COIN: COORDINATING IMPUTATION AND ADAPTATION FOR FINANCIAL FORECASTING WITH TIME-SERIES FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Missing history and changing statistical properties jointly affect financial forecasting with pretrained time-series foundation models (TSFMs), while reconstruction-based imputation selection need not improve future predictions. We introduce COIN, a hierarchical decision method that coordinates imputation candidates and adaptation using only subsequently observed forecasting targets. Eight lightweight imputations produce separate model forecasts; the data layer combines these forecasts according to past predictive losses, and the adaptation layer selects among no adaptation, residual correction, and two strengths of LoRA output. Backbone parameters remain frozen, and LoRA is trained once per scenario; only weights, residual states, and scores change during rolling prediction. Experiments cover 11 financial data sources and four TSFMs. On a development set with three initial missingness rates, COIN reduces mean scaled absolute error by 2.87%–3.67% relative to pipelines using four pretrained imputation methods. With configurations frozen on 22 additional series, average error decreases for all four backbones relative to an STL–Kalman pipeline with matched LoRA capability. Further diagnostics support forecast-based candidate evaluation and a mean–tail tradeoff from anchoring. Development-selected pruning reduces rolling latency by 38.9% on eight workloads, with slightly higher tail errors. Gains remain dependent on gap location and timely feedback.

1 INTRODUCTION

Time-series foundation models (TSFMs), including TimesFM, Chronos, Kairos, and Sundial, reduce the need to train a new forecasting model for every application (Das et al., 2024; Ansari et al., 2024; 2025; Feng et al., 2025; Liu et al., 2025). Financial applications nevertheless present two interacting difficulties: historical records contain gaps, and the relationship between past observations and future targets changes over time. A deployed forecaster must issue predictions from incomplete history and can revise its decisions only after new observations arrive. We study this rolling setting for numerical observation gaps on an already defined trading or publication calendar. Market closures, non-trading states, and values not yet released are not automatically treated as observations to impute.

A single imputation assumption is unlikely to suit prices, trading volumes, interest rates, and macroeconomic indicators across frequencies. Financial series exhibit heterogeneous periodicity, heavy tails, volatility clustering, and regime changes (Cont, 2001; Engle, 1982; Bollerslev, 1986; Ang & Timmermann, 2012; Andersen & Bollerslev, 1997). Simple imputers are inexpensive, whereas trained imputers can represent richer temporal dependencies (Moritz & Bartz-Beielstein, 2017; Cao et al., 2018; Du et al., 2023; Tashiro et al., 2021; Cheng et al., 2024). However, accurately reconstructing missing history does not ensure accurate future forecasts. Task-oriented imputation explicitly recognizes this mismatch (Wang et al., 2024); for a frozen TSFM, candidate usefulness also depends on how that particular model responds to a completed history. On confirmation data, reconstruction and first-window forecast rankings correlate only 0.251, with disjoint best sets in 81.439% of scenarios. Over 20 windows, filling artificial gaps with truth ranks best against the eight imputations in only 16.288% of scenarios (Appendix D.6).

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.

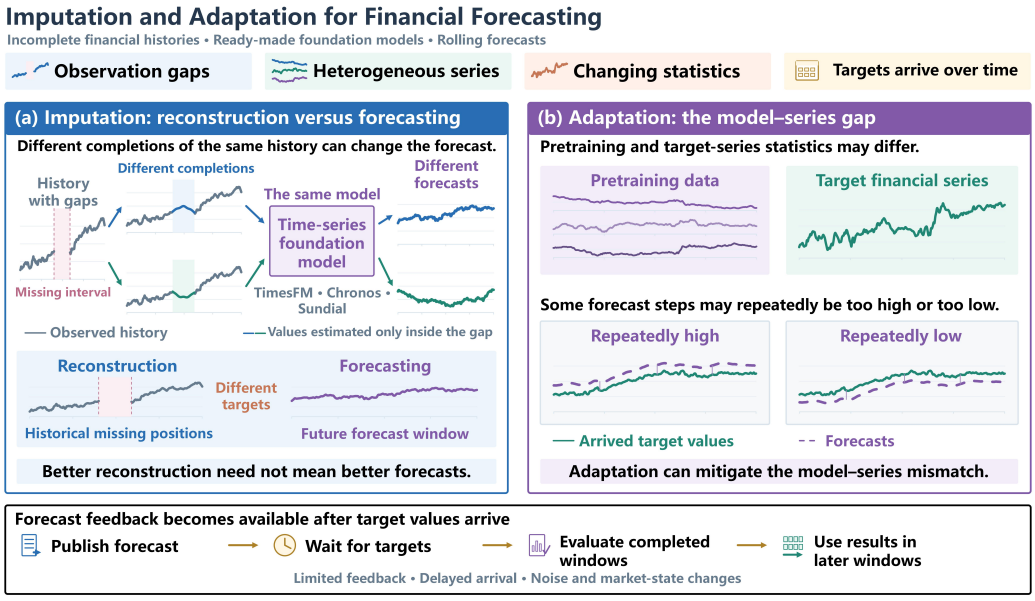

Figure 1: Financial forecasting with incomplete history. (a) Different completions can produce different forecasts from the same backbone; reconstruction and forecasting evaluate different target positions. (b) Model–series mismatch can cause repeated forecast bias. Arriving targets provide feedback only for subsequent windows. Trajectories are illustrative.

Adaptation is coupled to imputation because changing the completed input changes both the backbone prediction and the usefulness of any correction. Normalization, non-stationarity handling, and parameter-efficient adaptation address aspects of distribution mismatch (Kim et al., 2022; Liu et al., 2022; Hu et al., 2022). Methods such as TAFAS and COSA exploit subsequently observed targets, while DoubleAdapt studies incremental adaptation for stock prediction (Zhao et al., 2023; Kim et al., 2025; Im & Kwon, 2026). These developments motivate a further question: *without access to the true values of historical gaps, how can arriving prediction feedback coordinate useful input completions and subsequent adaptation?*

COIN first combines forecasts from eight lightweight completions using past predictive losses, then selects one of five adaptation branches. Base and LoRA models share data weights; backbone parameters remain frozen, and LoRA is trained once on initial history. Online updates affect only low-dimensional states, and adapted losses never update data weights. Thus, "joint decision" denotes hierarchical coordination toward a forecasting objective, rather than global optimization of missing values and model parameters.

Our contributions are: (i) separating forecast-output weighting from adaptation choice, with shared weights and scenario-level fixed LoRA; (ii) evaluating both decisions on arriving targets without historical missing-value labels; and (iii) experiments separating pipeline performance, matched adaptation, component effects, feedback dependence, and cost. Applicability requires arriving feedback, persistence in relative performance, and resources for multiple evaluations, rather than finance-specific mechanisms. Appendix A expands the motivation and related-work distinctions.

2 Related Work

Imputation for a downstream forecaster. Statistical imputers encode local, level, seasonal, or state-space assumptions (Moritz & Bartz-Beielstein, 2017). BRITS, SAITS, and CSDI learn temporal completion patterns (Cao et al., 2018; Du et al., 2023; Tashiro et al., 2021); NuwaTS, MOMENT, Timer, and Moirai offer pretrained capabilities (Cheng et al., 2024; Goswami et al., 2024; Liu et al., 2024; Woo et al., 2024). GRU-D instead incorporates missingness into a trained predictor (Che et al., 2018). The closest task-oriented study, Wang et al. (2024), estimates the downstream utility

of imputing missing training labels and combines imputed values before training a predictor. COIN addresses missing *forecast-time history*, evaluates candidates online using arriving targets, and combines *forecast outputs* of a frozen backbone. These differences concern the missing object, feedback timing, combination location, and parameter updates, rather than whether downstream utility matters.

Adaptation with observed targets. RevIN and Non-stationary Transformers are trained components for handling distribution shift (Kim et al., 2022; Liu et al., 2022). LoRA introduces trainable low-rank increments (Hu et al., 2022), while Tent's entropy objective primarily targets classification (Wang et al., 2021). TAFAS calibrates a frozen forecaster at its input and output, and COSA learns a context-aware output adapter from arriving targets (Kim et al., 2025; Im & Kwon, 2026). DoubleAdapt jointly adapts data and models through meta-learning for stock trends (Zhao et al., 2023). COIN applies adaptation to a combination of imputation-conditioned forecasts and selects whether to intervene. Its residual states are smoothed errors, not trained linear layers, and its LoRA parameters receive no online gradient updates.

Online expert learning. Weighted Majority and Hedge motivate feedback-based expert weighting (Littlestone & Warmuth, 1994; Freund & Schapire, 1997); Fixed Share and adaptive-regret methods address changing comparators (Herbster & Warmuth, 1998; Daniely et al., 2015; Adamskiy et al., 2016). COIN uses full-information feedback because every candidate prediction can be evaluated when targets arrive. Its discounted, clipped, anchor-relative losses and anchor floor differ from the standard algorithms, so their regret guarantees are not claimed here. The adaptation selector forms a separate decision layer.

3 Problem definition and framework

Let x_t denote a value and $m_t \in \{0, 1\}$ its observation mask on a task-specific calendar. At origin τ_k, the forecaster reads L historical positions and predicts H future positions:

$$C_k = (x_{\tau_k - L+1}, \ldots, x_{\tau_k}), \qquad y_k = (x_{\tau_k+1}, \ldots, x_{\tau_k+H}). \tag{1}$$

The information set $\mathcal{I}_k$ contains only observations, masks, targets, and preparation results available by τ_k. An observed value to the right of an internal historical gap may be used if it has already arrived; no future target or hidden historical truth may be used. A *scenario* is one complete rolling run for a series, backbone, and missingness condition. Experiments use $K = 20$ non-overlapping windows, with successive origins separated by H.

The task objective is to reduce $K^{-1} \sum_{k=1}^{K} \ell(\pi(\mathcal{I}_k), y_k)$ for a policy π. This objective is not directly available when forecasts are issued. The internal decision loss combines scaled errors:

$$\ell(p, y) = \kappa_1 \frac{\mathrm{MAE}(p, y)}{s} + \kappa_2 \frac{\mathrm{RMSE}(p, y)}{s} + \kappa_3 \, \mathrm{sAPE}_{\varepsilon}(p, y), \tag{2}$$

where s is the observed initial-history seasonal-naive MAE and $(\kappa_1, \kappa_2, \kappa_3) = (0.45, 0.30, 0.25)$. The percentage denominator is floored at $0.05s$. Unobserved targets are excluded. Appendix F specifies all constants; evaluation metrics are distinct from this loss.

For imputer ϕ_j, let $\widehat{C}_{k,j} = \phi_j(C_k, M_k)$ denote its completed candidate history, with observed values unchanged. After initialization, candidate construction retains the initial fills and appends newly available observations. Each candidate is evaluated by the frozen backbone f_θ and by $f_{\theta+\Delta}$, where the LoRA increment Δ is trained once before rolling prediction:

$$p_{k,j}^{\mathrm{base}} = f_\theta(\widehat{C}_{k,j}), \qquad p_{k,j}^{\mathrm{lora}} = f_{\theta+\Delta}(\widehat{C}_{k,j}). \tag{3}$$

The expert set contains eight imputers and an additional anchor formed from existing outputs, giving $J = 9$. Nonnegative shared weights summing to one define

$$P_k = \sum_{j=0}^{J-1} w_{k,j} p_{k,j}^{\mathrm{base}}, \qquad Q_k = \sum_{j=0}^{J-1} w_{k,j} p_{k,j}^{\mathrm{lora}}. \tag{4}$$

The adaptation layer selects a single output branch based on P_k and Q_k. Data weights determine the mixtures on which adaptation acts. Both layers use arriving targets but maintain separate scores and comparators (Figure 2; Appendix B).

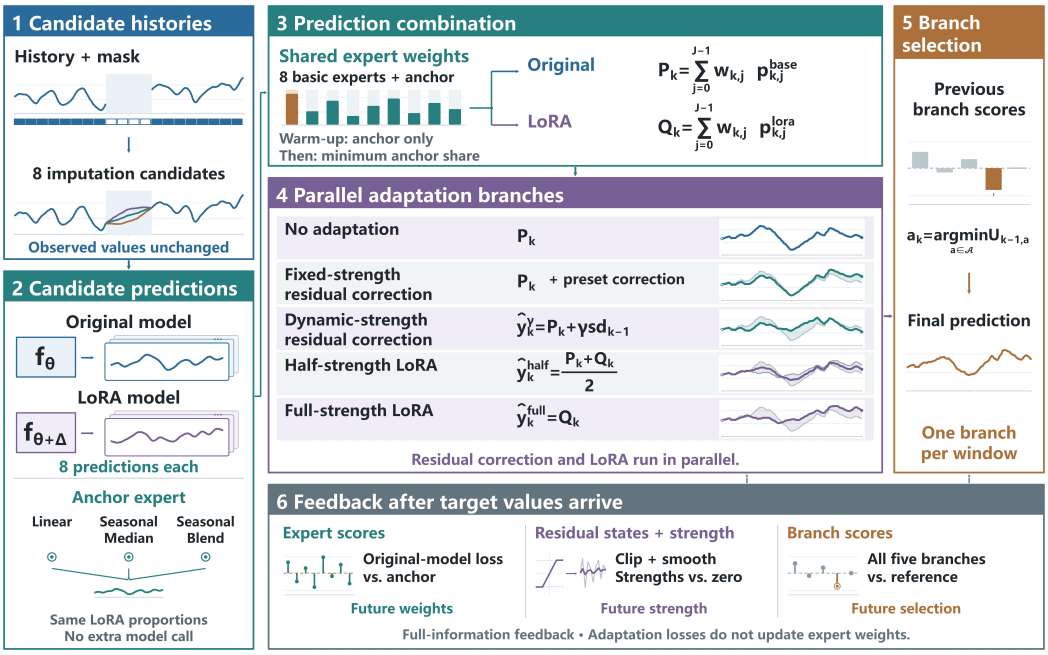

Figure 2: COIN rolling prediction and feedback. Steps 1–3 generate imputation-conditioned base/LoRA forecasts and combine them with shared weights; steps 4–5 select one of five parallel outputs. Step 6 updates states after target arrival. Base parameters stay frozen, LoRA is trained once before deployment, and adapted losses never update data weights.

