A VERIFIER FROM THE PAST: TEST-TIME COMPUTE FOR TIME SERIES FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time compute (TTC) improves language models by spending additional inference compute to generate, verify, and select among candidate outputs, but this recipe has not transferred directly to time series forecasting. A language model can verify a proof or vote across samples, but a forecaster must be causal; it cannot use future information to score a candidate. We propose *Backtest-Gated Augmentation Averaging* (BGAA), a TTC algorithm for time series forecasting that builds a causal verifier from the series's own history, using rolling-origin backtests to score structured views of the context, discard those that underperform the untransformed context, and average the survivors' predictive distributions. BGAA improves every tested TSFM–benchmark combination, reducing MASE by 0.5–1.1% on GIFT-Eval, 3.2–5.9% on fev-bench, and 1.9–5.1% on BOOM, where a 313M model with TTC outperforms the 1B and 2.5B models in its family. We further show that useful views have structured, model-dependent effects consistent with a prior-transport interpretation. On the 53 cross-fittable GIFT-Eval configs, the deployed policy gains only 0.4–1.5% CRPS, versus 1.4–4.1% when choosing the best view for each dataset configuration and 17–19% when choosing the best view separately for each test window. This gap suggests that a major remaining challenge is window-level view selection.

1 INTRODUCTION

A time series foundation model (TSFM) is pretrained once but deployed across many diverse series. Improving performance usually requires larger models or more training. *Test-time compute* offers another route: keep the model frozen and spend additional inference-time compute to improve forecast accuracy. Language models do this by thinking longer (Nye et al., 2021; Wei et al., 2022) or wider, sampling and selecting among multiple candidates (Snell et al., 2025). Thinking wider requires a *verifier*, a signal for which candidate is better. Language modeling has several: program synthesizers run tests (Li et al., 2022), provers check steps, reward models score answers (Cobbe et al., 2021), and majority vote exploits agreement across correct chains (Wang et al., 2023).

These verifiers don't transfer to time series. A numeric forecast has no syntax to violate, nor tests to fail. A reward model would need labeled forecasts for the deployment series, which are unavailable at inference time. Without a causal verifier, TSFMs are deployed with a single forward pass.

A forecaster has something a language model does not: the series' *own past*. A language prompt provides only a single instance of the task, whereas a forecasting context contains earlier prediction problems with observed outcomes. We rebuild the deployment task from the context alone and score candidates by rolling-origin backtest, reading nothing past the last observed point.

We introduce *Backtest-Gated Augmentation Averaging* (BGAA), a test-time compute algorithm for time series forecasting. BGAA generates several structured views of the same context, obtains a candidate forecast from each using the same frozen model, and scores them using a rolling-origin backtest on the observed history. Candidates that outperform the untransformed input on the historical backtest are retained and aggregated. Because past performance is only a proxy for future accuracy, the verifier is a *surrogate*, but it is causal and available for every series at test time.

Why should structured views give different, *sometimes better*, forecasts from the same observations? Under a Prior Fitted Network idealization, a frozen TSFM predicts under a distribution induced by

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.

