]
	Transformer	MSE	0.8794 [0.8699, 0.8929]	0.7539 [0.7151, 0.7846]
	MLP	NLL	0.8985 [0.8800, 0.9098]	0.8899 [0.8754, 0.9003]
	MLP	MSE	0.8034 [0.7789, 0.8332]	0.7543 [0.7232, 0.7975]
spl	Transformer	NLL	0.9007 [0.8844, 0.9122]	0.8191 [0.7783, 0.8407]
	Transformer	MSE	0.8810 [0.8684, 0.8963]	0.8011 [0.6929, 0.8119]
	MLP	NLL	0.8863 [0.8660, 0.9053]	0.7933 [0.7542, 0.8110]
	MLP	MSE	0.8211 [0.7811, 0.8436]	0.7207 [0.6032, 0.7473]

Table 2. Output-head intervention under MSE training with exponential activation for selected ω. Entries report median MSE and MAE over $n = 10$ seeds for the learned head and a ridge readout from the final-layer representation. Positive gain indicates improvement. MAE improves consistently, while MSE effects are mixed.

		MSE			MAE		
Model	ω	Original	Probe	Gain	Original	Probe	Gain
Transformer	30	4.422	4.382	-1.39%	0.873	0.836	$+5.28\%$
	178	7.470	7.469	-0.11%	0.987	0.958	$+1.29\%$
	365	5.158	5.128	$+0.47\%$	0.856	0.853	$+0.66\%$
MLP	30	4.384	4.397	-1.34%	0.856	0.834	$+3.89\%$
	178	7.433	7.446	-0.87%	0.994	0.954	$+3.05\%$
	365	5.192	5.123	-0.01%	0.864	0.859	$+1.18\%$

3.3. Explicit Latent Filters in Long-Cycle Regimes

For long-cycle dynamics ($\omega \geq 365$), Section 3.2 reveals a simple mechanism: Ridge R^2 for decoding h_{t+1} jumps to approximately 0.9 from 0 after ℓ^2 normalization, and changes little afterwards. Thus, most latent-state information is extracted by a single nonlinear normalization step applied after a learned linear projection:

$$\hat{h}_{i,t+1} = \underbrace{\frac{x_i W_{\text{in}}^\top}{\|x_i W_{\text{in}}^\top\|_2}}_{\ell^2\text{-normalized embedding}} \underbrace{B}_{\text{ridge probe}}, \tag{3}$$

where W_{in} is the learned linear embedding and B is the probe readout. We refer to this composition as an *explicit latent-state filter*.

Causal Perturbation of the Normalized Embedding In Appendix H, we show that the learned projection and the ℓ^2 normalization step are jointly necessary. For $\omega = 365$, we ablate the component aligned with the ridge-probe direction at different stages and propagate the perturbed activation through the rest of the network. Ablating the post-ℓ^2-norm representation causes a broad downstream collapse in decodability shown in Section 3.2, with only partial recovery at the FFN, whereas later ablations have much weaker effects. This identifies the ℓ^2-normalized embedding as the earliest stage at which the latent-state signal becomes causally relevant for downstream computation. Further details are descried in Appendix H.

References

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Andersen, T. G., Bollerslev, T., Diebold, F. X., and Labys, P. Modeling and forecasting realized volatility. *Econometrica*, 71(2):579–625, 2003.

Barigozzi, M., Hallin, M., and Soccorsi, S. Identification of global and local shocks in international financial markets via general dynamic factor models. *Journal of Financial Econometrics*, 17(3):462–494, 2019.

Belinkov, Y. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Belrose, N., Schneider-Joseph, D., Ravfogel, S., Cotterell, R., Raff, E., and Biderman, S. Leace: Perfect linear concept erasure in closed form. *arXiv preprint arXiv:2306.03819*, 2023.