Two assumptions govern applicability: targets eventually arrive with known availability times, and recent forecasting errors retain some information about subsequent windows. Abrupt regime shifts can invalidate the second assumption. Publication delays and revisions require real-time information accounting (Ang & Timmermann, 2012; Croushore & Stark, 2001); our feedback-delay experiment tests only part of this problem.

4 COIN

4.1 Candidate construction and anchor

The eight imputers are Linear, Mean, Median, Forward, Backward, Nearest, Seasonal Median, and Seasonal Blend, covering local, level, and seasonal assumptions. Only missing positions are filled; original observations and masks are retained. Rolling histories append arriving observations while initial fills stay fixed. All 20 confirmation windows retain initial gaps in context; Section 5.5 separately examines ongoing gaps.

The anchor combines the Linear (lin), Seasonal Median (sm), and Seasonal Blend (sb) predictions:

$$p_{k,0}^{\text{base}} = \xi p_{k,\text{lin}}^{\text{base}} + (1-\xi)\frac{p_{k,\text{sm}}^{\text{base}} + p_{k,\text{sb}}^{\text{base}}}{2}. \tag{5}$$

The LoRA anchor shares ξ. Retrospective initial-history validation selects ξ, falling back to Linear when mixing is unsupported. Candidates are completed before slicing historical validation inputs, so these inputs may use later initial-history observations, including validation targets; final test targets remain excluded. LoRA instead recomputes each completion from its training prefix. Appendix F specifies these distinct preparation boundaries. The anchor adds no rolling model call.

4.2 FORECAST COMBINATION

COIN retains each completion as an input hypothesis. Output weighting differs from averaging histories before a nonlinear backbone, and smooths candidate switching at the cost of retaining weaker forecasts and requiring separate inference.

When targets arrive, all base forecasts are scored, including experts with low current weights. With $\ell_{k,j} = \ell(p_{k,j}^{\text{base}}, y_k)$, the relative loss and its discounted accumulation are

$$r_{k,j} = \text{clip}\left(\frac{\ell_{k,j} - \ell_{k,0}}{\max(\ell_{k,0}, \delta)}, -c, c\right), \qquad R_{k,j} = \lambda R_{k-1,j} + r_{k,j}. \tag{6}$$

Here $\delta > 0$ protects the denominator, c clips extreme relative errors, and λ discounts older evidence. Negative relative loss indicates improvement over the anchor. Before window k, prior shares μ_j and sensitivity η yield

$$q_{k,j} = \frac{\mu_j e^{-\eta R_{k-1,j}}}{\sum_{i=0}^{J-1} \mu_i e^{-\eta R_{k-1,i}}}, \qquad w_{k,j} = (1-\rho) q_{k,j} + \rho \mathbf{1}\{j = 0\}. \tag{7}$$

Warm-up uses the anchor alone; subsequently, ρ reserves its minimum share. With delayed feedback, the score state includes only losses already received, rather than all preceding windows. Relative scaling, clipping, and the anchor floor limit the effect of extreme or sparse observations, but may slow useful adaptation.

Base and LoRA models share weights scored from base forecasts only. This keeps P_k and Q_k comparable and reduces state, while forgoing LoRA-specific weighting. Adapted losses never update $R_{k,j}$.

4.3 ADAPTATION BRANCHES

Five parallel branches produce (i) P_k, (ii) fixed-strength residual correction, (iii) dynamic-strength residual correction, (iv) $(P_k + Q_k)/2$, or (v) Q_k. They are alternatives, not sequential corrections: LoRA outputs are not additionally residual-corrected. Half strength averages *outputs*; it does not halve the LoRA parameter increment.

For dynamic correction, the horizon-specific residual state is updated after target arrival:

$$e_k = \text{clip}\left(\frac{y_k - P_k}{s}, -c_e, c_e\right), \qquad d_k = \beta d_{k-1} + (1-\beta) e_k. \tag{8}$$

The first nonempty feedback initializes $d = e$; later updates follow Eq. 8. Missing coordinates have zero innovation; empty windows skip updates. For a finite strength set $\mathcal{G}$ containing zero, candidate outputs are $\widehat{y}_k^{\gamma} = P_k + \gamma s d_{k-1}$. The strength is selected using cumulative relative loss against zero correction. All strengths can be evaluated without another model call. The fixed-strength branch maintains a separate smoothed, normalized, clipped residual using its *own corrected output* as the error reference, and applies a preset shrinkage with an enable/disable decision based on recent performance. The dynamic branch instead measures the bias of P_k itself.

LoRA trains low-rank factors in $W' = W + (\alpha_{\text{lora}}/r_{\text{lora}})BA$ (Hu et al., 2022), rotating completions on initial history. The increment stays fixed thereafter; residual updates and branch selection require no online backpropagation.

Each branch $a \in \mathcal{A}$ receives a discounted cumulative relative-loss score $U_{k,a}$ against a predetermined reference branch. Selection uses only available past scores:

$$a_k = \arg\min_{a \in \mathcal{A}} U_{k-1,a}, \qquad \widehat{y}_k = \widehat{y}_k^{a_k}. \tag{9}$$

The reference branch resolves insufficient feedback and ties. All branches are evaluated when targets arrive, so this is full-information selection rather than bandit feedback. Hard selection preserves an identifiable intervention but cannot exploit arbitrary mixtures of adaptation outputs.

4.4 FEEDBACK, SAFEGUARDS, AND COMPUTATION

Prediction precedes target arrival and updates. Expert, strength, and branch scores have distinct comparators (Appendix B). Warm-up, clipping, conservative correction, and fallback stabilize limited feedback without guaranteeing per-window improvement.

Full COIN evaluates eight histories through both model states (16 evaluations, potentially batched). Anchors reuse outputs; unselected LoRA outputs still incur cost. Preparation occurs once. "Lightweight" refers to imputers and state updates, not the entire pipeline. Models reset between scenarios.

5 EXPERIMENTS

5.1 PROTOCOL AND EVALUATION

Data and rolling prediction. The development set contains 38 series from 11 financial sources, four backbones, and initial block-missingness rates of 10%, 20%, and 30%, yielding 456 scenarios. It supports configuration selection, broad comparisons, and feedback analyses. The confirmation set contains 22 additional series, two per source, selected by length, observed fraction, and a fixed hash order without using prediction errors. It uses frozen configurations, 20% initial missingness, and paired seeds 1701–1703, yielding 264 scenarios. Sources span minute-level markets to annual financial indicators (Appendix C).

Each scenario has 20 non-overlapping windows. Artificial initial gaps protect the final H historical positions; native gaps remain. Methods share masks, origins, targets, and arriving observations. Preparation excludes final test targets (Appendix C). Publication vintages are not reconstructed, and confirmation series may share assets with development series. Evaluation concerns within-source transfer, not strict asset independence or guaranteed exclusion from pretraining.

Backbones and controls. We use Chronos-2, Kairos-23M, Sundial, and TimesFM-2.5, truncating context to each model's limit. NuwaTS and MOMENT use imputation interfaces; Timer and Moirai use segmented extrapolation to complete history. Statistical controls are Linear, Mean, Forward, Kalman, STL–Kalman (STL below), and GP–RBF. All preserve observed values and exclude the forecast interval from imputation. The broad comparison measures these particular pipelines, with full adaptation included only in COIN.

Three-branch controls share no adaptation and both residual corrections; five-branch controls add half/full LoRA. Linear and STL train dedicated LoRA with COIN's training budget, origins, and target masks, using their own completed inputs. STL+COSA uses the official adapter with full-window feedback and masked loss. Appendix C specifies training settings, seasonality, and numerical fallbacks.

Metrics and inference. The broad comparison reports MASE, sMAPE (%), and the within-scenario MAE rank among its 11 pipelines, averaged equally over development scenarios. MASE scales error by the seasonal naive error of observed initial history (Hyndman & Koehler, 2006). Confirmation instead uses

$$\mathrm{rMAE}_u = 100 \frac{\mathrm{MAE}_u}{\max(\mathrm{MAE}_{\mathrm{Linear}}, 10^{-8}\sigma_0)}, \tag{10}$$

where MAE averages window MAEs over evaluable targets and σ_0 is the lower-bounded standard deviation of observed initial history. No confirmation scenario triggers the denominator floor. We average seeds and backbones within each series, series within each source, and then weight the 11 sources equally. Relative MSE (rMSE) is auxiliary. Development feedback analyses instead use an equal-weight composite of MSE, MASE, and sMAPE relative to Linear; these measures should not be conflated.

Two prespecified comparisons are full COIN versus STL+dedicated LoRA, and COIN without LoRA versus STL+three branches. Positive paired differences mean reference minus COIN. We use 6,000 paired source-cluster bootstrap resamples for 95% intervals, and exact two-sided sign-flip tests on 11 source-level differences with Holm adjustment. Intervals are not multiplicity-adjusted; same-source series and seeds are not independent sources (Diebold & Mariano, 1995; Demšar, 2006; Cameron & Miller, 2015). Other comparisons are descriptive.

Table 1: Development-set downstream performance (456 scenarios, equally weighted). Rows are ordered by increasing MASE. Ranks compare MAE among these 11 pipelines. Shading identifies COIN; bold marks the lowest mean, not significance.

Pipeline	MASE	sMAPE (%)	Mean rank
COIN	**4.516**	**18.802**	**5.151**
STL–Kalman	4.605	19.029	5.446
Linear	4.626	19.059	5.795
GP–RBF	4.632	18.944	6.129
MOMENT	4.649	18.945	5.739
Moirai	4.654	19.062	6.053
Forward	4.655	19.122	6.175
NuwaTS	4.682	19.032	5.961
Timer	4.688	19.085	6.208
Kalman	4.689	19.146	6.289
Mean	5.203	19.565	7.053

5.2 Comparison with existing imputation pipelines

Table 1 compares full COIN with ten existing imputation pipelines on identical downstream backbones and targets. COIN has the lowest mean MASE, sMAPE, and rank. Its mean MASE is 2.87%–3.67% lower than those of the four pretrained imputation pipelines. These are complete-pipeline differences, combining candidate weighting and adaptation; they cannot be attributed to the imputer alone. The development data also informed configuration choices.

With common three-branch adaptation on development data, COIN without LoRA achieves MASE 4.591 versus 4.631–4.668 for the four pretrained pipelines and 4.587 for STL. Thus, average gains depend on the reference and metric, and are not uniformly larger than a strong statistical imputer. Frozen confirmation comparisons provide a separate assessment.

5.3 Matched adaptation and component effects

Table 2: Frozen-configuration confirmation and ablations (264 scenarios; sources equally weighted). Each block is ordered by increasing rMAE (Linear = 100). Dedicated LoRA is imputer-specific; shared LoRA reuses COIN's increment. Shading identifies COIN; bold marks the lowest mean, not significance.

Pipeline	rMAE	rMSE
Linear + dedicated LoRA	99.366	99.084
STL + dedicated LoRA	99.602	99.445
Linear + three branches	99.742	99.662
STL + three branches	99.987	100.098
STL + COSA	100.058	100.102
NuwaTS + three branches	102.644	116.221
COIN	**98.478**	**97.577**
Nine equal experts (original start)	98.864	98.162
COIN without LoRA	98.929	98.184
Data combination only	99.089	98.399
Linear + shared LoRA	99.300	99.074

Table 2 aligns adaptation capabilities and isolates components. Full COIN reduces rMAE by 1.124 points relative to STL+dedicated LoRA, with a 95% interval of $[0.174, 2.356]$. Without LoRA, the reduction versus STL+three branches is 1.058 points, with interval $[0.197, 2.127]$. Both comparisons improve the mean for each of the four backbones and for 16/22 series. However, both Holm-adjusted tests give $p = 0.0684$, above the prespecified 0.05 threshold. The evidence supports an average improvement direction, not consistent superiority across scenarios.

Squared error has a similar direction, but COIN's sMAPE is slightly higher (30.674 vs. 30.635). Relative to Linear+dedicated LoRA, the descriptive rMAE gain is 0.888 points ($[0.197, 1.751]$). Figure 4(a) shows both dedicated-LoRA comparisons by backbone; source sensitivity appears in Appendix D.8.

Ablations fix forecasts, LoRA state, and feedback order. COIN improves by 0.611 points over data combination alone and 0.822 over Linear+shared LoRA. Removing LoRA costs 0.451 points ($[0.047, 1.105]$). Dynamic versus equal weighting gains 0.386 points, but its interval $[-0.129, 0.987]$ crosses zero: ensembling itself contributes (Appendix D).

Matched-warm-up diagnostics reveal a mean–tail tradeoff (Figure 4(b)). Exponential weighting has lower mean rMAE than COIN (98.245 vs. 98.478), but higher mean positive excess over Linear (0.730 vs. 0.397). The excess reduction is 0.333 points ($[0.095, 0.669]$), with Holm $p = 0.0439$ in a separate exploratory five-comparison family. Appendix D.7 retains all mean/tail alternatives and tests; these are forecasting-error diagnostics, not trading-risk guarantees.