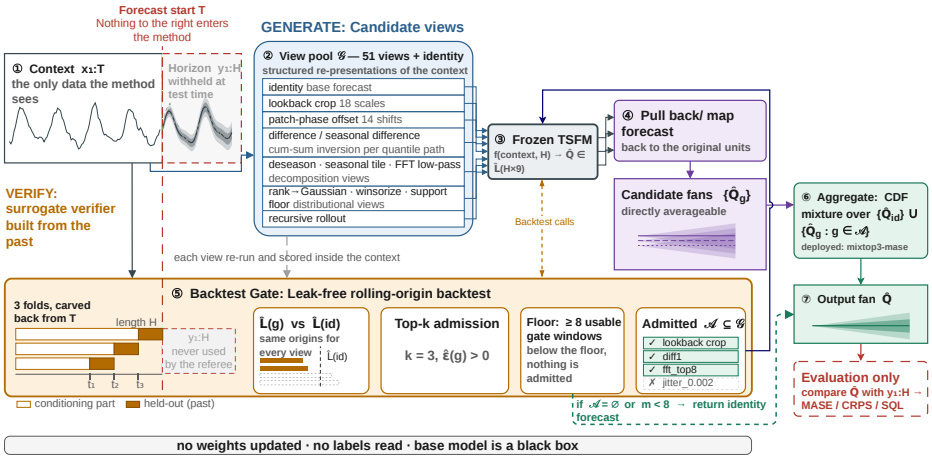


Figure 1: **BGAA system overview. (1)** The observed context is the only input available at test time. **(2)** A fixed pool of transformed views is generated, and **(3)** each is forecast with the same frozen TSFM. **(4)** Forecasts are mapped back to the original observation space. **(5)** A leak-free rolling-origin backtest scores the views against the base and admits the best candidates. **(6)** The admitted forecasts are aggregated with the base forecast to produce **(7)** the final predictive fan.

pretraining (Müller et al., 2022). For fixed pointwise, monotone transforms, forecasting a transformed context and mapping quantiles back is equivalent to prediction under a transported trajectory distribution (Proposition 1). Our deployed views are more general, so we treat this as an illustrative interpretation and test for systematic, model-dependent effects in Section 6.

Test-time compute on TSFMs needs safeguards against degradation. The base forecast remains a candidate, and the backtest acts as a *referee*: a view is admitted only if it beats the base on held-out windows (Section 3). This makes additional inference compute selective, since views are used only when the historical backtest indicates improvement over the base.

We test CHRONOS-2, TIREX-2, PATCHTST-FM-R1, and TOTO-2-313M on GIFT-Eval, BOOM, and fev-bench. All 12 model–benchmark combinations improve with no per-model re-tuning: 0.5 to 1.1% MASE on GIFT-Eval, 1.9 to 5.1% on BOOM, 3.2 to 5.9% on fev-bench. The gain often exceeds the spacing between model sizes: on BOOM, test-time compute on TOTO-2-313M beats the 1B and 2.5B models of its own family. On GIFT-Eval, the `high` TIREX-2 rung achieves lower MASE at lower inference FLOPs than TOTO-2-2.5B (Section 5). Simple view ensembling does not explain this gain: at matched compute on CHRONOS-2, random selection from the same pool worsens MASE by 4.8%, while BGAA improves it by 1.0%.

We also measure the remaining headroom. On the 53 GIFT-Eval dataset configurations that support cross-fitting, BGAA improves CRPS by 0.4–1.5%. Cross-fitted selection of the best single view per configuration raises this to 1.4–4.1%, while choosing the best view separately for each forecast window using its realized outcome gives 17–19%. This gap motivates better window-level verification, while leaving open gains from better candidate views.

Our contributions are four-fold:

1. **Test-time compute for TSFMs** (Section 3). BGAA needs only a quantile-forecast API, updates no weights, reads no future labels, and improves every model we test on every benchmark.
2. **Why augmented views help** (Section 6). Under the Prior Fitted Network idealization, fixed pointwise monotone views correspond to transported trajectory distributions. This motivates three predictions about structured, model-specific view effects, which Section 6 shows them for all four models.
3. **Evidence at scale** (Sections 5 and 6). Across 4 models and 3 benchmarks, we measure the accuracy–FLOPs frontier and show that the gains arise from structured, model-dependent view effects and effective backtest-based selection, rather than simply ensembling more forecasts.
4. **Verification headroom** (Section 7). Using cross-fitted and label-based diagnostics, we quantify the remaining gap at the window-level, showing substantial window-specific headroom.

2 RELATED WORK

Time series foundation models (TSFMs). TSFMs are models pretrained on large, heterogeneous corpora and deployed zero-shot. Given a context, they forecast an unseen series with no task-specific training. They differ in how series are tokenized and pretrained (Ansari et al., 2024; 2025; Wen et al., 2026; Cohen et al., 2025; Woo et al., 2024; Das et al., 2024; Kamarthi and Prakash, 2024; Goswami et al., 2024). We study a complementary question: how to improve forecasts from an already-trained TSFM by spending additional inference compute while keeping its weights fixed.

Test-time compute in language modeling. Self-consistency, best-of-n with a reward model, verifier-guided search and compute-optimal inference (Wang et al., 2023; Cobbe et al., 2021; Snell et al., 2025; Lightman et al., 2023) generate candidates and use different signals to select among them. Forecasting complicates candidate selection: the future observations needed to evaluate a forecast are unavailable at prediction time, and agreement among sampled forecasts does not by itself establish accuracy.

