TRUNCATE TO ACCELERATE: FROM FORECAST RECOVERABILITY TO EFFICIENT TIME-SERIES FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

In large language models, intermediate representations have been used both to study how task-relevant information evolves with depth and to accelerate inference. In time-series foundation models (TSFMs), however, these questions remain largely unexplored. We address this gap through layerwise probing, representation geometry, and lightweight adaptation across three state-of-the-art TSFMs: Chronos-2, TimesFM-3, and TiRex, evaluated on a collection of datasets with diverse sampling frequencies. First, layerwise forecasting readouts reveal a forecasting tunnel: across nearly all model-dataset pairs, intermediate depths already recover performance within 5% of the final-block readout. Next, we find that functional saturation coexists with continued geometric evolution. Finally, intermediate states can contain recoverable forecast information, but feeding them directly into the model's native forecasting pathway degrades performance relative to the full model. Lightweight affine alignment substantially reduces this gap and exposes a measured performance-compute frontier: aligned Chronos-2 remains within a median 3.1% of the full model after retaining only 4 of its 12 blocks, while running $2.43\times$ faster, with the achievable tradeoff varying substantially across architectures. Moreover, cuts inside the forecasting tunnel are typically better tolerated at matched depth, linking early recoverability to practical truncation. Together, these results suggest that much of the forecast-relevant information becomes recoverable before the final block and offer a path toward both understanding and accelerating time-series foundation models.

1 INTRODUCTION

Time-series foundation models (TSFMs) have recently achieved strong zero-shot forecasting performance across a broad range of domains (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Ansari et al., 2025; Auer et al., 2025). This capability increasingly relies on deep and computationally expensive backbones. At inference time, these models typically execute their full sequence of representation-transforming stages before producing a forecast. It is not clear, however, that every one of these stages still improves forecasting performance; identifying where performance saturates could therefore provide a direct route to faster inference.

In large language models, substantial depth can be removed with limited degradation (Fan et al., 2019; Gromov et al., 2025; Men et al., 2024). For TSFMs, Wiliński et al. (2025) locate removable blocks through similarity between intermediate representations. Representational similarity, however, need not coincide with forecasting redundancy. We therefore ask the functional question directly: *at what depth does near-final forecasting performance become recoverable, and can that early recoverability be converted into actual truncation?*

We study this in three state-of-the-art TSFMs with distinct architectures: the encoder-only transformer Chronos-2 (Ansari et al., 2025), the decoder-only transformer TimesFM-3 (Das et al., 2024), and the xLSTM-based TiRex (Auer et al., 2025), across 14 datasets spanning five sampling frequencies. The analysis proceeds in four steps. First, we introduce the *sustained forecasting tunnel*, a task-level criterion for locating the depth at which layerwise forecasting readouts reach near-final performance. We then compare this functional notion of depth with representation geometry, using

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

debiased linear Centered Kernel Alignment (CKA) (Song et al., 2012; Kornblith et al., 2019; Murphy et al., 2024; Cortes et al., 2024) and effective rank (Roy & Vetterli, 2007). Next, we test whether representations inside the tunnel are compatible with the model's pretrained forecasting pathway. Finally, we physically truncate the models and measure the resulting performance-compute frontier to determine how early recoverability translates into inference savings.

Our contributions are:

1. **A functional measure of forecasting depth.** We introduce a sustained forecasting tunnel for TSFMs, which identifies the earliest depth after which all subsequent forecasting readouts remain near the final-block readout. At our primary 5% tolerance, the tunnel begins before the final block in 40 of 42 model-dataset pairs.
2. **Recoverability versus geometry.** We show that the onset of the forecasting tunnel is not marked by CKA similarity regimes or effective-rank changes, indicating that geometric redundancy and forecasting redundancy provide different criteria for depth reduction.
3. **From recoverability to native-pathway compatibility.** We show that intermediate representations can support near-final forecasting readouts while remaining poorly matched to the model's pretrained forecasting pathway. A label-free affine alignment substantially improves this compatibility across all three architectures.
4. **From recoverability to acceleration.** We physically truncate the models and measure the resulting performance-compute frontier. At matched cut depths, tunnel membership consistently stratifies truncation tolerance, linking the recoverability diagnostic to actual depth reduction. Aligned Chronos-2 remains within a median 3.1% of the full model while retaining only 4 of 12 blocks and running $2.43\times$ faster; TiRex reaches a $1.71\times$ speedup with an 8.5% median MASE increase while retaining 7 of 12 blocks, while TimesFM-3 exhibits more modest gains.

The remainder of the paper reviews related work (Section 2), introduces our methods and hypotheses (Section 3), presents the results (Section 4), and concludes with limitations (Section 5); experimental details and additional results are provided in Appendices A and B, respectively.

2 Related Work

Time-series foundation models. TSFMs have expanded rapidly in scale and scope, targeting zero-shot, general-purpose forecasting across heterogeneous domains (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Ansari et al., 2025; Auer et al., 2025).

Layerwise analysis and probing. Linear probes measure how readily task-relevant information can be decoded from representations at different depths (Alain & Bengio, 2018). Intermediate features can carry information that the final representation does not (Evci et al., 2022), and in supervised image classifiers linear-probe performance can saturate before the final layer. This is dubbed the "tunnel effect" (Masarczyk et al., 2023). In language models, the logit lens (nostalgebraist, 2020) applies the pretrained output head directly to intermediate states, and the tuned lens (Belrose et al., 2025) first passes them through a learned affine map. Our forecasting hard cut parallels the logit lens, while our label-free affine adapter has a close analogue in the tuned lens for language models. Representation geometry gives a complementary view: CKA measures similarity between layer representations (Kornblith et al., 2019; Cortes et al., 2024), and effective rank characterizes their dimensional structure (Garrido et al., 2023; Skean et al., 2025). Murphy et al. (2024) show that standard CKA is inflated when samples are few relative to the representation dimension, even for independent random representations, so we use debiased CKA (Song et al., 2012).

Efficient inference and depth reduction. In language models, depth can be removed or bypassed with limited degradation through structured layer dropping (Fan et al., 2019), layer pruning (Men et al., 2024; Gromov et al., 2025), and early exiting (Teerapittayanon et al., 2017; Schwartz et al., 2020; Elhoushi et al., 2024). These complement broader compression methods such as quantization (Frantar et al., 2023) and knowledge distillation (Hinton et al., 2015; Sanh et al., 2020). For time series, Feofanov et al. (2026) show that simple classifiers trained on intermediate TSFM representations can outperform those trained on the final-layer representation. Wiliński et al. (2025) identify block-like redundancy in TSFMs with CKA and use it to guide pruning. We instead characterize removable depth directly through forecasting performance.

3 METHOD

We first introduce the notation and four hypotheses that structure our experiments (Section 3.1), followed by the layerwise readouts used to measure forecast recoverability (Section 3.2). We then introduce the hard cut and alignment procedures used to test native-pathway compatibility and acceleration (Section 3.3).

3.1 SETUP AND HYPOTHESES

A forecasting model maps its input through a sequence of L blocks. Let $h^{(0)}$ denote the representation entering the first block, and $h^{(\ell)}$ the output of block ℓ:

$$h^{(\ell)} = F_\ell\Big(h^{(\ell-1)}\Big), \qquad \ell = 1, \dots, L, \tag{1}$$

where F_ℓ is the transformation implemented by block ℓ (a transformer block in Chronos-2 and TimesFM-3, and an sLSTM block in TiRex). The model then produces its forecast through its *native forecasting pathway*:

$$\hat{y} = g\Big(h^{(L)}\Big), \tag{2}$$

where g comprises any post-block normalization, the pretrained forecasting head, and the model's inverse target transformation. We index depth only by block outputs, so any final normalization is included in g rather than treated as a separate depth.

Recoverability. At each depth ℓ, we fit a lightweight readout q_ℓ on the training split and measure its validation loss:

$$\mathcal{R}_\ell^{\mathrm{val}} = \frac{1}{N_{\mathrm{val}}} \sum_{i=1}^{N_{\mathrm{val}}} \mathcal{L}\Big(q_\ell\Big(h_i^{(\ell)}\Big), y_i\Big), \tag{3}$$

where N_{val} is the number of validation examples, $h_i^{(\ell)}$ the depth-ℓ representation of validation example i, y_i its target, and $\mathcal{L}$ the per-example forecasting loss. Simple readouts can saturate early in deep image classifiers, a phenomenon known as the tunnel effect (Masarczyk et al., 2023); we ask whether the same occurs in time-series forecasting. Rather than taking the first depth that reaches near-final performance, we define a stricter *sustained forecasting tunnel*, beginning at the earliest depth after which every subsequent readout remains within a tolerance ϵ of the final-block readout:

$$\ell_{\mathrm{tunnel}}(\epsilon) = \min\Big\{\ell \in \{0, \dots, L\} : \mathcal{R}_j^{\mathrm{val}} \le (1+\epsilon)\,\mathcal{R}_L^{\mathrm{val}} \;\; \forall j \in \{\ell, \dots, L\}\Big\}. \tag{4}$$

This sustained criterion prevents an isolated early dip from opening a tunnel. We use $\epsilon = 0.05$ as the primary criterion and report $\epsilon \in \{0.02, 0.08\}$ as sensitivity analyses. The reference is the *final-block readout*, not the native forecast. Thus, ℓ_{tunnel} marks where later blocks cease to substantially improve forecasting performance recoverable by the readout; it does not indicate where a truncated model would match the full model.

Representation geometry. If forecasting performance saturates at ℓ_{tunnel}, does representation geometry saturate with it? We track debiased linear CKA between depths (Song et al., 2012; Kornblith et al., 2019; Murphy et al., 2024; Cortes et al., 2024), which measures similarity between two representations, and entropy effective rank (Roy & Vetterli, 2007), which characterizes how the representation space expands or contracts with depth. If geometry stabilizes at the tunnel entrance, functional and geometric saturation coincide; if it continues to evolve, representation refinement persists after forecasting performance has saturated.

Native-pathway compatibility. A layerwise readout establishes only recoverability, while direct truncation requires something stronger: the model's own pathway g, trained on $h^{(L)}$, must also operate effectively on $h^{(\ell)}$. Even when $\mathcal{R}_\ell^{\mathrm{val}} \approx \mathcal{R}_L^{\mathrm{val}}$, directly applying g to $h^{(\ell)}$ can fail because $h^{(\ell)}$ may contain recoverable forecast information while remaining mismatched to the representation expected by g. We therefore insert a lightweight alignment map before the pathway:

$$\hat{y}_\ell = g\Big(A_\ell\Big(h^{(\ell)}\Big)\Big), \tag{5}$$

where A_ℓ is fit to reproduce the full model's forecast (Section 3.3). The degradation from directly applying g to $h^{(\ell)}$, relative to the full model, measures the compatibility gap, while the improvement from A_ℓ measures how much of that gap can be corrected by a lightweight transformation.

From compatibility to acceleration. If $g\big(A_\ell(h^{(\ell)})\big)$ approaches the full model's forecasting performance, blocks $\ell+1,\dots,L$ can be skipped at inference time. Sweeping ℓ then traces a performance-compute frontier.

We test four hypotheses:

(H1) Early forecast recoverability. The forecasting tunnel can begin well before the final block, with its entrance varying across models and datasets.

(H2) Geometry keeps evolving inside the tunnel. CKA and effective rank continue to change after ℓ_{tunnel} without commensurate gains in forecasting performance accessible to the readout.

(H3) Recoverability precedes native-pathway compatibility in TSFMs. At depths where near-final forecasting performance is already recoverable, directly applying g to $h^{(\ell)}$ degrades performance relative to the full model, while lightweight alignment reduces much of this gap.

(H4) Truncate to accelerate. Aligned truncation exposes a performance-compute frontier in which truncation depth trades forecasting performance against inference cost.

3.2 Layerwise forecasting readouts

We probe each model at the token positions read by its native forecasting head. Let H be the horizon, d the hidden width, P the output patch length, and $Q=9$ the number of quantile levels $\{0.1,\dots,0.9\}$ used across all three models. Chronos-2 forecasts from $K=\lceil H/P\rceil$ forecast-slot tokens, while TiRex forecasts from the last input token of each of its K inference passes. For both, a shared linear readout $q_\ell:\mathbb{R}^d\to\mathbb{R}^{Q\times P}$ is applied to each forecast-producing state, and the K predicted patches are concatenated. For TimesFM-3, a linear readout $q_\ell:\mathbb{R}^d\to\mathbb{R}^{Q\times H}$ predicts the full horizon from the model's last context token. Readouts are trained with the pinball loss in each model's normalized target space, with weight decay selected per depth on validation (Appendix A.6). Because target spaces differ, absolute losses are compared only within a model.

3.3 Hard cut and native-pathway alignment

At depth $\ell<L$, the *hard cut* feeds the intermediate representation directly into the pretrained forecasting pathway, $g(h^{(\ell)})$. To test whether a simple representation mismatch explains the resulting degradation, we insert *a label-free affine adapter*, $A_\ell(h)=h+\Delta_\ell h+b_\ell$, initialized at the identity and fitted with the backbone frozen to minimize the squared error between $g(A_\ell(h^{(\ell)}))$ and the full model's own forecast $g(h^{(L)})$. We additionally retain the linear readout q_ℓ as a supervised reference for how much forecast information is recoverable from the same representation. A nonlinear adapter variant is reported in Appendix A.7.

4 Results

We first locate the forecasting tunnel (Section 4.1, H1) and compare it with representation geometry (Section 4.2, H2). We then test whether the pretrained pathway can use early states (Section 4.3, H3) and whether aligned truncation accelerates inference (Section 4.4, H4).

Experimental setup. We evaluate Chronos-2, TimesFM-3, and TiRex on 14 datasets spanning five sampling frequencies, from 5-minute to daily: M4 (Makridakis et al., 2020), Electricity (Trindade, 2015; Godahewa et al., 2021), Uber TLC (FiveThirtyEight, 2015; Godahewa et al., 2021), Wind Farms (Godahewa et al., 2021), BOOM (Cohen et al., 2025), SG Carpark (Qiao et al., 2026), Coastal T-S (Qiao et al., 2026), LOOP Seattle (Cui et al., 2018; 2019), Electricity (15-minute) (Trindade, 2015; Lai et al., 2018; Godahewa et al., 2021), SZ Taxi (Jiang et al., 2024), London Smart Meters (D., 2019; Godahewa et al., 2021), M5 (Howard et al., 2020), Wiki Daily (Ansari et al., 2024), and Monash Traffic (Lai et al., 2018; Godahewa et al., 2021). Context length and horizon are fixed at

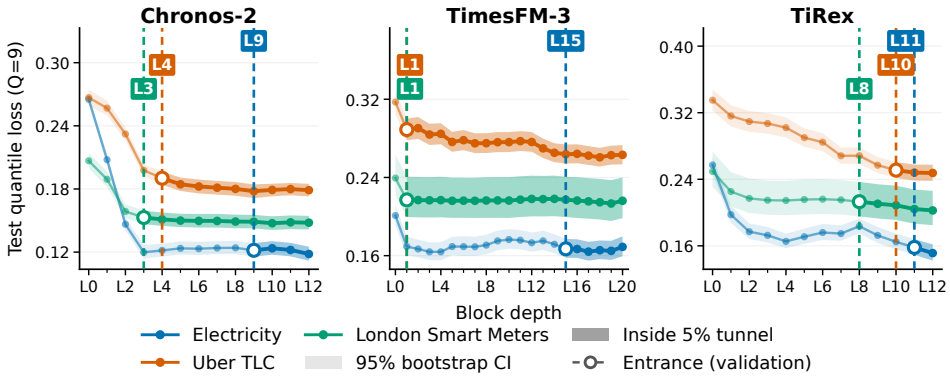


Figure 1: **Layerwise Q=9 forecasting loss for Electricity, Uber TLC, and London Smart Meters.** Test loss is shown across depth for each architecture. The shaded region begins at the sustained 5% tunnel entrance, selected using validation loss only; test loss does not affect selection and is shown to assess whether the validation-selected tunnel persists on held-out data. Bands show 95% series-level cluster-bootstrap intervals ($B = 5000$). Complete results are in Appendix B.1.1.

