TAILCAST: RELEASE-ORDER TAIL EXTRAPOLATION FOR FROZEN TIME-SERIES FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series foundation models forecast zero-shot, yet they expose their predictive distribution only on a finite grid of native quantiles—Chronos-Bolt and TimesFM-2.5 stop at 0.90, Chronos-2 at 0.99—while tail decisions are defined above it. Extending the profile is hard because the frozen model supplies no local shape there and validating outcomes are rare by construction, and because a label may arrive only after later forecasts have gone out. Existing post-hoc layers estimate the tail from scarce exceedances, discarding the conditional shape the backbone already encodes. We introduce TAILCAST, which treats that conditional quantile profile as an issue-specific anchor and learns only a small residual correction to it, continuing the top native spacing in log-survival coordinates above the grid. The residual is estimated at two adaptation scales, whose thirteen candidates are combined by a learning-rate-free rule over released pinball losses, each token scored against the state that issued it; a weighted isotonic projection, provably non-increasing for the reported score, returns the ladder. Across four frozen backbones, two benchmark groups holding 696 cases, and three scenarios, TAILCAST attains the lowest tail-weighted pinball loss of the thirteen methods compared, by 6.7% and 8.9% relative to the strongest of the other twelve on each group, issuing finite, noncrossing ladders under every delay and missingness condition tested. Because it reads only native quantiles, targets, and release timestamps, the layer drops onto any frozen backbone without retraining, bringing such tail levels within reach of operational alerting.

1 INTRODUCTION

Time-series foundation models forecast a new series without task-specific training (Ansari et al., 2024a;b; 2025; Das et al., 2024; Google Research, 2025; 2026), but their probabilistic output is a fixed grid of native quantiles: Chronos-Bolt and TimesFM-2.5 stop at 0.90 and Chronos-2 stops at 0.99, while an operational alert may be defined at 0.995 or 0.999. A practical calibration layer must therefore extend the profile above the native grid and adapt to a moving target distribution, without retraining the backbone or inspecting its internals—under feedback that is itself delayed, since the outcome validating a forecast issued at origin t may be reported only after forecasts at $t{+}1, \ldots, t{+}D$ have gone out.

Two obstacles make this harder than fitting a residual distribution. The requested levels lie where the frozen model supplies no information about the local shape, so the tail is not estimable from exceedances that are rare by definition; and delayed feedback breaks the standard online comparison, because the state that scores a label is no longer the state that produced the prediction, so a layer must either leak later information into an assessment of a decision already made, or discard late labels and forfeit most of its feedback. Neither obstacle is fully addressed by existing layers: conformal methods calibrate one coverage level at a time and return no profile above the grid (Vovk et al., 2005; Romano et al., 2019; Gibbs & Candès, 2021; Zaffran et al., 2022; Gibbs & Candès, 2024; Xu & Xie, 2021), extreme-value methods supply a tail shape but must estimate it from those same scarce exceedances (Balkema & de Haan, 1974; Pickands, 1975; Coles, 2001), and neither treats the release order of labels as part of the estimator.

We address both by *not* discarding the frozen model's conditional information and by *making the feedback order an explicit part of the estimator*. TAILCAST treats the frozen profile as an anchor

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.

rather than a target to be overwritten: each level is tied to the value the backbone would have issued had it been inside its grid, and only the residual around that anchor is learned. Above the native endpoint the tie continues the slope of the top native spacing in log-survival coordinates, which is exact when the upper tail is affine there—as a conditional exponential tail is—and, unlike a fitted tail model, applies where no exceedance has yet been observed; §5.3 separates the anchor from the continuation it uses.

The residual is where the data enters, and one adaptation rate cannot serve it: residual quantiles over recent history react quickly but cannot resolve $p = 0.999$ from a short history, while anchored quantile trackers accumulate exceedance information at a controlled rate but respond slowly to a level break. TAILCAST therefore carries two adaptation scales of one residual rule—dyadic residual windows and anchored quantile trackers—and combines them by AdaHedge (de Rooij et al., 2014), whose rate is read from the observed mixability gap rather than tuned, so neither a memory length nor a step size has to be chosen. Two properties of the resulting layer hold for any finite stream: a weighted isotonic projection using the same level weights λ_p that define the reported loss returns a noncrossing ladder without increasing that score (Proposition 1), and the delay-aware comparison prices outstanding feedback through a total-variation term that grows largest exactly when labels arrive in bursts (Proposition 2).

Across 696 cases, 58 series, four frozen backbones, three horizons, three scenarios, and five levels, TAILCAST attains the lowest tail-weighted pinball loss on both benchmark groups—6.7% and 8.9% below the strongest of twelve calibrators—and is the lowest-loss method in all seven delay and missingness conditions we test, issuing finite noncrossing ladders at every evaluated origin where the worst comparator crosses in 84.5% of Benchmark A and 80.4% of Benchmark B cases. Two qualifications carry through §5: calibration and tail accuracy rank the methods differently, and the delay advantage is a slope rather than a constant, comparable to the most delay-tolerant comparator rather than uniquely flat.

Our contributions are as follows.

- **A native-anchored parameterization of the tail** (§4.1): a score-space anchor that inherits the frozen model's conditional location and spacing, returns native levels unchanged, and extends the profile above the grid by a log-survival continuation. Removing the anchor or its fitted slope costs 17.14% and 21.98% of the reported loss, above every control that changes how the residual is estimated or aggregated, with uniform weighting the sole exception because it removes the aggregation rule itself.
- **A two-scale residual rule with learning-rate-free aggregation** (§4.2–§4.3): two adaptation scales of one residual rule, combined per horizon and level by AdaHedge, requiring no choice of memory length or step size. Removing either family degrades the reported loss significantly, and collapsing the rate range to a single window and a single step is itself significantly worse, so the two scales are complementary rather than redundant.
- **Two provable properties of the output and feedback layers** (§3.2, §4.4; proofs in Appendix A): the noncrossing projection and the delay-aware comparison are deterministic bounds that hold for any finite stream, and both are checked empirically—$200\,000$ random ladders for the projection, every evaluated origin for the ladders issued.
- **An evaluation that resolves the tail by level, scenario, backbone, and feedback condition** (§5): a paired protocol in which every method sees identical anchors, release counters, and masks, reported per level, scenario, and backbone, with separate sweeps over delay, missingness, and bursty arrival.

2 RELATED WORK

Forecasting with frozen foundation models. Time-series foundation models are pretrained and applied zero-shot, so their predictive distribution is fixed at inference (Ansari et al., 2024a;b; 2025; Woo et al., 2024; Das et al., 2024; Google Research, 2025; 2026; Rasul et al., 2023; Garza et al., 2023), and benchmarks standardize archives and metrics for that setting (Godahewa et al., 2021; Shchur et al., 2025; Aksu et al., 2024). Since the native output is a finite set of levels, any requested level outside it must come from a post-hoc layer applied identically across backbones whose grids differ; TAILCAST consumes only the quantile grid, the targets, and the release timestamps.

Calibrating a probabilistic forecast. Scoring rules and calibration supply the vocabulary for judging a predictive distribution (Gneiting & Raftery, 2007; Gneiting et al., 2007; Gneiting, 2011), and quantile-weighted rules concentrate that judgment on the tail (Gneiting & Ranjan, 2011); recent work calibrates the tail levels specifically (Allen et al., 2025). Regression calibration rescales the levels a base model already supplies (Kuleshov et al., 2018), and conformal prediction converts a forecast into an interval with distribution-free coverage under exchangeability, through adaptive variants that track shift (Vovk et al., 2005; Romano et al., 2019; Gibbs & Candès, 2021; Zaffran et al., 2022; Gibbs & Candès, 2024; Xu & Xie, 2021; 2023; Lade et al., 2026) and relaxations that trade exchangeability for weaker assumptions (Barber et al., 2023; Guan, 2023). All of them calibrate a predictive distribution the base model already determines, whereas TAILCAST extends that distribution above the native grid into a coherent profile, the object our tail-weighted pinball loss scores.

Tail extrapolation above the native grid. Peaks-over-threshold extrapolation models the tail directly (Balkema & de Haan, 1974; Pickands, 1975; Coles, 2001), and quantile regression with flexible bases estimates the conditional quantile function (Koenker & Bassett, 1978; Meinshausen, 2006; Park et al., 2022; Pasche & Engelke, 2024). Both supply a tail shape above the grid at the price of scarce exceedances and discarded conditional information. Our anchor makes the opposite trade: it inherits the shape the frozen model implies and spends the sample on the residual.

Online aggregation, quantile tracking, and delayed feedback. Hedge-style aggregation gives sublinear regret when feedback is immediate (Cesa-Bianchi & Lugosi, 2006; Herbster & Warmuth, 1998), and AdaHedge removes the learning rate by reading it from the mixability gap (de Rooij et al., 2014). Multi-quantile tracking follows each level independently and is the closest relative of our tracker family (Angelopoulos et al., 2023; Ding et al., 2025); TAILCAST keeps that behaviour as one family of candidates, anchors it to the current native forecast, and aggregates it by a bounded-loss rule. Delayed feedback breaks the standard expert comparison, since the state that scores a label need not be the state that produced it (Joulani et al., 2013); our delay term prices that gap inside the bound.

Enforcing coherence. Isotonic regression and rearrangement are standard fixes for quantile crossing (de Leeuw et al., 2009; Chernozhukov et al., 2010). Since the weighting decides whether the projection can hurt the score it is meant to improve, we project with the level weights of the reported loss.

3 PROBLEM SETUP

We fix notation, the issue-time information available, and the evaluation protocol; §4 gives the estimator.

3.1 SETTING AND RELEASE ORDER

Origins are indexed by t, with issue timestamps $o_0 < \cdots < o_{n-1}$. At origin t and horizon h a frozen forecaster supplies native quantiles $\{\widehat{q}_{g,t,h} : g \in \mathcal{G}\}$ for the target $y_{u_{t,h}}$ realized at $u_{t,h}$, and we must issue a ladder over a requested level set $\mathcal{P} = \{p_1 < \cdots < p_P\}$ that may extend above the native endpoint $p_0 = \max \mathcal{G}$. Throughout, $\mathcal{P} = \{0.90, 0.95, 0.99, 0.995, 0.999\}$: Chronos-Bolt and TimesFM-2.5 support levels up to 0.90 and Chronos-2 up to 0.99, so the two highest requested levels lie outside every grid we consider.