Test-time augmentation (TTA). TTA is standard in vision, where transformed inputs are evaluated and their predictions aggregated, with the aggregation rule as important as the augmentations (Chen et al., 2020; Shanmugam et al., 2021; Lyle et al., 2020). For forecasting, transformed contexts enable producing multiple candidate forecasts; the question remains which views help for a time series.

Forecast combination and label-free adaptation. Combining forecasts is a longstanding approach (Bates and Granger, 1969; Clemen, 1989; Timmermann, 2006), including the choice between averaging quantiles and probabilities (Lichtendahl et al., 2013; Gneiting and Raftery, 2007). Our use of forecast combination builds on this literature; the question here is which forecasts from transformed inputs to include in the combination. Conformal and recalibration methods use observed data to adapt predictive distributions or restore calibration (Gibbs and Candès, 2021; Stankevičiūtė et al., 2021), whereas our problem is candidate selection before aggregation.

Inference-time compute for frozen TSFMs. Two concurrent methods spend inference compute on a frozen forecaster. Hua et al. (2026) widen a saturating sample distribution by perturbing the input, but their main aggregator takes the median over all views, so harmful views are not filtered. Their selective variant ranks views against ground truth, and thus is limited to settings with labels. Pandey et al. (2026) instead edit a single context, suppressing the most disruptive patch when a meta-feature gate fires, with the gate fit by cross-validation over a *subset* of GIFT-Eval. BGAA instead selects among transformed-input forecasts using rolling-origin backtests on observed history, requiring neither future labels nor a separately trained cross-dataset gate (Section 3).

3 METHOD: BACKTEST-GATED AUGMENTATION AVERAGING

Setup and notation. We consider univariate forecasting for this work. Multivariate and covariate-aware forecasting are out of scope. Let $x_{1:T} = (x_1, \ldots, x_T) \in \mathbb{R}^T$ be the history observed up to the *forecast origin* T, the last observed time step (so T is also the history length); let H be the forecast horizon and $y_{1:H} = x_{T+1:T+H} \in \mathbb{R}^H$ the unobserved target. Such a history–target pair is a *forecast window*. Since time steps are counted within each series, history lengths vary across series and windows; we index windows by w, writing $x^{(w)}_{1:T_w}$, T_w, and $y^{(w)}_{1:H}$, and drop w when discussing a single window. A frozen forecaster f returns

$$f(x_{1:T}, H) = \widehat{Q} \in \mathbb{R}^{H \times |\mathcal{L}|}, \qquad \mathcal{L} = \{0.1, \ldots, 0.9\}, \tag{1}$$

a matrix with rows indexed by horizon steps $h \in \{1, \ldots, H\}$ and columns by quantile levels $\tau \in \mathcal{L}$; its entry $\widehat{Q}_{h,\tau}$ is the predicted τ-quantile at step h. We call $f(x_{1:T}, H)$ the *base forecast*.

A *view* g modifies how the history is presented to the forecaster or how its forecast is returned to the original forecast space. Formally, $g\colon f \mapsto f_g$ maps the frozen forecaster to a forecaster f_g with the same interface, $f_g(x_{1:T}, H) \in \mathbb{R}^{H \times |\mathcal{L}|}$. Let $\mathcal{G}$ be the fixed pool of candidate views, and $\widehat{Q}^g = f_g(x_{1:T}, H)$ the *candidate forecast* of view g, with entries $\widehat{Q}^g_{h,\tau}$ indexed as for $\widehat{Q}$. The *identity view* $\mathrm{id} \notin \mathcal{G}$ leaves the history and forecast unchanged, $f_{\mathrm{id}} = f$, so $\widehat{Q}^{\mathrm{id}} = \widehat{Q}$ is the base forecast.

A configuration, or *config*, c defines a finite set $\mathcal{W}_c$ of evaluation forecast windows w, all with the same horizon H. BGAA selects a single set of views $\widehat{\mathcal{A}}_c \subseteq \mathcal{G}$ per config and holds it fixed across

$\mathcal{W}_c$. Selection uses only the observations $\mathcal{O}_c$ available before the first forecast origin in $\mathcal{W}_c$, from which we sample a finite set $\mathcal{W}_c^{\text{gate}}$ of historical *gate windows*. This pools historical evidence across gate windows and amortizes selection cost across evaluation forecasts.