Table 1: **Sustained 5% forecasting-tunnel entrances for the Q=9 readout.** L0 denotes the representation before the first model block, while L1, L2, and so on denote the outputs of successive blocks. Entrances are selected from validation loss only.

Dataset	Chronos-2	TimesFM-3	TiRex	Dataset	Chronos-2	TimesFM-3	TiRex
M4	L8	L16	L12	LOOP Seattle	L9	L14	L9
Electricity	L9	L15	L11	Electricity (15-min)	L10	L1	L10
Uber TLC	L4	L1	L10	SZ Taxi	L2	L1	L0
Wind Farms	L3	L0	L11	London Smart Meters	L3	L1	L8
SG Carpark	L10	L12	L11	M5	L0	L0	L0
Coastal T-S	L1	L5	L1	Wiki Daily	L2	L0	L1
BOOM	L3	L0	L8	Monash Traffic	L11	L12	L12
Median depth	**29%**	**5%**	**79%**	**Before final**	**14/14**	**14/14**	**12/14**

$C = 512$ and $H = 64$. All three models receive identical windows and splits, and all pretrained backbones remain frozen. Selection uses validation data only, and the test split is evaluated once. We report MASE using each dataset's seasonal period, with 95% confidence intervals from 5,000 series-level cluster-bootstrap resamples, paired whenever two pathways are compared. We time inference end to end on an A100 over three independent repetitions, using batch size 1 for the main results and batch size 256 for the appendix throughput analysis (Appendix B.4). Full model configurations, dataset descriptions, split construction, optimization details, and each dataset's documented pretraining status for each model are in Appendix A.

4.1 Forecasting before the final block

We first ask *whether near-final forecasting performance becomes recoverable from intermediate representations*. For convenience, we denote the representation before the first model block as L0 and the outputs of successive blocks as L1, L2, and so on, with the final block at L12 for Chronos-2 and TiRex and L20 for TimesFM-3.

Figure 1 shows the layerwise $Q = 9$ mean quantile loss, while Table 1 summarizes the sustained 5% tunnel entrance across all datasets. Among the 42 model-dataset pairs, 40 enter the tunnel before the final block: 14/14 for Chronos-2, 14/14 for TimesFM-3, and 12/14 for TiRex; the two exceptions are TiRex on M4 and Monash Traffic. The depth at which the tunnel begins varies across models and datasets.

The validation-selected tunnel entrances also remain close to final-block performance on held-out test data. Across all 42 model-dataset pairs, the median test-loss increase at the selected entrance

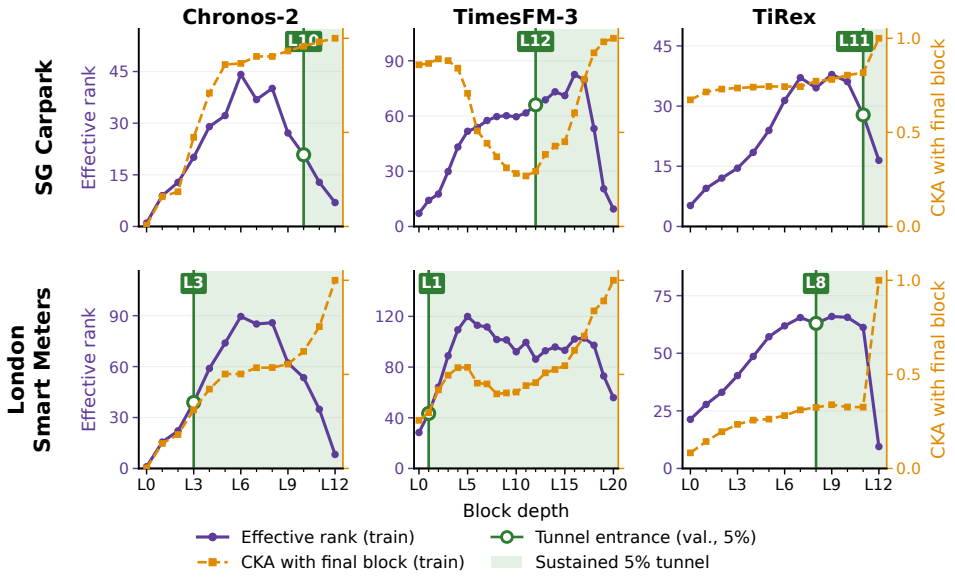


Figure 2: **Representation geometry and forecast recoverability.** Effective rank and debiased linear CKA between each depth and the final block, computed on training representations, for SG Carpark (top) and London Smart Meters (bottom). Columns correspond to Chronos-2, TimesFM-3, and TiRex. Shading marks the validation-selected 5% forecasting tunnel.

is 1.3% (interquartile range (IQR): -0.7% to 4.3%), with a maximum increase of 9.8%. Full held-out results are reported in Appendix B.1.2. We additionally repeat the analysis with a substantially lower-capacity, median-only ($Q = 1$) readout and obtain highly consistent tunnel entrances relative to the $Q = 9$ analysis (Appendix B.1.3).

Overall, *near-final forecasting performance can be recovered before the final block*, but how early this occurs depends on the architecture, dataset, and tolerated degradation; with $\epsilon = 0.08$, the tunnel begins before the final block in all model-dataset pairs, whereas under the stricter $\epsilon = 0.02$ criterion this holds for 36/42 pairs (Appendix B.1.4). We use the 5% criterion for our main analysis and next ask *whether representation geometry also saturates once near-final forecasting performance becomes recoverable*.

4.2 Geometry and Forecast Recoverability

Figure 2 shows that the forecasting tunnel does not consistently align with changes in representation geometry. On SG Carpark, the Chronos-2 tunnel begins several blocks after both the effective-rank peak and a major CKA transition; for TimesFM-3, effective rank continues to increase after the tunnel entrance; and for TiRex, substantial CKA evolution continues after the tunnel has already begun. Similarly, for London Smart Meters, the tunnel begins at L3, L1, and L8 for Chronos-2, TimesFM-3, and TiRex, respectively, while effective rank and similarity to the final representation continue to evolve at later depths. These patterns persist under the stricter 2% tolerance (Appendix B.2).

Taken together, these results show that *forecasting performance can saturate while geometry keeps evolving*. Moreover, geometry alone neither locates the tunnel nor defines a unique truncation depth: the effective-rank peak identifies a single layer with no direct link to forecast quality, while CKA can reveal several blocks of mutually similar layers, each providing a possible cut point (see Appendix B.2 for Chronos-2). It does, however, flag a potential challenge for truncation: at the tunnel entrance, CKA with the final block is below 0.5 in 55% of model-dataset pairs, so *the pretrained head may receive a substantially different representation*.

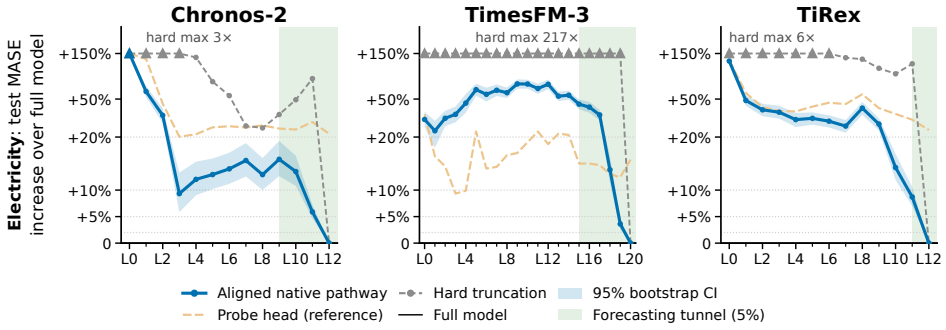


Figure 3: **Native-pathway compatibility across depth on Electricity.** Test MASE increase over the full model for the hard cut, affine-aligned pathway, and supervised probe reference. Shading marks the forecasting tunnel; bands show 95% cluster-bootstrap intervals. The y-axis is linear to $+20\%$ and log-scaled above; triangles mark clipped values. Complete results are in Appendix B.3.

Table 2: **Native-pathway compatibility across depth.** Median test MASE increase over the full model across 14 datasets at 25%, 50%, and 75% of network depth, corresponding to L3/L6/L9 for Chronos-2 and TiRex and L5/L10/L15 for TimesFM-3. $\times$ denotes ratios to the full-model MASE.

	Hard cut			Aligned pathway			Probe reference		
Model	25%	50%	75%	25%	50%	75%	25%	50%	75%
Chronos-2	+44.2%	+20.1%	+17.1%	+4.4%	+2.3%	+2.2%	+13.9%	+11.8%	+11.6%
TimesFM-3	53×	121×	160×	+69.8%	+67.3%	+47.9%	+16.8%	+17.0%	+14.8%
TiRex	+121.3%	+104.7%	+80.0%	+13.4%	+10.6%	+6.2%	+14.5%	+13.2%	+12.8%

4.3 Native-pathway compatibility across depth

Using the hard cut and label-free affine adapter introduced in Section 3.3, Figure 3 shows a clear separation between forecast recoverability and native-pathway compatibility. On Electricity, the hard cut never stays within 5% of the full model before the final block, while the affine adapter closes a median 73%, over 99%, and 82% of the hard-cut gap across intermediate depths for Chronos-2, TimesFM-3, and TiRex, respectively; the particularly large value for TimesFM-3 reflects its extremely poor hard-cut baseline. For TimesFM-3, the same supervised readout q_ℓ used in Section 4.1 remains substantially closer to full-model performance than the aligned pathway on Electricity, and this distinction persists across datasets; for Chronos-2 and TiRex, however, the aligned pathway outperforms both the hard cut and the supervised probe at every depth reported in Table 2. A nonlinear bottleneck adapter changes little relative to the affine adapter for Chronos-2 and TiRex, with a median difference of only -0.2 percentage points in test MASE, whereas it performs substantially worse for TimesFM-3 (Appendix B.3).

We conclude that *recoverability precedes native-pathway compatibility in TSFMs*, and that lightweight alignment substantially improves compatibility across all three architectures. The extent of recovery is architecture-dependent: for Chronos-2 and TiRex, a depth-specific affine map can bring intermediate states close to native performance several blocks before the final one, while for TimesFM-3 it removes most of the large hard-cut degradation but leaves a substantial residual gap to the full model.

4.4 Truncate to accelerate

The forecasting tunnel establishes that near-final forecast information often becomes recoverable well before the final block, while Section 4.3 shows that alignment can make such intermediate states substantially more compatible with the native pathway. We now ask whether this early recoverability can be converted into actual inference savings.

Table 3: **Operating points of aligned truncation.** Median test MASE increase across 14 datasets and end-to-end speedup at batch size 1. For TimesFM-3 at L19, the adapter overhead slightly exceeds the saving from removing one block. Full layer-by-layer results are reported in Appendix B.4.

Model	Layer	ΔMASE	Speedup	Layer	ΔMASE	Speedup	Layer	ΔMASE	Speedup
Chronos-2	L9	+2.2%	1.28×	L6	+2.3%	1.78×	L4	+3.1%	2.43×
TimesFM-3	L19	+5.4%	0.99×	L10	+67.3%	1.19×	L0	+31.3%	1.38×
TiRex	L11	+1.8%	1.09×	L7	+8.5%	1.71×	L3	+13.4%	3.93×

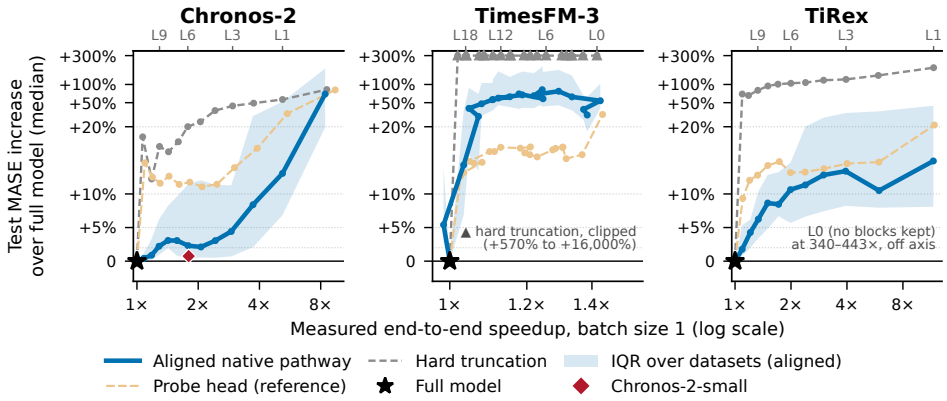


Figure 4: **Performance-compute frontier of truncation.** Median test MASE increase across 14 datasets versus end-to-end speedup at batch size 1. Points denote aligned cut depths; the star is the full model and the diamond is Chronos-2-small; bands show interquartile ranges.

How much does it cost to remove blocks? Figure 4 shows the resulting *performance-compute frontier*, obtained by physically truncating each model, verifying agreement with the offline aligned pathway in mean test MASE to within 10^{-4} relative error (Appendix A.9), and measuring end-to-end speedup over the full model. Aligned Chronos-2 remains within a median 3.1% of the full model from L11 through L4, where it runs $2.43\times$ faster. TiRex exhibits a more gradual tradeoff, reaching $1.71\times$ speedup at an 8.5% median MASE increase when truncated after L7. TimesFM-3 provides more modest gains at batch size 1: even removing all blocks reaches only $1.38\times$, indicating that a substantial fraction of its end-to-end runtime lies outside the Transformer blocks (Table 3). Chronos-2-small (Ansari et al., 2025) provides a strong separately pretrained baseline, running $1.80\times$ faster at only $+0.8\%$ median MASE; at a comparable speed, aligned truncation after L6 reaches $1.78\times$ with a $+2.3\%$ increase, while avoiding the need to train a separate smaller model and providing multiple speed-performance tradeoffs from a single pretrained Chronos-2 checkpoint. Complete layer-by-layer performance and speedup results are reported in Appendix B.4.

Does the forecasting tunnel identify where truncation is well tolerated? Because inference speed depends only on the cut depth, we compare datasets at the same cut according to whether that depth lies inside or before their validation-selected tunnel. Despite being defined only from layerwise readouts, without evaluating a truncated model, the tunnel consistently stratifies truncation performance. After L4, aligned Chronos-2 incurs a median $+2.2\%$ MASE increase on datasets for which L4 lies inside the tunnel, compared with $+14.2\%$ when it lies before the tunnel; after L6, the corresponding TiRex increases are $+4.4\%$ and $+12.3\%$. Truncation is cheaper inside the tunnel at all depths where both groups are present for Chronos-2 and TiRex, and at 14 of 15 such depths for TimesFM-3 (Table 14). For TimesFM-3, however, substantial degradation remains even inside the tunnel, indicating that early recoverability alone does not ensure native-pathway compatibility. Thus, the forecasting tunnel identifies datasets that better tolerate a given cut, even though its entrance need not define the best truncation depth.