5.4 Continued updates and feedback delay

On development data, continued and frozen policy states share four warm-up windows and identical arriving history. Their subsequent 16-window composite scores are 98.365 and 99.037, a 0.673-point improvement with descriptive source-cluster interval $[0.021, 1.248]$. Only weights, residuals, and branch-state updates differ.

Delaying completed-window feedback by one or three windows increases the 20-window composite from 98.720 to 98.972 and 99.202. Figure 3 shows model-dependent, sometimes non-monotonic sensitivity. Only policy feedback is delayed; history arrives on schedule. This tests decision feedback, not the full effects of publication delays or revisions.

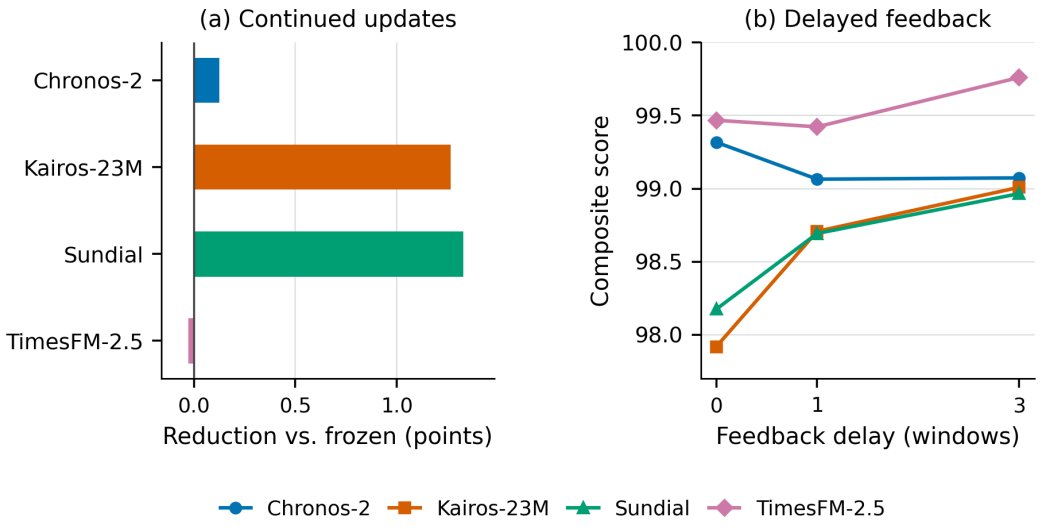

Figure 3: Development-set feedback analysis. Left: frozen minus continued-update score over the 16 post-warm-up windows (positive favors updating). Right: 20-window composite score under delayed policy feedback (lower is better). Each backbone aggregates 38 series and three missingness rates.

5.5 Ongoing missingness

Across eight sources (one series each), Chronos-2/Sundial, and three seeds, C0 retains native gaps; C1 adds 20% initial missingness; C2/C3 add 20% random-block/window-final missingness. New gaps share Linear fills; hidden targets enter neither history nor feedback. All use the matched C0 Linear denominator (Appendix C).

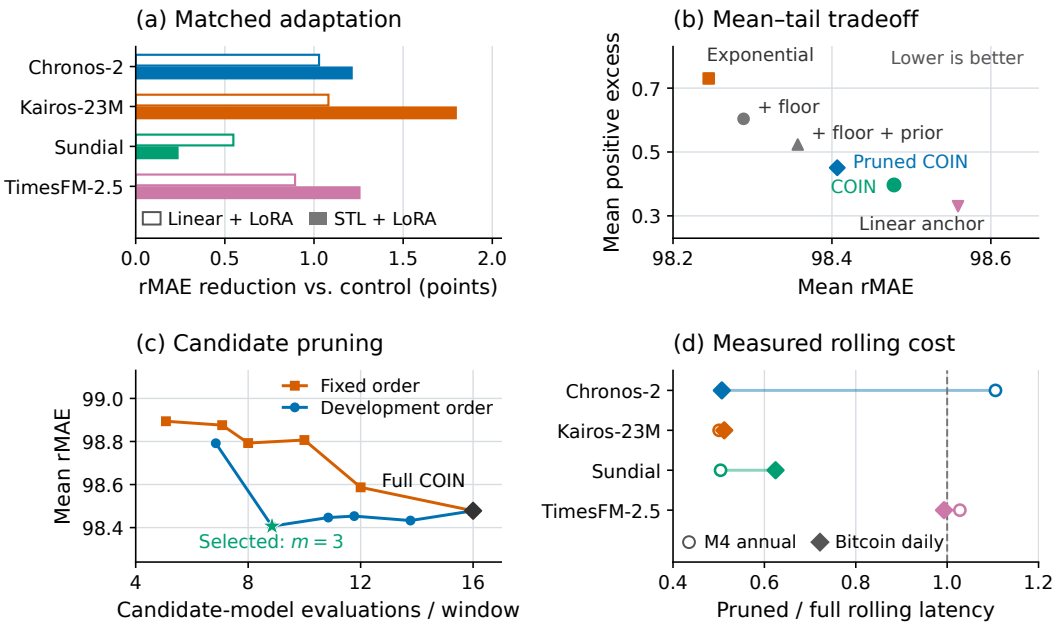

Figure 4: Confirmation and cost diagnostics. (a) Source-averaged rMAE reductions versus dedicated-LoRA controls. (b) All six mean–tail rules; excess is $\max(\text{rMAE} - 100, 0)$. (c) Complete pruning grids including anchor-support evaluations; the star marks development selection. (d) All eight workloads, three repeats; ratios below one mean faster rolling prediction. Panels (a)–(c) use 264 confirmation scenarios; (d) excludes preparation and training. Descriptive point estimates; intervals and tests: Tables 8, 16–17, and 23.

Relative to C1, C2 increases COIN's error by 3.094/0.903 points for Chronos-2/Sundial; C3 increases it by 26.323/18.998 (Appendix D, Figure 6). Recent gaps remove both input information and feedback. Paired intervals versus STL+dedicated LoRA cross zero in C2 and C3; this two-backbone study does not establish stable superiority under ongoing missingness.

5.6 End-to-end cost and limitations

Eight workloads on an RTX 4060 Laptop GPU (8 GiB) combine M4 annual/Bitcoin daily with four backbones (three repeats). End-to-end timing covers preparation and 20 windows, excluding loading; no caches are used. COIN averages 31.336 s versus 8.870 s for STL+dedicated LoRA (3.53-fold); without LoRA, 16.132 s. Appendix E reports all pipelines, initialization, and memory.

Development-selected pruning retains three candidates plus anchor support: 4.424 generated histories on average. A separate rolling-only pass reduces latency from 862.74 to 527.17 ms/window (38.9%), excluding preparation and training. Confirmation rMAE is 98.407 versus 98.478 ($[-0.229, 0.067]$ for the difference), but worst-three error rises by 0.262 points. Figure 4(c,d) shows the full grid and workload dependence; Appendix E.1 gives selection and uncertainty details.

6 Conclusion

COIN coordinates imputation-conditioned forecasts and adaptation using arriving targets. Matched comparisons support average error reductions; diagnostics expose the value of forecast-based evaluation, anchoring's mean–tail tradeoff, and pruning's accuracy–latency choice. Benefits depend on timely feedback, recent observations, and computational resources, without implying universal dominance or globally optimal joint learning.

AI USE STATEMENT

AI tools assisted with translation, editing, bibliographic checking, and LaTeX typesetting. Numerical results are reported from the experimental records.

REPRODUCIBILITY STATEMENT

Sections 3–4 define COIN. Appendices B–E document the procedure, experimental protocols, supplementary comparisons, and measured costs. Appendix F specifies the numerical settings, preparation boundaries, LoRA interfaces, randomization, and artifact verification. The accompanying supplement provides machine-readable configurations, sequence manifests and numerical segments, preparation records, source snapshots, and result tables.

REFERENCES

Dmitry Adamskiy, Wouter M. Koolen, Alexey Chernov, and Vladimir Vovk. A Closer Look at Adaptive Regret. *Journal of Machine Learning Research*, 17(23):1–21, 2016. URL https://jmlr.org/papers/v17/13-533.html.

Torben G. Andersen and Tim Bollerslev. Intraday periodicity and volatility persistence in financial markets. *Journal of Empirical Finance*, 4(2–3):115–158, 1997. doi: 10.1016/s0927-5398(97)00004-2. URL https://doi.org/10.1016/S0927-5398(97)00004-2.

Andrew Ang and Allan Timmermann. Regime Changes and Financial Markets. *Annual Review of Financial Economics*, 4(1):313–337, 2012. doi: 10.1146/annurev-financial-110311-101808. URL https://doi.org/10.1146/annurev-financial-110311-101808.

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. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=gerNCVqqtR.

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. arXiv:2510.15821, 2025. URL https://arxiv.org/abs/2510.15821v1. Version 1.

Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986. doi: 10.1016/0304-4076(86)90063-1. URL https://doi.org/10.1016/0304-4076(86)90063-1.

A. Colin Cameron and Douglas L. Miller. A Practitioner's Guide to Cluster-Robust Inference. *Journal of Human Resources*, 50(2):317–372, 2015. doi: 10.3368/jhr.50.2.317. URL https://doi.org/10.3368/jhr.50.2.317.

Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. BRITS: Bidirectional Recurrent Imputation for Time Series. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6776–6786, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/734e6bfcd358e25ac1db0a4241b95651-Abstract.html.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific Reports*, 8(1):6085, 2018. doi: 10.1038/s41598-018-24271-9. URL https://doi.org/10.1038/s41598-018-24271-9.

Jinguo Cheng, Chunwei Yang, Wanlin Cai, Yuxuan Liang, Qingsong Wen, and Yuankai Wu. NuwaTS: a Foundation Model Mending Every Incomplete Time Series. arXiv:2405.15317, 2024. URL https://arxiv.org/abs/2405.15317v3. Version 3, revised 2024-10-02.

Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6(1):3–73, 1990. URL https://www.scb.se/contentassets/ca21efb41fee47d293bbee5bf7be7fb3/stl-a-seasonal-trend-decomposition-procedure-based-on-loess.pdf.

R. Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1(2):223–236, 2001. doi: 10.1080/713665670. URL https://doi.org/10.1080/713665670.

Dean Croushore and Tom Stark. A real-time data set for macroeconomists. *Journal of Econometrics*, 105(1):111–130, 2001. doi: 10.1016/s0304-4076(01)00072-0. URL https://doi.org/10.1016/S0304-4076(01)00072-0.

Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly Adaptive Online Learning. In *Proceedings of the International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1405–1411. PMLR, 2015. URL https://proceedings.mlr.press/v37/daniely15.html.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167. PMLR, 2024. URL https://proceedings.mlr.press/v235/das24c.html.

Janez Demšar. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7:1–30, 2006. URL https://jmlr.org/papers/v7/demsar06a.html.

Francis X. Diebold and Roberto S. Mariano. Comparing Predictive Accuracy. *Journal of Business & Economic Statistics*, 13(3):253–263, 1995. doi: 10.1080/07350015.1995.10524599. URL https://doi.org/10.1080/07350015.1995.10524599.

Wenjie Du, David Côté, and Yan Liu. SAITS: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023. doi: 10.1016/j.eswa.2023.119619. URL https://doi.org/10.1016/j.eswa.2023.119619.

Robert F. Engle. Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, 50(4):987–1007, 1982. doi: 10.2307/1912773. URL https://doi.org/10.2307/1912773.

Kun Feng, Shaocheng Lan, Yuchen Fang, Wenchao He, Sihan Lu, Shuqi Gu, Lintao Ma, Xingyu Lu, and Kan Ren. Kairos: Toward Adaptive and Parameter-Efficient Time Series Foundation Models. arXiv:2509.25826, 2025. URL https://arxiv.org/abs/2509.25826v3. Version 3, revised 2026-05-14.

Yoav Freund and Robert E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. doi: 10.1006/jcss.1997.1504. URL https://doi.org/10.1006/jcss.1997.1504.

Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. TimeGPT-1. arXiv:2310.03589, 2023. URL https://arxiv.org/abs/2310.03589v3. Version 3, revised 2024-05-27.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. Monash Time Series Forecasting Archive. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/eddea82ad2755b24c4e168c5fc2ebd40-Abstract-round2.html.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A Family of Open Time-series Foundation Models. In *Proceedings of the International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16115–16152. PMLR, 2024. URL https://proceedings.mlr.press/v235/goswami24a.html.

Mark Herbster and Manfred K. Warmuth. Tracking the Best Expert. *Machine Learning*, 32(2): 151–178, 1998. doi: 10.1023/a:1007424614876. URL https://doi.org/10.1023/A:1007424614876.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2106.09685.

Rob J. Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, Melbourne, Australia, 3rd edition, 2021. URL https://otexts.com/fpp3/.

Rob J. Hyndman and Anne B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006. doi: 10.1016/j.ijforecast.2006.03.001. URL https://doi.org/10.1016/j.ijforecast.2006.03.001.

Jeonghwan Im and Hyuk-Yoon Kwon. COSA: Context-aware Output-Space Adapter for Test-Time Adaptation in Time Series Forecasting. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/2a8ce71baac4c89bf9ff479d8240c7d9-Abstract-Conference.html.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960. doi: 10.1115/1.3662552. URL https://doi.org/10.1115/1.3662552.