Bollerslev, T. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A.,

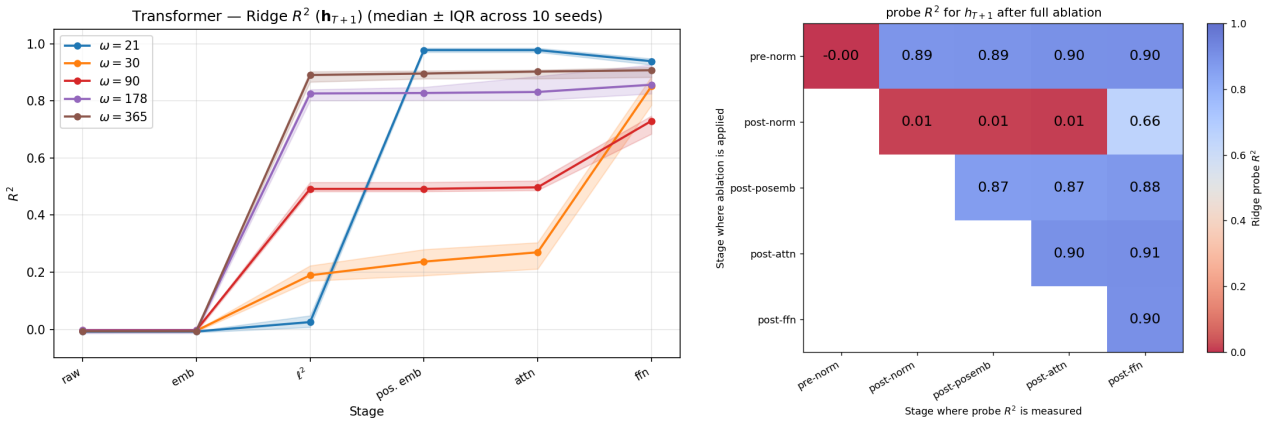


Figure 1. Stage-wise emergence and causal relevance of latent-state decodability in the Transformer. Left: latent-state decodability emerges at different architectural stages depending on the volatility period ω. Right: for $\omega = 365$, ablating the post-ℓ^2-norm representation produces the strongest downstream collapse in decodability, identifying it as the earliest causally relevant stage.

Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Chan, L., Garriga-Alonso, A., Goldowsky-Dill, N., Greenblatt, R., Nitishinskaya, J., Radhakrishnan, A., Shlegeris, B., and Thomas, N. Causal scrubbing: A method for rigorously testing interpretability hypotheses. AI Alignment Forum, 2022.

Chughtai, B., Chan, L., and Nanda, N. A toy model of universality: Reverse engineering how networks learn group operations. In *International Conference on Machine Learning*, pp. 6243–6267. PMLR, 2023.

Durbin, J. and Koopman, S. J. *Time series analysis by state space methods*. Oxford University Press (UK), 2012.

Engle, R. F. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pp. 987–1007, 1982.

Fama, E. F. and French, K. R. Common risk factors in the returns on stocks and bonds. *Journal of financial economics*, 33(1):3–56, 1993.

Geiger, A., Wu, Z., Potts, C., Icard, T., and Goodman, N. D. Finding alignments between interpretable causal variables and distributed neural representations. In *Proceedings of Machine Learning Research*, 2024.

Geiger, A., Ibeling, D., Zur, A., Chaudhary, M., Chauhan, S., Huang, J., Arora, A., Wu, Z., Goodman, N. D., Potts, C., and Icard, T. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 2025.

Harvey, A., Ruiz, E., and Shephard, N. Multivariate stochastic variance models. *The Review of Economic Studies*, 61 (2):247–264, 1994.

Hashemi, B., Corominas, R., and Giacchetto, A. Can transformers do enumerative geometry? In *International Conference on Learning Representations*, volume 2025, pp. 70047–70067, 2025.

Heimersheim, S. and Nanda, N. How to use and interpret activation patching. *arXiv preprint arXiv:2404.15255*, 2024.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Hewitt, J. and Liang, P. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

Huang, X., Jackson, B., and Lee, K.-H. From black box to bijection: Interpreting machine learning to build a zeta map algorithm. *arXiv preprint arXiv:2511.12421*, 2025.

Jacquier, E., Polson, N. G., and Rossi, P. E. Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics*, 20(1):69–87, 2002a.

Jacquier, E., Polson, N. G., and Rossi, P. E. Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics*, 20(1):69–87, 2002b.

Kim, S., Shephard, N., and Chib, S. Stochastic volatility: likelihood inference and comparison with arch models. *The review of economic studies*, 65(3):361–393, 1998.

Kitagawa, G. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

Li, T., Liu, Z., Shen, Y., Wang, X., Chen, H., and Huang, S. Master: Market-guided stock transformer for stock price forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 162–170, 2024.

Lim, B., Arık, S. Ö., Loeff, N., and Pfister, T. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International journal of forecasting*, 37(4):1748–1764, 2021.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

Nix, D. A. and Weigend, A. S. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pp. 55–60. IEEE, 1994.

Shephard, N. Statistical aspects of arch and stochastic volatility. In *Time series models*, pp. 1–68. Chapman and Hall/CRC, 1996.

Taylor, S. J. *Modelling financial time series*. Wiley, 1986.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vig, J., Gehrmann, S., Belinkov, Y., Qian, S., Nevo, D., Singer, Y., and Shieber, S. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33: 12388–12401, 2020.

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., and Sun, L. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

Yang, L., Ng, T. L. J., Smyth, B., and Dong, R. Html: Hierarchical transformer-based multi-task learning for volatility prediction. In *Proceedings of the web conference 2020*, pp. 441–451, 2020.

Zeng, Z., Kaur, R., Siddagangappa, S., Rahimi, S., Balch, T., and Veloso, M. Financial time series forecasting using cnn and transformer. *arXiv preprint arXiv:2304.04912*, 2023.