5 Conclusion and Limitations

Together, these results *complete the progression from layerwise analysis to acceleration*. Rather than relying on geometric similarity alone to identify removable depth in time-series foundation models, we introduce the forecasting tunnel, a direct task-level criterion based on forecasting performance and a single tolerance parameter. It shows that near-final forecast information can become recoverable well before the final block even while representation geometry continues to evolve. We then ask whether these early representations can be reused by the model's pretrained forecasting pathway: direct reuse often fails, but lightweight alignment substantially improves compatibility. Finally, physical truncation converts this compatibility into a measured performance-compute frontier, *allowing different speed-performance operating points from a single pretrained checkpoint*.

Across three state-of-the-art TSFMs and 14 datasets spanning five sampling frequencies, the forecasting tunnel begins before the final block in 40 of 42 model-dataset pairs at our primary 5% tolerance, and remains pre-final in 36 of 42 pairs even under the stricter 2% tolerance. Representation geometry nevertheless continues to evolve inside the tunnel, showing that forecasting saturation need not coincide with geometric stabilization. Direct reuse of intermediate states through the pretrained forecasting pathway rarely succeeds, whereas lightweight affine alignment substantially improves compatibility across all three architectures. Physical truncation then turns this improved compatibility into measurable inference savings: Chronos-2 remains within a median 3.1% of the full model after retaining only 4 of its 12 blocks while running $2.43\times$ faster, and TiRex reaches $1.71\times$ speedup with an 8.5% median MASE increase while retaining 7 of 12 blocks. Importantly, at matched cut depths, datasets for which the cut lies inside the forecasting tunnel generally tolerate truncation better, directly linking early recoverability to practical depth reduction. TimesFM-3 provides the complementary case: substantial degradation can remain even inside the tunnel, showing that recoverability alone does not guarantee native-pathway compatibility. Its batch-size-1 speedups are also modest, with truncation before the first Transformer block reaching only $1.38\times$, consistent with a substantial fraction of end-to-end runtime lying outside the Transformer blocks. Chronos-2-small remains a strong separately pretrained baseline, while truncation provides multiple post-hoc operating points from a single pretrained checkpoint.

Our readouts and adapters are deliberately simple, making our estimates of recoverability and compatibility conservative with respect to more expressive mappings. For TimesFM-3, the affine adapter leaves a substantial gap to the full model, and richer alignment, for example over the full token sequence, may close more of it. Adapters are also fitted per dataset; whether alignment transfers across datasets, allowing a single truncated model at each depth, remains open. With 14 datasets, we can establish an association between forecast recoverability and truncation tolerance but not determine how reliably it generalizes or supports cut-depth selection. Larger benchmarks such as GIFT-Eval (Aksu et al., 2024) and fev-bench (Shchur et al., 2026) could test whether the forecasting tunnel can serve as a reliable indicator of truncation tolerance and guide cut-depth selection. Several datasets in our study are documented pretraining data for at least one model (Appendix A.3), so frozen-backbone experiments cannot fully separate generalizable structure from effects of pretraining exposure; extending the analysis to fev-bench, which is held out from the pretraining of Chronos-2 and TimesFM-3, would provide a cleaner test. Finally, we study univariate forecasting at a single context length and horizon, with speed measured on one GPU type; multivariate and covariate-aware forecasting, other horizons and context lengths, and additional hardware remain important directions for future work.

Together, these findings mark a step toward TSFMs that are both easier to understand internally and more efficient to deploy, by connecting where forecast information becomes recoverable to where computation can be removed.

AI use statement

In this work, we used generative AI tools for writing assistance, such as improving word choice, correcting grammar, and improving the flow and transitions between sentences. Generative AI tools also assisted in drafting the Appendix sections describing the latency protocol and compute accounting; the underlying experimental procedures, measurements, and reported values were determined and verified by the authors, and all generated text was reviewed and revised by the authors. We also used generative AI tools for coding, with their behavior constrained by author-written instruction

files. All AI-generated code was reviewed and debugged by the authors and validated through tests written both by the authors and with AI assistance. Additionally, generative AI tools were used to implement plotting and table-generation code for the paper, under the authors' review. Generative AI tools were also used to assist in writing scripts for collecting latency and compute information, such as device metadata, runtime records, and speedup measurements; all such code was reviewed by the authors.

We have not used generative AI tools to generate research ideas, conduct the literature review, design methods, select time-series foundation models or datasets, design experiments, or draft sections of the main paper. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

Ethics statement

This work uses three publicly released pretrained forecasting models and 14 public datasets and involves no human subjects. Most datasets contain aggregated counts, sensor measurements, or system metrics, including traffic, parking, ride demand, wind power, oceanographic, and cloud-observability data. The two electricity-consumption datasets contain anonymized readings from individual clients or households; we use them only in the form released by their providers and make no attempt at re-identification. Our methods analyze and truncate existing models rather than introduce new capabilities, and can reduce the computational cost of inference. Because truncation trades forecasting performance against speed, with the attainable tradeoff varying across models and datasets, truncated models should be validated on the target data before use in consequential decisions.

Reproducibility statement

All three models are public checkpoints, used frozen; the exact checkpoints and package versions are listed in Appendix A.1. All 14 datasets are publicly available, and Appendix A.2 gives their sources and every preprocessing step. The construction of windows and splits, including the fixed random seed used for random choices, is described in Appendix A.4; all three models are evaluated on identical windows, checked against a stored reference for each dataset. The target spaces, readout training, alignment procedure, and metrics are specified in Appendices A.5, A.6, A.7, and A.8, including all hyperparameter grids and selection rules.

The tunnel definition and all selection rules use validation data only and were fixed before the corresponding test results were computed. The comparison of truncation inside and before the tunnel (Section 4.4) is a post hoc analysis. A pre-specified per-window equivalence check between the offline aligned pathway and the physically truncated implementation was not satisfied at every operating point; however, mean test MASE agrees to within 10^{-4} relative error. Appendix A.9 reports this check together with the latency protocol. Appendix A.10 lists the hardware, software, and compute.

References

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Gift-eval: A benchmark for general time series forecasting model evaluation, 2024. URL https://arxiv.org/abs/2410.10393.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2018. URL https://arxiv.org/abs/1610.01644.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series, 2024. URL https://arxiv.org/abs/2403.07815.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami,

Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting, 2025. URL https://arxiv.org/abs/2510.15821.

Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. Tirex: Zero-shot forecasting across long and short horizons with enhanced in-context learning, 2025. URL https://arxiv.org/abs/2505.23719.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens, 2025. URL https://arxiv.org/abs/2303.08112.

Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean Ogier du Terrail, Anna-Monica Toon, Kan Wang, Stephan Xie, Zongzhe Xu, Viktoriya Zhukova, David Asker, Ameet Talwalkar, and Othmane Abou-Amal. This time is different: An observability perspective on time series foundation models, 2025. URL https://arxiv.org/abs/2505.14766.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment, 2024. URL https://arxiv.org/abs/1203.0550.

Zhiyong Cui, Ruimin Ke, and Yinhai Wang. Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. *arXiv preprint arXiv:1801.02143*, 2018.

Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

Jean-Michel D. Smart meters in London. Kaggle, https://www.kaggle.com/datasets/jeanmidev/smart-meters-in-london, 2019. Refactored from UK Power Networks' Low Carbon London smart meter data.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting, 2024. URL https://arxiv.org/abs/2310.10688.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean Wu. Layerskip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12622–12642. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.681. URL http://dx.doi.org/10.18653/v1/2024.acl-long.681.

Utku Evci, Vincent Dumoulin, Hugo Larochelle, and Michael C. Mozer. Head2toe: Utilizing intermediate representations for better transfer learning, 2022. URL https://arxiv.org/abs/2201.03529.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout, 2019. URL https://arxiv.org/abs/1909.11556.

Vasilii Feofanov, Songkang Wen, Jianfeng Zhang, Lujia Pan, and Ievgen Redko. Mantisv2: Closing the zero-shot gap in time series classification with synthetic data and test-time strategies, 2026. URL https://arxiv.org/abs/2602.17868.

FiveThirtyEight. Uber tlc foil response. https://github.com/fivethirtyeight/uber-tlc-foil-response, 2015. Uber trip data obtained from the New York City Taxi and Limousine Commission through a Freedom of Information Law request.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann Lecun. Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank, 2023. URL https://arxiv.org/abs/2210.02885.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive, 2021. URL https://arxiv.org/abs/2105.06643.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers, 2025. URL https://arxiv.org/abs/2403.17887.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Addison Howard, inversion, Spyros Makridakis, and vangelis. M5 forecasting - accuracy. https://www.kaggle.com/competitions/m5-forecasting-accuracy, 2020. Kaggle.

Rob J. Hyndman and Anne B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL https://www.sciencedirect.com/science/article/pii/S0169207006000239.

Jiawei Jiang, Chengkai Han, Wenjun Jiang, Wayne Xin Zhao, and Jingyuan Wang. Libcity: A unified library towards efficient and comprehensive urban spatial-temporal prediction, 2024. URL https://arxiv.org/abs/2304.14343.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited, 2019. URL https://arxiv.org/abs/1905.00414.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks, 2018. URL https://arxiv.org/abs/1703.07015.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.04.014. URL https://www.sciencedirect.com/science/article/pii/S0169207019301128. M4 Competition.

Wojciech Masarczyk, Mateusz Ostaszewski, Ehsan Imani, Razvan Pascanu, Piotr Miłoś, and Tomasz Trzciński. The tunnel effect: Building data representations in deep neural networks, 2023. URL https://arxiv.org/abs/2305.19753.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect, 2024. URL https://arxiv.org/abs/2403.03853.

Alex Murphy, Joel Zylberberg, and Alona Fyshe. Correcting biased centered kernel alignment measures in biological and artificial neural networks, 2024. URL https://arxiv.org/abs/2405.01012.

nostalgebraist. Interpreting GPT: the logit lens. LessWrong, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens. Blog post.

Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's time: Towards the next generation of time series forecasting benchmarks, 2026. URL https://arxiv.org/abs/2602.12147.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020. URL https://arxiv.org/abs/1910.01108.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. The right tool for the job: Matching model and instance complexities, 2020. URL https://arxiv.org/abs/2004.07453.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting, 2026. URL https://arxiv.org/abs/2509.26468.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models, 2025. URL https://arxiv.org/abs/2502.02013.

Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13(47):1393–1434, 2012. URL http://jmlr.org/papers/v13/song12a.html.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. Branchynet: Fast inference via early exiting from deep neural networks, 2017. URL https://arxiv.org/abs/1709.01686.

Artur Trindade. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015. DOI: https://doi.org/10.24432/C58C86.

Michał Wiliński, Mononito Goswami, Willa Potosnak, Nina Żukowska, and Artur Dubrawski. Exploring representations and interventions in time series foundation models, 2025. URL https://arxiv.org/abs/2409.12915.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers, 2024. URL https://arxiv.org/abs/2402.02592.

A Experimental Details

A.1 Models

All three models are used as released, with frozen float32 parameters, and run in univariate mode. Table 4 summarizes the architecture and readout positions of each model. For every model, depth is indexed as $\mathrm{L}0, \ldots, \mathrm{L}L$: $\mathrm{L}0$ denotes the representation entering the first block, and $\mathrm{L}\ell$ the output of block ℓ. Any normalization applied after the final block belongs to the native forecasting pathway g and does not define an additional depth. We still probe its output as a diagnostic, but exclude it from the forecasting-tunnel definition and the main CKA analysis. All models are evaluated at the same nine quantile levels $\{0.1, \ldots, 0.9\}$, verified against each model's native implementation.

Table 4: **Models and readout positions.** L: number of blocks. d: hidden width. K: forecast-producing states per window at $H = 64$, each covering P horizon steps.

	Chronos-2	TimesFM-3	TiRex
Checkpoint	`amazon/chronos-2`	`google/timesfm-3.0-pytorch`	`NX-AI/TiRex`
Package	`chronos-forecasting` 2.3.1	`timesfm` 3.0.2	`tirex-ts` 1.4.2
Backbone	encoder-only Transformer	decoder-only Transformer	xLSTM (sLSTM blocks)
L / d	12 / 768	20 / 1280	12 / 512
Input patch	16	32	32
Readout states	K=4 forecast-slot tokens	last context token (index 15)	last input token (index 63) of each of K=2 passes
P per state	16	64	32
L0	tokens entering block 1	output of the input residual block	output of the input patch embedding
Head input	T5-style RMS normalization of L12	L20	final RMSNorm of L12
Native head	residual MLP block	linear, $1280 \to 64 \cdot 9$	residual MLP block, $512 \to 2048 \to 32 \cdot 9$
Target space	standardize, then arcsinh	detrend, then RevIN	per-pass location/scale
Readout q_ℓ	$\mathbb{R}^{768} \to \mathbb{R}^{9\times 16}$, shared	$\mathbb{R}^{1280} \to \mathbb{R}^{9\times 64}$	$\mathbb{R}^{512} \to \mathbb{R}^{9\times 32}$, shared

Chronos-2. We run one encoder pass with $K = \lceil H/16 \rceil = 4$ forecast slots, the model's native layout for $H = 64$, and read the four slot states. Because encoder attention is bidirectional, the slot states depend on K, so all Chronos-2 features come from this $K = 4$ pass. The native head predicts 21 quantiles, and we use the nine that match the other two models. Its final normalization is a T5-style layer norm, i.e. an RMS normalization without mean subtraction or bias.

TimesFM-3. With $C = 512$ and input patch 32, the context occupies 16 tokens. The native head reads only the last one (index 15), and one output patch of 64 steps covers the whole horizon, so $K = 1$. Its forecast is decoded by inverting RevIN with that token's statistics, clipping, and adding back the context trend.

TiRex. TiRex left-pads every context with missing values to its training length of 2048, so $C = 512$ becomes 48 masked and 16 real patches. We use the package's default inference path, which produces $H = 64$ in two passes: the second pass appends one patch of missing values, rather than predicted values, to the context. Each pass forecasts 32 steps from token 63, with its own location and scale. We use the package's PyTorch backend throughout. Its sLSTM recurrence runs in bfloat16 with pointwise operations in float32, causing hidden states to vary slightly with GPU batch size; we therefore extract all features at a fixed batch size of 256. The second pass's L0 state is identical across windows because the appended patch is fully masked, so it carries no window-specific information.

Native-pathway check. For every model and dataset, applying the native pathway to the hooked final-block state reproduces the package's own forecast up to floating-point accumulation error.

A.2 DATASETS

Table 5 lists the 14 datasets. Every dataset is used univariately: a multivariate dataset contributes each selected *variate*, i.e. each channel, as a separate series. Several datasets come from three public collections: the Monash Time Series Forecasting Archive (Godahewa et al., 2021); the UCI Machine Learning Repository, a public dataset archive hosted by the University of California, Irvine; and TIME, a recent forecasting benchmark (Qiao et al., 2026). Series lengths are reported before windowing. London Smart Meters, M5, and Wiki Daily are additionally reduced to a fixed-seed random sample of 1,394 eligible series (marked † in Table 5; Appendix A.4).