HyunGi Kim, Siwon Kim, Jisoo Mok, and Sungroh Yoon. Battling the Non-stationarity in Time Series Forecasting via Test-time Adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(17):17868–17876, 2025. doi: 10.1609/aaai.v39i17.33965. URL https://doi.org/10.1609/aaai.v39i17.33965.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021. doi: 10.1016/j.ijforecast.2021.03.012. URL https://doi.org/10.1016/j.ijforecast.2021.03.012.

Nick Littlestone and Manfred K. Warmuth. The Weighted Majority Algorithm. *Information and Computation*, 108(2):212–261, 1994. doi: 10.1006/inco.1994.1009. URL https://doi.org/10.1006/inco.1994.1009.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2205.14415.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative Pre-trained Transformers Are Large Time Series Models. In *Proceedings of the International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32369–32399. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24cb.html.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A Family of Highly Capable Time Series Foundation Models. In *Proceedings of the International Conference on Machine Learning*, volume 267 of

Proceedings of Machine Learning Research, pp. 39295–39317. PMLR, 2025. URL https://proceedings.mlr.press/v267/liu25be.html.

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. doi: 10.1016/j.ijforecast.2019.04.014. URL https://doi.org/10.1016/j.ijforecast.2019.04.014.

Steffen Moritz and Thomas Bartz-Beielstein. imputeTS: Time Series Missing Value Imputation in R. *The R Journal*, 9(1):207–218, 2017. doi: 10.32614/RJ-2017-009. URL https://journal.r-project.org/articles/RJ-2017-009/.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *International Conference on Learning Representations*, 2023. URL https://dblp.org/rec/conf/iclr/NieNSK23.html.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. URL https://gaussianprocess.org/gpml/chapters/.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. Lag-Llama: Towards Foundation Models for Probabilistic Time Series Forecasting. arXiv:2310.08278, 2023. URL https://arxiv.org/abs/2310.08278v3. Version 3, revised 2024-02-08.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24804–24816, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/cfe8504bda37b575c70ee1a8276f3486-Abstract.html.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Tent: Fully Test-time Adaptation by Entropy Minimization. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2006.10726.

Zhixian Wang, Linxiao Yang, Liang Sun, Qingsong Wen, and Yi Wang. Task-oriented Time Series Imputation Evaluation via Generalized Representers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 137403–137431, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f88264fcc54775ee1706116e90fe351a-Abstract-Conference.html.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified Training of Universal Time Series Forecasting Transformers. In *Proceedings of the International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53140–53164. PMLR, 2024. URL https://proceedings.mlr.press/v235/woo24a.html.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are Transformers Effective for Time Series Forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):11121–11128, 2023. doi: 10.1609/aaai.v37i9.26317. URL https://doi.org/10.1609/aaai.v37i9.26317.

Lifan Zhao, Shuming Kong, and Yanyan Shen. DoubleAdapt: A Meta-learning Approach to Incremental Learning for Stock Trend Forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3492–3503, 2023. doi: 10.1145/3580305.3599315. URL https://doi.org/10.1145/3580305.3599315.

A Extended context and related-work distinctions

This appendix provides background, methodological details, and supplementary evidence. Appendix B gives the method specification and procedure; Appendix C records the experimental protocol; Appendices D and E provide supplementary analyses and measured costs. Appendix F specifies numerical implementation settings and reproducibility artifacts.

A.1 Why the financial setting motivates these decisions

Financial missingness has several causes. Closures, suspensions, no-trade states, and not-yet-released indicators carry information about availability; transmission interruptions, recording omissions, and unsynchronized data feeds can instead create numerical observation gaps. COIN addresses the latter once the calendar and availability rules have been established. An imputed value is a model input and never becomes an observed label merely because it has been filled.

The diversity of target variables motivates retaining several completion assumptions. Prices, volumes, rates, and macroeconomic indicators differ in scale, frequency, persistence, and seasonal structure. Intraday and other periodic patterns may matter (Andersen & Bollerslev, 1997), while heavy tails and volatility clustering motivate limiting the influence of isolated errors (Cont, 2001; Engle, 1982; Bollerslev, 1986). Regime changes motivate discounted rather than permanent performance accumulation (Ang & Timmermann, 2012). Scheduled market or indicator updates provide feedback, but releases and revisions require actual-availability timestamps (Croushore & Stark, 2001).

Table 3: Financial properties, corresponding COIN mechanisms, and their limitations. These motivations do not make the mechanisms exclusive to finance.

Property	Mechanism	Limitation
Heterogeneous gaps	Observation mask and availability boundary	Calendar and publication semantics must be established first.
Diverse dynamics	Eight imputation candidates	More candidates do not guarantee better forecasts.
Different scales	Scaled and anchor-relative losses	Performance depends on the scale estimate and floors.
Regime changes	Discounted expert and branch scores	Sudden shifts can invalidate recent feedback.
Extreme errors	Loss/residual clipping and anchor floor	Genuine persistent changes may be tracked slowly.
Delayed targets	Predict–observe–update ordering	Missing feedback limits online decisions.
Repeated operation	One-time LoRA; state-only online updates	Multi-candidate inference still incurs cost.

A.2 Foundation models and specialized forecasters

TimesFM uses patch-based decoder-only pretraining (Das et al., 2024); Chronos scales and quantizes values into tokens, and Chronos-2 broadens the forecasting setting (Ansari et al., 2024; 2025). Kairos employs adaptive patching and multi-scale representation, while Sundial uses continuous-valued generative modeling with flow matching (Feng et al., 2025; Liu et al., 2025). TimeGPT-1 and Lag-Llama also investigate cross-source pretraining (Garza et al., 2023; Rasul et al., 2023). Moirai unifies universal forecasting, and MOMENT and Timer cover broader time-series tasks (Goswami et al., 2024; Liu et al., 2024; Woo et al., 2024).

These approaches reduce task-specific training compared with dedicated architectures such as Temporal Fusion Transformer, PatchTST, and linear baselines (Lim et al., 2021; Nie et al., 2023; Zeng et al., 2023). They nevertheless differ in input representation and output generation. COIN there-

fore evaluates each completion through the particular downstream backbone rather than assigning a backbone-independent imputation quality score.

A.3 Distinguishing nearby formulations

Rules in imputeTS operate at missing numerical positions without downstream feedback (Moritz & Bartz-Beielstein, 2017). BRITS, SAITS, and CSDI learn reconstruction or generative completion mechanisms (Cao et al., 2018; Du et al., 2023; Tashiro et al., 2021). Pretrained imputers reuse learned weights; missingness-aware predictors such as GRU-D incorporate masks and elapsed time into a task model (Che et al., 2018). These objectives do not directly minimize the forecasting loss of an unrelated frozen TSFM.

The task-oriented formulation of Wang et al. (2024) studies missing training labels, estimates downstream gains offline through generalized representers, combines imputed values at individual positions, and then trains a prediction model. COIN studies missing input history at forecast time, measures gains from subsequently arriving targets, combines forecasts, and keeps backbone parameters frozen. Its contribution is this online hierarchical decision arrangement, not the general idea that imputation should be judged by task utility.

TAFAS and COSA provide precedent for using arriving targets in forecasting adaptation (Kim et al., 2025; Im & Kwon, 2026). COSA concatenates predictions and recent-target context, applies a linear residual adapter, and controls it through a learned gate. COIN's residual branches instead keep horizon-specific smoothed error states, and adaptation operates after multiple candidate forecasts have been combined. DoubleAdapt adapts both data and parameters through incremental meta-learning (Zhao et al., 2023), whereas COIN fixes LoRA after preparation and makes an online branch choice.

Full-information expert learning supplies the weighting perspective (Littlestone & Warmuth, 1994; Freund & Schapire, 1997). Tracking and adaptive-regret methods motivate attention to changing performance (Herbster & Warmuth, 1998; Daniely et al., 2015; Adamskiy et al., 2016), but COIN's clipping, discounting, relative losses, priors, and floor are additional design choices. No inherited regret bound or universal optimality of this update rule is asserted.

B Detailed method and rolling procedure

B.1 Notation

Table 4: Notation used in the main text and procedure.

Symbol	Meaning
x_t, m_t	Series value and observation mask.
τ_k, L, H, K	Origin, context length, forecast horizon, and number of windows.
C_k, M_k, y_k	Historical context, its mask, and future target.
$\mathcal{I}_k, \pi, \ell, s$	Available information, policy, decision loss, and initial-history scale.
$\phi_j, \widehat{C}_{k,j}$	Imputer and its completed historical candidate.
θ, Δ	Frozen backbone parameters and one-time LoRA increment.
$p_{k,j}^{\mathrm{base}}, p_{k,j}^{\mathrm{lora}}$	Expert forecasts with the base and LoRA models.
$J, w_{k,j}, P_k, Q_k$	Expert count, shared weights, and the two forecast mixtures.
$r_{k,j}, R_{k,j}$	Anchor-relative loss and discounted accumulation.
$d_k, \mathcal{G}$	Dynamic residual state and finite correction-strength set.
$\mathcal{A}, U_{k,a}, a_k$	Branch set, branch score, and chosen branch.

B.2 Candidate definitions and information boundary

The initial history is copied into each candidate with the same original mask. Linear interpolates between available observations. Mean and Median use the corresponding location statistics of observed initial values. Forward carries the nearest earlier observation; Backward uses an already

available later observation within the historical input; Nearest uses the closest observation in time. Seasonal Median takes a robust center over visible values at the corresponding seasonal position. Seasonal Blend combines a seasonal estimate with Linear. During deployment, right-side observations must have arrived by the forecast origin. Retrospective anchor validation and prefix-truncated LoRA training have distinct preparation boundaries (Appendix F).

Local interpolation may miss turning points; location statistics suppress dynamics and may be affected by level changes; neighbor-based rules can lag trends or introduce jumps; seasonal methods need sufficient, stable repeating structure. Candidate diversity matters most when completion assumptions generate different model responses. In smooth regions, several candidates can be nearly identical. The anchor of Eq. 5 supplies a validated starting point and a comparator without adding another rolling inference call.

Original fills remain fixed as new observations are appended. This differs from rerunning all imputers at each origin, and implies that initial input diversity can decay as gaps leave the context. In the ongoing-missingness study, new gaps receive a shared Linear fill rather than eight new distinct completions.

B.3 Scoring and branch semantics

The data layer's loss is evaluated on *unadapted* expert outputs. Its relative scores are normalized against the anchor, clipped, discounted, and converted to continuous weights. Sharing weights across the two models makes the comparison between P_k and Q_k conditional on the same candidate allocation. It also restricts the LoRA model from independently preferring different completions.

Table 5: The three score families have different comparators and update different decisions.

Score	Evaluated prediction	Comparator	Decision
Expert	Base forecast for each expert	Anchor expert	Shared data weights
Strength	Output at each residual strength	Zero correction	Dynamic strength
Branch	Output of each adaptation branch	Reference branch	Single final branch

Dynamic residuals estimate the scaled error of P_k, separately at each horizon position. Fixed residuals estimate the error left after their own correction. Unobserved target coordinates receive zero innovation and their states decay; an entirely unobserved window skips feedback updates. The first nonempty feedback initializes each state directly. Every branch output and every candidate strength must be determined before its target is observed and retained for subsequent scoring. Updating a residual first and then retrospectively using it to score the same window would violate this order.

The five branches are parallel. Selecting the half-LoRA output changes the mixture between P_k and Q_k, not the trained matrices. The fixed and dynamic residual branches retain independent states, and no residual correction is stacked on a LoRA branch. The reference branch and zero strength provide fallback options rather than a guarantee against degradation.

B.4 Complete rolling procedure

Table 6 summarizes preparation and each rolling iteration. Saving pre-feedback outputs ensures that all three score families evaluate the predictions actually available at the corresponding origin.

B.5 Safeguards and computational accounting

Warm-up prevents one or two windows from causing aggressive switching, but postpones use of useful alternatives. The anchor floor limits early concentration, but prevents complete concentration on a superior expert. Clipping protects against isolated extremes, but can blunt a legitimate persistent change. Conservative strength and reference-branch fallbacks can avoid unsupported corrections while missing useful ones. These are explicit tradeoffs, not universal safety guarantees.

Table 6: COIN rolling procedure. All prediction decisions precede observation of their targets.

Input: Values and observation masks; frozen f_θ; eight imputers; K windows.
Output: Issued forecasts $\widehat{y}_1, \ldots, \widehat{y}_K$.

Preparation (once per scenario).
1. Preserve original observed values and masks. Build eight initial-history completions and determine the anchor ratio ξ by initial-history validation.
2. Train LoRA increment Δ on initial history while rotating candidate inputs. Keep both θ and Δ fixed thereafter.
3. Initialize expert scores, independent residual states, strength scores, and branch scores; specify warm-up and fallback rules.
For each window $k = 1, \ldots, K$**:**
4. Append only observations available by τ_k to each candidate, and truncate to the allowed context. Keep original fills fixed.
5. Evaluate each candidate with the base and LoRA models. Construct both anchor forecasts from existing outputs.
6. Compute shared $w_{k,j}$ from available past expert scores; form P_k and Q_k. During warm-up, use the anchor.
7. Construct all residual-strength outputs and all five branches from past states. Select a_k from available branch scores, apply fallback rules if needed, and issue $\widehat{y}_k$. Save the forecasts used for scoring.
8. Wait for target availability. Score all saved expert, strength, and branch outputs only on observed targets. Update expert scores, residual states, strength scores, and branch scores for subsequent forecasts.

The full pipeline evaluates eight candidates through each of two model states every window. Batching can reduce runtime without changing this count. Anchor forecasts, residual-strength candidates, and branch selection reuse these outputs. Prediction caching supports controlled comparisons when inputs and model states are identical, but is excluded from the end-to-end timing experiment. At a scenario boundary the implementation restores the original model state to prevent LoRA carryover. Logs retain origins, weights, branch choices, and realized losses; failed configurations are recorded separately.