Zhang, F. and Nanda, N. Towards best practices of activation patching in language models: Metrics and methods. *arXiv preprint arXiv:2309.16042*, 2023.

Zhao, J., Chu, F., Xie, L., Che, Y., Wu, Y., and Burke, A. F. A survey of transformer networks for time series forecasting. *Computer Science Review*, 60:100883, 2026.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

A. Architectural Details

This section provides the full architectural details for the Transformer model and the MLP baseline used in the experiments.

A.1. Transformer Architecture

We model the conditional variance of asset returns using a Transformer encoder (Vaswani et al., 2017) that operates across assets. Given a panel of past returns

$$x \in \mathbb{R}^{N \times T},$$

where $x_i \in \mathbb{R}^T$ denotes the return history of asset i over the previous T time steps, each asset is treated as one token whose feature vector is its return trajectory.

The Transformer consists of three components: a linear embedding layer, a single Transformer encoder layer, and an output head. We use one attention head and one encoder layer throughout. Dropout with rate 0.1 is applied in the model.

Linear embedding. For each asset i, the return history $x_i \in \mathbb{R}^T$ is first mapped into a d_{model}-dimensional representation by a linear projection,

$$\tilde{e}_i = x_i W_{\text{in}}^\top, \tag{4}$$

where $W_{\text{in}} \in \mathbb{R}^{d_{\text{model}} \times T}$. We then normalize the embedded representation by its ℓ^2 norm,

$$\tilde{e}_{i,\text{norm}} = \frac{\tilde{e}_i}{\|\tilde{e}_i\|_2}, \tag{5}$$

and add a learnable asset-specific embedding,

$$e_i = \tilde{e}_{i,\text{norm}} + p_i, \tag{6}$$

where $p_i \in \mathbb{R}^{d_{\text{model}}}$ identifies asset i. In all experiments, we set $d_{\text{model}} = 64$.

Transformer encoder. Let

$$H = [e_1, \ldots, e_N]$$

denote the matrix of asset-token representations. The tokens are processed by a single Transformer encoder layer with one self-attention head, followed by a feed-forward network with GELU activation and residual connections. Specifically,

$$z_i = \left[\text{GELU}\left((h_i + \text{Attn}(H))\, W_1^\top + b_1\right)\right] W_2^\top + b_2, \tag{7}$$

where $\text{Attn}(H)$ denotes the self-attention operator and GELU denotes the Gaussian error linear unit (Hendrycks & Gimpel, 2016). We do not include layer normalization in the main specification, since it had little effect on forecasting performance in our experiments and removing it simplifies the interpretation of hidden representations.

Output head. A linear prediction head maps the final representation of each asset into a scalar variance forecast. For asset i,

$$s_i = z_i W_{\text{head}}^\top + b_{\text{head}}, \tag{8}$$

and the predicted next-period squared return is

$$\widehat{r}_{i,T+1}^2 = f(s_i), \tag{9}$$

where $W_{\text{head}} \in \mathbb{R}^{1 \times d_{\text{model}}}$ and $b_{\text{head}} \in \mathbb{R}$. We consider both exponential and softplus choices for the output activation f to ensure positive variance forecasts.

A.2. MLP Baseline

We use an MLP baseline to distinguish latent-state recovery through nonlinear univariate forecasting from recovery due to attention-based cross-asset aggregation. Unlike the Transformer, the MLP processes each asset independently and does not include asset-specific embeddings, self-attention, or cross-asset token interactions.

For each asset i, the input is its length-T return history $x_i \in \mathbb{R}^T$. The MLP consists of three hidden layers with GELU activations with width 64. Formally, the model computes

$$h_i^{(0)} = x_i, \tag{10}$$

$$a_i^{(\ell)} = h_i^{(\ell-1)} W_\ell^\top + b_\ell, \qquad \ell = 1, 2, 3, \tag{11}$$

$$h_i^{(\ell)} = \text{GELU}\left(a_i^{(\ell)}\right), \qquad \ell = 1, 2, 3, \tag{12}$$

$$z_i = h_i^{(3)}. \tag{13}$$

The output layer follows the same prediction-head structure as in the Transformer. Specifically,

$$s_i = z_i W_{\text{head}}^\top + b_{\text{head}}, \tag{14}$$

and the predicted next-period squared return is

$$\widehat{r}_{i,T+1}^2 = f(s_i), \tag{15}$$

where f is chosen to be either the exponential or softplus function to ensure positive squared return forecasts. The dropout configuration is kept the same as in the Transformer.