Table 5: **Datasets.** m: seasonal period used by MASE. Series: number of series after dataset-specific construction or filtering but before the fixed-seed subsampling marked by †. Lengths are in time steps at the frequency used.

Dataset	Freq.	m	Series	Length (min / max)
M4	1H	24	414	748 / 1,008
Electricity	1H	24	321	26,304
Uber TLC	1H	24	262	4,344
Wind Farms	1H	24	337	1,715 / 8,784
BOOM	1H	24	356	664 / 13,159
SG Carpark	1H (from 15min)	24	354	≈3,623
Coastal T-S	1H	24	48 (24 stations)	2,733 / 8,784
LOOP Seattle	5min	288	323	105,120
Electricity (15-minute)	15min	96	370	16,032 / 140,256
SZ Taxi	15min	96	156	2,976
London Smart Meters	30min	48	5,560†	288 / 39,648
M5	1D	7	30,490†	124 / 1,969
Wiki Daily	1D	7	100,000†	2,741
Monash Traffic	1H	24	862	17,544

M4. The 414 hourly series of the M4 forecasting competition, anonymized and drawn from several domains (Makridakis et al., 2020).

Electricity. Hourly electricity load from 321 clients (2012 to 2014). It is derived from the UCI dataset (Trindade, 2015), which records 370 clients every 15 minutes. The Monash archive distributes an hourly aggregate of 321 of these clients, the version used by Lai et al. (2018) (Godahewa et al., 2021).

Uber TLC. Hourly Uber pickup counts across New York City locations, derived from data released by the New York City Taxi and Limousine Commission (FiveThirtyEight, 2015; Godahewa et al., 2021).

Wind Farms. Hourly power production from 337 Australian wind farms (Godahewa et al., 2021).

BOOM. Monitoring metrics from Datadog production systems (Cohen et al., 2025). We use the 378 queries recorded hourly and, for each query, select the first variate with at most 20% missing values and at least one complete window, retaining 356 queries.

SG Carpark. Available-lot counts from 354 public carparks in Singapore (2025), from TIME (Qiao et al., 2026). We aggregate the 15-minute data to hourly means, requiring at least three of the four samples in an hour and never filling missing values.

Coastal T-S. Hourly measurements from 24 moored instruments, i.e. sensors anchored at fixed points, along the Australian coast, from TIME (Qiao et al., 2026). We use temperature and salinity, giving 48 series. The temperature and salinity series from each station form one bootstrap cluster, leaving 24 independent clusters and producing the widest confidence intervals in the benchmark.

LOOP Seattle. Five-minute freeway speeds from 323 sensor stations in the Seattle area during 2015 (Cui et al., 2018; 2019).

Electricity (15-minute). The UCI dataset itself (Trindade, 2015): all 370 clients at the native 15-minute resolution (2011 to 2014) (Godahewa et al., 2021). It shares its source with Electricity and serves as a *frequency control*: it keeps the domain and most meters fixed while changing the sampling rate. With $C = 512$, its context spans 5.3 days, compared with 21 days for hourly Electricity.

SZ Taxi. Fifteen-minute road speeds on 156 roads in Shenzhen during January 2015, computed from taxi GPS trajectories. We use the version distributed by LibCity (Jiang et al., 2024).

London Smart Meters. Half-hourly electricity consumption from 5,560 London households (November 2011 to February 2014), collected by the electricity distributor UK Power Networks in its Low Carbon London smart-meter trial (D., 2019; Godahewa et al., 2021). We use the Monash version that retains missing values.

M5. Daily unit sales from 30,490 Walmart item-store series (2011 to 2016) from the M5 forecasting competition (Howard et al., 2020), including the competition's final evaluation period.

Wiki Daily. Daily page views from 100,000 English Wikipedia articles (2015 to 2022), compiled by Ansari et al. (2024).

Monash Traffic. Hourly road occupancy rates from 862 sensors on San Francisco Bay Area freeways (2015 to 2016). We use the version distributed by the Monash archive (Godahewa et al., 2021), which was also used by Lai et al. (2018).

A.3 Pretraining provenance

Because all three models were pretrained on large public corpora, some evaluation datasets may overlap their pretraining data. For every model-dataset pair, we record what the model's own documentation establishes using three categories:

- **Listed**: the dataset appears in the documented pretraining corpus.
- **Held out**: the documentation explicitly excludes the dataset, or a benchmark containing it, from the relevant real-data pretraining corpus.
- **Not listed**: the dataset appears in neither category. This means only that the available documentation does not identify it as training data; it does *not* establish that the model never saw it.

For each cell, we additionally distinguish evidence for the exact dataset from evidence covering only the same source at a different resolution or in a different form, marked *source*.

Sources. *Chronos-2* lists its real univariate pretraining datasets in Table 6 of Ansari et al. (2025). Its authors also state that no dataset or task from fev-bench (Shchur et al., 2026) or Chronos Benchmark II (Ansari et al., 2024) was used for training. *TimesFM-3* has no dedicated paper; its model card describes its training corpus as the GIFT-Eval pretraining collection (Aksu et al., 2024), excluding datasets that overlap fev-bench, together with Wikipedia page views through November 2023, Google Trends queries through the end of 2022, and unspecified synthetic and augmented data. *TiRex* documents its real pretraining data in Tables 5 and 6 of Auer et al. (2025): Chronos training datasets and a subset of the GIFT-Eval pretraining collection, supplemented with synthetic data. Its authors state that the training data do not overlap Chronos Benchmark II. Benchmark membership was checked against the published task lists: Chronos Benchmark II contains M5 and Traffic, while fev-bench contains LOOP Seattle (5-minute), SZ Taxi (15-minute), M5, and 15 tasks derived from BOOM.

Notes.

- *BOOM**: fev-bench contains 15 tasks derived from BOOM. Chronos-2 excludes fev-bench from training, and TimesFM-3 excludes fev-bench overlaps from its GIFT-Eval pretraining component.

Table 6: **Pretraining provenance.** Listed: in the documented pretraining corpus. Held out: explicitly excluded through a dataset or benchmark-level holdout. Not listed: identified by neither, which does not imply unseen. *source*: evidence covers the same source at a different resolution or in a different form. Evidence: [C2] Chronos-2 Table 6; [BII] Chronos Benchmark II exclusion; [fev] fev-bench exclusion; [GE] GIFT-Eval pretraining collection, via the TimesFM-3 model card; [WP] Wikipedia page views, via the TimesFM-3 model card; [T5] TiRex Table 5. * fev-bench contains tasks derived from BOOM; see Notes.

Dataset	Chronos-2	TimesFM-3	TiRex
M4	listed [C2]	not listed	listed [T5]
Electricity	listed [C2]	not listed	listed [T5]
Uber TLC	listed, *source* [C2]	listed [GE]	listed [T5]
Wind Farms	listed [C2]	listed, *source* [GE]	listed [T5]
BOOM	not listed*	not listed*	not listed
SG Carpark	not listed	not listed	not listed
Coastal T-S	not listed	not listed	not listed
LOOP Seattle	held out [fev]	held out [fev]	not listed
Electricity (15-minute)	listed [C2]	not listed	listed [T5]
SZ Taxi	held out [fev]	held out [fev]	not listed
London Smart Meters	listed [C2]	listed [GE]	listed [T5]
M5	held out [BII, fev]	held out [fev]	held out [BII]
Wiki Daily	listed [C2]	listed, *source* [WP]	listed [T5]
Monash Traffic	held out [BII]	listed [GE]	held out [BII]
Listed / held out / not listed	7 / 4 / 3	5 / 3 / 6	7 / 2 / 5

We could not establish whether those tasks share series with the 356 hourly queries used here, so we conservatively record BOOM as not listed.

- *Wind Farms, TimesFM-3*: the GIFT-Eval pretraining collection contains the same wind farms at minute resolution; we evaluate hourly values.
- *Wiki Daily, TimesFM-3*: the model card names Wikipedia page views through November 2023 as a training source, which covers the time span of our 2015 to 2022 data, but does not identify this particular dataset.
- *Held out, TimesFM-3*: the fev-bench exclusion is stated specifically for the GIFT-Eval pretraining component. Because the model card does not specify its synthetic and augmented data further, TimesFM-3 has the least complete provenance documentation of the three models.
- *Monash Traffic*: the pretraining corpora of Chronos-2 and TiRex contain other data from the same California PeMS traffic-sensor system, but not this dataset.
- *Listed* establishes documented pretraining overlap, at the level of the dataset or, where marked *source*, of its source. It does not imply that the specific windows used in our evaluation were presented during training.

A.4 Windows and splits

Windows. Every example consists of a context of $C = 512$ steps followed by a target of the next $H = 64$ steps. Within each series, candidate forecast origins are spaced H steps apart from the beginning of the series, so targets from different windows never overlap. A window is discarded if its context or target contains a missing value, or if its context is near-constant, with standard deviation below 10^{-6}.

Temporal split. A series is eligible if it contains at least three valid windows. For each eligible series, the last valid window is reserved as its candidate test window, the second-to-last as its candidate validation window, and all earlier windows as candidate training windows. Thus, every training target precedes the validation target, which precedes the test target, with no target time step shared across splits. Each series selected for validation and test contributes exactly one window to each split.

Budgets. Each dataset uses 1,394 training windows and, by default, 262 validation and 262 test windows. Validation and test series are sampled from the eligible series; for eligible series not selected, the final two windows remain unused. Training windows are drawn from all retained eligible series and subsampled to 1,394 by round-robin over series, so every series contributes one window before any contributes a second and long series cannot dominate. Four datasets deviate from the default validation/test budget:

- SG Carpark and BOOM use all 354 eligible series, giving 354 validation and 354 test windows each.
- Coastal T-S uses all 48 eligible series, giving 48 validation and 48 test windows.
- SZ Taxi uses all 156 eligible series, giving 156 validation and 156 test windows.

Series cap. London Smart Meters, M5, and Wiki Daily have 5,555, 28,491, and 100,000 eligible series, respectively, more than the 1,394 training-window budget can cover. Before constructing the splits, we retain a fixed-seed random sample of 1,394 eligible series without using their data values. No series-level cap is applied to the remaining datasets.

Eligibility. Most datasets retain all series after filtering. The exceptions are Wind Farms, with 328 of 337 series eligible, and BOOM, with 354 of the 356 selected metric queries eligible.

Bootstrap clusters. Confidence intervals are obtained by resampling whole clusters. The cluster is one series by default; for SG Carpark it is one carpark, for BOOM one metric query, and for Coastal T-S one station, grouping its temperature and salinity series and yielding 24 independent clusters. Datasets with fewer independent clusters consequently have greater bootstrap uncertainty.

Identical windows across models. All three models are evaluated on exactly the same windows. For each dataset, the windows are fixed once and stored as a reference containing the split counts, the series identity of every test window, and a checksum of the test contexts. Every model run is checked against this reference and stops on any mismatch. All random choices use seed 0.

A.5 Target spaces

Each model normalizes its input internally, and we train its readouts in the same normalized space as its native head. All statistics come from the context $x_{1:C}$ only, never from the future $y_{1:H}$.

- **Chronos-2:** $z_t = \operatorname{arcsinh}\big((y_t - \mu)/\sigma\big)$, with μ and σ the context mean and standard deviation.
- **TimesFM-3:** $z_t = (y_t - \tau_t - \mu_{15})/\sigma_{15}$, where τ_t is a least-squares linear trend extrapolated from the context (set to zero unless the residual standard deviation is below half the context standard deviation), and μ_{15}, σ_{15} are the running statistics the model computes at token 15, whose state its head reads.
- **TiRex:** $z_t = (y_t - m_k)/s_k$ for the 32 steps forecast by pass $k \in \{1, 2\}$, with m_k, s_k the mean and standard deviation of that pass's context ($s_k = 0$ is replaced by $|m_k| + 10^{-5}$).

Predictions are mapped back to raw units by the models' own inverses: $\mu + \sigma \sinh(\hat{z}_t)$ for Chronos-2, $\sigma_{15}\hat{z}_t + \mu_{15}$ clipped to the model's value range and then $+\tau_t$ for TimesFM-3, and $s_k\hat{z}_t + m_k$ for TiRex. All three are monotone, so they preserve quantile order, and MASE is computed in raw units (Appendix A.8). Because the target spaces differ, readout losses are compared only within a model.

A.6 Readout training

Inputs. At each depth, readout features are standardized per dimension using means and standard deviations estimated from the training windows only. For Chronos-2 and TiRex, one standardization is shared across the K readout states, matching the shared readout.

Objective. Readouts are trained with pinball loss over the nine quantile levels and all $H = 64$ horizon steps in each model's target space (Appendix A.5). We report mean pinball loss for all three models. The Chronos-2 optimization code sums rather than averages over quantile levels, differing

from the reported convention only by a constant factor $(2Q)$; this does not affect weight-decay selection or the tunnel entrance.

Optimization. Each readout is a linear layer trained with full-batch AdamW for 300 epochs at learning rate 10^{-2}, with weight decay applied to the weights but not the bias. We retain the final-epoch parameters without early stopping. For each model, dataset, and depth, every weight-decay candidate starts from the same seeded initialization.

Weight-decay selection. For every model, dataset, and depth, we train one readout for each weight decay in

$$\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 0.3, 1, 3, 10, 30, 45, 65, 90\}$$

on the full training split and select the value with the lowest validation loss. The selected readout is not retrained on training and validation data jointly, and the test split is used only for final evaluation. We restrict the grid to values satisfying $\eta\lambda < 1$, where η is the learning rate and λ the weight decay: AdamW's decoupled decay multiplies the current weights by $1 - \eta\lambda$ at each update, which reaches zero at $\eta\lambda = 1$ and becomes negative beyond it.

Largest weight decay. When the largest value in the grid is selected, we compare its validation loss with the best constant-forecast loss, obtained in closed form from the per-step training quantiles. This provides a diagnostic for distinguishing an effectively uninformative readout from a regularization optimum that may lie beyond the search range.

Exception: LOOP Seattle. For LOOP Seattle, the largest weight decay was selected in the 10^{-2} runs. We therefore use learning rate 10^{-3} for all three models and augment the grid with $\{150, 300, 450, 650, 900, 950, 975, 990\}$, keeping $\eta\lambda < 1$. All other readout-training settings remain unchanged. As for every other dataset, the no-information reference is the closed-form constant forecast.

A.7 Alignment training

Adapter and objective. The adapter of Section 3.3 has $\Delta_\ell \in \mathbb{R}^{d\times d}$ and $b_\ell \in \mathbb{R}^d$, i.e. $d^2 + d$ parameters per depth: 0.59M for Chronos-2, 1.64M for TimesFM-3 and 0.26M for TiRex. Like the readout, one adapter is shared across all states read by the head (the four forecast slots of Chronos-2 and the two pass states of TiRex). It minimizes

$$\frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \left\| g\big(A_\ell(h_i^{(\ell)})\big) - g\big(h_i^{(L)}\big) \right\|_2^2 \tag{6}$$

over the training windows, with both forecasts in the model's normalized output space (Appendix A.5). The targets are the full model's own forecasts, so no future values are used for fitting or selection. The selected adapter is not retrained, and the test split is read once.