B.6 Design choices and computational implications

Combining outputs preserves each completion as a separate input hypothesis; averaging completed histories before a nonlinear backbone is a different operation. Continuous weighting also reduces abrupt switching between noisy candidate rankings, at the expense of retaining some weaker forecasts and requiring separate candidate inference.

Data scores use only base-model forecasts, while both models use the resulting weights. This makes P_k and Q_k comparable under the same candidate allocation and reduces online state. It also forgoes independently optimizing candidate weights for the LoRA model. No adapted-output loss is used to update $R_{k,j}$.

For an adapted weight matrix, LoRA uses $W' = W + (\alpha_{\mathrm{lora}}/r_{\mathrm{lora}})BA$ (Hu et al., 2022). Only the low-rank factors are trained. Initial-history training rotates through candidate inputs, covering multiple completion assumptions. The increment then stays fixed throughout the scenario; residual updates and branch selection handle subsequent changes without online backpropagation.

Each window follows predict, wait, observe, and update. Expert scores compare base forecasts with the anchor; strength scores compare corrected outputs with zero correction; branch scores compare the five outputs with the reference branch. These distinct scores must not be interpreted as one unified reward. Warm-up, conservative correction, clipping, and fallback branches stabilize limited feedback but do not guarantee improvement on every window.

The full method requires 16 candidate-model evaluations per window: eight histories through both base and LoRA models, potentially batched. Anchors reuse outputs, but unselected LoRA branches still incur inference cost. Anchor validation and LoRA training occur once. "Lightweight" describes

imputers and state updates, not the complete pipeline. Models reset between scenarios; origins, weights, choices, and losses are logged (Appendix B).

C Experimental protocol and implementation

C.1 Sources and selection

Table 7: Data sources, series counts, and confirmation frequencies/horizons. Counts are development/confirmation; frequency and H describe confirmation data.

Source	Dev.	Conf.	Frequency	H
Binance	2	2	1, 15 minutes	48
FNSPID	2	2	Business day	30
M4 Finance	10	2	Annual, monthly	4, 12
Futures sample	2	2	1 minute	48
Indian market	2	2	Business day	30
Monash Bitcoin	2	2	Daily	30
MTBench	2	2	5 minutes	48
OpenMedallion	2	2	Monthly	12
PortBench	2	2	Daily	30
TIME Finance	10	2	Monthly, quarterly	12, 8
Twelve Data	2	2	Hourly	48
Total	38	22		

Series lengths range from 186 to 4,096 points. Binance covers cryptocurrency quotations; FNSPID covers stock prices and volumes; the futures sample uses opening and closing prices; the Indian-market sample uses adjusted prices and volumes. Monash Bitcoin includes discussion and transaction measures, MTBench includes quotations and trade counts, OpenMedallion includes market and composite indicators, PortBench includes equity and commodity prices, TIME Finance includes economic uncertainty, and Twelve Data includes quotations and momentum measures. M4 uses only the Finance category (Makridakis et al., 2020); Bitcoin comes from the Monash archive (Godahewa et al., 2021).

The confirmation selection excludes known duplicates and conspicuous anomalous records, using length, observed proportion, and a fixed hash order rather than forecasting performance. Two series are selected per existing source. Bitcoin variables can refer to the same asset, and cross-shard relationships of anonymous assets cannot be fully verified. Historical publication versions are not reconstructed for economic indicators or every validation origin. Consequently, neither asset independence nor a strict real-time vintage evaluation is claimed.

Development scenarios are $38 \times 4 \times 3 = 456$; confirmation scenarios are $22 \times 4 \times 3 = 264$. The latter three-way factor denotes paired seeds, whereas the former denotes missingness rates. Configurations are frozen before confirmation. Artificial masking operates on observed initial-history values; native missingness and the last-H protection interval follow the main-text protocol. Arriving real observations are shared by every comparison method.

C.2 Baseline interfaces and capability alignment

Kalman filtering, STL decomposition, and Gaussian processes motivate the corresponding statistical controls (Kalman, 1960; Cleveland et al., 1990; Rasmussen & Williams, 2006). NuwaTS and MOMENT are called through imputation interfaces; Timer and Moirai complete gaps through segmented extrapolation interfaces. This evaluation concerns those interfaces followed by the same TSFMs, not every possible use of those pretrained models.

Common three-branch adaptation includes no correction, fixed residuals, and dynamic residuals. Five-branch adaptation adds half/full LoRA. Dedicated Linear and STL LoRA models share COIN's training origins, sample selection, target mask, 24 steps, batch size 2, rank 8, and scale 16. Kairos

uses learning rate 0.0002; the other models use 0.0001. COIN rotates completion inputs; the dedicated controls retain their single completion. Linear+shared LoRA instead reuses COIN's trained increment and five-branch selector while fixing the input to Linear, thereby isolating a different component from dedicated LoRA training.

Confirmation STL periods are 4 for quarterly, 12 for monthly, 52 for weekly, 7 for daily, and 5 for business-day series. Annual, minute-level, and hourly series with unclear trading calendars receive non-seasonal handling, affecting 108/264 scenarios. Development statistical baselines retain their existing frequency-inference configuration.

STL+COSA uses the official adapter and learning-rate adjustment function, ported to full-window feedback, initial-history normalization, and a masked loss. The development-selected schedule adapts every two windows, uses the mean of the latest five target batches as context, and takes three update steps per adaptation. This produces 27 gradient updates per confirmation scenario.

Kairos and Sundial are evaluated per candidate with paired random numbers. Chronos and TimesFM use batched calls checked against single-sample calls. Non-finite forecasts fall back to the last genuinely observed value; non-finite training restores frozen weights. Finite extreme values are retained. The confirmation runs report zero scenario-level numerical failures; this does not make failures impossible under other configurations.

C.3 Aggregation and statistical comparisons

For evaluable target positions $\mathcal{O}_k$ in window k, the MAE of method u is

$$\mathrm{MAE}_u = \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{h\in\mathcal{O}_k}|y_{k,h}-\widehat{y}^{u}_{k,h}|}{|\mathcal{O}_k|}. \tag{11}$$

Raw absolute errors are not averaged across variables of different units. MASE and sMAPE in Table 1 are direct metric values. The rank is determined separately by MAE within each development scenario, with average ranks for ties, and then averaged equally over 456 scenarios.

For confirmation, rMAE is normalized to the matched Linear scenario through Eq. 10; the denominator floor does not activate in any of the 264 runs. Seed and backbone averages are taken within a series, then series averages within a source, then an equal average over 11 sources. rMSE serves as an auxiliary error measure. Development feedback analyses use their separate three-metric composite and scenario weighting. A composite-point change is therefore not an rMAE-point change.

Paired bootstrap resampling draws 11 source clusters with replacement, retaining within-source pairing, for 6,000 replicates. The two primary comparisons use exact two-sided sign-flip tests of the 11 source-level mean differences with Holm correction. Bootstrap intervals are 95% intervals without multiplicity correction. Positive differences mean reference minus COIN. The two corrected p-values are both 0.0684 even though their unadjusted bootstrap intervals exclude zero; the different procedures should be reported together rather than selectively interpreted.

Rolling evaluation separates each forecast origin from its subsequently observed targets, following the temporal ordering of forecast evaluation (Hyndman & Athanasopoulos, 2021).

C.4 Special-purpose protocols

The continued-update contrast shares the same first four windows for warm-up and evaluates the subsequent 16. History acquisition is unchanged when policy states are frozen. The delay contrast holds history acquisition fixed while delaying completed-window feedback by one or three windows, and evaluates the 20-window composite. These two panels use different evaluation intervals.

The ongoing-gap study uses eight prespecified sources, one confirmation series each, two backbones, and three seeds. C1 contributes 48 reused runs; C0, C2, and C3 contribute 144 additional runs. C2 and C3 reuse C1's initial completions, reference configuration, and LoRA state. Their new gaps are filled by a common Linear rule. Artificially hidden targets remain unavailable to history and decision updates; their original values are used only for offline scoring. Normalization to the same C0 Linear MAE makes degradation across conditions visible. C0 still contains native missingness.

D SUPPLEMENTARY RESULTS AND DIAGNOSTIC INTERPRETATION

D.1 VARIATION ACROSS DATA SOURCES AND BACKBONES

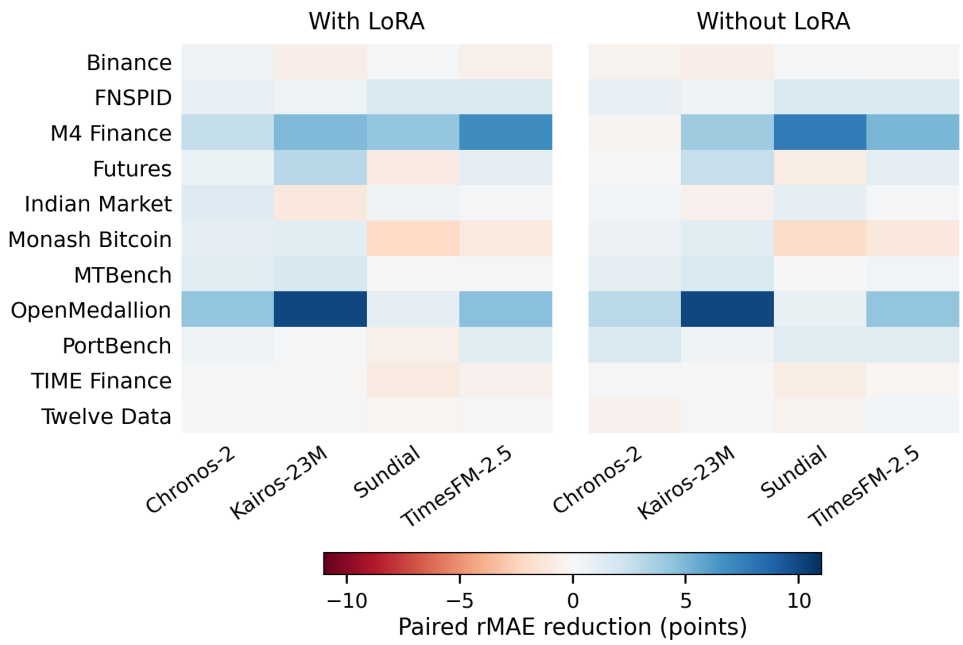

Figure 5: Paired confirmation improvement by source and backbone. Each cell averages two series and three seeds. Left: STL+dedicated LoRA minus full COIN. Right: STL+three branches minus COIN without LoRA. Both panels share a zero-centered linear scale; blue positive values denote rMAE improvement.

The full method improves the cross-backbone mean on 8/11 sources, and the no-LoRA variant on 7/11. Both improve 16/22 series in their respective prespecified comparisons. M4 Finance and OpenMedallion exhibit larger gains, while a source can have different effects for different backbones. This is consistent with candidate usefulness depending jointly on completion assumptions and model response, and argues against reading an aggregate improvement as a per-series guarantee.

D.2 SUMMARY OF REPORTED PAIRED EFFECTS

Table 8: Reported paired effects. Confirmation rows measure reference minus COIN in rMAE points. The last row is a development composite contrast on 16 post-warm-up windows and is not directly comparable in scale.

Comparison	Reduction	95% interval
STL+dedicated LoRA vs. COIN	1.124	$[0.174, 2.356]$
STL+three branches vs. no-LoRA COIN	1.058	$[0.197, 2.127]$
No-LoRA COIN vs. COIN	0.451	$[0.047, 1.105]$
Equal weighting vs. COIN	0.386	$[-0.129, 0.987]$
Frozen vs. continued policy updates	0.673	$[0.021, 1.248]$

The first two rows are the prespecified confirmation tests (both Holm-adjusted $p = 0.0684$). Other effects describe components or development behavior. The improvement of full COIN over data combination only is 0.611 points, and over Linear+shared LoRA is 0.822 points. The shared-LoRA comparison holds the trained increment fixed and should not be substituted for the dedicated-LoRA comparison.

Dynamic weighting's interval versus equal weighting crosses zero, so the study does not establish that this particular weighting rule is uniformly superior. The residual branches also provide small aggregate increments: development composite scores are 98.790 without fixed residuals and 98.817 without dynamic residuals, compared with 98.720 for the complete method. Neither result proves superadditive cooperation between components.

D.3 Alternative decision rules (S1)

Table 9 compares decision rules on the same 264 confirmation scenarios. All methods share archived forecasts and target masks. The equal-weight control includes the anchor and starts equally weighted at the first window; it differs from the matched-warm-up controls below. Hedge uses sensitivity 1 and discount 0.95; the flat mixture uses 4 and 0.75. Each pair was selected from $\eta \in \{0.25, 1, 4\}$ and $\lambda \in \{0.5, 0.75, 0.95\}$ on 44 development scenarios. The flat rule combines 45 outputs: five adaptation branches for each of nine input experts, with independent residual states. All paired intervals cross zero; neither superiority nor equivalence is established.

Table 9: Alternative decision rules; differences are method minus COIN in rMAE points. Ordered by increasing rMAE. Shading identifies COIN; it does not denote significance.