We issue in score space. With $b_{t,h} = \widehat{q}_{p_0,t,h}$ the native upper boundary and $a_h > 0$ a scale estimated from fit-visible targets, the score and issued ladder are

$$z_{t,h} = \frac{y_{u_{t,h}} - b_{t,h}}{a_h}, \qquad Q_{t,h,p} = b_{t,h} + a_h \widetilde{Q}_{t,h,p}. \tag{1}$$

The map is positive and affine, so it preserves ordering and the residual sign. All methods share $b_{t,h}$ and a_h, so differences between them reflect how the profile is formed. The scale a_h is the interquartile range of fit-visible scores, with a standard-deviation fallback and a numerical floor, frozen at the end of the fit block.

Let $\mathcal{F}_t$ be the frozen quantiles and the labels released by origin t; an issued ladder must be $\mathcal{F}_t$-measurable. Let $r_{t,h}$ be the feedback timestamp of the outcome for origin t and horizon h, o the

ordered issue timestamps, and $D \geq 0$ an additional delay in origins. The release counter of the token issued at origin t is

$$R_{t,h} = \max\{t+1,\ \mathrm{searchsorted}_{\mathrm{left}}(o, r_{t,h}) + D\}. \tag{2}$$

The max enforces causality: a token is never released before the origin after its issue. Before issuing at origin t the engine processes every pending token with $R_{i,h} \leq t$, ordered by $(R_{i,h}, \text{insertion index})$, scoring each against the state stored at its issue while the update advances the *current* estimator, so a late label is never credited to the issued profile with information unavailable when it was issued. A token whose feedback never arrives is never released and contributes no update, remaining in the evaluation mask; Figure 7 draws the geometry.

3.2 Evaluation protocol and metrics

For a stream of n origins let $F = \lfloor 0.4n \rfloor$ and $C = \lfloor 0.6n \rfloor$. The first F origins are the fit block: initialization uses only valid records $i < F$ whose feedback is visible by the fit boundary, $R_{i,h} \leq F$, and the resulting scale and state are frozen. Origins $F, \ldots, C-1$ are a warm-up block replayed through the same release queue, contributing state but no evaluation loss. Evaluation begins at C, with a mask set by target validity alone, independent of feedback arrival. The split is fixed before any method is applied, so all methods in a case see identical anchors, release counters, and masks.

For horizon h, let $\mathcal{E}_h$ be the evaluation origins and $N_h = |\mathcal{E}_h|$. With $\widehat{C}_{h,p} = N_h^{-1} \sum_{t \in \mathcal{E}_h} \mathbf{1}\{y_{u_{t,h}} \leq Q_{t,h,p}\}$ we report the *profile calibration error* and *tail-weighted pinball loss*

$$\mathrm{PCE}_h = \frac{1}{P} \sum_{p \in \mathcal{P}} \left| \widehat{C}_{h,p} - p \right|, \qquad \mathrm{TP}_h = \frac{1}{N_h} \sum_{t \in \mathcal{E}_h} \sum_{p \in \mathcal{P}} \lambda_p\, \ell_p\!\left(\frac{y_{u_{t,h}} - Q_{t,h,p}}{a_h} \right), \tag{3}$$

where $\ell_p(e) = \max\{pe, (p-1)e\}$ is the pinball loss and

$$\lambda_p = \frac{x(p)}{\sum_{q \in \mathcal{P}} x(q)}, \qquad x(p) = \log \frac{1}{1-p}. \tag{4}$$

PCE measures calibration alone; TP measures tail accuracy and is the score we optimize. The weights λ_p grow with the survival level, so a shallow error at 0.999 costs more than the same error at 0.90, the ordering a tail decision implies, and losses are never clipped. The task is thus to issue, at each origin and horizon, a nondecreasing ladder over $\mathcal{P}$ minimizing TP under the release order of Eq. (2). We also track the fractions of non-monotone ladders and finite entries, neither used for fitting. Reported numbers average horizons within a case, then cases with equal weight (one series, backbone, and scenario); uncertainty is a cluster bootstrap over data sources.

4 Method

TailCast is a post-hoc layer with four components: a *native anchor* that extends the frozen quantile grid into a score-space profile over $\mathcal{P}$; two *candidate families* that estimate the residual of that profile at different adaptation scales; a per-level *expert rule* that aggregates them from released pinball losses; and a weighted *isotonic projection* that returns the issued ladder. Each issued token saves its anchor, candidates, and weights, so a release scores exactly what was issued and no component reads an outcome that has not been released. All rules are stated in the score space of Eq. (1); §4.1–§4.3 develop the issue-time stages in the order they run, §4.4 states the feedback guarantee, and Figure 2 (Appendix C) traces both phases.

4.1 A native anchor and its residual

The frozen model already encodes a conditional shape, so the cheapest estimate of a level above its grid is a continuation of that shape rather than a fresh estimate from the residual sample. Write $x(p) = \log\{1/(1-p)\}$ and let $g^- < p_0$ be the second-highest native level. With $\bar{g}(p) = \max\{g \in \mathcal{G} : g \leq p\}$,

$$\nu_{t,h,p} = \begin{cases} \left(\widehat{q}_{\bar{g}(p),t,h} - b_{t,h}\right)/a_h, & p \leq p_0, \\ \dfrac{\left[\widehat{q}_{p_0,t,h} - \widehat{q}_{g^-,t,h}\right]_+}{a_h \left[x(p_0) - x(g^-)\right]} \left[x(p) - x(p_0)\right], & p > p_0. \end{cases} \tag{5}$$

Below the endpoint the anchor is the native predecessor quantile, so a level the backbone already supports passes through unchanged. Above it, the anchor converts the spacing between the two highest native quantiles into a slope per unit of x and extends it to p. If, conditionally on the issue information, the upper quantile function is affine in $x(p)$ above g^-—which includes a conditional exponential survival tail—and the two highest native quantiles are correct, Eq. (5) returns the correct level for every $p > p_0$ (Appendix A), and §5.3 tests that assumption directly by replacing the continuation with a flat endpoint.

Two properties of the anchor matter for the rest of the design. Its slope is non-negative by construction, so the anchor cannot introduce a crossing of its own. And because it depends only on the frozen forecast, it is available at issue time even where the residual sample contains no exceedance at all. The estimator therefore learns the level-specific residual $e_{i,h,p} = z_{i,h} - \nu_{i,h,p}$ of a released token, comparable across series and backbones because the anchor has absorbed the conditional location and the model's own spacing. A natural family of candidates is thus a location-shifted anchor $V = \nu + \hat{e}$, and the design question is how $\hat{e}$ should be estimated.

4.2 Two adaptation scales of one residual rule

The residual distribution is locally stable, so recent residuals inform the next one, and it drifts, so a long history built under a different regime misleads. No single memory length serves both, and at the extreme level the tension is a hard constraint: a window of m residuals resolves a p-quantile only through the conformal rank $k = \min\{m, \lceil (m+1)p \rceil\}$, which at $p = 0.999$ needs $m \geq 999$ before the rank stops being clipped at the sample maximum.

The first family uses the dyadic histories $\mathcal{W} = \{32, 64, 128, 256, 512, 1024, \infty\}$, spanning short through growing memory. On a window of m released residuals, with $e_{(k)}$ the kth order statistic,

$$V^{(m)}_{t,h,p} = \nu_{t,h,p} + e^{(m)}_{(k)}, \qquad k = \min\big\{m, \lceil (m+1)p \rceil\big\}, \tag{6}$$

and an empty history contributes the bare anchor. Clipping the rank to the largest observed residual makes the family safe at high levels and unresponsive at the same time, since at $p = 0.999$ the clipped candidates for short windows coincide; only the expanding candidate $m = \infty$ accumulates enough residuals to resolve the top level without clipping, and it is correspondingly slow to move.

The second family adapts the offset directly. For step scales $\alpha_j = s_0 2^{-j}$, $j = 0, \ldots, 5$, with s_0 the robust scale of fit-visible scores, tracker j predicts $V^{(j)}_{t,h,p} = \nu_{t,h,p} + \theta_{j,h,p}$ and on the release of token i updates

$$\theta_{j,h,p} \leftarrow \theta_{j,h,p} + \alpha_j \Big\{ \mathbf{1}\big[z_{i,h} > V^{(j),\mathrm{iss}}_{i,h,p}\big] - (1-p) \Big\}. \tag{7}$$

The increment is an exceedance residual: positive when the score came in above the issued level, negative when it fell below, and zero on average when the level is correct. Summing Eq. (7) over releases gives an exact identity (Appendix A), so a bounded state implies that the tracker's own exceedance frequency converges to $1 - p$. The dyadic ladder of rates spans a factor of 32 in response speed: the fastest tracker reacts to a local change, while the slowest integrates a sustained level error that a window of the same length would forget.

Neither family dominates—windows need no step size but cannot extrapolate past their rank, trackers move continuously but must be told how fast—so TailCast treats them as $7 + 6 = 13$ candidates of one residual rule and lets the released losses decide, with no memory length or step size to set. §5.3 shows that removing either family costs accuracy.

4.3 Released-loss aggregation and ordered output

For candidate k, level p, and a released token i, the loss is $\ell_{i,k,p} = \ell_p(z_{i,h} - V^{\mathrm{iss}}_{i,h,k,p})$, evaluated on the stored candidate so that the loss corresponds to what was issued. A separate controller is maintained per horizon and level, with cumulative released loss $L_{r-1,k,p}$ and mixability gap $\Delta_{r-1,p}$; before the rth release,

$$u_{r,k,p} \propto \exp\big(-\eta_{r,p} L_{r-1,k,p}\big), \qquad \eta_{r,p} = \begin{cases} \infty, & \Delta_{r-1,p} = 0, \\ \log K / \Delta_{r-1,p}, & \Delta_{r-1,p} > 0, \end{cases} \tag{8}$$

with $K = 13$; at infinite rate the ties among the minimum cumulative losses share the weight equally. This is AdaHedge (de Rooij et al., 2014). The rate is a measurable property of the stream rather than a hyperparameter: when the candidates agree the gap is small and the aggregate concentrates on the leader, and when they disagree it hedges. The realized rates vary by less than a factor of two across levels (Appendix B).

At issue, the stored weights mix the candidates into $v_{t,h,p} = \sum_k w^{\mathrm{iss}}_{t,h,p,k} V_{t,h,k,p}$, which need not be monotone in p because different candidates can rank the levels differently. We therefore issue

$$\widetilde{Q}_{t,h,\cdot} = \Pi_{\uparrow,\lambda}\big(v_{t,h,\cdot}\big), \qquad Q_{t,h,p} = b_{t,h} + a_h \widetilde{Q}_{t,h,p}, \tag{9}$$

using the same weights λ_p that define the reported loss. The guarantee below makes that projection free with respect to the reported score.