Chronos-2 and TiRex. Their heads are nonlinear, so the adapter is fitted iteratively: full-batch AdamW at learning rate 10^{-3} for 1,500 epochs, starting from $\Delta_\ell = 0$, $b_\ell = 0$ (the hard cut). Weight decay acts on Δ_ℓ only and therefore pulls the adapter toward the identity. We train one adapter per weight decay in $\{0, 0.1, 1, 10, 30, 100\}$, evaluate the validation error every 25 epochs, and keep the weight decay and epoch with the lowest validation error. Epoch 0, the hard cut, is always a candidate, so the adapter is used only if it improves on the hard cut on validation data. These optimizer settings were fixed before the full runs, on validation data only, at the tunnel entrances of Chronos-2 and TiRex on Electricity: of three settings (learning rate 10^{-2} for 300 epochs, 3×10^{-3} for 1,000, and 10^{-3} for 1,500), only the last converged for every fit, i.e. selected a checkpoint before 90% of the epochs and a weight decay below the grid maximum.

TimesFM-3. Its head is a single linear map W, so the objective reduces to a ridge problem in the metric $W^\top W$ with a closed-form solution:

$$\min_{\Delta_\ell,\, b_\ell} \sum_i \left\| (I + \Delta_\ell) h_i^{(\ell)} + b_\ell - h_i^{(L)} \right\|_{W^\top W}^2 + \lambda \left\| \Delta_\ell \right\|_F^2, \tag{7}$$

with b_ℓ unpenalized and $\lambda = \kappa\, g_{\max} m_{\max}$, where $g_{\max}$ and $m_{\max}$ are the largest eigenvalues of the centred training Gram matrix and of $W^\top W$, and $\kappa \in \{10^{-12}, 10^{-11}, \ldots, 10^{2}\}$. The hard cut is an explicit candidate; the candidate with the lowest validation error in the same metric is kept, with ties going to the larger λ.

Nonlinear variant. The nonlinear variant adds a bottleneck branch, $A_\ell(h) = h + \Delta_\ell h + b_\ell + W_2\,\mathrm{GELU}(W_1\,\mathrm{LN}(h) + c)$, where LN is a layer normalization without learned parameters, $W_1 \in \mathbb{R}^{64\times d}$ is initialized randomly with a fixed seed and $W_2 \in \mathbb{R}^{d\times 64}$ at zero, so the branch contributes nothing at initialization. It uses the same objective and iterative protocol for all three models, with weight decay also on W_1 and W_2. We tested it on Electricity at 25, 50 and 75% of depth under an adoption rule fixed in advance and evaluated on validation data only; the rule was not met, and the full run on all 14 datasets agrees (Appendix B.3). For TimesFM-3, the iterative fit, with or without the branch, does not fit even the training windows (on Electricity, training R^2 between -314 and -6 against the full model's forecast), whereas the closed-form affine adapter fits them almost exactly ($R^2 \approx 0.997$) but not the validation windows (R^2 between -3.0 and -0.6). We therefore use the affine adapter for all results.

A.8 Metrics and uncertainty

Pinball loss. For quantile level τ, the pinball loss of a prediction $\hat{z}$ for a target z is $\rho_\tau(z - \hat{z})$ with $\rho_\tau(u) = \max\{\tau u, (\tau - 1)u\}$. The readout loss of a window is the mean of ρ_τ over the nine quantile levels and the H horizon steps, in the model's target space (Appendix A.5); a dataset's loss is the mean over its windows. This loss selects the weight decay and defines the forecasting tunnel.

MASE. Forecast accuracy in raw units is measured by the mean absolute scaled error (Hyndman & Koehler, 2006) of the median forecast. For a window with context x, future y and median forecast $\hat{y}$:

$$\mathrm{MASE} = \frac{\frac{1}{H}\sum_{t=1}^{H} |y_t - \hat{y}_t|}{\frac{1}{C-m}\sum_{t=m+1}^{C} |x_t - x_{t-m}|}, \tag{8}$$

where m is the dataset's seasonal period (Table 5). The denominator, the error of a seasonal naive forecast, is computed from the window's own context, so every window has its own scale and no future value enters it. It is floored at 10^{-8}. A dataset's MASE is the mean over its test windows. The full model, every readout and every truncated model are scored on the same windows with the same denominators, so their MASE values are directly comparable. This in-context scaling differs from benchmarks such as GIFT-Eval and fev-bench, which scale by the series' history before the forecast, so our absolute MASE values are not comparable to published leaderboards.

Confidence intervals. Windows of the same series are correlated, so we resample whole clusters: series, or carparks, metric queries and stations for SG Carpark, BOOM and Coastal T-S (Appendix A.4). Each of $B = 5{,}000$ replicates draws as many clusters as the dataset has, with replacement, and recomputes the window mean. The 95% interval spans the 2.5th to 97.5th percentiles of the replicates. The resampled clusters depend only on the dataset and the seed (0), so every depth, model pathway and metric of a dataset uses the same replicates. Comparisons are therefore paired: the interval of a difference, such as a depth against the final block, or of a ratio, such as a truncated model's MASE over the full model's, is computed within each replicate.

A.9 Latency protocol

What is timed. We time each model's public forecasting interface end to end, from a context array in host memory to quantile forecasts returned to host memory: `predict_quantiles` for Chronos-2, `predict_batch` for TimesFM-3, and `forecast` with default two-pass inference for TiRex. This is our headline measurement. As a secondary measurement, we time the forward pass alone with inputs already on the GPU. Each configuration is measured at batch sizes 1 (latency) and 256 (throughput setting). We time the full model and, at every depth, the hard cut, the aligned model and the readout used as a replacement head. Execution time depends on an adapter's shape rather than its learned values, so adapters are timed with random weights.

Inputs. For complete contexts of fixed length, the timed computation does not depend on the input values, so we use synthetic random-walk contexts of length $C = 512$ rather than the 14 evaluation datasets. Four fixed batches are cycled so that consecutive calls receive different inputs.

Procedure. All timings use one full NVIDIA A100 GPU rather than a partitioned device, with TF32 disabled and no compilation or mixed-precision inference. Model parameters and surrounding computation use float32; TiRex's recurrence internally uses bfloat16, as in all our experiments. Each configuration runs in a fresh process, so removed blocks leave no cached state, and configurations run in random order. After 20 warm-up calls, we time 100 calls with GPU synchronization before and after each call and record the median. Only truncated configurations that passed our verification of physical truncation are timed.

Speedup. Within each job, speedup is the full model's median runtime divided by the configuration's median runtime. The full model is timed at the beginning, middle and end of each job, and the median of the three is the baseline; the three differ by a median of 1.0% and at most 5.4%. We repeat the procedure in three independent jobs (TiRex's depths are split over two jobs per repetition, each with its own baseline) and report the median of the per-job speedups.

Full-model runtimes. At batch size 1, one end-to-end forecast takes 19.4 ms for Chronos-2, 97.5 ms for TimesFM-3 and 499 ms for TiRex. For TimesFM-3, the forward pass alone takes 49.3 ms, so about half of its end-to-end time is spent outside the backbone and cannot be removed by truncation; this limits its end-to-end speedup at batch size 1.

Physical versus offline truncation. Forecasting performance along the frontier (Figure 4 and Table 13) is computed by applying the aligned pathway to the cached representations of each depth, while speed is measured on physically truncated models. We verify that both describe the same model. At every depth of every model, the physically truncated network reproduces the full model's state at that depth bitwise, on synthetic inputs and on 256 Electricity windows against the cached features; the removed blocks are never executed; and the physically truncated aligned model reproduces the offline aligned forecast within a relative tolerance of 10^{-4}. In addition, 419 truncated configurations, covering every model-dataset pair, were evaluated physically on the complete test splits; their test MASE agrees with the offline pathway to within 7.9×10^{-5} relative, so no reported result depends on which pipeline is used. Individual forecast values agree less tightly. For TiRex, even the unmodified model differs between the two pipelines by up to 11% of the mean forecast magnitude on single values, consistent with its bfloat16 recurrence; for TimesFM-3, the aligned pathway differs by up to 5% while the unmodified model agrees. The element-wise check we had specified in advance for these test-split evaluations (relative tolerance 10^{-4}) therefore rejected 96 of the 419 configurations. We replaced it, uniformly for all models, by the requirement that test MASE agree within 10^{-4} relative, which every configuration satisfies.

A.10 Compute

All experiments ran on a shared academic GPU cluster. Each GPU job used one NVIDIA A100 (40 GB); building windows and computing representation geometry need no GPU. The software stack was Python 3.11, PyTorch 2.12 with CUDA 13.2, `chronos-forecasting` 2.3.1, `timesfm` 3.0.2 and `tirex-ts` 1.4.2. The pretrained backbones are never trained, so all computation is inference plus the fitting of small readouts and adapters. Table 7 reports GPU time by stage. Over half of it is latency measurement; the probing analyses themselves take under 3 GPU-hours. The nonlinear adapter, which we did not adopt, accounts for 18.4 of the 48.9 GPU-hours.

Table 7: **Compute by stage**, in A100 GPU-hours from the scheduler's accounting, summed over all GPU jobs including failed and cancelled ones. Feature extraction runs inside the readout jobs. Alignment, truncation and latency include both the affine and the nonlinear adapter.

Stage	GPU-hours
Feature extraction, layerwise readouts ($Q = 9$ and $Q = 1$) and geometry	2.9
Alignment (all depths, models and datasets)	13.8
Truncation checks and evaluation	0.5
Latency (three repetitions)	26.0
Development, smoke tests and superseded runs	5.7
Total	48.9

B Additional Results

B.1 Additional forecast-recoverability results

B.1.1 Complete layerwise forecasting curves

Figures 5, 6 and 7 show the complete nine-quantile ($Q = 9$) layerwise forecasting results for all three models and all 14 datasets. Tunnel entrances are selected from validation loss using the sustained 5% criterion, while test loss is shown only for held-out evaluation.

Pretraining exposure and the tunnel entrance. We find no consistent relationship between a dataset's documented pretraining exposure (Table 6; cells marked *source* count as listed) and where its tunnel begins. Within each model, the median normalized tunnel entrance ℓ_{tunnel}/L for listed versus held-out or not-listed datasets is 0.33 versus 0.25 for Chronos-2, 0.05 versus 0.25 for TimesFM-3, and 0.83 versus 0.67 for TiRex. The direction therefore differs across models, and exact permutation tests do not distinguish the groups ($p \geq 0.58$).

B.1.2 Held-out performance at the tunnel entrance

Table 8 reports held-out performance at each validation-selected tunnel entrance. Across all 42 model-dataset pairs, the median increase in $Q = 9$ test loss relative to the final-block readout is 1.3% (interquartile range -0.7% to 4.3%), and 34 of the 42 pairs stay within 5% on test, both at the entrance and at every later block. By model, the median increase is 2.8% for Chronos-2, 0.8% for TimesFM-3 and 2.1% for TiRex. The largest increase is 9.8%, for TimesFM-3 on Uber TLC; negative values, where the entrance readout is better on test than the final-block readout, occur for all three models. The median-forecast MASE changes by a median of $+1.3\%$.

B.1.3 Sensitivity to readout capacity

We repeat the analysis with a median-only ($Q = 1$) readout, whose output is a ninth of the size of the $Q = 9$ readout, trained with the same settings as the $Q = 9$ readout of each dataset (for LOOP Seattle, the lower learning rate and extended grid of Appendix A.6). Table 9 compares the tunnel entrances. Of the 42 pairs, 31 have exactly the same entrance, 37 differ by at most one block, and the median absolute difference is zero blocks. The largest difference is TimesFM-3 on Coastal T-S, where the $Q = 1$ entrance moves from L5 to L18; Coastal T-S has the fewest bootstrap clusters (24). The next largest are TimesFM-3 on LOOP Seattle (four blocks earlier with $Q = 1$) and Chronos-2 on M4 (three blocks earlier).

B.1.4 Sensitivity to the tunnel tolerance

We repeat the analysis with $\epsilon \in \{0.02, 0.05, 0.08\}$, using the same validation curves. Table 10 lists the resulting entrances. The tunnel begins before the final block in 36 of the 42 pairs at $\epsilon = 0.02$, 40 at $\epsilon = 0.05$, and all 42 at $\epsilon = 0.08$. Tightening the tolerance to 2% moves the median entrance from 29% to 67% of model depth for Chronos-2 and from 5% to 65% for TimesFM-3, and loosening it to 8% moves TiRex from 79% to 67%.

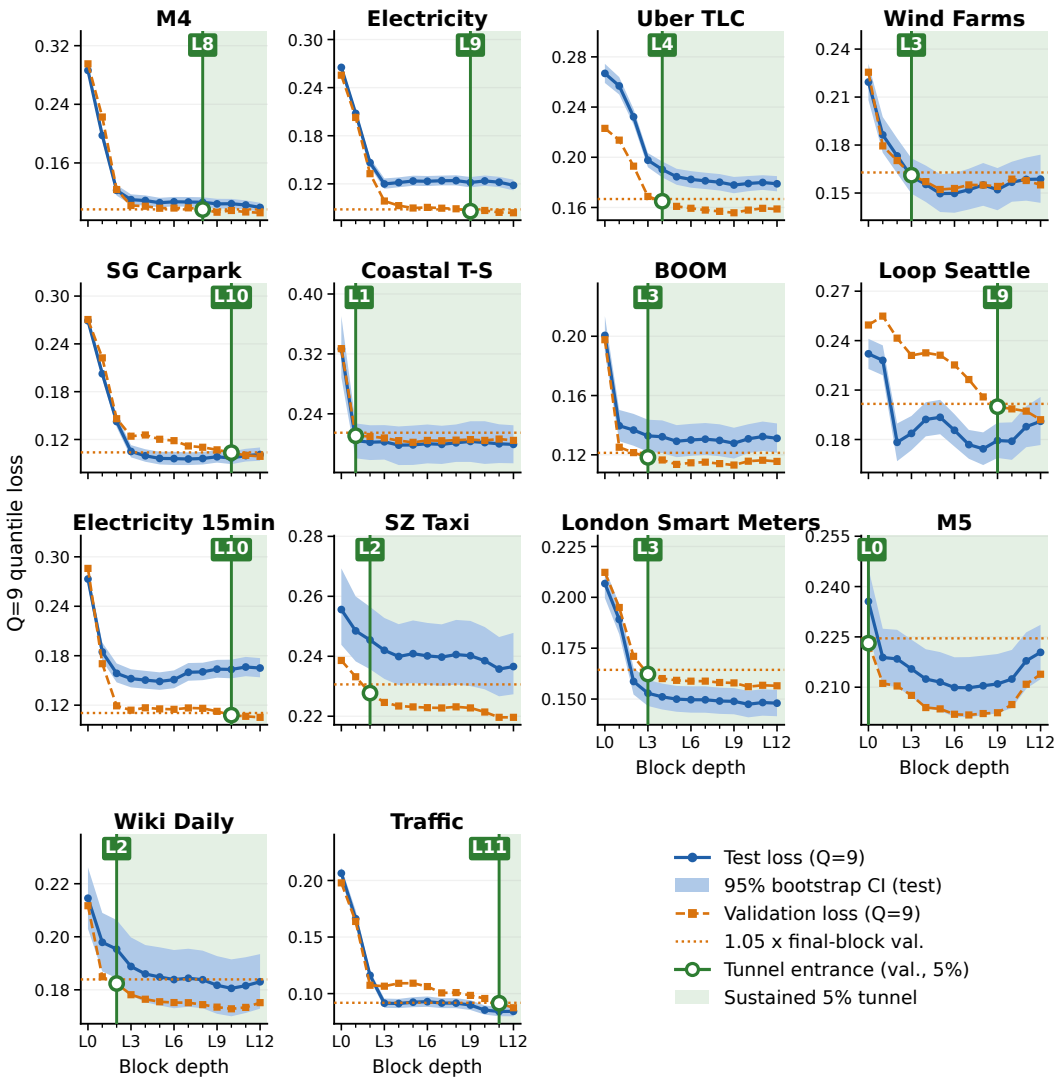


Figure 5: **Chronos-2 layerwise Q=9 forecasting loss across all datasets.** The tunnel entrance is selected from validation loss using the sustained 5% criterion. Bands show 95% series-level cluster-bootstrap intervals ($B = 5000$).