Method	rMAE	Difference	95% interval
Discounted Hedge	98.313	-0.165	$[-0.608, 0.203]$
Flat 45 experts	98.352	-0.126	$[-0.604, 0.365]$
COIN	98.478	0.000	$[0.000, 0.000]$
Equal nine (original start)	98.864	0.386	$[-0.129, 0.987]$

As a descriptive supplementary statistic, the mean positive excess $\max(\mathrm{rMAE} - 100, 0)$ is 0.397 for COIN, 0.615 for Hedge, 0.768 for the flat mixture, 0.361 without LoRA, and 0.224 for Linear+three branches. This statistic was not a primary endpoint and does not measure investment risk.

D.4 Matched-adaptation development results (S2)

Table 10 reports all 11 pipelines with common identity/fixed/dynamic residual branches and no LoRA. Metrics average equally over 456 development scenarios; MAE ranks use these 11 pipelines with average ranks for ties. The best mean depends on the metric: STL leads MASE, MOMENT leads sMAPE, and COIN without LoRA leads mean rank. Confirmation sMAPE is 30.674 for COIN and 30.635 for STL+dedicated LoRA despite favorable rMAE and rMSE directions. Development configuration selection and metric dependence limit claims of general dominance.

Table 10: Development comparison with matched three-branch adaptation; lower is better. Ordered by MASE, then sMAPE for ties. Shading identifies COIN; it does not denote significance. Darker cells mark column minima.

Pipeline	MASE	sMAPE (%)	Mean rank
STL–Kalman	**4.587**	18.921	5.433
COIN without LoRA	4.591	18.853	**5.283**
Linear	4.607	18.955	5.795
GP–RBF	4.614	18.874	6.101
MOMENT	4.631	**18.811**	5.715
Forward	4.635	19.004	6.270
Moirai	4.636	18.929	6.011
NuwaTS	4.661	18.926	5.943
Timer	4.668	18.947	6.197
Kalman	4.668	19.045	6.281
Mean	5.174	19.299	6.971

D.5 Feedback and ongoing-gap boundaries

The developmental 20-window composite rises from 98.720 with immediate feedback to 98.972 with one-window delay and 99.202 with three-window delay. Continued updating over the 16 post-warm-up windows yields 98.365 versus 99.037 with frozen states. Kairos and Sundial are more sensitive in Figure 3; some conditions are non-monotonic. The descriptive interval of the update contrast is not independent confirmation because the development data were also used for configuration choices.

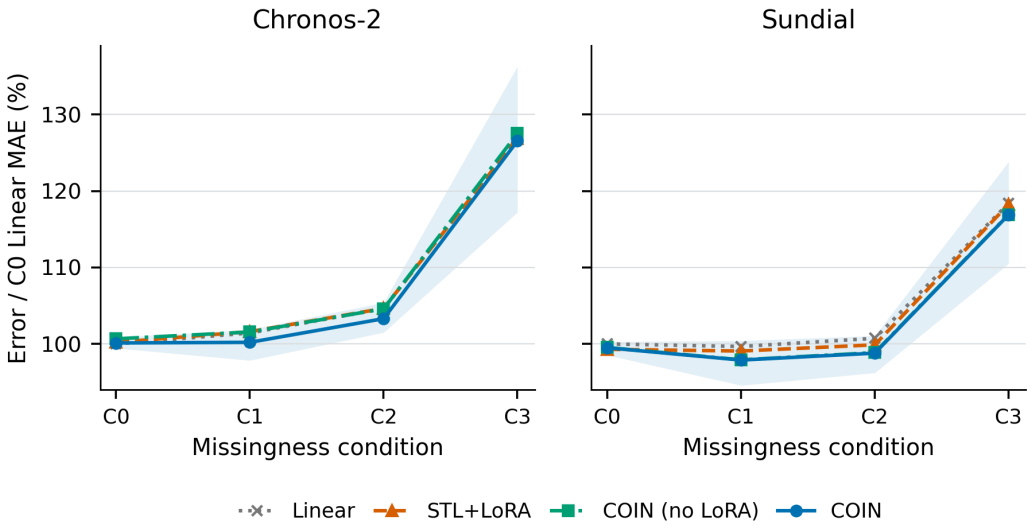

Figure 6: Targeted ongoing-missingness study. Error is expressed relative to the same C0 Linear MAE for each matched series, backbone, and seed, unlike the per-scenario denominator in Table 2. Shading denotes COIN's 95% source-cluster interval. C0 includes native gaps; C3 removes the most recent observations.

In the targeted ongoing-gap study, C2 increases COIN error relative to C1 by 3.094 points for Chronos-2 and 0.903 for Sundial. The corresponding C3 increases are 26.323 and 18.998. Paired COIN–STL+dedicated LoRA intervals cross zero under both C2 and C3. Recent gaps remove high-value input positions and usable feedback together, while the shared Linear fill reduces candidate distinctions. These results limit the scope of the initial-gap protocol rather than establish a general solution to sustained missingness.

D.6 Reconstruction versus forecasting rankings

For each confirmation scenario, we evaluate the eight basic imputers at the first forecasting window. Reconstruction MAE uses only artificially hidden positions inside the backbone's actual input context; native gaps, observed positions, the anchor, and adapted outputs are excluded. Forecast MAE evaluates each corresponding base-model prediction on that window's observed targets. Reconstruction is computed from the original float64 completions before the model's float32 conversion. Spearman correlation uses average ranks for exact ties. Best-candidate sets are disjoint only when no imputer attains both minimum errors, allowing ties in either set.

All 264 scenarios are evaluable. Table 11 shows weak positive rank agreement and frequent differences between optimal sets, supporting forecast-based evaluation of completions. The 81.439% disagreement is not a COIN win rate, and the positive correlation does not imply that reconstruction contains no useful information. Hidden initial values are used only for this offline diagnostic, never for COIN's decisions.

Truth completion and downstream utility. We additionally replace only artificial initial gaps with their true values, retaining observed values, native-gap handling, initial normalization, and fore-

Table 11: First-window ranking diagnostics, with 95% source-cluster intervals.

Backbone	Rank correlation	Disjoint best sets (%)
Chronos-2	0.253 $[0.124, 0.397]$	77.273 $[66.667, 87.879]$
Kairos-23M	0.263 $[0.096, 0.441]$	90.909 $[83.333, 96.970]$
Sundial	0.237 $[0.022, 0.440]$	80.303 $[66.667, 92.424]$
TimesFM-2.5	0.252 $[0.084, 0.410]$	77.273 $[65.152, 87.879]$
All	0.251 $[0.133, 0.362]$	81.439 $[75.379, 86.742]$

casting boundaries. This provides a ninth *base-model input*, not an adapted competitor. Ranking by full-20-window MAE gives truth completion mean rank 4.428, with source interval $[3.966, 4.886]$; it is best in 16.288% of scenarios ($[9.470, 23.106]$). These are properties of the specified candidate set, not a test against an assumed random probability of $1/9$.

Table 12 separates online base-output rules from offline references. Reconstruction selection minimizes hidden-gap error in the first effective context and then holds that single imputer for all 20 windows. Hindsight references choose using future targets. They are unavailable online, and a best-single-candidate reference is not a bound on convex mixtures or adapted outputs.

Table 12: Base-model forecasting and offline references on the same 264 scenarios. No LoRA or residual correction is used. Rows are ordered by rMAE; all full-run means use the same Linear denominator.

Rule	rMAE	Information used
Hindsight best per window	89.939	Future targets
Hindsight best per scenario	96.854	Future targets
Exponential data layer	98.746	Arrived targets
COIN data layer	99.089	Arrived targets
Eight-candidate average	99.578	Observed history
Truth-filled history	99.766	Hidden initial values
Reconstruction-selected	99.839	Hidden initial values
Linear	100.000	Observed history
Hindsight worst per scenario	106.290	Future targets

Table 13: All seven planned exploratory contrasts for completion utility. Differences are first minus second in rMAE points; intervals are paired source-bootstrap 95% intervals without multiplicity adjustment.

First minus second	Difference	95% interval
Truth-filled history − Hindsight best per scenario	2.912	$[1.854, 4.152]$
Truth-filled history − Eight-candidate average	0.188	$[-0.471, 0.854]$
Truth-filled history − COIN data layer	0.677	$[-0.044, 1.710]$
Truth-filled history − Exponential data layer	1.021	$[-0.126, 2.581]$
Reconstruction-selected − Eight-candidate average	0.261	$[-0.328, 0.886]$
Reconstruction-selected − Expected random single candidate	-0.866	$[-2.037, -0.120]$
Reconstruction-selected − Hindsight best per scenario	2.985	$[2.013, 4.139]$

Truth filling has rMAE 99.766, versus 100 for Linear; their difference is -0.234 with interval $[-0.751, 0.278]$. Reconstruction selection exceeds the COIN data layer by 0.750 points ($[0.178, 1.630]$), but this comparison changes both the scoring objective and fixed-single versus dynamic-combination decisions. The corresponding average-output contrast is 0.489 ($[-0.106, 1.192]$), and exponential weighting minus the COIN data layer is -0.343 ($[-1.012, 0.272]$). Table 13 also shows that reconstruction selection improves over the expected random single-candidate loss by 0.866 points, while its contrast with averaging outputs crosses zero.

Reconstruction therefore contains useful information without reliably identifying the best forecasting input. Truth completion is not demonstrably worse than simple averaging. The random reference averages *candidate losses*, whereas an ensemble averages *predictions* before scoring.

D.7 Matched warm-up and worse-scene performance

The following post hoc diagnostics reuse the 264 confirmation scenarios. They do not add independent test sources. All rules use the same first two feedback windows and identical forecasts during this warm-up, then maintain their own adaptation states. The equal-nine rule includes the anchor; equal-eight excludes it after warm-up. Ordinary exponential weighting uses the same composite decision loss but removes anchor-relative normalization, clipping, anchor prior, and reserved anchor share. Its sensitivity 4 and discount 0.75 were selected on the same 44 development scenarios; this is distinct from S1's discounted Hedge control. Table 14 reports every matched rule. None of the three paired comparisons is significant after their separate Holm correction ($p = 0.9199$).

Table 14: Matched-warm-up rules on the confirmation cohort. Ordered by increasing rMAE.

Rule	rMAE	Difference	95% interval
Exponential (matched)	98.245	-0.233	$[-0.801, 0.333]$
COIN	98.478	0.000	$[0.000, 0.000]$
Equal nine (matched)	98.768	0.290	$[-0.133, 0.825]$
Equal eight (matched)	98.775	0.297	$[-0.150, 0.834]$

We define deterioration as scene rMAE > 105, relative to unadapted Linear. Each source has 24 scenes. For each method separately, we rank its scenes by rMAE, average its worst three within each source, and then average the 11 sources equally. Thus the worst-three statistic compares method-specific worst subsets, not errors on a common selected subset. Paired source bootstrap resampling (6,000 replicates, seed 20260916) retains these source-level summaries. Table 15 reports all tested safeguards, including the clipping ablation; intervals are descriptive and unadjusted for multiple diagnostics.

Table 15: Mean and worse-scene rMAE. Rate is the percentage of scenes with rMAE > 105; the last column is the worst-three difference versus COIN with its 95% interval. Ordered by mean rMAE.

Rule	Mean	Rate (%)	Worst 3	Worst-3 difference
Exponential (matched)	98.245	3.030	103.652	1.717 $[0.573, 3.304]$
No anchor floor	98.363	0.758	102.155	0.220 $[0.120, 0.322]$
Uniform prior	98.389	0.758	102.303	0.368 $[0.014, 0.703]$
No expert clipping	98.474	0.758	101.896	-0.039 $[-0.235, 0.147]$
COIN	98.478	0.758	101.935	0.000 $[0.000, 0.000]$
Equal nine (matched)	98.768	3.030	103.343	1.408 $[0.459, 2.478]$

COIN has 2/264 deteriorating scenes, compared with 8/264 for both matched equal-nine and ordinary exponential weighting. Their deterioration-rate differences are 2.273 percentage points, with intervals $[0.379, 4.167]$ and $[0.000, 4.924]$, respectively. Ordinary exponential weighting achieves lower mean error, but higher worst-three error. Removing the anchor floor slightly lowers mean rMAE by 0.115 points, while increasing worst-three error by 0.220; replacing the anchor prior with a uniform prior gives corresponding changes of -0.089 and $+0.368$. Removing expert-loss clipping shows no clear worst-three benefit or harm. These results identify a mean–worst-scenario tradeoff associated with anchoring rather than validate every safeguard. They neither guarantee per-scene improvement nor establish control of trading losses.

Continuous excess loss and the full tail panel. To avoid choosing different worst-scene subsets, we also evaluate $\max(\mathrm{rMAE}_u - 100, 0)$ on *every* shared scenario and average within sources, then across sources. This measures positive forecasting-error excess over Linear. For complementary tail summaries, P90/P95 use linear interpolation within each source's 24 scenarios; the worst-10% mean

(CVaR10) averages the highest two values and 0.4 of the third, divided by 2.4. Quantiles and CVaR still use method-specific orderings.

The six-rule panel includes matched exponential weighting, adding only the 0.2 anchor floor, adding both that floor and the 0.5 anchor prior, full COIN, development-selected pruning (Appendix E.1), and fixing the anchor coefficient to $\xi = 1$. All other adaptation states are replayed in their original causal order. The fixed-Linear-anchor change affects 142/264 scenarios and does not alter imputation, normalization, or LoRA training boundaries. The matched exponential rule differs from the S1 control in Section D.3, so their excess losses must not be interchanged.

Table 16: Complete mean–tail panel on 264 shared confirmation scenarios (11 sources equally weighted). All columns are lower-is-better; rows are sorted by mean rMAE. Excess is the mean positive excess over Linear, in rMAE points. Shading identifies COIN; it does not denote significance. Darker cells mark column minima.