Proposition 1 (Weighted pinball contraction)**.** *Let $p_1 < \cdots < p_P$ and $\lambda_p > 0$, and let $\Pi_{\uparrow,\lambda}(v) = \arg\min_{q_1 \le \cdots \le q_P} \sum_p \lambda_p (q_p - v_p)^2$. Then for every $v \in \mathbb{R}^P$ and every outcome $z \in \mathbb{R}$,*

$$\sum_{p \in \mathcal{P}} \lambda_p \, \ell_p\big(z - [\Pi_{\uparrow,\lambda}(v)]_p\big) \;\le\; \sum_{p \in \mathcal{P}} \lambda_p \, \ell_p(z - v_p). \tag{10}$$

The proof (Appendix A) uses the block structure of the isotonic solution: within each constant block the fitted value is the λ-weighted mean, so the convex part of the pinball loss contracts by Jensen's inequality, and the pooling condition makes the remaining linear part non-positive after summation by parts. The bound constrains the aggregate over levels and is tied to λ, so it does not extend to the unweighted projection, a difference §5.3 quantifies in realized loss: weighting buys the guarantee, not the score. Because the guarantee covers the aggregate and not each level separately, we apply the projection without a further safeguard. The ladder is finite whenever the anchor, scale, and candidates are finite, and noncrossing by construction.

4.4 Delayed feedback

Under Eq. (2), the state that scores a token is not the state that issued it, so the standard expert comparison does not apply directly. Index the token-bearing releases of one horizon after gate initialization by r, let $u_{r,p}$ be the weights immediately before the rth release and $w^{\mathrm{iss}}_{r,p}$ the weights stored when those tokens were issued, set $B_{r,p} = \max_k \ell_{r,k,p} - \min_k \ell_{r,k,p}$, and write $\mathrm{TV}(w, u) = \frac{1}{2}\sum_k |w_k - u_k|$.

Proposition 2 (Delay-aware expert comparison)**.** *For any such finite release sequence, the total weighted pinball loss of* TailCast *on the released tokens obeys*

$$\sum_r \sum_p \lambda_p \ell_p\big(z_r - \widetilde{Q}_{r,p}\big) - \sum_p \lambda_p \min_k \sum_r \ell_{r,k,p} \;\le\; \sum_p \lambda_p \Big(2\Delta_{R,p} + \sum_r B_{r,p}\, \mathrm{TV}\big(w^{\mathrm{iss}}_{r,p}, u_{r,p}\big)\Big). \tag{11}$$

The proof combines the AdaHedge potential argument on the release sequence of one controller, convexity of the pinball loss—which bounds the loss of the issued mixture by its candidate-weighted value—and a total-variation bound between the issued and current weight vectors; Proposition 1 transfers the comparison through the projection (Appendix A). The second term is the price of outstanding feedback: when a burst of labels arrives at once, the aggregate moves far from the stored weights, and the term grows with that movement. It is therefore a design quantity rather than a nuisance—under heavy delay, hedging matters more than concentration—and §5.4 measures the resulting slope directly. The proposition covers released tokens from initialization onward and makes no claim about tokens whose outcomes never arrive, nor about coverage under drift. The implementation respects the same boundary: Appendix B verifies that predictions issued before a cutoff are unchanged when later outcomes are perturbed, and that never-released outcomes cannot influence any prediction.

5 Experiments

We evaluate three questions: whether TailCast attains the lowest tail-weighted pinball loss against the twelve comparators (§5.2), which design decisions produce that result (§5.3), and how the layer behaves under delay, missingness, and cost pressure (§5.4); §5.1 fixes the benchmarks, backbones, and comparators.

Table 1: Main comparison on both benchmark groups. TP is the tail-weighted pinball loss and ΔTP the excess over TAILCAST; PCE is the profile calibration error; cross is the percentage of cases in which a method issues a non-monotone ladder. Lower is better for all four. Methods are sorted by TP on Benchmark A; the ordering of Benchmark B differs, as §5.2 notes. † marks a paired cluster-bootstrap interval over data sources that excludes zero. The uncalibrated grid (*Native*) repeats its top quantile above the grid.

	Benchmark A (276 cases)			Benchmark B (420 cases)			
Method	TP	ΔTP	PCE	TP	ΔTP	PCE	cross (%)
TAILCAST	0.05715	0.00000	0.01377	0.06254	0.00000	0.01436	0.0
Rolling	0.06124	0.00409†	0.02238	0.06869	0.00615†	0.02247	0.0
MultiQT	0.06284	0.00569†	0.00920	0.07243	0.00989†	0.00718	0.0
TCP-RM	0.06298	0.00582†	0.01875	0.08963	0.02708†	0.02297	11.7
LC-CP	0.06334	0.00618†	0.02561	0.08916	0.02662†	0.03210	0.0
TCP	0.06351	0.00635†	0.02420	0.07321	0.01067†	0.02416	0.0
IQF	0.06856	0.01141†	0.03419	0.09040	0.02786†	0.03596	0.0
POT-GPD	0.06896	0.01180†	0.03027	0.09344	0.03089†	0.03820	0.0
ISQF	0.06899	0.01183†	0.03465	0.09053	0.02799†	0.03640	0.0
SPCI-QRF	0.08495	0.02780†	0.05116	0.10906	0.04652†	0.05115	0.0
BC-ACI	0.10660	0.04944†	0.08312	0.12263	0.06008†	0.10184	84.5
CondEVT	0.11692	0.05976†	0.05221	0.13368	0.07113†	0.04896	0.2
Native	0.42788	0.37072†	0.36408	0.69948	0.63694†	0.40976	0.0

5.1 DATA, BACKBONES, AND COMPARATORS

Benchmark A holds 23 series of air quality, renewable power, and operational signals (Beijing multi-site (Chen, 2017) and city-level $\mathrm{PM}_{2.5}$; NYC taxi demand, a CPU utilization trace, and machine temperature from the Numenta Anomaly Benchmark (Lavin & Ahmad, 2015); aggregated renewable generation); Benchmark B holds 35 series of financial levels, streamflow discharge, and renewable meteorology across many sites. The groups are reported separately because they differ in autocorrelation and in how often the requested tail is realized. Each series is paired with four frozen backbones used exactly as released—Chronos-2, Chronos-Bolt-Small, TimesFM-2.5-200M, and TimesFM-3.0, context length 128, horizons $\{1, 6, 24\}$—giving 92 frozen archives in Benchmark A and 140 in Benchmark B. Three scenarios are applied: *natural* keeps the original targets; *level shift* adds 1.5 times a fit-visible robust spread to evaluation targets; and *scale shift* maps $y \mapsto c + 2(y - c)$ for a fit-visible center c. Both shifts act on evaluation targets only, leaving forecasts, release counters, and masks untouched. The design yields 276 and 420 cases (696 in total), each at five levels and three horizons, under $D = 3$ with no added missingness.

Twelve comparators share the score space, spanning the main families of online distributional calibration; Appendix B lists them with their update rules. We single out no reference method: Table 1 orders them by tail-weighted pinball loss, and every margin below is measured against the best of the twelve on the group in question. Within a case all methods see the same anchors, release counters, and masks, and all intervals are paired cluster bootstraps over data sources.

5.2 MAIN COMPARISON

Table 1 reports both groups. TAILCAST attains the lowest tail-weighted pinball loss of the thirteen methods on each group—0.05715 on Benchmark A and 0.06254 on Benchmark B—and every margin in the table is supported by a paired interval that excludes zero. Rolling is the strongest comparator on both groups, at 0.06124 and 0.06869, with MultiQT next on each; TAILCAST is 6.7% below the best of the twelve on Benchmark A and 8.9% on Benchmark B, and 16.9% and 30.9% below the median comparator. The gap to the uncalibrated grid is an order of magnitude larger, because the two highest requested levels lie outside every native grid.

Below the two leaders the order is not stable across groups—TCP-RM is fourth on Benchmark A and sixth on Benchmark B, where LC-CP and TCP overtake it—so per-case statistics complement the averages. TAILCAST places first in 49.3% and 46.0% of individual cases and in the top three in

72.5% and 78.3%, with mean ranks 2.65 and 2.40 against the twelve comparators. Calibration and tail loss rank the methods differently: MultiQT attains the lower profile calibration error, 0.00920 on Benchmark A and 0.00718 on Benchmark Bagainst 0.01377 and 0.01436 for TAILCAST, yet loses on the weighted score. Ladder coherence separates the methods more sharply than either average: the worst offender issues a non-monotone ladder in 84.5% of Benchmark A cases and 80.4% of Benchmark B cases, whereas TAILCAST issues none, and every entry of its ladders is finite throughout.

5.3 DESIGN CONTROLS

We isolate each design decision on the 696 control cases, changing one thing at a time and keeping everything else fixed. Appendix Figure 1b gives the ratio of paired mean losses to the complete method; Appendix Table 6 lists every control with its absolute difference and interval.

The anchor carries the largest effects. Dropping it—replacing the frozen model's profile by a zero anchor so that every candidate must estimate the level from residuals—costs 0.01035 (17.14%), and removing only the fitted slope of the log-survival continuation costs 0.01328 (21.98%). Both exceed the effect of removing either candidate family (0.00069, 0.00672), the single-scale collapse (0.00346), and every aggregation control except uniform weighting (0.01841), which discards the aggregation rule rather than the anchor. The continuation itself is tested directly by extending the top native quantile flatly, costing 0.00080: it helps, but by less than having an anchor at all.

The two adaptation scales are complementary rather than redundant. Removing the tracker family leaves 0.06109 and removing the windows leaves 0.06713, against 0.06041 for the complete method; both differences are significant and neither family reproduces the other's behaviour. The window-only pool is the better single-family variant by loss but has the worst calibration error of any control that keeps the anchor (0.01789), while the tracker-only pool is its mirror image: the best calibration error in the table (0.01247) and the worst loss among the scale controls (0.00672), worse than the single-scale collapse (0.00346). Figure 5 shows how the rule allocates weight without ever being told the requested level: at $p = 0.90$ the windows carry 76.0% of the mass, because exceedances are frequent enough for a short window to be informative while the trackers still lag, and the share falls to 48.7% at $p = 0.999$, where the windows spend most of their time clipped at their sample maximum and the accumulated exceedance signal in the trackers resolves the level.

The aggregation and output layers contribute smaller but consistent gains. Uniform weighting is the worst variant (0.01841), and follow-the-leader is clearly worse (0.00047), as expected when an aggregate commits to the current best candidate and is then punished by delay; fixed share is worse but not significantly so (0.00135), and a single gate shared across the five levels costs 0.00101, since it sees the same candidates and losses and can express only one confidence for all five levels. Removing the projection increases the loss by 0.00025 and produces crossings in 20.7% of cases, which is the effect Proposition 1 bounds. The unweighted projection also restores monotonicity and is marginally better here (-0.00005, a paired difference whose interval excludes zero but which is about a tenth of a percent of the loss), so weighting buys the guarantee rather than the score: only the λ-weighted form is covered by the proof, and the random-ladder check reports no increase for the unweighted form either (0 of 200 000 trials). The unclipped conformal rank $\lceil (m+1)p \rceil$ does not help either (0.00020): it extrapolates past the largest observed residual and over-shoots exactly where the sample is thinnest. Pool size is a second-order choice: halving the candidate count is statistically indistinguishable from the complete method (0.00003), while extending it is marginally better (-0.00033), so the seven-by-six pool trades that small effect for the per-origin cost of the extra candidates.