B. Spillover matrix A

We report the spillover matrix A from Equation (2) for all volatility periods $\omega \in \{21, 30, 90, 178, 365, 730\}$, denoted as A_ω, considered in the paper.

$$A_{21} = \begin{pmatrix} 0.9556 & -0.1398 & 0.0983 & 0.1905 & -0.1280 & 0.0710 \\ 0.1398 & 0.9556 & -0.2398 & 0.0852 & -0.0405 & 0.0304 \\ -0.0983 & 0.2398 & 0.9556 & 0.1167 & -0.0731 & 0.0276 \\ -0.1905 & -0.0852 & -0.1167 & 0.9556 & -0.0810 & -0.1523 \\ 0.1280 & 0.0405 & 0.0731 & 0.0810 & 0.9556 & -0.2387 \\ -0.0710 & -0.0304 & -0.0276 & 0.1523 & 0.2387 & 0.9556 \end{pmatrix}$$

$$A_{30} = \begin{pmatrix} 0.9781 & -0.0986 & 0.0693 & 0.1343 & -0.0903 & 0.0500 \\ 0.0986 & 0.9781 & -0.1692 & 0.0601 & -0.0286 & 0.0215 \\ -0.0693 & 0.1692 & 0.9781 & 0.0823 & -0.0515 & 0.0194 \\ -0.1343 & -0.0601 & -0.0823 & 0.9781 & -0.0571 & -0.1074 \\ 0.0903 & 0.0286 & 0.0515 & 0.0571 & 0.9781 & -0.1684 \\ -0.0500 & -0.0215 & -0.0194 & 0.1074 & 0.1684 & 0.9781 \end{pmatrix}$$

$$A_{90} = \begin{pmatrix} 0.9976 & -0.0331 & 0.0233 & 0.0451 & -0.0303 & 0.0168 \\ 0.0331 & 0.9976 & -0.0568 & 0.0202 & -0.0096 & 0.0072 \\ -0.0233 & 0.0568 & 0.9976 & 0.0276 & -0.0173 & 0.0065 \\ -0.0451 & -0.0202 & -0.0276 & 0.9976 & -0.0192 & -0.0360 \\ 0.0303 & 0.0096 & 0.0173 & 0.0192 & 0.9976 & -0.0565 \\ -0.0168 & -0.0072 & -0.0065 & 0.0360 & 0.0565 & 0.9976 \end{pmatrix}$$

$$A_{178} = \begin{pmatrix} 0.9994 & -0.0167 & 0.0118 & 0.0228 & -0.0153 & 0.0085 \\ 0.0167 & 0.9994 & -0.0287 & 0.0102 & -0.0048 & 0.0036 \\ -0.0118 & 0.0287 & 0.9994 & 0.0140 & -0.0087 & 0.0033 \\ -0.0228 & -0.0102 & -0.0140 & 0.9994 & -0.0097 & -0.0182 \\ 0.0153 & 0.0048 & 0.0087 & 0.0097 & 0.9994 & -0.0286 \\ -0.0085 & -0.0036 & -0.0033 & 0.0182 & 0.0286 & 0.9994 \end{pmatrix}$$

$$A_{365} = \begin{pmatrix} 0.9999 & -0.0082 & 0.0057 & 0.0111 & -0.0075 & 0.0041 \\ 0.0082 & 0.9999 & -0.0140 & 0.0050 & -0.0024 & 0.0018 \\ -0.0057 & 0.0140 & 0.9999 & 0.0068 & -0.0043 & 0.0016 \\ -0.0111 & -0.0050 & -0.0068 & 0.9999 & -0.0047 & -0.0089 \\ 0.0075 & 0.0024 & 0.0043 & 0.0047 & 0.9999 & -0.0139 \\ -0.0041 & -0.0018 & -0.0016 & 0.0089 & 0.0139 & 0.9999 \end{pmatrix}$$

$$A_{730} = \begin{pmatrix} 1.0000 & -0.0041 & 0.0029 & 0.0056 & -0.0037 & 0.0021 \\ 0.0041 & 1.0000 & -0.0070 & 0.0025 & -0.0012 & 0.0009 \\ -0.0029 & 0.0070 & 1.0000 & 0.0034 & -0.0021 & 0.0008 \\ -0.0056 & -0.0025 & -0.0034 & 1.0000 & -0.0024 & -0.0044 \\ 0.0037 & 0.0012 & 0.0021 & 0.0024 & 1.0000 & -0.0070 \\ -0.0021 & -0.0009 & -0.0008 & 0.0044 & 0.0070 & 1.0000 \end{pmatrix}$$

C. Optimality of Gaussian NLL for Conditional Variance

Theorem 1. *Fix an asset i and condition on the latent state $\boldsymbol{h}_{T+1}$. Under the observation model in Equation (1),*

$$r_{i,T+1} \mid \boldsymbol{h}_{T+1} \sim \mathcal{N}(0, \exp(h_{i,T+1}))\,.$$

Consider the per-asset Gaussian negative log-likelihood

$$\ell\big(\widehat{r}^2_{i,T+1}; r_{i,T+1}\big) = \log \widehat{r}^2_{i,T+1} + \frac{r^2_{i,T+1}}{\widehat{r}^2_{i,T+1}}, \qquad \widehat{r}^2_{i,T+1} > 0.$$