Rule	Mean	Excess	P90	P95	CVaR10
Exponential (matched)	**98.245**	0.730	102.156	103.106	103.956
Exponential + floor	98.289	0.604	101.798	102.510	103.224
Exponential + floor + prior	98.357	0.523	101.592	102.204	102.781
Pruned COIN (3 retained)	98.407	0.451	101.181	101.763	102.405
COIN	98.478	0.397	101.105	101.532	102.104
Fixed Linear anchor	98.559	**0.330**	**100.967**	**101.363**	**101.845**

Table 17: All five exploratory excess-loss contrasts, ordered as in Table 16. Positive differences favor COIN. Intervals are pointwise 95%; raw source sign-flip and five-comparison Holm values are both shown. Earlier testing families are unchanged.

Rule	Excess difference (95% CI)	Raw p	Holm p
Exponential (matched)	0.333 $[0.095, 0.669]$	0.0088	0.0439
Exponential + floor	0.207 $[0.039, 0.453]$	0.0273	0.1094
Exponential + floor + prior	0.127 $[-0.010, 0.357]$	0.2119	0.2119
Pruned COIN (3 retained)	0.054 $[0.013, 0.099]$	0.0459	0.1377
Fixed Linear anchor	-0.067 $[-0.146, -0.009]$	0.0664	0.1377

Against matched exponential weighting, COIN reduces excess loss by 0.333 points ($[0.095, 0.669]$), with exploratory Holm $p = 0.0439$. Adding the floor and prior produces intermediate tail-error point estimates, while mean rMAE rises. This is a measured tradeoff: fixed Linear anchoring has still lower tail-error estimates but higher mean rMAE than COIN, and not all pairwise tail contrasts survive correction. The original deterioration threshold remains $\mathrm{rMAE} > 105$; at thresholds 103, 104, 105, 106, and 110, COIN has 6, 3, 2, 1, and 0 events, versus 24, 15, 8, 7, and 2 for matched exponential weighting and 20, 12, 8, 5, and 1 for matched equal weighting. The complete threshold panel is retained in the supplement; smaller sensitivity-test p-values do not replace the original threshold.

D.8 Source sensitivity and separate state updates

Leaving out one source at a time preserves positive COIN improvements versus dedicated Linear LoRA (0.576–1.020 points) and dedicated STL LoRA (0.736–1.271 points). These are within-source robustness checks, not tests on new sources. In the separate three-comparison family comprising these two controls and original-start equal weighting, Holm-adjusted p-values are 0.0967, 0.0967, and 0.2412. This family differs from the two prespecified primary comparisons, whose adjusted values remain 0.0684.

To distinguish online state contributions, confirmation runs share the first four windows, then freeze either expert weights, branch scores, or residual states at the snapshot taken after window four. History still grows identically; window-five predictions are identical before later state trajectories

diverge. Table 18 evaluates windows 5–20 using each scene's full-20-window Linear MAE denominator. This scale differs from the development composite in Figure 3. Only the residual-state contrast excludes zero in its unadjusted interval, and none survives the separate three-comparison Holm correction. Effects cannot be added as a decomposition of the complete method.

Table 18: Separate state freezing on confirmation windows 5–20. Ordered by increasing rMAE.

Policy	rMAE	Difference (95% interval)	Holm p
COIN	102.192	0.000 $[0.000, 0.000]$	1.0000
Freeze branch scores	102.320	0.128 $[-0.051, 0.305]$	0.3457
Freeze residual states	102.377	0.185 $[0.017, 0.346]$	0.1934
Freeze weights	102.669	0.477 $[-0.030, 1.293]$	0.3457

D.9 Effects over the rolling horizon

Table 19 retains all period-level contrasts against three controls, using the full-20-window Linear MAE denominator throughout. The first two windows verify shared initialization for the matched ensemble controls. Subsequent advantages vary over time and by comparator; ordinary exponential weighting is better late in the run. All 20 windows retain artificially missing initial-history positions inside the actual model contexts, so these period effects cannot be attributed to initial gaps having completely left the context. Period-specific intervals are exploratory and unadjusted.

Table 19: Period-wise rMAE differences (control minus COIN) and 95% intervals.

Windows	Equal nine	Exponential	Linear + LoRA
1–2	0.000 $[0.000, 0.000]$	0.000 $[0.000, 0.000]$	0.163 $[-0.121, 0.480]$
3–4	0.602 $[-0.359, 1.763]$	0.150 $[-0.713, 1.118]$	0.091 $[-1.188, 1.927]$
5–8	0.959 $[-0.149, 2.241]$	-0.130 $[-1.010, 0.640]$	1.119 $[0.077, 2.394]$
9–12	-0.060 $[-0.918, 0.804]$	0.483 $[-0.472, 1.717]$	0.150 $[-0.392, 0.712]$
13–16	0.274 $[-0.520, 1.363]$	-0.782 $[-1.804, 0.099]$	1.766 $[0.439, 3.849]$
17–20	-0.024 $[-0.446, 0.501]$	-0.812 $[-1.879, -0.005]$	1.277 $[0.310, 2.569]$
3–20	0.322 $[-0.148, 0.916]$	-0.259 $[-0.890, 0.370]$	0.968 $[0.214, 1.914]$
1–20	0.290 $[-0.133, 0.825]$	-0.233 $[-0.801, 0.333]$	0.888 $[0.197, 1.751]$

D.10 Additional series with frozen configurations

A further 11 series, one from each existing source, were selected before inference by length, observed fraction, and a frozen hash order, excluding previous series, identified duplicate entities, exact numerical duplicates, and mechanical calendar fields. With four backbones and seed 2701, this yields 44 new scenarios. The protocol retains 20 windows, 20% initial missingness, original COIN settings, the already selected exponential rule, and matched dedicated Linear LoRA with 24 training steps and batch size two. All 44 runs completed without numerical fallback.

Table 20 reports every tested method. COIN improves on matched equal weighting and Linear+dedicated LoRA in 6/11 source means; ordinary exponential weighting again has the lowest overall mean. Every paired difference interval includes zero. This small, single-seed extension supports a similar directional pattern but provides neither new-source generalization nor independent confirmation of the worst-three diagnostic.

E Measured end-to-end computational cost

The machine has an NVIDIA RTX 4060 Laptop GPU with 8 GiB. The eight workloads combine one M4 annual series and one Bitcoin daily series with the four backbones. Each execution performs imputation, initial reference selection, LoRA training where applicable, 20 rolling windows, selection, and feedback updates. No forecast cache is read. The no-LoRA version neither constructs

Table 20: Additional-series evaluation (44 scenarios); sources equally weighted. Ordered by increasing rMAE. Shading identifies COIN; it does not denote significance.

Method	rMAE	Difference	95% interval
Exponential (matched)	99.385	-0.234	$[-1.096, 0.455]$
COIN	99.619	0.000	$[0.000, 0.000]$
Linear + dedicated LoRA	99.925	0.307	$[-0.341, 1.058]$
Equal nine (matched)	100.015	0.396	$[-0.018, 0.868]$

Table 21: Additional-series rMAE by backbone.

Backbone	COIN	Equal nine	Exponential	Linear + LoRA
Chronos-2	100.365	99.994	100.246	100.741
Kairos-23M	97.744	97.869	96.038	99.201
Sundial	99.429	100.653	99.741	99.281
TimesFM-2.5	100.937	101.543	101.514	100.479

Table 22: Measured costs on eight common workloads, with three executions each. Means are reported for time, maximum allocated GPU memory for memory. Total includes initialization and excludes loading. Ordered by total time.

Method	Init. (s)	Window (ms)	Total (s)	MiB
Linear + three branches	0.001	91.4	1.828	1280.9
STL + three branches	0.096	90.1	1.898	1280.9
STL + COSA	0.096	92.3	1.941	1280.9
STL + dedicated LoRA	4.332	226.9	8.870	1319.7
COIN without LoRA	9.741	319.6	16.132	1280.9
COIN	15.509	791.3	31.336	1319.7

nor trains LoRA. Mean per-window time includes the first window; memory includes the loaded backbone. Rounding of displayed components need not exactly recover displayed totals.

Full COIN's total is 3.53 times STL+dedicated LoRA's, and unused LoRA outputs still cost inference time because all branches are evaluated before selection. On the same workload subset, full COIN, no-LoRA COIN, and STL+dedicated LoRA have rMAE 96.409, 98.130, and 99.126. The accuracy–cost tradeoff is therefore measured on a shared workload but only two series; it does not estimate the complete confirmation set's gain. Workloads with long update intervals may accommodate the extra calls, while throughput-sensitive settings require separate measurement.

E.1 Development-selected pruning and paired rolling latency

This optional variant restricts the standalone basic experts while retaining the anchor and its nonzero constituent predictions. It reuses LoRA trained on all eight completions, and does not change the full method. We test retained counts $m \in \{2, 3, 4, 5, 6, 8\}$ under two orders. The fixed order is Linear, Seasonal Blend, Forward, Seasonal Median, Nearest, Median, Mean, Backward. The development order is Seasonal Median, Nearest, Mean, Backward, Seasonal Blend, Median, Forward, Linear, ranked by source-equal mean original COIN weights on the 456 development scenarios. The subset prior assigns 0.5 to the anchor and $0.5/m$ to each standalone expert, with anchor floor 0.2.

The selection rule uses development data only: choose the smallest m whose rMAE is no more than 0.10 points above full COIN and whose deterioration rate rises by no more than one percentage point, breaking ties by actual support count and then the fixed order. It selects the development order with $m = 3$ (archived as development_K3). Full and selected development rMAE are 98.806 and 98.818, respectively. This is development-only selection within a post hoc diagnostic round, not a new independent confirmation study. Table 23 retains the full grid.

Table 23: Complete pruning grid on 264 confirmation scenarios. Each order is sorted by rMAE. Difference is pruned minus full COIN, with pointwise paired 95% intervals. Evaluation counts include anchor support and both model states; they are not batched backend-call counts. Shading marks the development-selected variant.

Variant	rMAE	Difference (95% CI)	Eval./window
Fixed, $m = 8$	98.478	0.000 $[0.000, 0.000]$	16.000
Fixed, $m = 6$	98.587	0.109 $[0.015, 0.222]$	12.000
Fixed, $m = 4$	98.793	0.315 $[0.013, 0.817]$	8.000
Fixed, $m = 5$	98.807	0.329 $[0.034, 0.815]$	10.000
Fixed, $m = 3$	98.876	0.398 $[0.040, 0.998]$	7.076
Fixed, $m = 2$	98.894	0.416 $[0.049, 1.020]$	5.076
Development, $m = 3$	98.407	-0.071 $[-0.229, 0.067]$	8.848
Development, $m = 6$	98.433	-0.045 $[-0.144, 0.036]$	13.773
Development, $m = 4$	98.447	-0.031 $[-0.115, 0.051]$	10.848
Development, $m = 5$	98.453	-0.025 $[-0.084, 0.037]$	11.773
Development, $m = 8$	98.478	0.000 $[0.000, 0.000]$	16.000
Development, $m = 2$	98.792	0.314 $[0.040, 0.728]$	6.848

The selected variant generates 4.424 histories on average after adding anchor dependencies, requiring 8.848 candidate-model evaluations rather than 16. Its confirmation mean rMAE is 98.407 versus 98.478, with paired difference -0.071 ($[-0.229, 0.067]$); this does not establish equivalence. Deteriorations increase from 2 to 3, and source-averaged worst-three error from 101.935 to 102.197, a 0.262-point difference ($[0.057, 0.489]$). Table 16 additionally reports its full tail panel.

For measured latency, the same eight original timing workloads undergo three alternating full/pruned executions each (48 executions). Forecasts are recomputed without a prediction cache; CUDA synchronization brackets rolling inference and decisions. Loading, initial imputation, historical validation, and LoRA training are excluded. Arithmetic mean times are 862.74 and 527.17 ms/window: a 38.9% reduction, ratio 0.611. The geometric mean of workload ratios is 0.681. These contemporaneous rolling-only timings are separate from the earlier end-to-end measurements in Table 22.

Table 24: Paired rolling-only latency in ms/window, averaged over three repeats. Rows are ordered by pruned/full ratio. Each workload retains four or five generated histories after anchor support is included.

Backbone	Workload	Full	Pruned	Ratio
Kairos-23M	M4 annual	909.86	455.74	0.501
Sundial	M4 annual	1855.66	935.30	0.504
Chronos-2	Bitcoin daily	343.77	174.23	0.507
Kairos-23M	Bitcoin daily	937.90	480.32	0.512
Sundial	Bitcoin daily	1839.10	1148.30	0.624
TimesFM-2.5	Bitcoin daily	747.16	742.05	0.993
TimesFM-2.5	M4 annual	199.24	204.79	1.028
Chronos-2	M4 annual	69.26	76.58	1.106

Chronos-2 on the short M4 workload is slightly slower, and TimesFM gains little. Thus pruning offers a measured workload-dependent speed–tail tradeoff, not uniform acceleration or reduced training cost. A separate development-selected dispersion gate chose threshold zero and reproduced the static pruned outputs while requiring 12.178 evaluations per window because it first computed all eight base forecasts; no additional gating benefit is claimed.

F IMPLEMENTATION SETTINGS AND REPRODUCIBILITY

F.1 LOSS, SCALE, AND ONLINE DECISIONS

The constants below specify the configuration used for the frozen confirmation experiments. Let b be the length of initial history, q the seasonal lag, and $\mathcal{P} = \{t : q \le t < b,\ m_t = m_{t-q} = 1\}$. The fixed loss scale is the mean of $|x_t - x_{t-q}|$ over $\mathcal{P}$, floored at 10^{-8}, and is set to 1 if $\mathcal{P}$ is empty. Imputed values do not contribute. GluonTS 0.14.4 determines q; examples are annual 1, quarterly 4, monthly 12, weekly 1, daily 1, business-day 5, hourly 24, and five-minute 288. These lags differ from the STL decomposition periods in Appendix C.