5.4 ROBUSTNESS AND COST

Figure 6 varies delay and missingness on 69 cases per condition over 7 conditions: no delay, $D \in \{1, 3, 6\}$ additional origins, 10% and 30% permanently missing feedback, and a burst condition in which labels arrive in batches. TAILCAST has the lowest tail-weighted pinball loss in every condition, by 6.3% at zero delay, 7.7% at $D = 6$, and 5.4% under burst, relative to the best remaining comparator. Its loss grows by 2.55% from zero delay to $D = 6$, against 4.10% for MultiQT and 2.21% for TCP, so the slope is comparable to the most delay-tolerant comparator rather than uniquely flat. Delay hurts every method, as Proposition 2 predicts through the total-variation term. Missing

feedback is comparatively benign for the loss, because a token whose label never arrives simply does not update the state while the evaluation mask is independent of feedback availability. The burst condition is the hardest, and it is also where our margin is thinnest in relative terms: the aggregate moves several steps at once, which is exactly the regime the total-variation term describes.

Calibration tells a complementary story. MultiQT keeps the lowest PCE throughout, by a wide margin at zero delay (0.00651 against 0.01252), and the burst condition inflates PCE for every method, to 0.02301 for TAILCAST. The two scores therefore disagree in a specific and interpretable way: a per-level tracker that reacts to every exceedance separately keeps the average level error small, and the same reactivity is what makes it slower to correct the extreme tail, which is what the weighted score measures. Appendix Table 7 gives the numeric values.

TAILCAST spends 0.573 ms predicting and 0.190 ms updating on Benchmark A, against 0.135 ms and 0.102 ms for MultiQT; CondEVT needs 0.470 ms and SPCI-QRF 1.389 ms per prediction (Appendix Table 8). The layer is therefore mid-field on cost, and the two-scale pool is the reason. All quantities are computed online with bounded memory—a FIFO for the window family, one scalar per tracker level, one weight vector per level and horizon—so the per-origin cost does not grow with the stream length.

6 CONCLUSION

We presented TAILCAST, a post-hoc layer that extends a frozen time-series foundation model above its native quantile grid without retraining it. The design inverts the usual tail layer: rather than estimate the tail from scarce exceedances, it treats the frozen profile as an issue-specific anchor—exact for a conditional exponential upper tail, and returning native levels unchanged—and learns only the residual around it, continuing the top native spacing in log-survival coordinates. Two adaptation scales of that residual are combined by a learning-rate-free expert rule, and the mixture is projected onto the nondecreasing cone under the weights of the reported loss, which the projection provably cannot increase.

Three findings carry across the evaluation. Across two benchmark groups, four backbones, three scenarios, five levels, and seven feedback conditions, TAILCAST attains the lowest tail-weighted pinball loss of the thirteen methods compared—by 6.7% and 8.9% relative to the best of the others on each group—and issues no crossing or non-finite ladder. Calibration and tail accuracy disagree: a per-level tracker attains the lower profile calibration error yet loses on the weighted score, so both are reported side by side rather than traded off implicitly. And the cost of delay is predictable: the total-variation term of Proposition 2 grows exactly when labels arrive in bursts, leaving a delay slope comparable to the most delay-tolerant comparator rather than uniquely flat.

Limitations. Three limitations bound these claims. The layer does not improve the backbone's conditional median, so a series on which the frozen profile is badly misspecified stays hard. Above the grid it inherits the anchor's assumption that the tail is affine in log-survival coordinates, and the flat-endpoint control measures how much that slope is worth. And the delay term grows with the distance between the issued weights and those that score them, so the guarantee is weakest precisely in the bursty regime that is hardest for every method we tested.

Future work. The anchor's continuation could use how the native spacing varies with the level rather than a single top-end estimate, and the aggregation rule could model the total-variation term explicitly instead of hedging against it. Because the decomposition assumes only that a backbone exposes a quantile grid, it extends naturally to other frozen predictors and to tail levels beyond those studied here.

REPRODUCIBILITY STATEMENT

Detailed algorithm descriptions, hyperparameters, and defaults are given in Appendix B, together with the executable protocol checks of Table 2. Code will be released upon acceptance. All methods are evaluated on identical stored forecasts, release counters, and evaluation masks, and no backbone was retrained or fine-tuned for any experiment in this paper. Appendices A–C document the proofs, the implementation and evaluation protocol, the checks, and the additional results.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks with required disclosure: writing parts of the experimental code that produces the results reported in §5 and Appendix B. We have not used generative AI tools to generate synthetic data, to develop theoretical models or conceptual frameworks, to formulate or prove mathematical claims, to propose or refine hypotheses, to design the research methodology or experiments, or to interpret results, and the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for the following tasks with recommended disclosure: editing the manuscript to improve readability and grammar, creating and modifying the scientific figures, and formatting references.

We have reviewed all AI-assisted work. The propositions and proofs of Appendix A were checked against their statements and against the executable protocol checks of Table 2; the AI-assisted code was validated against those checks and against the reproduction record of Appendix B; and all AI-edited text and figures were reviewed and corrected by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

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

Sam Allen, Jonathan Koh, Johan Segers, and Johanna Ziegel. Tail calibration of probabilistic forecasts. *Journal of the American Statistical Association*, 120(552):2796–2808, 2025. doi: 10.1080/01621459.2025.2506194. URL https://doi.org/10.1080/01621459.2025.2506194.

Anastasios N. Angelopoulos, Emmanuel J. Candès, and Ryan J. Tibshirani. Conformal PID control for time series prediction. In *Advances in Neural Information Processing Systems*, volume 36, pp. 23047–23074, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/47f2fad8c1111d07f83c91be7870f8db-Abstract-Conference.html.

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*, 2024a. ISSN 2835-8856. URL https://openreview.net/forum?id=gerNCVqqtR.

Abdul Fatir Ansari, Caner Turkmen, Oleksandr Shchur, and Lorenzo Stella. Fast and accurate zero-shot forecasting with Chronos-Bolt and AutoGluon. AWS Machine Learning Blog, December 2024b. URL https://aws.amazon.com/blogs/machine-learning/fast-and-accurate-zero-shot-forecasting-with-chronos-bolt-and-autogluon/. Official release article, December 2, 2024.

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 preprint arXiv:2510.15821, 2025. URL https://arxiv.org/abs/2510.15821.

A. A. Balkema and L. de Haan. Residual life time at great age. *The Annals of Probability*, 2(5): 792–804, 1974. doi: 10.1214/aop/1176996548. URL https://doi.org/10.1214/aop/1176996548.

Rina Foygel Barber, Emmanuel J. Candès, Aaditya Ramdas, and Ryan J. Tibshirani. Conformal prediction beyond exchangeability. *The Annals of Statistics*, 51(2):816–845, 2023. doi: 10.1214/23-AOS2276. URL https://doi.org/10.1214/23-AOS2276.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006. doi: 10.1017/CBO9780511546921. URL https://doi.org/10.1017/CBO9780511546921.

Song Chen. Beijing multi-site air quality. UCI Machine Learning Repository, 2017. URL https://archive.ics.uci.edu/dataset/501/beijing+multi+site+air+quality+data. Dataset.

Victor Chernozhukov, Iván Fernández-Val, and Alfred Galichon. Quantile and probability curves without crossing. *Econometrica*, 78(3):1093–1125, 2010. doi: 10.3982/ECTA7880. URL https://www.econometricsociety.org/doi/10.3982/ECTA7880.

Stuart Coles. *An Introduction to Statistical Modeling of Extreme Values*. Springer Series in Statistics. Springer, 2001. doi: 10.1007/978-1-4471-3675-0. URL https://link.springer.com/book/10.1007/978-1-4471-3675-0.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st 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.

Jan de Leeuw, Kurt Hornik, and Patrick Mair. Isotone optimization in R: Pool-adjacent-violators algorithm (PAVA) and active set methods. *Journal of Statistical Software*, 32(5):1–24, 2009. doi: 10.18637/jss.v032.i05. URL https://www.jstatsoft.org/article/view/v032i05.

Steven de Rooij, Tim van Erven, Peter D. Grünwald, and Wouter M. Koolen. Follow the leader if you can, hedge if you must. *Journal of Machine Learning Research*, 15(37):1281–1316, 2014.

Tiffany Ding, Isaac Gibbs, and Ryan J. Tibshirani. Calibrated multi-level quantile forecasting. arXiv preprint arXiv:2512.23671, 2025. URL https://arxiv.org/abs/2512.23671.

Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. TimeGPT-1. arXiv preprint arXiv:2310.03589, 2023. URL https://arxiv.org/abs/2310.03589.

Isaac Gibbs and Emmanuel Candès. Adaptive conformal inference under distribution shift. In *Advances in Neural Information Processing Systems*, volume 34, pp. 1660–1672, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/0d441de75945e5acbc865406fc9a2559-Abstract.html.

Isaac Gibbs and Emmanuel J. Candès. Conformal inference for online prediction with arbitrary distribution shifts. *Journal of Machine Learning Research*, 25(162):1–36, 2024. URL https://jmlr.org/papers/v25/22-1218.html.

Tilmann Gneiting. Quantiles as optimal point forecasts. *International Journal of Forecasting*, 27(2):197–207, 2011. doi: 10.1016/j.ijforecast.2009.12.015. URL https://doi.org/10.1016/j.ijforecast.2009.12.015.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Tilmann Gneiting and Roopesh Ranjan. Comparing density forecasts using threshold- and quantile-weighted scoring rules. *Journal of Business & Economic Statistics*, 29(3):411–422, 2011. doi: 10.1198/jbes.2010.08110. URL https://doi.org/10.1198/jbes.2010.08110.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E. Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society, Series B*, 69(2):243–268, 2007. doi: 10.1111/j.1467-9868.2007.00587.x. URL https://doi.org/10.1111/j.1467-9868.2007.00587.x.

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://openreview.net/forum?id=wEc1mgAjU-.

Google Research. TimesFM 2.5: Official release notes and inference documentation. TimesFM project repository, 2025. URL https://github.com/google-research/timesfm#update---sept-15-2025. Release dated September 15, 2025; accessed September 11, 2026.

Google Research. TimesFM 3.0: Official release notes and inference documentation. TimesFM project repository, 2026. URL https://github.com/google-research/timesfm. Release dated August 2026; accessed September 11, 2026.