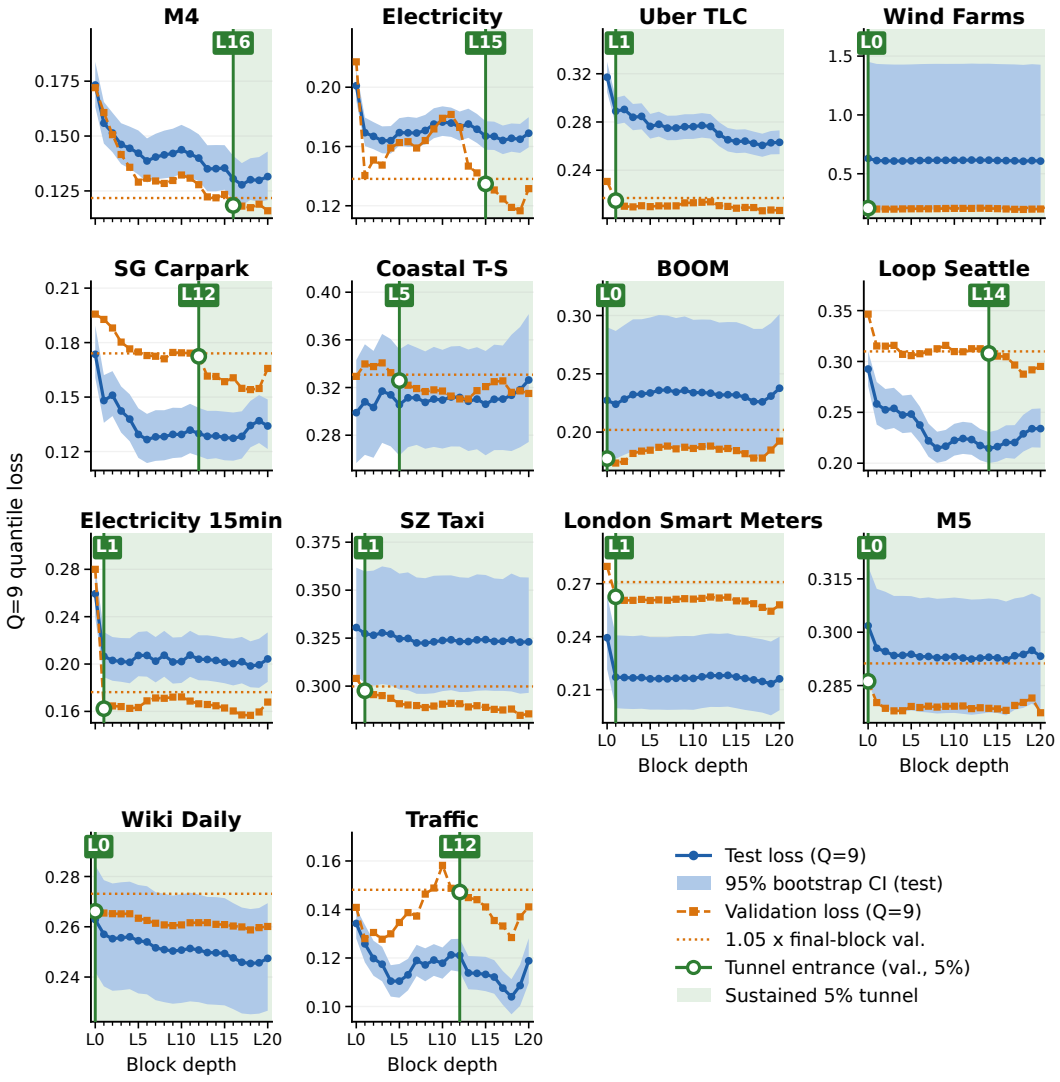


Figure 6: **TimesFM-3 layerwise Q=9 forecasting loss across all datasets.** The tunnel entrance is selected from validation loss using the sustained 5% criterion. Bands show 95% series-level cluster-bootstrap intervals ($B = 5000$).

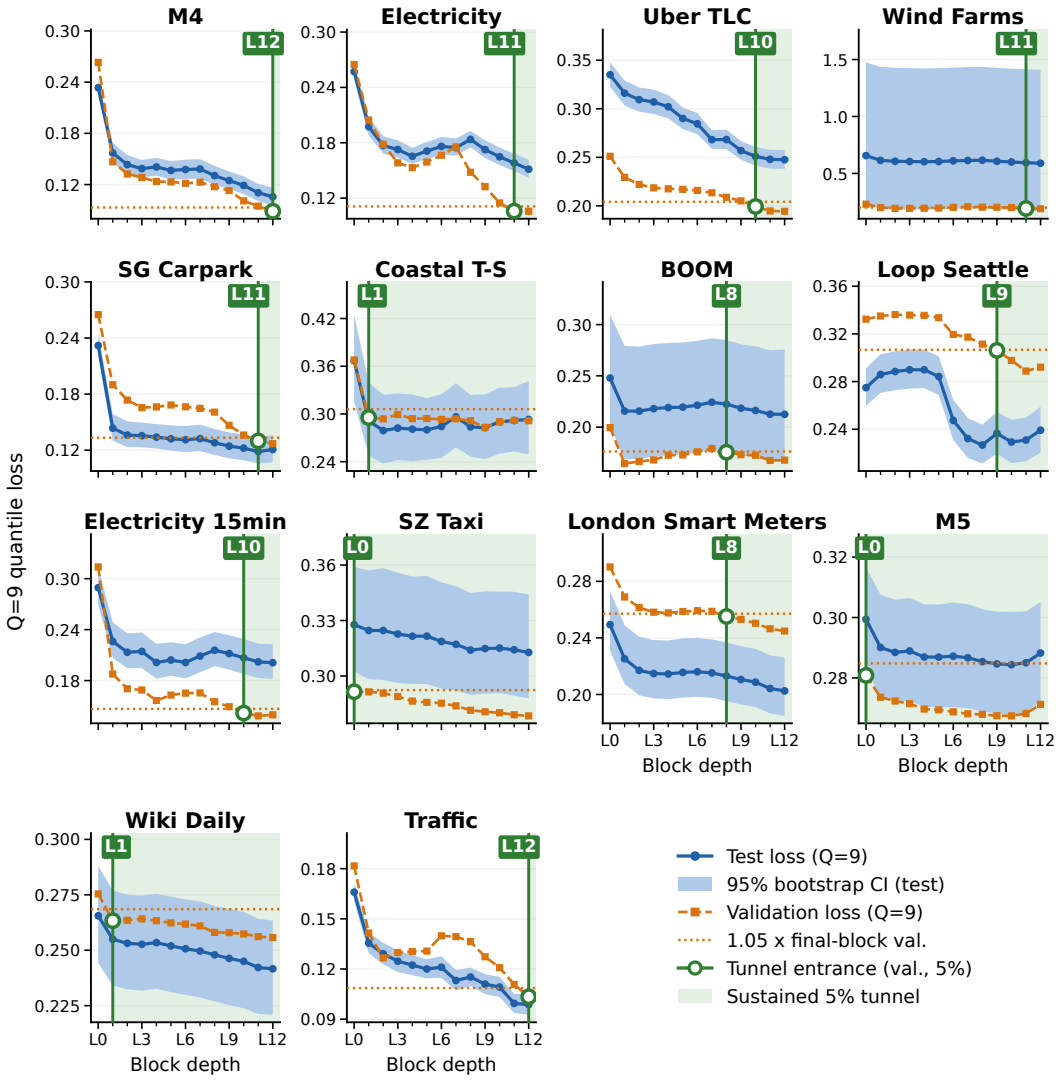


Figure 7: **TiRex layerwise Q=9 forecasting loss across all datasets.** The tunnel entrance is selected from validation loss using the sustained 5% criterion. Bands show 95% series-level cluster-bootstrap intervals ($B = 5000$).

B.2 Additional geometry results

Geometry estimators. We compute geometry from the same forecast-producing states used by the readouts, using the training split. Chronos-2 contributes its four forecast-slot states per window, TimesFM-3 its last context token, and TiRex the corresponding state from the first inference pass. We use debiased linear CKA (Kornblith et al., 2019; Cortes et al., 2024; Murphy et al., 2024) and effective rank (Roy & Vetterli, 2007). Training-split CKA provides substantially more samples than the test split; as a robustness check, the dominant CKA boundary differs by at most one block between the two splits in 37 of 42 model-dataset pairs. Chronos-2's L0 forecast-slot states contain no window-specific information, so CKA involving L0 is excluded.

Geometric landmarks. We summarize each profile by the effective-rank peak, the dominant CKA boundary, the change in CKA and effective rank at the tunnel entrance, and the sharpest adjacent CKA change. The dominant CKA boundary is the split of L1 to LL into two contiguous regions that maximizes within-region relative to between-region CKA. A landmark is considered near the tunnel entrance if it lies within $\max(1, \mathrm{round}(0.1L))$ blocks. These descriptors are post hoc summaries rather than selection criteria.

The geometric landmarks do not consistently coincide with the forecasting tunnel entrance (Table 11). Instead, substantial geometric change often continues inside the tunnel: the sharpest adjacent CKA change lies within it in 34 of 42 model-dataset pairs. The final block retains a median 7%, 19%, and 9% of peak effective rank for Chronos-2, TimesFM-3, and TiRex, respectively. The same pattern holds under the stricter 2% tolerance, whose entrances lie at or after the 5% entrances: the sharpest adjacent CKA change lies inside the tunnel in 30 of 42 pairs (16.1 expected by chance), and the effective-rank peak lies at or after the entrance in 23 of 42 (16.9 expected). Because a larger tolerance can only move the entrance earlier, both counts can only grow for tolerances above 5%. Thus, forecasting performance can saturate well before representation geometry does.

Figure 8 shows the full Chronos-2 CKA matrices. Several datasets display blocks of mutually similar depths separated by sharper transitions, so CKA suggests several candidate cut points rather than a single one, and none of them is tied to forecast quality.

B.3 Additional compatibility results

Figures 9, 10 and 11 show the test MASE increase of every readout at every depth for all 14 datasets, in the format of Figure 3. They also show a forecast-trained adapter for Chronos-2 and TiRex: the same affine map as in Section 3.3, trained instead on the $Q = 9$ pinball loss against the true future through the frozen pathway, with the iterative protocol of Appendix A.7. At the tunnel entrance its median test MASE increase is no lower than the label-free adapter's ($+6.7\%$ against $+5.5\%$ for Chronos-2, $+8.7\%$ against $+6.3\%$ for TiRex), although it is slightly lower at a few early depths (Chronos-2 L1; TiRex L1, L3 and L4).

Depth profiles. Aligned Chronos-2 has a median test MASE increase between $+0.4\%$ (L11) and $+4.4\%$ (L3) at every depth from L3 onwards. For TimesFM-3, the aligned pathway stays between roughly $+30\%$ and $+82\%$ up to L17 and comes within 15% only at L18 ($+14.3\%$) and L19 ($+5.4\%$). TimesFM-3's head is linear and reads the residual stream without a final normalization, which is consistent with, but not shown to cause, this behaviour.

Nonlinear adapter. Table 12 repeats Table 2 with the nonlinear adapter of Appendix A.7. For Chronos-2 and TiRex, the median change in test MASE over all 168 depth-dataset pairs of each model is -0.2 percentage points, and the share of the hard-cut gap closed moves from 85 to 87% (Chronos-2) and from 90 to 91% (TiRex). The affine map therefore captures almost all of the compatibility that a lightweight adapter can restore. For TimesFM-3 the nonlinear adapter is far worse than the affine one, because it must be fitted iteratively and this fit does not reproduce the full model's forecasts even on the training windows (Appendix A.7).

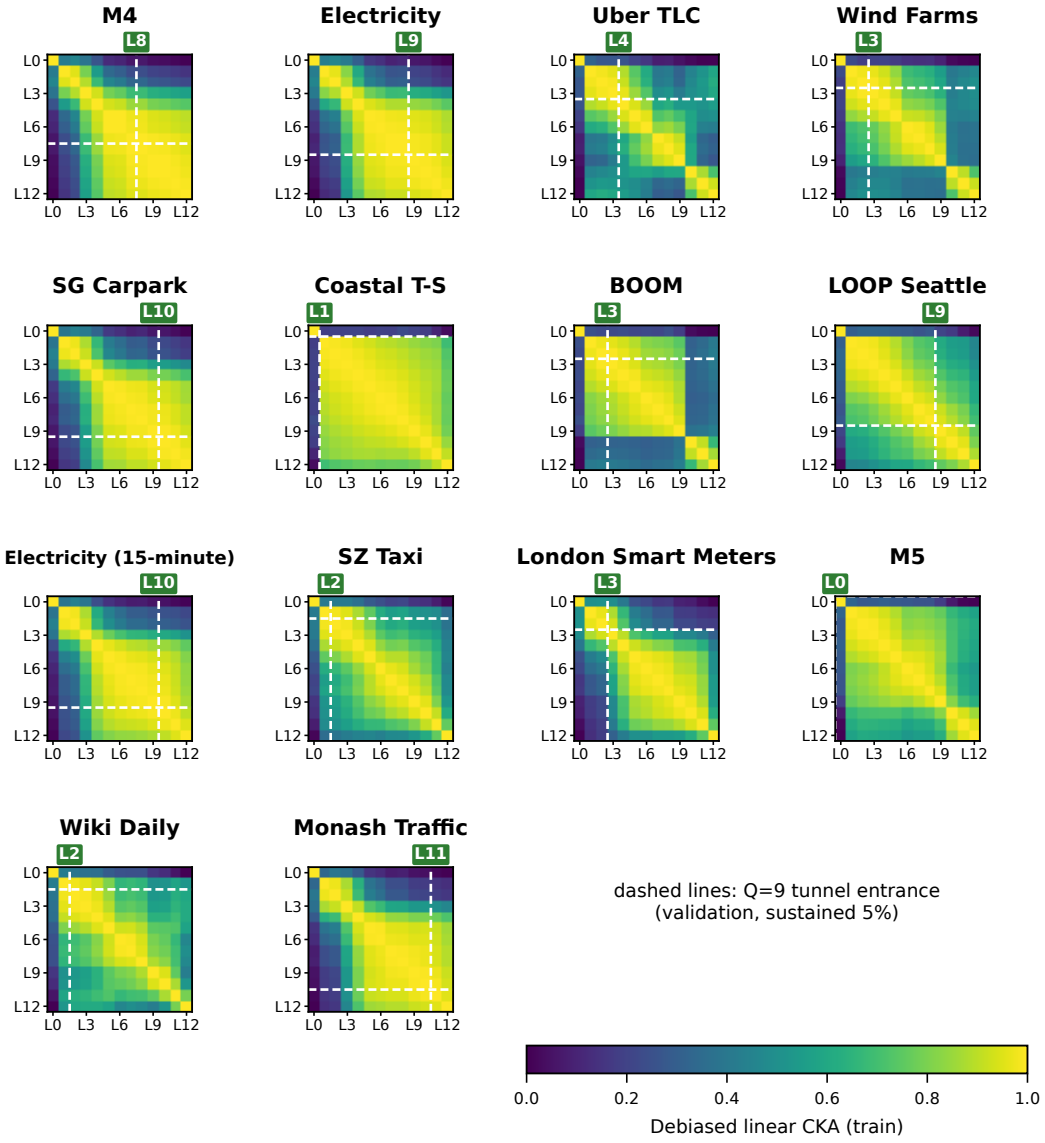


Figure 8: **Chronos-2 debiased linear CKA across all depths and datasets.** CKA is computed on training representations; dashed lines mark the validation-selected sustained 5% tunnel entrance.