Then the conditional expected loss

$$\mathbb{E}\big[\ell\big(\widehat{r}^2_{i,T+1}; r_{i,T+1}\big) \mid \boldsymbol{h}_{T+1}\big]$$

is uniquely minimized at

$$\widehat{r}^2_{i,T+1} = \exp(h_{i,T+1}).$$

Proof. Conditioning on $\boldsymbol{h}_{T+1}$, we have

$$\mathbb{E}\big[r^2_{i,T+1} \mid \boldsymbol{h}_{T+1}\big] = \mathrm{Var}(r_{i,T+1} \mid \boldsymbol{h}_{T+1}) = \exp(h_{i,T+1}).$$

Therefore,

$$\mathbb{E}\big[\ell\big(\widehat{r}^2_{i,T+1}; r_{i,T+1}\big) \mid \boldsymbol{h}_{T+1}\big] = \log \widehat{r}^2_{i,T+1} + \frac{\exp(h_{i,T+1})}{\widehat{r}^2_{i,T+1}}.$$

Define

$$J(v) = \log v + \frac{\exp(h_{i,T+1})}{v}, \qquad v > 0.$$

Then

$$J'(v) = \frac{1}{v} - \frac{\exp(h_{i,T+1})}{v^2} = \frac{v - \exp(h_{i,T+1})}{v^2}.$$

Thus the unique critical point is

$$v = \exp(h_{i,T+1}).$$

Moreover,

$$J''(v) = -\frac{1}{v^2} + \frac{2\exp(h_{i,T+1})}{v^3},$$

so at $v = \exp(h_{i,T+1})$,

$$J''(\exp(h_{i,T+1})) = \frac{1}{\exp(2h_{i,T+1})} > 0.$$

Hence $\widehat{r}^2_{i,T+1} = \exp(h_{i,T+1})$ uniquely minimizes the conditional expected Gaussian NLL. □

D. Consistent Probe–Readout R^2 Gap Under MSE Loss

In Table 1, we reported a substantial gap between the hidden-state probe R^2 and the linear head R^2 at $\omega = 365$. Figure 2 shows that this gap persists across all volatility periods $\omega \in \{30, 90, 178, 365, 730\}$ for both the Transformer and the MLP under MSE training. For both architectures, the hidden probe R^2 is consistently higher than the linear head R^2. The gap narrows as ω increases, and both probe R^2 and head R^2 improve with longer volatility cycles. The Transformer exhibits a consistently smaller alignment gap than the MLP, especially at $\omega = 30$ and $\omega = 90$, suggesting that under MSE training its learned head remains better aligned with the latent-state direction encoded in the hidden representation.

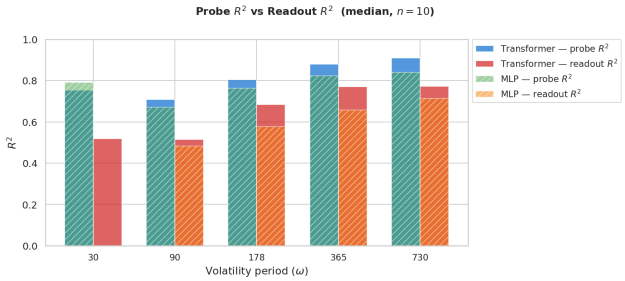


Figure 2. Probe R^2 and linear head R^2 for decoding $\boldsymbol{h}_{T+1}$ from the final representation under MSE training, shown across volatility periods for the Transformer and MLP (median over $n = 10$ seeds). In both architectures, probe R^2 remains above head R^2 across all periods, with smaller in long-cycle regimes. The R^2 in Transformer are consistently higher.

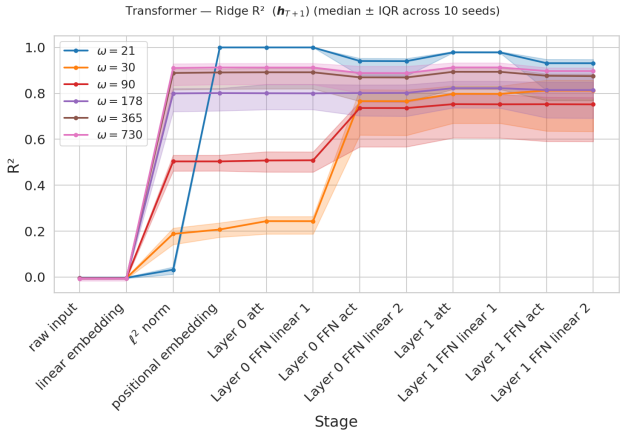


Figure 4. Stage-wise ridge probe R^2 for decoding h_{T+1} from intermediate representations in a two-layer Transformer across volatility periods. The main gains in decodability occur before or within the first layer, while the second layer yields only limited additional improvement.