Leying Guan. Localized conformal prediction: A generalized inference framework for conformal prediction. *Biometrika*, 110(1):33–50, 2023. doi: 10.1093/biomet/asac040. URL https://doi.org/10.1093/biomet/asac040.

Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32(2):151–178, 1998. doi: 10.1023/A:1007424614876. URL https://link.springer.com/article/10.1023/A:1007424614876.

Pooria Joulani, Andras Gyorgy, and Csaba Szepesvari. Online learning under delayed feedback. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1453–1461. PMLR, 2013. URL https://proceedings.mlr.press/v28/joulani13.html.

Roger Koenker and Gilbert Bassett. Regression quantiles. *Econometrica*, 46(1):33–50, 1978. doi: 10.2307/1913643. URL https://doi.org/10.2307/1913643.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2796–2804. PMLR, 2018. URL https://proceedings.mlr.press/v80/kuleshov18a.html.

Ankit Lade, Sai Krishna J., and Indar Kumar. Bias-corrected adaptive conformal inference for multi-horizon time series forecasting. arXiv preprint arXiv:2604.13253, 2026. URL https://arxiv.org/abs/2604.13253.

Alexander Lavin and Subutai Ahmad. Evaluating real-time anomaly detection algorithms: The Numenta anomaly benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 38–44. IEEE, 2015. doi: 10.1109/ICMLA.2015.141. URL https://github.com/numenta/NAB.

Nicolai Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(35): 983–999, 2006. URL https://jmlr.org/papers/v7/meinshausen06a.html.

Youngsuk Park, Danielle C. Maddix, François-Xavier Aubet, Kelvin Kan, Jan Gasthaus, and Yuyang Wang. Learning quantile functions without quantile crossing for distribution-free time series forecasting. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 8127–8150, 2022. URL https://proceedings.mlr.press/v151/park22a.html.

Olivier C. Pasche and Sebastian Engelke. Neural networks for extreme quantile regression with an application to forecasting of flood risk. *The Annals of Applied Statistics*, 18(4):2818–2839, 2024. doi: 10.1214/24-AOAS1907. URL https://doi.org/10.1214/24-AOAS1907.

James Pickands, III. Statistical inference using extreme order statistics. *The Annals of Statistics*, 3(1): 119–131, 1975. doi: 10.1214/aos/1176343003. URL https://doi.org/10.1214/aos/1176343003.

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 preprint arXiv:2310.08278, 2023. URL https://arxiv.org/abs/2310.08278.

Yaniv Romano, Evan Patterson, and Emmanuel Candès. Conformalized quantile regression. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/5103c3584b063c431bd1268e9b5e76fb-Abstract.html.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting. arXiv preprint arXiv:2509.26468, 2025. URL https://arxiv.org/abs/2509.26468.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005. doi: 10.1007/b106715. URL https://link.springer.com/book/10.1007/b106715.

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 41st 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.

Chen Xu and Yao Xie. Conformal prediction interval for dynamic time-series. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11559–11569. PMLR, 2021. URL https://proceedings.mlr.press/v139/xu21h.html.

Chen Xu and Yao Xie. Sequential predictive conformal inference for time series. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38707–38727. PMLR, 2023. URL https://proceedings.mlr.press/v202/xu23r.html.

Margaux Zaffran, Olivier Feron, Yannig Goude, Julie Josse, and Aymeric Dieuleveut. Adaptive conformal predictions for time series. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25834–25866. PMLR, 2022. URL https://proceedings.mlr.press/v162/zaffran22a.html.

A PROOFS

Proposition 1 (weighted pinball contraction). We work in score units; the affine map of Eq. (1) is positive, so it preserves the ordering of ladders and we may prove the statement for $b_{t,h} = 0$ and $a_h = 1$. Write the pinball loss as $\ell_p(z - q) = p(z - q) + (q - z)_+$ and let $q = \Pi_{\uparrow,\lambda}(v)$. An isotonic solution partitions the levels into consecutive blocks on which q is constant. Let B be such a block with fitted value c_B. Because c_B minimizes $\sum_{i\in B} \lambda_i(q_i - v_i)^2$ subject to a common value, c_B is the λ-weighted mean of v over B, so Jensen's inequality gives

$$\sum_{i\in B} \lambda_i (c_B - z)_+ \;\le\; \sum_{i\in B} \lambda_i (v_i - z)_+.$$

The pooling condition of isotonic regression states that for every proper prefix $B_{\le j}$ of a final block, $S_j = \sum_{i\in B_{\le j}} \lambda_i(v_i - c_B) \ge 0$, while the sum over the whole block vanishes. Summation by parts therefore yields

$$\sum_{i\in B} \lambda_i p_i (v_i - c_B) = \sum_{j\in B\setminus\{\max B\}} S_j\,(p_j - p_{j+1}) \;\le\; 0,$$

since $p_j < p_{j+1}$ and $S_j \ge 0$. Adding the two inequalities gives $\sum_{i\in B} \lambda_i \ell_{p_i}(z - c_B) \le \sum_{i\in B} \lambda_i \ell_{p_i}(z - v_i)$ for each block, and summing over blocks proves Eq. (10). The argument uses one common outcome z across all levels; it does not imply a decrease at each level separately, and indeed the weighted projection may move an individual level in the unfavorable direction. □

Proposition 2 (delay-aware comparison). Fix a horizon and let $H_{r,p} = \sum_k u_{r,k,p}\ell_{r,k,p}$ denote the release-time mixture loss for level p. For AdaHedge with a nonincreasing learning rate schedule, the potential argument (de Rooij et al., 2014) applied to the complete release sequence of one controller gives

$$\sum_{r=1}^{R} H_{r,p} - \min_k \sum_{r=1}^{R} \ell_{r,k,p} \;\le\; 2\Delta_{R,p}.$$

Let $w^{\mathrm{iss}}_{r,p}$ be the weights stored in the token released at step r. The pinball loss is convex in its prediction argument, so the issued mixture loses at most its candidate-weighted value, $\ell_p(z_r - \sum_k w^{\mathrm{iss}}_{r,k,p} V^{\mathrm{iss}}_{r,k,p}) \le \sum_k w^{\mathrm{iss}}_{r,k,p}\ell_{r,k,p}$. For a loss vector whose entries span a range $B_{r,p}$, the difference between its expectations under two probability vectors is bounded by $B_{r,p}\,\mathrm{TV}(w^{\mathrm{iss}}_{r,p}, u_{r,p})$. Combining the two bounds controls the issued mixture against the best fixed candidate on the released subsequence. Proposition 1 then replaces the issued mixture by the projected ladder $\widetilde{Q}_{r,p}$, because the projection cannot increase the weighted loss. Summing the three comparisons over levels and releases and collecting the constant $2\Delta_{R,p}$ per level proves Eq. (11). The statement is deterministic, applies to released tokens from gate initialization onward, and makes no claim about tokens whose outcomes are never released. □

Exactness of the anchor on an affine-in-x tail. Suppose that, conditionally on the information available at issue, the upper quantile function satisfies $q_p = q_{p_0} + \beta[x(p) - x(p_0)]$ for $p \ge g^-$ with $\beta \ge 0$, and that the two highest native quantiles $\widehat{q}_{p_0,t,h}$ and $\widehat{q}_{g^-,t,h}$ are correct. Then $\beta = (\widehat{q}_{p_0,t,h} - \widehat{q}_{g^-,t,h})/[x(p_0) - x(g^-)]$, and substituting into the continuation branch of Eq. (5) returns q_p for every $p > p_0$. A conditional exponential upper tail has this form, since its quantile is affine in $x(p) = \log\{1/(1-p)\}$. For other tail shapes the continuation is an approximation, which is why §5.3 and the flat-anchor row of Table 6 test it rather than assume it.

Tracker identity. For a fixed tracker (j, h, p), sum Eq. (7) over its first R token releases. Because α_j is fixed after the fit block,

$$\sum_{r=1}^{R} \Big\{ \mathbf{1}\big[z_{r,h} > V^{(j),\mathrm{iss}}_{r,h,p}\big] - (1-p) \Big\} = \frac{\theta_{R,j,h,p} - \theta_{0,j,h,p}}{\alpha_j}.$$

If the state remains bounded as R grows, the left side divided by R is $o(1)$, so the tracker's release-sequence exceedance frequency converges to $1-p$. Neither boundedness of θ nor the same conclusion for the aggregate follows from the update alone; the aggregate is a weighted mixture of thirteen such quantities, and its realized coverage is reported in Tables 4 and 5.

Table 2: Executable protocol checks. Each property is tested on the released archives or on random inputs as indicated.

Check	Quantity	Value
release-order measurability	cutoff origin	3000
missing-feedback isolation	outcomes suppressed	1395
finite ordered ladders	evaluated origins	4749
weighted projection	random trials	200 000
weighted projection	max weighted increase	0.0
weighted projection	unweighted increases	0
release bookkeeping	state advances	*pass*

Resolution of a finite window. Under exchangeability and an exogenous release schedule, the untruncated conformal rank $k = \lceil (m+1)p \rceil$ requires $k \leq m$, that is $m \geq p/(1-p)$, to have a finite endpoint. For $p = 0.999$ this requires $m \geq 999$, so every window in $\{32, \ldots, 512\}$ returns a clipped rank and cannot support a finite-sample $p = 0.999$ coverage claim; the expanding candidate is the only member of the family that can, once enough residuals have accumulated. This is the reason the rank is clipped rather than extrapolated, and the reason the tracker family is retained. □

B PROTOCOL, IMPLEMENTATION, AND CHECKS

Executable protocol checks. We verify the following properties directly on the released archives, with the results reproduced below. (i) *Release-order measurability*: perturbing every outcome whose release counter exceeds a cutoff origin leaves all predictions issued through that origin bitwise unchanged. (ii) *Missing feedback isolation*: altering outcomes that are never released leaves both the issued predictions and the evaluation mask unchanged; 1395 outcomes were suppressed in this check. (iii) *Finite ordered ladders*: every ladder issued by TAILCAST is finite and nondecreasing over 4749 evaluated origins. (iv) *Weighted projection*: over 200 000 random ladders and outcomes, the weighted projection never increased the weighted pinball loss (the observed maximum increase was exactly zero), while the unweighted projection increased it in 0 cases. The two are therefore not interchangeable as a guarantee, even though their losses on the evaluated streams differ by less than a tenth of a percent. (v) *Release bookkeeping*: a release advances the aggregation state from its current value; an implementation that restored issue-time weights instead would produce a different weight vector, which the check detects.