Equation 2 uses $(\kappa_1, \kappa_2, \kappa_3) = (0.45, 0.30, 0.25)$ and

$$\mathrm{sAPE}_\varepsilon(p, y) = \frac{1}{|\mathcal{O}|} \sum_{h \in \mathcal{O}} \frac{2|p_h - y_h|}{\max(|p_h| + |y_h|, 0.05s)}. \tag{12}$$

This term is not multiplied by 100. Reported sMAPE instead averages $200|p_h - y_h| / \max(|p_h| + |y_h|, 10^{-8})$. For Eq. 10, σ_0 is the observed initial-history population standard deviation with floor $10^{-8} \max(1, \mathrm{median}\,|x_{\mathrm{obs}}|)$; it is distinct from s. All score and residual states are initially zero. Feedback counts include only windows with at least one observed target.

Table 25: Frozen online decision settings.

Component	Setting
Data scores	Discount $\lambda = 0.75$, sensitivity $\eta = 8$, relative clipping $c = 0.2$, denominator floor $\delta = 10^{-8}$.
Prior and floor	Anchor $\mu_0 = 0.5$; each basic expert $\mu_j = 0.0625$; reserved anchor share $\rho = 0.2$.
Numerical softmax	Subtract the maximum logit, then clip to $[-40, 0]$ before exponentiation.
Data warm-up	Anchor only until two nonempty feedback windows arrive.
Residual states	Decay $\beta = 0.75$ and normalized innovation clipping $c_e = 0.5$; first update initializes the state directly.
Fixed correction	Strength 0.3; enable after at least two feedback windows if the mean of the latest at most three counterfactual relative losses is negative.
Dynamic strengths	$\mathcal{G} = \{0, 0.15, 0.30, 0.50, 0.75, 1\}$.
Strength scores	Discount 0.75, relative clipping $[-0.5, 0.5]$, denominator floor 10^{-8}.
Dynamic eligibility	At least four feedback windows; best allowed score below -0.001 and its latest-at-most-three mean relative loss no greater than zero.
Progressive strength cap	With n received feedback windows, restrict the grid to strengths $\le \min(1, 0.25(n - 4 + 1))$ before minimization.
Branch scores	Discount 0.75; relative clipping $[-0.5, 0.5]$; denominator floor 10^{-8}.
Branch warm-up and ties	Two nonempty feedback windows. Tolerance 10^{-12}; prefer the reference branch, otherwise declaration order.
Reference branch	Dynamic residual for Kairos and Sundial; fixed residual for Chronos-2 and TimesFM.

Branch declaration order is no adaptation, fixed residual, dynamic residual, half LoRA, and full LoRA. The mathematical anchor index is $j = 0$; the stored array order places the eight basic imputers first and the anchor last. Equations and stored arrays are mapped accordingly.

For an already issued window, define the relative strength loss by comparing each saved output $P_k + \gamma s d_{k-1}$ with P_k, dividing by $\max(\ell(P_k, y_k), 10^{-8})$ and clipping to $[-0.5, 0.5]$. Each strength score equals 0.75 times its old score plus this loss. The dynamic gate in Table 25 chooses the lowest-score eligible strength; otherwise it uses zero. The fixed gate analogously evaluates the saved counterfactual output with strength 0.3, even when correction was disabled. Its residual innovation, however, uses its own emitted branch output, whether or not that branch was selected as the final prediction. Branch scores compare each emitted branch output with the predetermined reference branch using the same relative denominator and clipping, followed by discount 0.75.

For partially observed windows, missing coordinates receive zero innovation, so their previous residual states decay. Entirely unobserved windows skip all feedback updates. The first nonempty update sets each residual state equal to its clipped innovation; subsequent updates use Eq. 8. Forecasts, counterfactual strengths, and branch outputs are saved before target arrival. Scoring never uses a state that has already incorporated the target being scored.

F.2 Candidate rules and anchor validation

Every imputer preserves observed values. Linear uses pandas linear interpolation with both-direction endpoint filling; an entirely missing column becomes zero. Mean and Median use observed values of the supplied prefix. Forward, Backward, and Nearest use the corresponding pandas fill/interpolation rule. Nonfinite leftovers fall back to Linear. Seasonal Median uses at most eight nearest observed same-phase positions, falling back to Linear if none exist. Seasonal Blend averages Seasonal Median and Linear with equal weights at missing positions only. During rolling prediction initial fills stay fixed, and any newly missing stream values receive shared Linear completion over the available concatenated history.

The anchor searches $\xi \in \{0, 0.25, 0.50, 0.75, 1\}$ in Eq. 5. Historical origins are the sorted values $b - oH$ for $o = 1, \ldots, 2K + 8$ with $b - oH > H$, retaining only target windows with observed values. At least eight earlier origins are reserved before two equal chronological evaluation folds. Each fold contains $\min(K, \lfloor (n_{\text{obs}} - 8)/2 \rfloor)$ origins, at least two, requiring at least 12 observable origins. The earlier reserved fold does not train a separate anchor model.

Candidate utility combines relative improvements on the two evaluation folds with weights 0.35 and 0.65. It uses a zero-mean Gaussian prior with standard deviation 0.02, population dispersion of window-relative improvements, and a standard error divided by the square root of the number of non-tied wins and losses, floored at 10^{-6}. The gain denominator floor is 10^{-12} and the win/loss tolerance is 10^{-9}. If μ_{post} and σ_{post} denote the resulting posterior parameters, utility is $(\mu_{\text{post}} - 0.5\sigma_{\text{post}}) \Pr(\text{improvement} > 0)$, with posterior-probability denominator floor 10^{-12}. Selection requires utility strictly above 0.0005; otherwise $\xi = 1$. Ties follow the ascending grid order. Per-scene records contain origins, candidate statistics, and the selected weight.

Preparation information boundary. Anchor selection is retrospective within initial history $[0, b)$. Candidates are completed over this whole interval and then sliced at historical validation origins. Consequently, interpolation can use later observations inside the initial history, including validation targets. No final evaluation-period observation enters preparation. This is not an origin-by-origin causal historical backtest. An input audit compared Linear with strict prefix filling across 66 series–seed combinations and 2,028 historical origins; 81 origins in nine combinations had different inputs. This checks Linear inputs only and does not quantify the effect on forecasting results.

LoRA instead recomputes each completion from the prefix ending at its training origin before taking the trailing context. Its targets are excluded from completion and input-normalization statistics. The loss scale s is estimated once from all observed initial history and shared across examples; it is not recomputed from each example's earlier prefix. These distinctions preserve the final test boundary while identifying the retrospective choices made during preparation.

F.3 LoRA optimization and model interfaces

All models use rank 8, scaling parameter 16, zero LoRA dropout, 24 optimizer steps, and mini-batches of two examples. AdamW uses betas $(0.9, 0.999)$, epsilon 10^{-8}, weight decay 0.01, and gradient norm clipping at 1. Only low-rank factors receive gradients. The backbone remains in eval-

uation mode, including during optimization. Factor initialization uses seed 1729; model parameters, optimizer states, and buffers reset between scenarios.

Table 26: Backbone-specific LoRA settings. Context is the training cap; trainable counts exclude the frozen backbone.

Backbone	Target suffixes	Rate	Context	Parameters
Chronos-2	q, k, v, o, wi, wo	10^{-4}	128	1,916,928
Kairos-23M	q, k, v, o, wi, wo	$2 \cdot 10^{-4}$	256	589,824
Sundial	q_proj, k_proj, v_proj, o_proj, down_proj, gate_proj, up_proj	10^{-4}	256	1,695,744
TimesFM-2.5	qkv_proj, out, ff0, ff1	10^{-4}	128	2,048,000

Training context length is $\min(L_{\text{cap}}, \max(32, 32\lfloor b/96 \rfloor))$. Candidate origins start at this length, advance by H, and end no later than $b - H$; target windows must contain an observed value. Up to 12 origins combine evenly spaced historical coverage and recent origins, filling duplicate selections from latest to earliest. The archived origin-selection function and actual origin lists specify the exact selection. At step t, example $j \in \{0, 1\}$ uses basic imputer $(2t + j) \bmod 8$, in the order Linear, Mean, Median, Forward, Backward, Nearest, Seasonal Median, Seasonal Blend. Temporal origins are sampled with replacement. Dedicated controls share the same origin draws and target masks. Missing targets are excluded; completed inputs are passed as fully observed contexts.

The differentiable objective retains weights 0.45/0.30/0.25 but uses smoothed normalized absolute errors $\sqrt{((p - y)/s)^2 + 10^{-8}}$ and a minibatch-pooled normalized RMSE with 10^{-8} inside the final square root. It is distinct from the unsmoothed online decision loss. Sundial uses a differentiable median of three samples with fixed sample seed 1729 during training, and 100 samples at inference. The source snapshot supplies the full masked loss and the actual injected module inventory.

Prediction context caps are 8,192 (Chronos-2), 2,048 (Kairos), 2,880 (Sundial), and 4,096 (TimesFM). Point forecasts are medians. In confirmation, Kairos and Sundial evaluate candidates individually with common random numbers; Chronos and TimesFM use deterministic batches. Development uses individual Kairos calls and batched calls for the other models, including Sundial with wrapper seed zero. These batching and randomization policies must be distinguished when reproducing the archived runs. Kairos routing-bias update rates are zero. Nonfinite batch predictions trigger individual retries; remaining nonfinite outputs fall back to the latest actually observed value. Finite extreme predictions are retained. Nonfinite training restores the frozen weights.

Each backbone applies its native normalization to the completed context. COIN does not use full-series mean/variance normalization. Sundial uses population standard deviation plus 10^{-5}; Chronos and Kairos retain native instance normalization. The differentiable TimesFM path uses its outer normalization and native patch statistics. Initial observed mean/variance stored separately serve COSA and the relative-error denominator floor, rather than the decision scale s. Inference uses four PyTorch threads and one interop thread; the measured GPU is specified in Appendix E.

F.4 Masks, randomization, versions, and artifacts

The sequence manifests enumerate all 38 development and 22 confirmation series: original series/entity identifiers, variables, source row indices, segment bounds, timestamps, frequency, horizon, seasonal lag, and SHA-256 hashes. Selected numerical segments retain native missing values. Development, confirmation, timing, and ongoing-gap subsets are marked separately; the 11-series extension has its own frozen manifest.

Initial artificial blocks have length $\min(50, H)$. The final evaluation interval and the last H positions of initial history are protected. Historical validation windows retain up to $\max(2, \lceil 0.1H \rceil)$ evenly spaced observed positions, so protected positions may interrupt blocks. The nominal missingness rate applies to injectable, naturally observed positions, not all series positions. All methods in a scenario share the resulting mask.

Confirmation seeds are 1701–1703. Training and prediction seed keys are respectively `train:model:series_id:seed` and `predict:model:series_id:seed`. The first

eight hexadecimal digits of SHA-256 are reduced modulo 2^{31}; each prediction window hashes its prediction seed and origin again. Initial masking separately hashes `confirm:series_id:seed`, takes the first eight digest bytes as a big-endian integer, and reduces modulo 2^{32}. Development mask keys are `series_id:ratio`, with ratios 010/020/030. Development training hashes model, dataset, and ratio separated by double underscores, reads the first four bytes little-endian, and reduces modulo 2^{31}; anchor seeds use the same fields separated by colons. The development wrapper seed is zero. Executable routines and realized origins are included with the configurations.

The frozen environment record specifies PyTorch 2.4.1, Transformers 4.57.6, NumPy 1.26.4, SciPy 1.11.4, pandas 2.1.4, PyArrow 24.0.0, and statsmodels 0.14.6. A later complete inventory additionally records PEFT 0.19.1, GluonTS 0.14.4, chronos-forecasting 2.2.2, and timesfm 2.0.0; this later inventory is distinguished from contemporaneously recorded versions. Checkpoint identifiers are listed in Table 27. The two Chronos revisions have identical model and configuration blobs; per-scene resolution between them was not recorded. Checkpoint inventories also retain file hashes and the NuwaTS checkpoint hash.

Table 27: Checkpoint snapshot identifiers.

Backbone	Snapshot identifier
Chronos-2	`29ec3766d36d6f73f0696f85560a422f50e8498c`
	`0f8a440441931157957e2be1a9bce66627d99c76`
Kairos-23M	`5123edcc9d6ee6f495bef78ec9fb63bb00ce878d`
Sundial	`3212e42564493f520593e5414af4367fc4b49226`
TimesFM-2.5	`1d952420fba87f3c6dee4f240de0f1a0fbc790e3`

The materials include configurations, sequence segments, preparation records, complete result tables, and source snapshots (including external SCOPE and imputation interfaces), plus new diagnostic protocols, development-only selections, and analysis code. Earlier verification records match 355 source files and 60 selected segments to their inventories. CPU replay from archived base/adapted forecasts matches 264 scenarios and 18,480 metric values with zero maximum prediction or metric discrepancy. Sixteen previously frozen source hashes also match; the larger 355-file inventory was created later and is not evidence of an earlier freeze. The separate 44-scenario extension includes new model inference and its frozen manifest checks. CPU replay verification does not constitute a fresh full GPU rerun or a clean-machine installation test. The new scalar summaries were independently recomputed from archived scenario records.