E. Less Structured Emergence in the MLP

In the Transformer, the stage at which the latent state $\boldsymbol{h}_{T+1}$ first becomes linearly decodable depends strongly on the volatility cycle as shown in Section 3.2. This stage-specific pattern is much less pronounced in a standard MLP. As shown in Figure 3a, latent-state decodability in the MLP typically arises in a more diffuse and less modular manner than the Transformer. It arises after the first nonlinear block and then improves more gradually across subsequent layers, rather than emerging sharply at architecture-specific components.

We also consider an MLP control with an ℓ^2-normalization layer inserted after the initial linear embedding, mirroring the corresponding preprocessing step in the Transformer. As shown in Figure 3b, this normalized MLP exhibits a similar pattern: in long-cycle regimes, the linear embedding together with ℓ^2-normalization accounts for most of the decodability of h_{t+1}. As ω decreases, the dominant contribution shifts from ℓ^2-normalization alone to a shared contribution from ℓ^2-normalization and the subsequent nonlinear activation.

F. Stage-Wise Results for Two-Layer Transformers

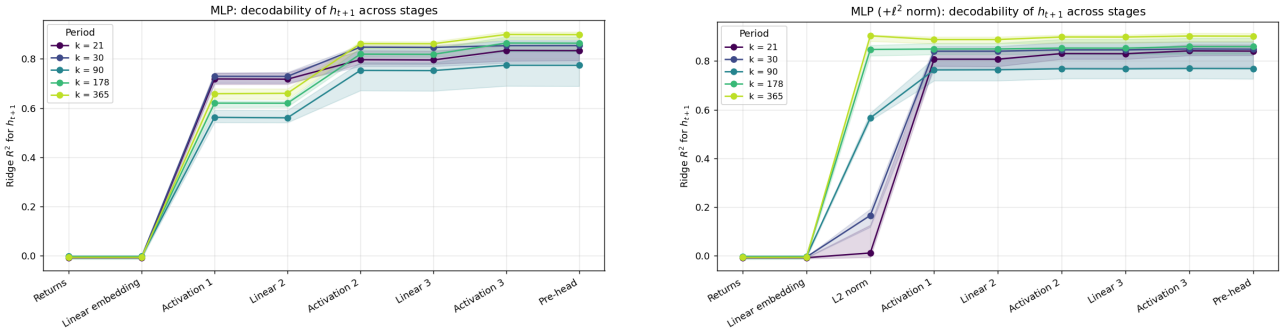
We also examine whether a deeper Transformer exhibits a different stage-wise emergence pattern. Figure 4 shows that the main decodability transitions still occur in the first layer. In our setting, increasing depth from one layer to two does not substantially change where the latent-state representation is formed.

G. Impact of Signal-to-Noise Ratio

The spectral radius ρ of the state-transition matrix A controls the persistence of the latent process $\boldsymbol{h}_t$, and hence its signal-to-noise ratio. Smaller ρ leads to faster decay and weaker recoverable structure, while ρ near 1 yields more persistent dynamics and higher latent-state decodability. Consistent with this, ridge probe R^2 rises sharply as ρ increases and then largely plateaus for the largest values; representative trajectories are shown in Figure 5, with quantitative results in Table 3.

Table 3. Mean ridge probe R^2 for decoding $\mathbf{h}_{T+1}$ from $\boldsymbol{z}$ across ρ for the Transformer and MLP ($n = 5$ seeds).

ρ	Transformer Mean R^2	MLP Mean R^2
0.9999	0.286	0.289
0.99999	0.873	0.872
0.999999	0.901	0.897
0.9999999	0.903	0.899



(a) Standard MLP.

(b) MLP with ℓ^2 normalization after the initial linear layer.

Figure 3. Stage-wise ridge probe R^2 for decoding $\boldsymbol{h}_{T+1}$ from intermediate MLP representations across volatility periods. **(a)** A standard MLP exhibits a smoother and less localized rise in decodability than the Transformer, with weaker dependence on the volatility period. **(b)** With an ℓ^2-normalization layer inserted after the initial linear embedding, the linear embedding together with ℓ^2-normalization accounts for most of the decodability of h_{t+1} in long-cycle regimes.