Fit, replay, and evaluation blocks, in full. For a stream of n origins the fit block is $F = \lfloor 0.4n \rfloor$ and the evaluation block begins at $C = \lfloor 0.6n \rfloor$; the intervening origins replay through the same release queue and contribute state but no evaluation loss. Initialization uses only valid records $i < F$ with $R_{i,h} \leq F$, so no label that would be released after the fit boundary enters the initial state. A horizon is admitted for a series only when at least eight such records exist, ordered by release counter before filling the histories. The split is computed from the archive alone, before any method is instantiated.

Exact candidate settings and defaults. The window family is $\{32, 64, 128, 256, 512, 1024, \infty\}$ with the finite-endpoint conformal rank clipped at the sample maximum; the largest finite window is the smallest power of two above the resolution limit derived in Appendix A. The tracker family uses $\alpha_j = s_0 2^{-j}$ for $j = 0, \ldots, 5$, where s_0 is the interquartile range of the fit-visible scores, with a standard-deviation fallback and a numerical floor when it degenerates. Each horizon and level has its own controller over $K = 13$ candidates, initialized with equal weight. The output uses the isotonic projection weighted by λ_p of Eq. (4). The level shift adds 1.5 times a fit-visible robust spread; the scale shift uses $c + 2(y - c)$ with a fit-visible center. The main protocol applies $D = 3$ with no artificial missingness; the burst condition releases labels in batches that follow the recorded lag.

Comparator details. MultiQT tracks per-level score quantiles online. Rolling takes a moving empirical quantile of recent scores. TCP and TCP-RM apply online conformal updates with fixed and modulated steps. LC-CP weights the conformal update by locality. BC-ACI uses a bounded conformal update. CondEVT couples the conditional profile with an extreme-value correction. POT-

Table 3: Tail-weighted pinball loss by frozen backbone (left block) and by horizon (right block) for the two benchmark groups.

Benchmark A (276 cases)								
Method	TFM-2.5	TFM-3.0	Bolt	Chronos-2	Method	h=1	h=6	h=24
TAILCAST	0.05515	0.05559	0.05731	0.06057	**TAILCAST**	0.03805	0.05391	0.07951
MultiQT	0.05723	0.05702	0.06124	0.07588	MultiQT	0.04186	0.05670	0.08996
Rolling	0.05626	0.05611	0.06241	0.07018	Rolling	0.04139	0.05716	0.08517
TCP	0.05733	0.05721	0.06522	0.07427	TCP	0.04337	0.05942	0.08772
TCP-RM	0.06174	0.06170	0.06105	0.06741	TCP-RM	0.04705	0.05901	0.08287
LC-CP	0.05830	0.05837	0.06586	0.07083	LC-CP	0.04183	0.05857	0.08962
Benchmark B (420 cases)								
Method	TFM-2.5	TFM-3.0	Bolt	Chronos-2	Method	h=1	h=6	h=24
TAILCAST	0.06341	0.06142	0.06299	0.06235	**TAILCAST**	0.04127	0.06733	0.07903
MultiQT	0.06662	0.06476	0.07018	0.08817	MultiQT	0.04640	0.07169	0.09921
Rolling	0.06892	0.06708	0.06566	0.07310	Rolling	0.04746	0.07219	0.08641
TCP	0.07295	0.07098	0.07059	0.07833	TCP	0.05128	0.07668	0.09167
TCP-RM	0.10771	0.10447	0.07130	0.07503	TCP-RM	0.07207	0.09258	0.10423
LC-CP	0.10394	0.10553	0.07054	0.07663	LC-CP	0.05814	0.09170	0.11764

GPD fits a generalized Pareto tail above a threshold. IQF, ISQF, and SPCI-QRF estimate the quantile function with spline or forest bases. Native reports the frozen grid, using its top quantile for every level above the grid. All methods receive the same score space, the same scale, the same fit and evaluation blocks, and the same release order within a case.

Per-backbone and per-horizon results. Table 3 reports the tail-weighted pinball loss split by frozen backbone and by horizon. TAILCAST is best on all four backbones and on all three horizons in both groups. The spread across backbones is informative: on Benchmark A the loss of TAILCAST ranges from 0.05515 for TimesFM-2.5 to 0.06057 for Chronos-2, a spread of 9.8% of the lower value, and the range across backbones is wider for most comparators. The horizon trend is driven by the forecast rather than by the calibrator: the loss at $h = 24$ is roughly twice the loss at $h = 1$ for most methods, and TAILCAST is the lowest-loss method at every horizon in both groups.

Uncertainty. All intervals in the paper are cluster bootstraps that resample data sources, computed on per-case losses. Figure 4 gives the paired interval of every comparator against TAILCAST on both groups, and each $Rolling$-led ranking in Table 1 is reproduced under that resampling.

Level-resolved tables. Tables 4 and 5 give coverage gaps and normalized pinball loss for every method and level, one table per group. The pattern of Figure 3 reproduces numerically: under the natural scenario the calibration gap of TAILCAST is 0.08 points at 0.90 and -0.20 at 0.999, while single-rate trackers lose substantially more at the intermediate levels and recover only at the extremes, where their fixed step saturates. The loss columns separate the methods most clearly at 0.999, the level that lies above every native grid in the study.

Control table. Table 6 lists every structural control with its paired interval and calibration error, in the same order as Figure 1b.

Realized aggregation rates. The median learning rate produced by Eq. (8) is 2.15, 2.20, 2.55, 2.68, and 3.38 at the five levels respectively, computed over the released-loss sequences of the control panel. The rates vary by less than a factor of two across levels and the maximum observed rate is two orders of magnitude above the median, which occurs when the candidates agree and the mixability gap is small. That is the behaviour the learning-rate-free rule is meant to produce: concentrated when the candidates agree, hedged when they do not, without a schedule to tune.

Reproduction record. The reported experiments were run with Python 3.12 and NumPy 2.1.3 on a 20-core CPU allocation with 20 worker processes and one BLAS thread per worker. Each case is identified by a hash of its archive digest, scenario, condition, method list, levels, and code revision,

Table 4: Level-resolved results on Benchmark A: coverage gap in percentage points (left block, zero is nominal) and normalized pinball loss (right block).

	Coverage gap (pp)					Normalized pinball loss				
Method	90	95	99	99.5	99.9	90	95	99	99.5	99.9
TAILCAST	-2.26	-1.64	-0.95	-0.71	-0.52	0.18602	0.12452	0.04651	0.03056	0.01247
Rolling	-4.49	-3.00	-1.39	-1.10	-0.91	0.19731	0.12868	0.04831	0.03317	0.01678
MultiQT	-0.90	-0.77	-0.70	-0.64	-0.61	0.22027	0.13715	0.04693	0.03045	0.01359
TCP	-5.15	-3.08	-1.08	-0.90	-1.30	0.20595	0.13267	0.04913	0.03369	0.01849
TCP-RM	-2.07	-2.05	-1.71	-1.49	-1.24	0.19628	0.12898	0.05032	0.03538	0.01953
LC-CP	-4.63	-3.35	-1.80	-1.51	-1.24	0.19807	0.13064	0.05056	0.03549	0.01912
POT-GPD	-0.41	-5.20	-2.28	-1.79	-1.24	0.21255	0.14597	0.05528	0.03800	0.02055
IQF	-6.74	-4.52	-2.32	-1.83	-1.09	0.22043	0.14383	0.05490	0.03762	0.01813
ISQF	-6.74	-4.51	-2.38	-1.90	-1.24	0.22043	0.14366	0.05515	0.03809	0.01905
SPCI-QRF	-6.42	-5.95	-4.69	-4.34	-3.87	0.22139	0.15459	0.07203	0.05607	0.04003
BC-ACI	-6.59	-7.75	-8.76	-9.04	-9.40	0.24129	0.16866	0.08821	0.07789	0.06906
CondEVT	-5.44	-4.53	-5.14	-5.08	-4.80	0.31514	0.24327	0.09593	0.07046	0.04567
Native	-32.63	-36.05	-37.05	-37.55	-37.95	0.48448	0.46119	0.41614	0.41429	0.41281

Table 5: Level-resolved results on Benchmark B: coverage gap in percentage points (left block, zero is nominal) and normalized pinball loss (right block).

	Coverage gap (pp)					Normalized pinball loss				
Method	90	95	99	99.5	99.9	90	95	99	99.5	99.9
TAILCAST	-2.22	-1.51	-1.08	-0.88	-0.69	0.18034	0.12793	0.05558	0.03848	0.01802
Rolling	-3.85	-2.89	-1.69	-1.36	-1.24	0.20186	0.13820	0.05873	0.04187	0.02136
MultiQT	-0.49	-0.55	-0.57	-0.58	-0.55	0.23333	0.15069	0.05870	0.04050	0.01851
TCP	-4.37	-3.06	-1.39	-1.17	-1.57	0.21474	0.14543	0.06169	0.04491	0.02410
TCP-RM	-2.04	-2.21	-2.33	-2.29	-2.09	0.21773	0.15665	0.08173	0.06500	0.04201
LC-CP	-4.40	-3.76	-2.83	-2.61	-2.36	0.22295	0.16201	0.08086	0.06225	0.03915
POT-GPD	-0.73	-5.61	-3.27	-2.73	-2.08	0.24048	0.17713	0.08219	0.06103	0.04048
IQF	-5.92	-4.55	-2.90	-2.45	-1.62	0.23960	0.17233	0.08055	0.06009	0.03495
ISQF	-5.92	-4.51	-2.96	-2.55	-1.79	0.23960	0.17188	0.08076	0.06046	0.03515
SPCI-QRF	-5.83	-5.59	-4.92	-4.66	-4.32	0.23377	0.17633	0.10019	0.08273	0.06444
BC-ACI	-7.50	-9.41	-10.98	-11.31	-11.68	0.27714	0.19687	0.10385	0.09067	0.07595
CondEVT	-5.35	-4.25	-4.77	-4.71	-4.44	0.34597	0.25507	0.11101	0.08630	0.06172
Native	-36.43	-40.17	-41.72	-42.22	-42.62	0.72094	0.72037	0.69276	0.69282	0.69286

and completed cases are reused on resubmission, so the panels are reproducible from the manifests. No backbone was retrained or fine-tuned for any experiment in this paper; the forecast archives were frozen before any calibrator was applied. Run manifests record every archive's SHA-256 digest, the scenario and condition list, the random seed, and the completion status, and the protocol checks of Table 2 are executed by a script in the released code.

C ADDITIONAL FIGURES

Table 6: Design controls, pooled over both benchmark groups and 696 cases. Each row changes one decision of TAILCAST and keeps everything else fixed. ΔTP is the paired difference to the complete method, so a negative value means a lower tail-weighted pinball loss. † marks a cluster-bootstrap interval that excludes zero.