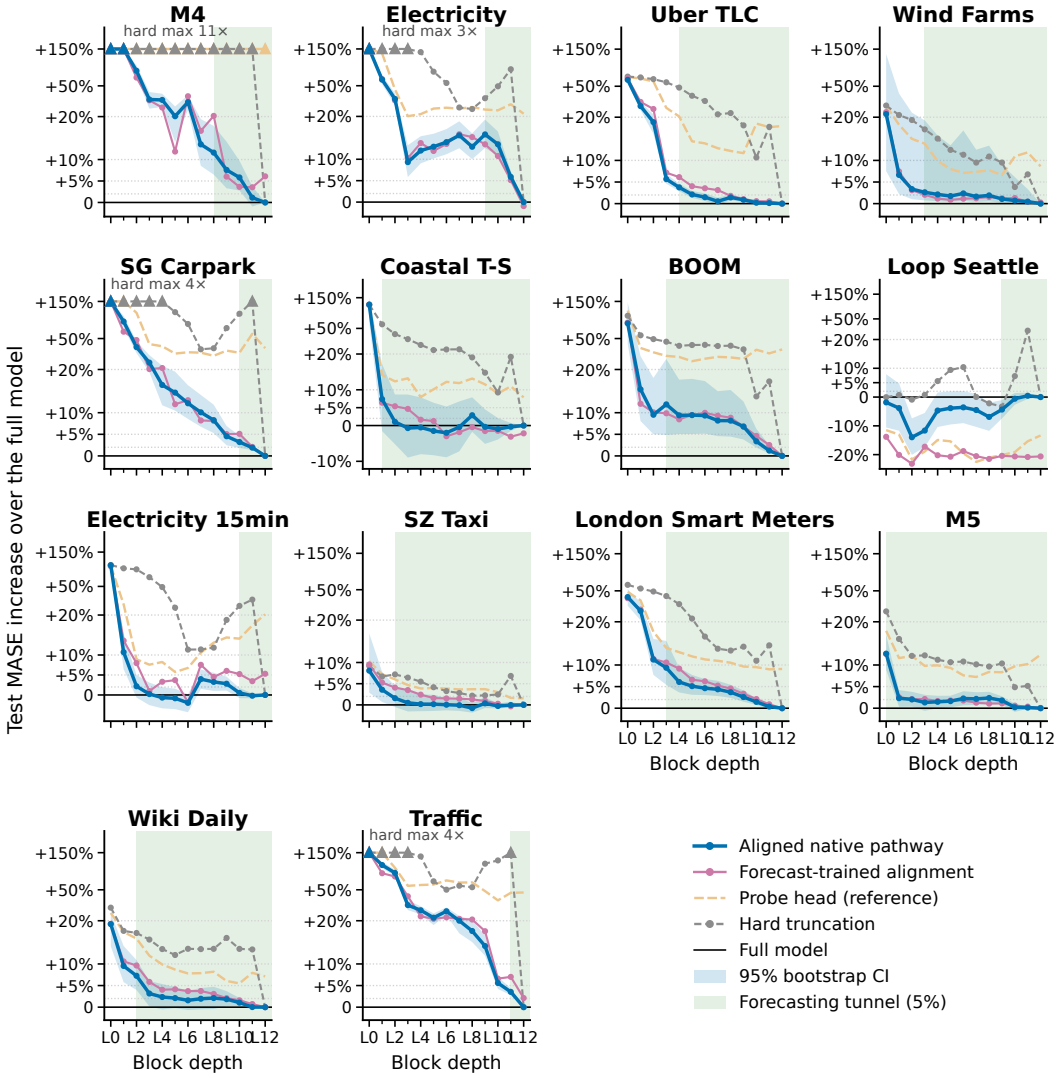


Figure 9: **Chronos-2: native-pathway compatibility across depth** on all 14 datasets. Test MASE increase over the full model for the hard cut, the aligned native pathway (with its 95% series-level cluster-bootstrap band), the forecast-trained adapter and the probe head (reference, dashed). The shaded region is the forecasting tunnel.

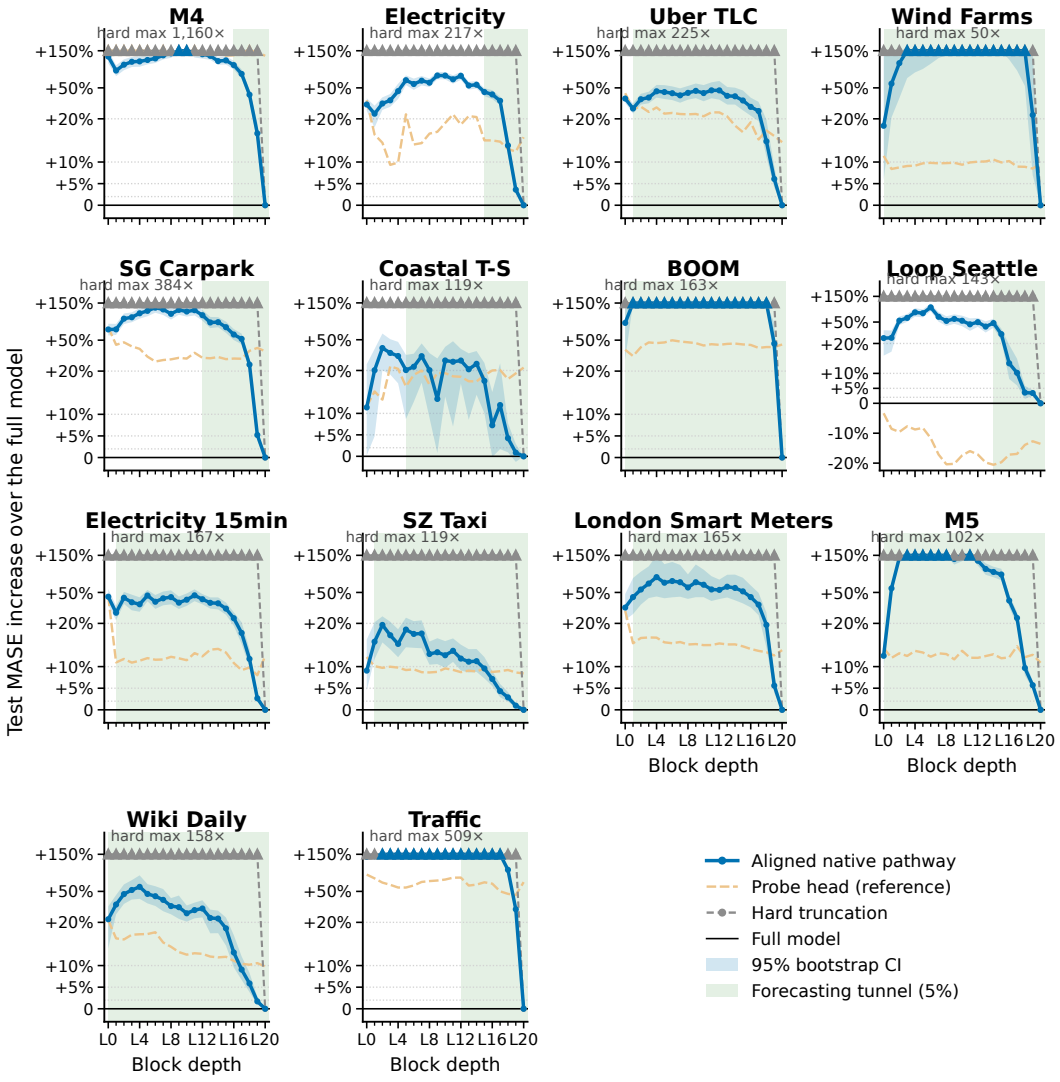


Figure 10: **TimesFM-3: native-pathway compatibility across depth** on all 14 datasets, in the format of Figure 9. A forecast-trained adapter is not shown: with TimesFM-3's linear head it spans exactly the function class of the probe head.

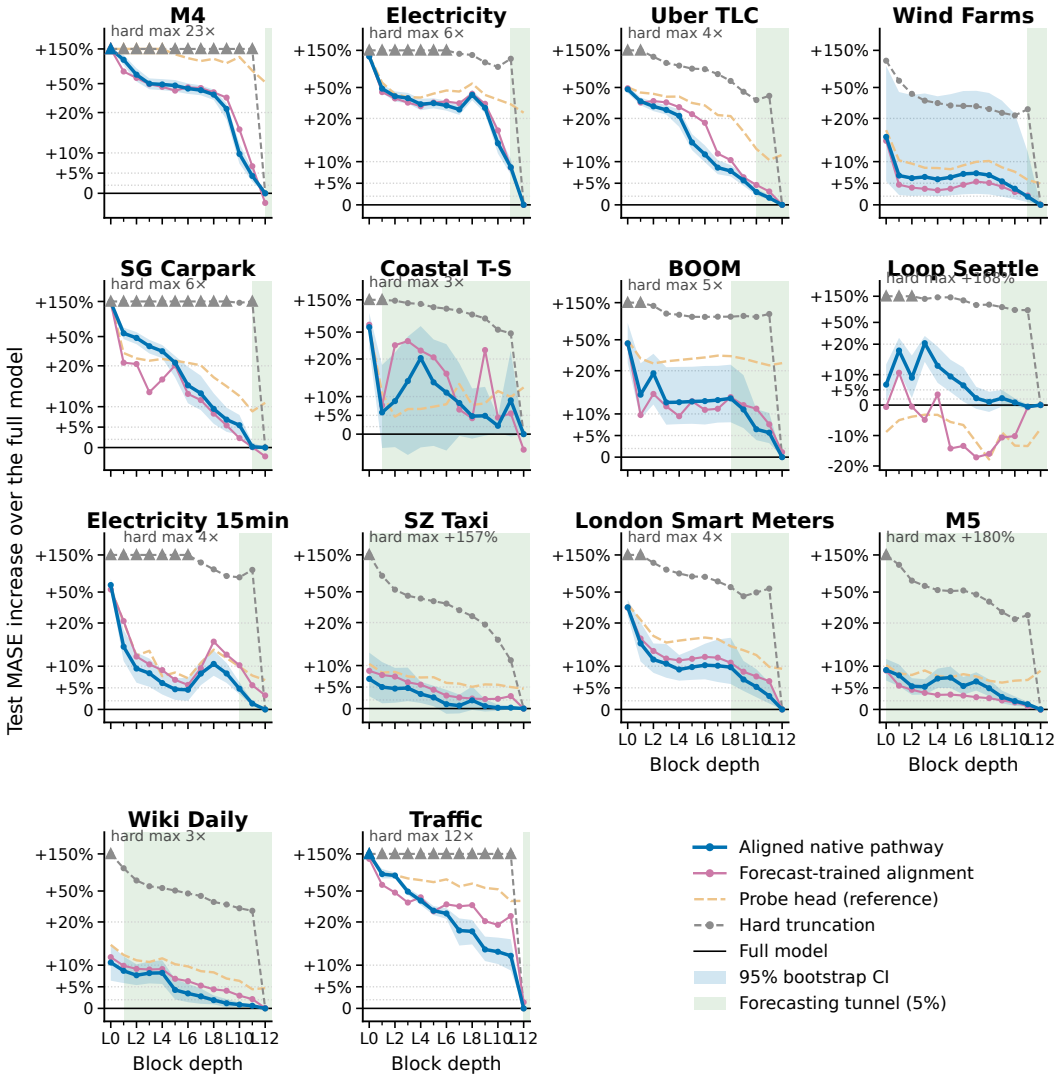


Figure 11: **TiRex: native-pathway compatibility across depth** on all 14 datasets, in the format of Figure 9.

B.4 ADDITIONAL TRUNCATION RESULTS

Complete frontier. Table 13 gives the complete numerical version of the performance-compute frontier in Figure 4: every aligned truncation depth of all three models, with the full model and Chronos-2-small as references. Forecasting performance is computed as described in Appendix A.9.

Throughput at batch size 256. For Chronos-2 and TiRex, speedups at batch size 256 closely track those at batch size 1; for example, Chronos-2 after L4 runs $2.43\times$ and $2.51\times$ faster, and TiRex after L6 $1.99\times$ and $1.96\times$. TimesFM-3 differs: its speedup at batch size 1 never exceeds $1.43\times$, whereas at batch size 256 it grows steadily with the number of removed blocks, to $2.70\times$ after L5 and $5.94\times$ with no block retained. This is consistent with a fixed per-call cost outside the backbone that dominates at batch size 1 (Appendix A.9). Its forecasting degradation is unchanged, so the larger throughput gains still come at the cost reported in Table 13. Chronos-2-small also benefits more from batching ($1.80\times$ at batch size 1, $3.28\times$ at 256).

Truncation inside and before the tunnel. Table 14 gives the depth-matched comparison of Section 4.4. At each cut depth, the datasets of a model are split according to whether that depth lies inside their validation-selected 5% tunnel ($\ell \geq \ell_{\text{tunnel}}$) or before it, and the median aligned test MASE increase is reported for each group. Because speedup depends only on the cut depth, both groups are compared at the same speedup. Depths are listed where both groups contain at least two datasets. Truncation is cheaper inside the tunnel at all 9 such depths for Chronos-2, all 12 for TiRex, and 14 of 15 for TimesFM-3; the exception is TimesFM-3 at L13, where only three datasets lie before the tunnel. The comparison is observational: datasets whose tunnel begins early may also be easier to forecast from intermediate states for other reasons.

Table 8: **Held-out performance at the validation-selected 5% tunnel entrance.** Δ test gives the relative change in $Q = 9$ test loss compared with the final-block readout, with a paired 95% series-level cluster-bootstrap interval ($B = 5000$). Negative values indicate lower test loss at the tunnel entrance. Entrances are selected using validation loss only; where the entrance is the final block, Δ test is zero by construction.