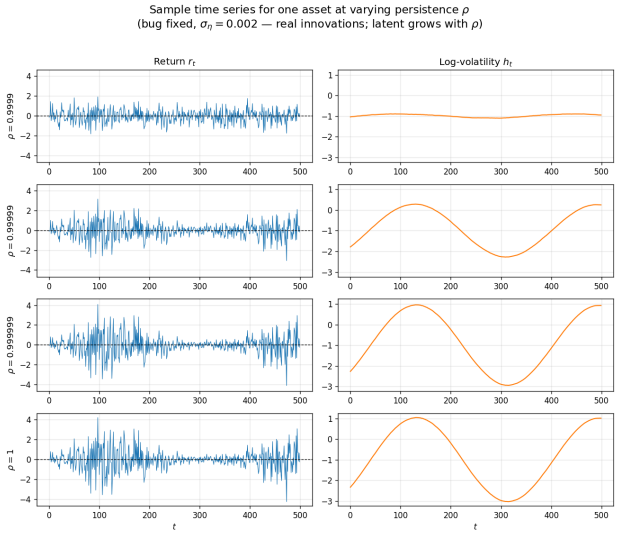


Figure 5. Representative latent trajectories for different spectral radii ρ. Smaller ρ leads to faster decay and weaker temporal structure, while ρ close to 1 produces more persistent latent dynamics.

H. Details on Explicit Latent Filters in Long-Cycle Regimes

Both components are essential To isolate which components of the input-embedding filter block are load-bearing, we compare three conditions:

1. the full filter, which applies the learned linear projection W_{in} followed by ℓ^2 normalization;

2. an ℓ^2-only baseline, which normalizes the raw return window directly and bypasses the learned projection;

3. and a linear-only baseline, which uses the projected representation without normalization.

As reported in Table 4, only the full composition yields high decodability of $\boldsymbol{h}_{t+1}$, while both ablated variants remain near zero. This shows that the learned projection and the normalization step are jointly necessary: neither one alone is sufficient to make the latent state linearly decodable.

Table 4. Ridge R^2 for probing h_{t+1} from transformer representations under three ablation conditions (median [IQR], $n = 10$ seeds).

Condition	$\omega = 365$	$\omega = 730$
Linear $\rightarrow$ ℓ^2 norm $\rightarrow$ Linear	0.890 [0.866, 0.903]	0.921 [0.914, 0.933]
ℓ^2 norm on raw returns $\rightarrow$ Linear	-0.004 [-0.005, -0.002]	-0.003 [-0.004, -0.002]
Linear embedding only $\rightarrow$ Linear	-0.006 [-0.009, -0.003]	-0.005 [-0.015, -0.000]

Causal perturbation of the post-norm representation. We next test whether the post-ℓ^2-norm embedding is causally upstream of the forecast. Following causal intervention methods from mechanistic interpretability, in the spirit of activation patching, causal tracing and concept-erasure (Meng et al., 2022; Heimersheim & Nanda, 2024; Belrose et al., 2023), we remove a varying fraction α of the component aligned with the latent-state direction at a chosen stage and then propagate the perturbed activation through the remainder of the network unchanged. We apply this intervention to the post-input-embedding, post-ℓ^2-norm, post-positional-embedding, post-attention, and post-FFN representations, and measure both the downstream change in latent-state decodability (Section 3.2) and the change in forecast error (Figure 6). For $\omega = 365$, perturbing the post-ℓ^2-norm representation produces the largest downstream collapse in decodability and the strongest increase in forecast MSE and MAE, whereas perturbations at later stages have substantially weaker effects. This identifies the ℓ^2-normalized embedding as the earliest stage at which the latent-state signal becomes causally relevant for the model's prediction.