Variant	TP	ΔTP	95% CI	PCE
extended ladder	0.06008	-0.00033†	[-0.00088, -0.00004]	0.01351
unweighted projection	0.06036	-0.00005†	[-0.00010, -0.00001]	0.01392
TAILCAST	0.06041	+0.00000	—	0.01413
coarse ladder	0.06044	+0.00003	[-0.00029, +0.00022]	0.01450
empirical rank	0.06060	+0.00020	[-0.00020, +0.00064]	0.01698
no projection	0.06065	+0.00025†	[+0.00016, +0.00041]	0.01426
follow-the-leader	0.06087	+0.00047†	[+0.00014, +0.00113]	0.01484
TAILCAST w/o trackers	0.06109	+0.00069†	[+0.00019, +0.00147]	0.01789
flat endpoint above grid	0.06121	+0.00080	[-0.00008, +0.00284]	0.01365
shared gate	0.06142	+0.00101†	[+0.00061, +0.00175]	0.01607
fixed share	0.06175	+0.00135	[-0.00081, +0.00636]	0.01449
single scale	0.06387	+0.00346†	[+0.00085, +0.00811]	0.01659
TAILCAST w/o windows	0.06713	+0.00672†	[+0.00041, +0.02148]	0.01247
no anchor	0.07076	+0.01035†	[+0.00206, +0.02549]	0.02109
no fitted slope	0.07368	+0.01328†	[+0.00092, +0.03686]	0.01858
uniform weights	0.07882	+0.01841†	[+0.00246, +0.05338]	0.03423

Table 7: Tail-weighted pinball loss (top) and profile calibration error (bottom) under seven feedback conditions, 69 cases each. Delay is additional beyond the recorded feedback timestamp; the burst condition releases labels in batches following the recorded lag.

	no delay	D=1	D=3	D=6	10% missing	30% missing	burst
TAILCAST	0.05642	0.05686	0.05731	0.05786	0.05724	0.05709	0.06255
MultiQT	0.06023	0.06047	0.06124	0.06270	0.06005	0.05867	0.06677
Rolling	0.06144	0.06183	0.06241	0.06305	0.06183	0.06031	0.06613
TCP	0.06438	0.06473	0.06522	0.06580	0.06529	0.06443	0.06948
TAILCAST	0.01252	0.01343	0.01423	0.01513	0.01421	0.01514	0.02301
MultiQT	0.00651	0.00648	0.00661	0.00675	0.00684	0.00760	0.01629
Rolling	0.01807	0.01851	0.01899	0.01973	0.01854	0.01722	0.02661
TCP	0.01931	0.01961	0.02001	0.02055	0.01883	0.01752	0.02802

Table 8: Mean wall-clock cost per call in milliseconds, averaged over cases of both groups. Prediction is the issue-time cost; update is the release-time cost. All methods run in the same process on the same archives.

	Benchmark A		Benchmark B	
Method	predict	update	predict	update
TCP-RM	0.001	0.006	0.001	0.005
Native	0.019	0.000	0.017	0.000
ISQF	0.033	0.044	0.029	0.043
IQF	0.056	0.004	0.052	0.004
TCP	0.088	0.002	0.081	0.001
BC-ACI	0.093	0.019	0.086	0.017
MultiQT	0.135	0.102	0.124	0.093
Rolling	0.331	0.003	0.315	0.003
CondEVT	0.470	0.059	0.430	0.054
LC-CP	0.486	0.002	0.465	0.002
TAILCAST	0.573	0.190	0.530	0.176
POT-GPD	0.722	0.006	0.657	0.006
SPCI-QRF	1.389	0.003	1.279	0.003

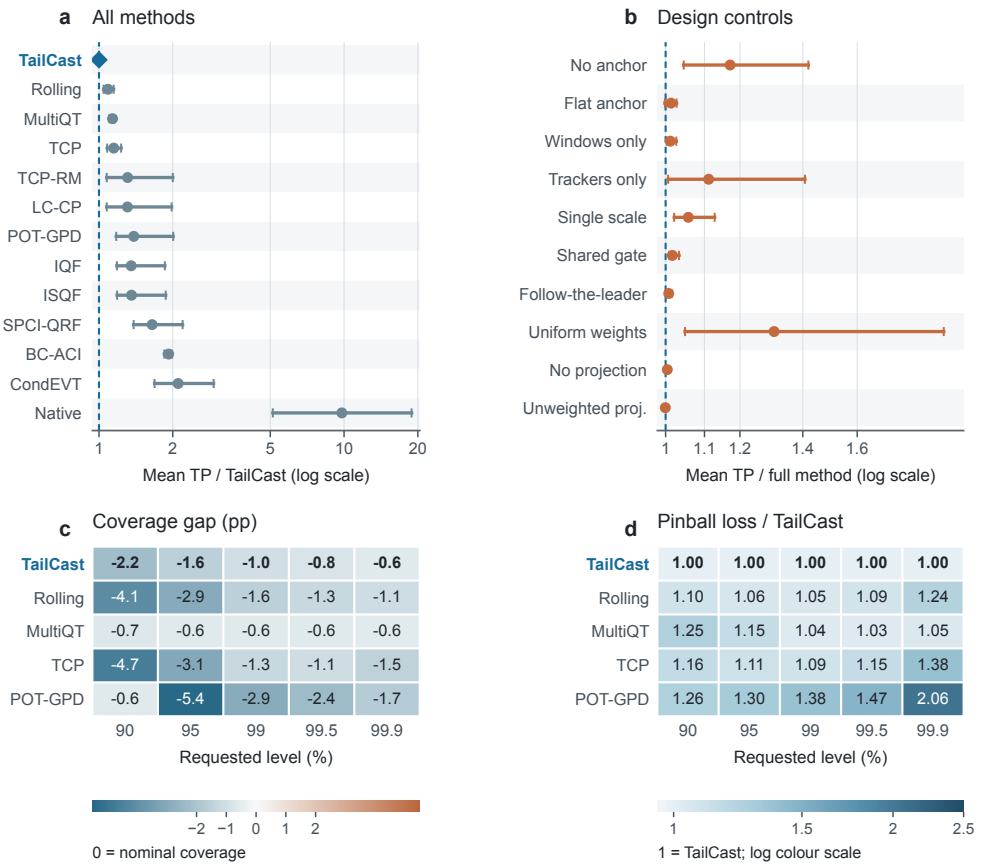


Figure 1: Core evidence. **(a)** Mean tail-weighted pinball loss (TP) relative to TAILCAST, pooled over 696 paired cases and nine data sources. **(b)** Mean TP of each design control relative to the complete method on 696 paired cases. Dots are ratios of case means; bars are percentile 95% intervals from 10,000 paired source-cluster bootstrap draws. Both horizontal axes are logarithmic, the reference is one, and all intervals are shown in full. **(c,d)** Coverage gap in percentage points and normalized pinball loss relative to TAILCAST at each level for five representative methods, pooled over both groups and all three scenarios. Zero gap is nominal; a loss ratio below one is better than TAILCAST. Horizons are averaged within cases and cases receive equal weight. Cells show the numerical means or ratios; (d) uses a log colour scale.

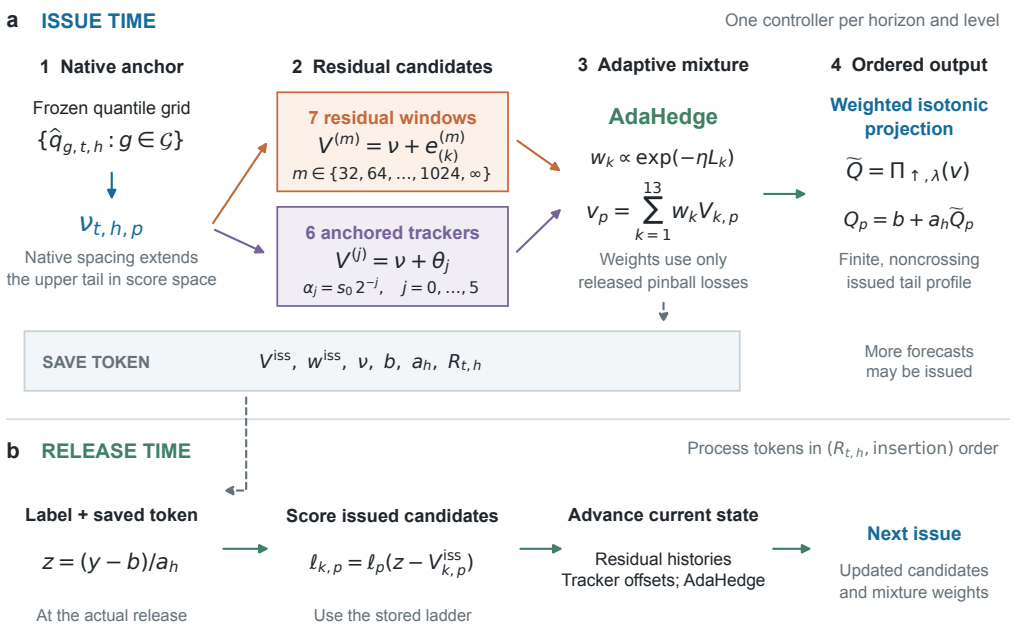


Figure 2: TAILCAST at **(a)** issue time and **(b)** release time. The frozen forecaster's quantile grid is converted to a score-space anchor shared by both candidate families: residual windows over dyadic histories of past residuals, and anchored quantile trackers with geometrically spaced step sizes. A per-level expert rule aggregates the thirteen candidates from released pinball losses, and a weighted isotonic projection returns the issued ladder. The token store is ordered by release counter; a released label scores the stored candidate ladders and then advances the current estimator for subsequent issues. Solid arrows denote computation; dashed arrows carry the issue-time snapshot through the token store. Time and horizon subscripts are suppressed in the candidate and aggregation formulas. No component reads an outcome that has not been released.