Dataset	Entr.	Δ test (%)	95% CI	Dataset	Entr.	Δ test (%)	95% CI
Chronos-2							
M4	L8	$+7.1$	$[5.0, 9.3]$	LOOP Seattle	L9	-6.1	$[-9.0, -3.1]$
Electricity	L9	$+3.0$	$[1.5, 4.7]$	Electricity (15-minute)	L10	-1.0	$[-2.7, 0.7]$
Uber TLC	L4	$+6.4$	$[5.3, 7.5]$	SZ Taxi	L2	$+3.7$	$[1.5, 6.6]$
Wind Farms	L3	$+1.0$	$[-3.9, 6.3]$	London Smart Meters	L3	$+3.3$	$[1.7, 5.0]$
SG Carpark	L10	-4.4	$[-6.0, -2.8]$	M5	L0	$+6.9$	$[4.8, 9.2]$
Coastal T-S	L1	$+2.5$	$[-4.4, 10.0]$	Wiki Daily	L2	$+6.7$	$[4.3, 9.4]$
BOOM	L3	$+1.3$	$[-1.9, 4.3]$	Monash Traffic	L11	-0.2	$[-0.9, 0.6]$
TimesFM-3							
M4	L16	-0.8	$[-2.8, 1.4]$	LOOP Seattle	L14	-8.3	$[-11.5, -5.0]$
Electricity	L15	-1.2	$[-2.9, 0.6]$	Electricity (15-minute)	L1	$+1.1$	$[-2.9, 5.3]$
Uber TLC	L1	$+9.8$	$[8.6, 11.1]$	SZ Taxi	L1	$+1.3$	$[0.1, 2.6]$
Wind Farms	L0	$+3.7$	$[1.4, 14.8]$	London Smart Meters	L1	$+0.4$	$[-0.9, 2.0]$
SG Carpark	L12	-3.1	$[-5.0, -1.3]$	M5	L0	$+2.9$	$[2.0, 4.0]$
Coastal T-S	L5	-6.3	$[-13.8, 2.0]$	Wiki Daily	L0	$+6.2$	$[3.9, 9.0]$
BOOM	L0	-4.4	$[-9.2, -0.1]$	Monash Traffic	L12	$+1.9$	$[-0.9, 4.9]$
TiRex							
M4	L12	0.0	$[0.0, 0.0]$	LOOP Seattle	L9	-1.1	$[-3.5, 1.5]$
Electricity	L11	$+4.5$	$[2.9, 6.0]$	Electricity (15-minute)	L10	$+2.8$	$[1.5, 4.1]$
Uber TLC	L10	$+1.4$	$[0.9, 1.9]$	SZ Taxi	L0	$+4.8$	$[3.1, 6.6]$
Wind Farms	L11	$+0.7$	$[0.2, 3.6]$	London Smart Meters	L8	$+5.2$	$[4.1, 6.3]$
SG Carpark	L11	-1.8	$[-3.5, -0.2]$	M5	L0	$+3.9$	$[3.0, 4.8]$
Coastal T-S	L1	-0.7	$[-8.0, 7.0]$	Wiki Daily	L1	$+5.5$	$[3.5, 7.7]$
BOOM	L8	$+4.7$	$[1.6, 7.7]$	Monash Traffic	L12	0.0	$[0.0, 0.0]$

Table 9: $Q = 1$ **versus** $Q = 9$ **sustained 5% tunnel entrances.** Each cell gives the difference in entrance depth, $\Delta\ell = \ell_{Q=1} - \ell_{Q=9}$, with the $Q = 1$ entrance in parentheses.

Dataset	Chronos-2	TFM-3	TiRex	Dataset	Chronos-2	TFM-3	TiRex
M4	-3 (L5)	-2 (L14)	0 (L12)	LOOP Seattle	0 (L9)	-4 (L10)	0 (L9)
Electricity	$+1$ (L10)	0 (L15)	0 (L11)	Electricity (15-minute)	0 (L10)	0 (L1)	0 (L10)
Uber TLC	$+1$ (L5)	0 (L1)	0 (L10)	SZ Taxi	0 (L2)	0 (L1)	0 (L0)
Wind Farms	0 (L3)	0 (L0)	0 (L11)	London Smart Meters	$+1$ (L4)	0 (L1)	0 (L8)
SG Carpark	$+2$ (L12)	-1 (L11)	0 (L11)	M5	0 (L0)	0 (L0)	0 (L0)
Coastal T-S	0 (L1)	$+13$ (L18)	0 (L1)	Wiki Daily	0 (L2)	0 (L0)	0 (L1)
BOOM	-1 (L2)	0 (L0)	$+1$ (L9)	Monash Traffic	0 (L11)	0 (L12)	0 (L12)
Exact match	**8/14**	**10/14**	**13/14**	$\lvert\Delta\ell\rvert \leq 1$	**12/14**	**11/14**	**14/14**

Table 10: **Sensitivity of the sustained tunnel entrance to ϵ.** Each entry gives the selected entrance for $\epsilon = 0.02$, 0.05, and 0.08. Median depth is the median of ℓ/L over the 14 datasets.

Dataset	2%	5%	8%	Dataset	2%	5%	8%
Chronos-2							
M4	L11	L8	L5	LOOP Seattle	L12	L9	L8
Electricity	L11	L9	L5	Electricity (15-minute)	L11	L10	L9
Uber TLC	L5	L4	L3	SZ Taxi	L4	L2	L1
Wind Farms	L11	L3	L3	London Smart Meters	L5	L3	L3
SG Carpark	L11	L10	L10	M5	L1	L0	L0
Coastal T-S	L3	L1	L1	Wiki Daily	L3	L2	L1
BOOM	L4	L3	L2	Monash Traffic	L12	L11	L11
Median depth	**67%**	**29%**	**25%**	**Before final**	**12/14**	**14/14**	**14/14**
TimesFM-3							
M4	L20	L16	L13	LOOP Seattle	L17	L14	L1
Electricity	L16	L15	L15	Electricity (15-minute)	L11	L1	L1
Uber TLC	L13	L1	L1	SZ Taxi	L5	L1	L0
Wind Farms	L15	L0	L0	London Smart Meters	L1	L1	L1
SG Carpark	L13	L12	L4	M5	L1	L0	L0
Coastal T-S	L18	L5	L4	Wiki Daily	L2	L0	L0
BOOM	L0	L0	L0	Monash Traffic	L15	L12	L11
Median depth	**65%**	**5%**	**5%**	**Before final**	**13/14**	**14/14**	**14/14**
TiRex							
M4	L12	L12	L11	LOOP Seattle	L10	L9	L8
Electricity	L11	L11	L11	Electricity (15-minute)	L10	L10	L9
Uber TLC	L11	L10	L8	SZ Taxi	L7	L0	L0
Wind Farms	L11	L11	L8	London Smart Meters	L11	L8	L2
SG Carpark	L12	L11	L10	M5	L1	L0	L0
Coastal T-S	L4	L1	L1	Wiki Daily	L8	L1	L0
BOOM	L11	L8	L1	Monash Traffic	L12	L12	L11
Median depth	**92%**	**79%**	**67%**	**Before final**	**11/14**	**12/14**	**14/14**

Table 11: **Forecast recoverability versus representation geometry.** Counts of model-dataset pairs. Entry-step statistics exclude the seven pairs whose tunnel begins at L0. Chance denotes the expected count under a uniformly placed landmark independent of the entrance. The 2% columns repeat the analysis with the entrances of the stricter 2% tolerance; entry-step statistics then exclude the one pair whose tunnel begins at L0.

	5% tunnel		2% tunnel	
Observation	Pairs	Chance	Pairs	Chance
Entry step is the largest adjacent CKA change	3/35	2.6	4/41	3.0
Entry step is the largest adjacent effective-rank change	2/35	2.6	2/41	3.0
Effective-rank peak is near the entrance	5/35	7.8	10/41	9.0
Dominant CKA boundary is near the entrance	8/35	7.8	12/41	9.0
Sharpest adjacent CKA change lies inside the tunnel	34/42	24.4	30/42	16.1
Sharpest adjacent CKA change is the step into the final block	25/42		25/42	
Effective-rank peak lies at or after the entrance	29/42	24.5	23/42	16.9
At least half of the effective-rank variation lies inside the tunnel	28/42		20/42	
CKA between the entrance and final block is below 0.5	23/42		16/42	

Table 12: **Affine versus nonlinear adapter at 25, 50 and 75% of network depth.** Median test MASE increase over the full model across all 14 datasets, with the number of datasets within 5% in parentheses. †Iterative fit; on Electricity it does not reproduce the full model's forecasts even on the training windows (training $R^2 < 0$).

Model	Depth	Affine	Nonlinear
Chronos-2	L3 (25%)	+4.4% (7)	+3.6% (8)
	L6 (50%)	+2.3% (9)	+2.2% (9)
	L9 (75%)	+2.2% (10)	+1.6% (11)
TimesFM-3	L5 (25%)	+69.8% (0)	7.1×† (0)
	L10 (50%)	+67.3% (0)	14.5×† (0)
	L15 (75%)	+47.9% (0)	20.5×† (0)
TiRex	L3 (25%)	+13.4% (1)	+15.3% (2)
	L6 (50%)	+10.6% (3)	+8.7% (5)
	L9 (75%)	+6.2% (5)	+6.2% (5)

Table 13: **Performance-compute frontier of aligned truncation, all depths.** "Lk" keeps blocks $1, \ldots, k$ and removes the rest; L0 keeps no block. MASE increase is the median over the 14 datasets of test MASE relative to the full model, with its interquartile range. Speedups are end to end, from a context in host memory to forecasts returned to host memory, relative to the full model at the same batch size: batch size 1 (B=1) measures single-forecast latency and is the headline setting; batch size 256 (B=256) measures throughput. Chronos-2-small is a separately pretrained model, not a truncation of Chronos-2.

Model	Cut	Test MASE increase		Speedup	
		Median	IQR	B=1	B=256
Chronos-2	L12 (full model)	0.0%		1.00×	1.00×
	L11	+0.4%	[0.0%, +1.2%]	1.08×	1.08×
	L10	+0.9%	[+0.2%, +3.3%]	1.18×	1.17×
	L9	+2.2%	[+0.9%, +6.2%]	1.28×	1.29×
	L8	+3.1%	[+2.0%, +8.2%]	1.42×	1.43×
	L7	+3.1%	[+0.8%, +9.6%]	1.57×	1.60×
	L6	+2.3%	[+0.4%, +11.5%]	1.78×	1.82×
	L5	+2.1%	[+0.5%, +12.1%]	2.06×	2.11×
	L4	+3.1%	[+0.5%, +11.4%]	2.43×	2.51×
	L3	+4.4%	[+0.7%, +11.3%]	2.91×	3.10×
	L2	+8.4%	[+2.1%, +30.0%]	3.72×	4.07×
	L1	+13.1%	[+6.8%, +51.9%]	5.19×	5.88×
	L0	+69.4%	[+19.9%, +183.8%]	8.41×	9.93×
Chronos-2-small	separate model	+0.8%	[0.0%, +5.9%]	1.80×	3.28×
TimesFM-3	L20 (full model)	0.0%		1.00×	1.00×
	L19	+5.4%	[+2.9%, +14.0%]	0.99×	1.04×
	L18	+14.3%	[+6.8%, +36.7%]	1.03×	1.09×
	L17	+29.7%	[+13.4%, +69.7%]	1.07×	1.15×
	L16	+40.3%	[+15.9%, +88.3%]	1.05×	1.20×
	L15	+47.9%	[+30.1%, +105.8%]	1.08×	1.26×
	L14	+56.0%	[+36.9%, +105.7%]	1.11×	1.34×
	L13	+56.4%	[+37.4%, +121.8%]	1.11×	1.42×
	L12	+62.7%	[+42.3%, +133.3%]	1.15×	1.51×
	L11	+59.5%	[+45.5%, +165.5%]	1.12×	1.60×
	L10	+67.3%	[+41.0%, +163.7%]	1.19×	1.71×
	L9	+69.9%	[+39.1%, +155.7%]	1.18×	1.85×
	L8	+58.1%	[+43.2%, +161.4%]	1.25×	2.03×
	L7	+65.1%	[+40.6%, +154.0%]	1.22×	2.21×
	L6	+81.9%	[+42.9%, +203.6%]	1.24×	2.44×
	L5	+69.8%	[+45.8%, +186.4%]	1.24×	2.70×
	L4	+77.3%	[+45.2%, +174.0%]	1.29×	3.11×
	L3	+62.4%	[+37.4%, +140.7%]	1.34×	3.49×
	L2	+53.8%	[+40.0%, +102.8%]	1.43×	4.25×
	L1	+38.7%	[+25.9%, +65.8%]	1.37×	4.65×
	L0	+31.3%	[+19.3%, +62.5%]	1.38×	5.94×
TiRex	L12 (full model)	0.0%		1.00×	1.00×
	L11	+1.8%	[+0.8%, +5.3%]	1.09×	1.09×
	L10	+4.3%	[+2.0%, +6.2%]	1.21×	1.19×
	L9	+6.2%	[+3.4%, +10.4%]	1.33×	1.32×
	L8	+8.7%	[+4.8%, +12.8%]	1.50×	1.48×
	L7	+8.5%	[+6.7%, +13.2%]	1.71×	1.69×
	L6	+10.6%	[+5.7%, +14.7%]	1.99×	1.96×
	L5	+11.3%	[+6.6%, +20.1%]	2.39×	2.33×
	L4	+12.8%	[+7.4%, +28.2%]	2.99×	2.88×
	L3	+13.4%	[+8.2%, +33.8%]	3.93×	3.76×
	L2	+10.5%	[+7.9%, +36.1%]	5.95×	5.43×
	L1	+14.9%	[+8.1%, +44.5%]	11.7×	9.83×
	L0	+46.2%	[+11.9%, +109.2%]	340×	50.2×

Table 14: **Aligned truncation inside versus before the forecasting tunnel.** Median test MASE increase over the full model, at the same cut depth, for datasets where that depth lies inside their validation-selected 5% tunnel and for datasets where it lies before the tunnel entrance. n gives the number of datasets in each group.

Model	Cut	n (inside / before)	Inside tunnel	Before tunnel
Chronos-2	L1	2 / 12	+4.8%	+21.5%
	L2	4 / 10	+1.8%	+15.1%
	L3	7 / 7	+2.6%	+9.3%
	L4	8 / 6	+2.2%	+14.2%
	L5	8 / 6	+1.9%	+13.8%
	L6	8 / 6	+1.9%	+13.1%
	L7	8 / 6	+1.8%	+11.8%
	L8	9 / 5	+2.4%	+8.3%
	L9	11 / 3	+1.8%	+4.5%
TimesFM-3	L0	4 / 10	+20.2%	+34.2%
	L1	8 / 6	+38.7%	+47.1%
	L2	8 / 6	+50.3%	+73.8%
	L3	8 / 6	+58.6%	+79.1%
	L4	8 / 6	+67.9%	+92.7%
	L5	9 / 5	+46.4%	+114.1%
	L6	9 / 5	+43.4%	+119.5%
	L7	9 / 5	+42.0%	+124.1%
	L8	9 / 5	+43.5%	+108.9%
	L9	9 / 5	+45.7%	+122.4%
	L10	9 / 5	+42.9%	+116.7%
	L11	9 / 5	+46.5%	+121.3%
	L12	11 / 3	+53.9%	+71.5%
	L13	11 / 3	+59.2%	+53.6%
	L14	12 / 2	+52.5%	+82.6%
TiRex	L0	2 / 12	+8.0%	+53.3%
	L1	4 / 10	+6.8%	+25.5%
	L2	4 / 10	+6.5%	+24.0%
	L3	4 / 10	+6.7%	+23.3%
	L4	4 / 10	+7.7%	+17.3%
	L5	4 / 10	+5.8%	+13.7%
	L6	4 / 10	+4.4%	+12.3%
	L7	4 / 10	+4.6%	+11.7%
	L8	6 / 8	+4.9%	+10.0%
	L9	7 / 7	+2.9%	+8.3%
	L10	9 / 5	+2.2%	+9.8%
	L11	12 / 2	+1.5%	+8.2%