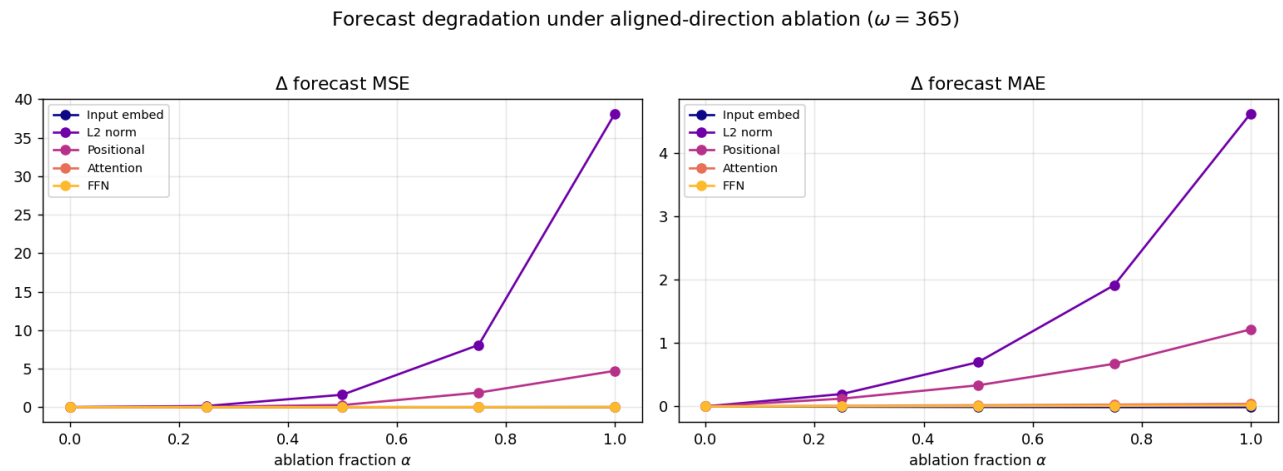


Figure 6. Forecast error change (ΔMSE and ΔMAE, ablated minus baseline) as a function of ablation strength α at ω=365, one curve per intervention stage (mean $\pm$ std, n=10 seeds). Intervening at after ℓ^2-norm causes the largest degradation; later stages other than positional embedding have negligible effect.

I. Related Work

Probing and causal interventions. Our stage-wise analysis builds on linear probing methods, which test what information is decodable from intermediate representations (Alain & Bengio, 2016; Hewitt & Liang, 2019; Belinkov, 2022). Because probe decodability alone does not establish that the model uses the decoded information, we complement probes with activation-level interventions, in the spirit of causal mediation, activation patching, causal scrubbing, and causal-abstraction approaches (Vig et al., 2020; Chan et al., 2022; Geiger et al., 2024; 2025; Zhang & Nanda, 2023). In contrast to many concept-level analyses, our probe target is the known latent state of the data-generating process.

Stochastic volatility and latent-state filtering. Stochastic-volatility models represent volatility as an unobserved stochastic state that must be inferred from noisy returns (Taylor, 1986; Shephard, 1996; Harvey et al., 1994; Kim et al., 1998; Jacquier et al., 2002b). Since returns depend nonlinearly on latent log-volatility, filtering and likelihood evaluation generally require approximation, simulation, or Bayesian methods, rather than closed-form Kalman filtering (Kitagawa, 1996; Durbin & Koopman, 2012). We use this classical latent-state setting not to introduce a new estimator, but as a controlled benchmark for testing whether neural forecasters internally recover information about the latent volatility state.

Transformers for time series and financial forecasting. A growing literature applies Transformer architectures to time-series forecasting, motivated by their ability to capture long-range temporal dependence and interactions across variables through self-attention (Wen et al., 2022; Zhao et al., 2026; Lim et al., 2021; Zhou et al., 2021). In finance, Transformers have been used for return prediction, market-risk forecasting, and volatility estimation (Yang et al., 2020; Zeng et al., 2023; Li et al., 2024). These applications are natural in multivariate financial settings, where asset returns and volatilities interact through common factors, market-wide shocks, and spillover effects (Fama & French, 1993; Andersen et al., 2003; Barigozzi et al., 2019). Most existing work, however, emphasizes predictive performance. Our focus is instead mechanistic: we ask whether a Transformer trained for volatility forecasting internally recovers the latent volatility state and where this computation emerges in the architecture.

Domain Knowledge Discovery Via Mechanistic Interpretability A growing line of work studies mechanistic interpretability not only as a tool for explaining model behavior, but also as a route to domain knowledge discovery. In enumerative geometry, Hashemi et al. (2025) train a Transformer to compute ψ-class intersection numbers and report interpretability evidence that the model implicitly captures structural objects. In combinatorics, Huang et al. (2025) propose a workflow in which a Transformer trained on paired Dyck paths is interpreted to derive a new algorithmic description of the well known zeta map. Our setting is aligned with this perspective, but focuses on latent-state recovery in stochastic dynamical systems rather than symbolic or combinatorial structure.

J. Limitations And Broader Implications

Limitations. First, all experiments are conducted in synthetic MSV models with known latent states. This makes controlled mechanistic analysis possible, but leaves open whether the same representational structure and stage-wise

emergence persist in real financial time series, which may include heavier tails, regime shifts, and model misspecification. Second, our evidence is still based primarily on linear probes, stage-wise readouts, and targeted aligned-direction interventions. These results establish latent-state recoverability and partial mechanism identification, but not a complete circuit-level account of how the computation is implemented. Third, the architectures studied here are intentionally small. While this supports interpretability, it remains unclear how well the same mechanisms scale to larger forecasting models. Finally, because the MLP baselines also recover substantial latent information in some regimes, our strongest Transformer-specific claim is not latent-state recovery itself, but rather the more localized stage-wise emergence and the explicit long-cycle filter structure.

Broader Implications. More broadly, our results suggest that financial time series may provide a useful benchmark for mechanistic interpretability under noisy observations and latent stochastic dynamics. Unlike many synthetic mechanistic tasks, stochastic-volatility models contain irreducible uncertainty and partial observability, forcing models to perform implicit filtering rather than exact algorithmic computation. Understanding how neural sequence models internally represent and manipulate latent-state information in such settings may provide a useful bridge between mechanistic interpretability, time-series modeling, and state-space inference.