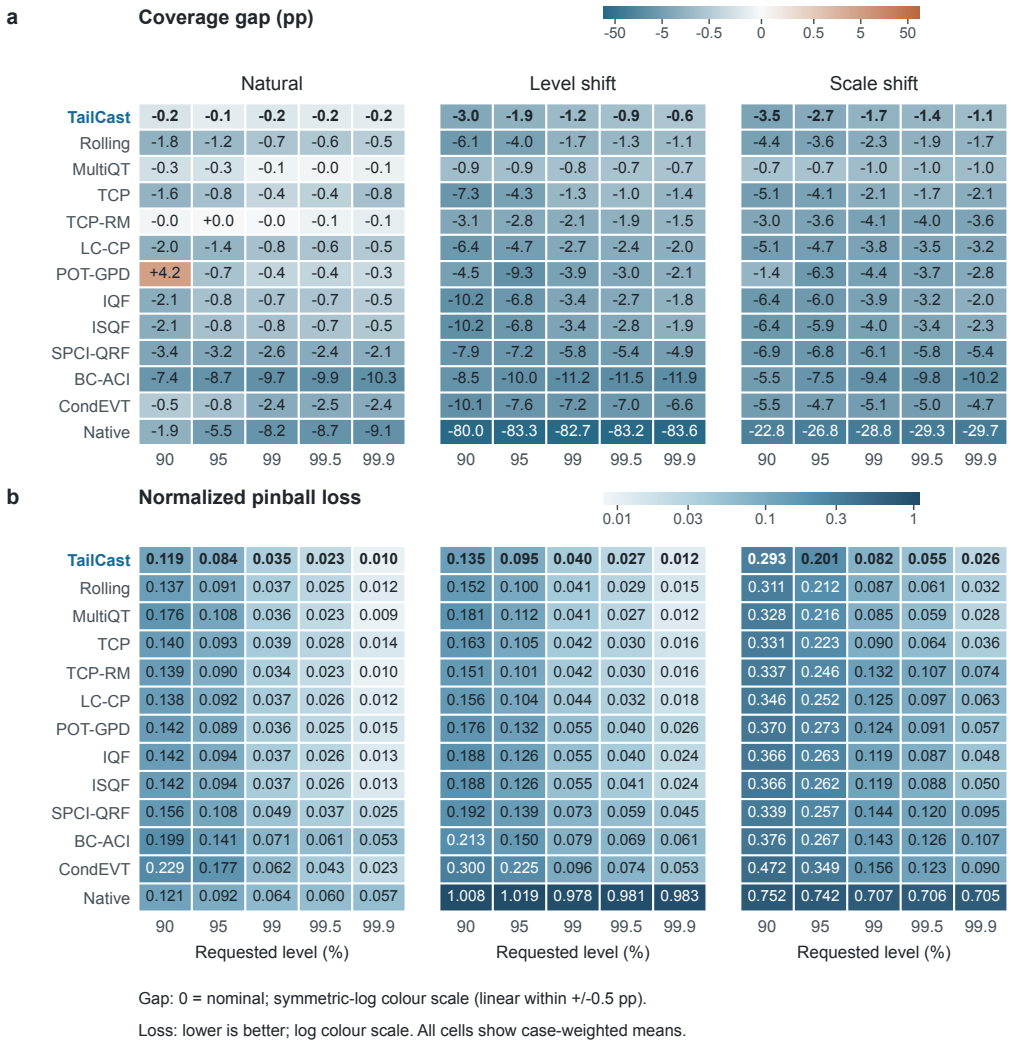


Figure 3: All thirteen methods at all five requested levels, pooled over both benchmark groups with one column per scenario (232 cases per scenario). **(a)** Coverage gap in percentage points; zero is nominal, negative values indicate undercoverage. The shared symmetric-log colour scale is linear within ± 0.5 percentage points. **(b)** Normalized pinball loss, with a shared log colour scale; lower is better. Each cell is the case-weighted mean after averaging horizons within cases. Numerical annotations preserve the values even where colours are similar. No methods, levels, or outlying means are omitted or clipped.

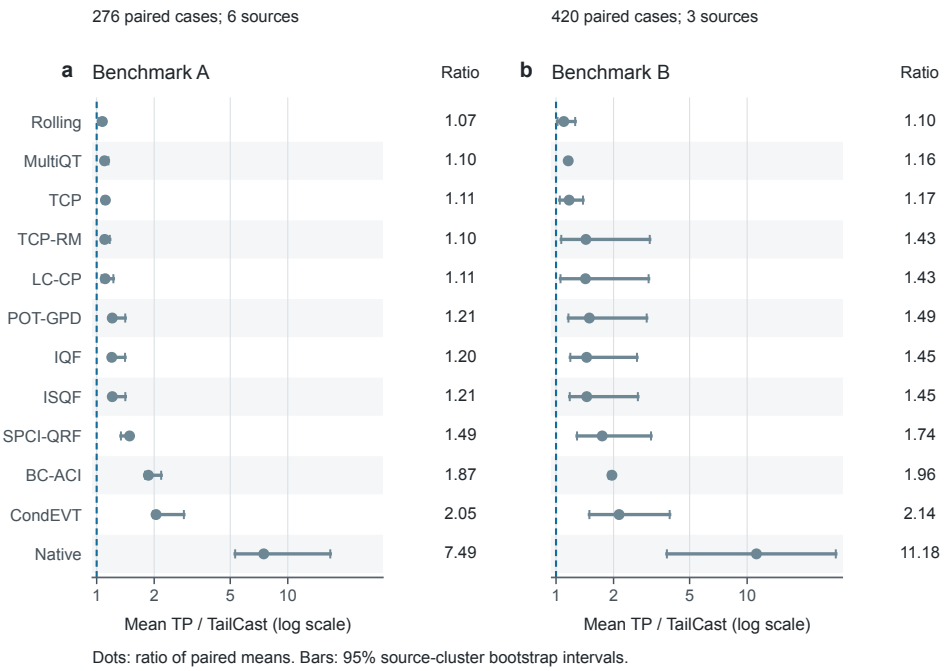


Figure 4: Paired comparisons within each benchmark group. Dots and numerical labels show each comparator's mean tail-weighted pinball loss divided by TAILCAST's mean over the same cases; bars show percentile 95% intervals from 10,000 source-cluster bootstrap draws, resampling both methods together. The common logarithmic axis includes every interval in full; one is the TAILCAST reference and larger ratios are worse. Benchmark A has 276 paired cases from six sources; Benchmark B has 420 from only three sources, which limits the source-level uncertainty assessment.

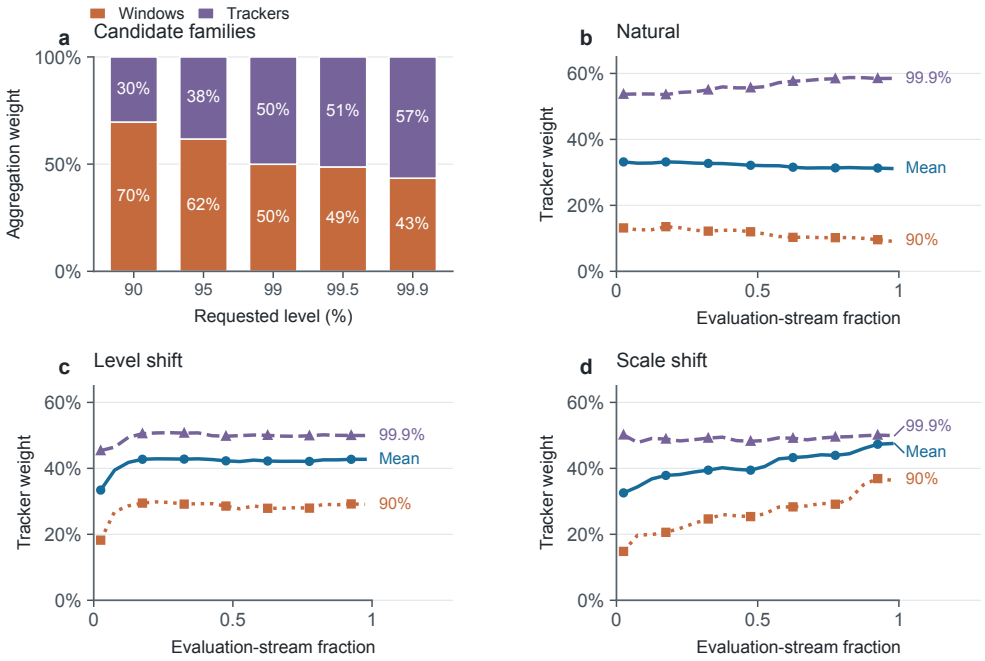


Figure 5: How the two adaptation scales are used. **(a)** Mean aggregation weights for the two candidate families on Benchmark A, averaging horizons within cases and then all 276 cases. Stacked bars show the full unit mass: windows dominate at 0.90, while trackers carry the majority at 0.999. **(b–d)** Tracker weights in the natural, level-shift and scale-shift scenarios from the released-feedback trace export, shown at the midpoints of 20 evaluation bins. Direct labels distinguish $p = 0.90$, $p = 0.999$ and the mean over all levels. The natural-scenario tail share rises while its all-level mean falls; shifted scenarios show different adaptation trajectories. Lines connect bin means and do not represent confidence intervals.



Figure 6: **(a)** Tail-weighted pinball loss and **(b)** profile calibration error across seven feedback conditions, with 69 cases per condition. Cells show equal-weight case means after averaging horizons; each metric has its own linear colour scale. Delay D is additional to the recorded feedback timestamp. TAILCAST has the lowest loss in every condition, while MultiQT has the lowest calibration error. Burst feedback increases calibration error for all four methods relative to the delay-only conditions.

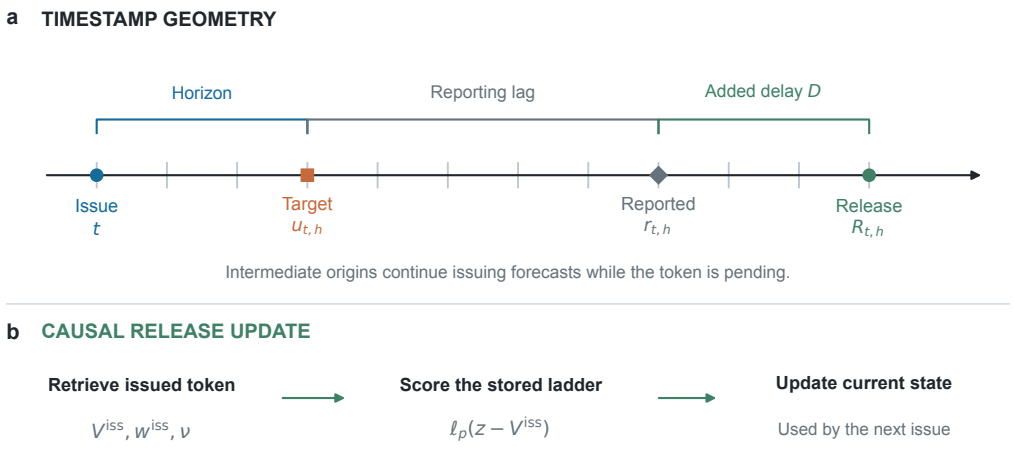


Figure 7: Release-order protocol. **(a)** Schematic timestamp geometry: forecasting horizon, reporting lag and additional delay D separate issue, target, report and release. The actual release counter follows Eq. (2); intervals in the schematic are not empirical durations. **(b)** On release, the engine retrieves the issue-time token, scores its stored candidates and advances the current estimator. Later issues use the updated state; predictions issued while feedback was pending are never rewritten.