Problem definition. For a loss ℓ, BGAA seeks a forecaster $\tilde{f}$, chosen using only the frozen black-box f and $\mathcal{O}_c$, whose *relative risk* $\mathcal{R}_c$ (its mean per-window loss divided by the base's, so each window counts equally regardless of scale) satisfies

$$\mathcal{R}_c(\tilde{f}) = \frac{1}{|\mathcal{W}_c^+|} \sum_{w \in \mathcal{W}_c^+} \frac{\ell\Big(\tilde{f}(x_{1:T_w}^{(w)}, H), y_{1:H}^{(w)}\Big)}{\ell\Big(f(x_{1:T_w}^{(w)}, H), y_{1:H}^{(w)}\Big)} < \mathcal{R}_c(f) = 1, \tag{2}$$

where $\mathcal{W}_c^+ \subseteq \mathcal{W}_c$ contains the windows with positive base loss. Since their targets are unobserved at selection time, BGAA scores each view by the gate-window surrogate $1 - \widehat{\epsilon}(g)$ (Equation (4)).

Method overview. BGAA generates alternative forecasts from a frozen model, evaluates them on historical forecasting tasks, and combines selected alternatives with the base forecast. Each historical task holds out an already-observed segment and forecasts it using only preceding observations. These backtests select views once per config, before any evaluation window is observed. If historical evidence is insufficient or no alternative improves over the base, BGAA returns the base forecast. Figure 1 illustrates the pipeline, and Algorithm 1 summarizes the procedure.

Candidate generation. The candidate generator produces alternative forecasts for the same target horizon using the frozen forecaster. A *view family* specifies a type of modification, with different parameter settings defining individual views. For example, cropping is a family; retaining the latest 50% or 75% of the history defines two views. A view need not be an invertible input transformation: it may modify the context, post-process a forecast, or make multiple model calls, provided its output is returned to the same timestamps and units as the target. The fixed pool $\mathcal{G}$ contains lookback crops, patch-phase offsets, differencing and seasonal differencing, decomposition and frequency-based views, distributional transforms, recursive rollout, and small perturbations. The complete catalog and pull-back rules are in Section B.

Selection using historical backtests. To evaluate a view without using any evaluation-window target, we recreate the forecasting problem inside an observed history. We index gate windows as $\mathcal{W}_c^{\text{gate}} = \{w_1, \ldots, w_M\}$ and write $x^{(j)} = x^{(w_j)}$ for the observed history of w_j. Each gate window provides $n_f = 3$ backtests, one per rolling origin t_{jr}, $r \in \{1, \ldots, n_f\}$; at each origin, we forecast the next H observations using only $x_{1:t_{jr}}^{(j)}$. For the loss ℓ, instantiated below, the empirical loss of view g is

$$\widehat{L}_j(g) = \frac{1}{n_f} \sum_{r=1}^{n_f} \ell\Big(f_g(x_{1:t_{jr}}^{(j)}, H), x_{t_{jr}+1:t_{jr}+H}^{(j)}\Big). \tag{3}$$

For point selection, ℓ is the mean absolute error of the median forecast. Because the MASE scale factor is common to a view and the base within a window, their relative comparison is unchanged, so we call this the MASE gate. The distributional alternative uses mean pinball loss over $\mathcal{L}$, a finite-quantile surrogate for CRPS, and is called the CRPS gate.

Windows with $\widehat{L}_j(\text{id}) = 0$ are excluded because relative improvement would be undefined. With $\mathcal{J} = \{j \in \{1, \ldots, M\} : \widehat{L}_j(\text{id}) > 0\}$ and $m = |\mathcal{J}|$, each candidate view receives the score

$$\widehat{\epsilon}(g) = \frac{1}{m} \sum_{j \in \mathcal{J}} \frac{\widehat{L}_j(\text{id}) - \widehat{L}_j(g)}{\widehat{L}_j(\text{id})}. \tag{4}$$

Thus, $\widehat{\epsilon}(g) > 0$ means g improves over the base on average across usable historical windows.

The deployed rule selects up to the $k = 3$ views with the largest positive scores,

$$\widehat{\mathcal{A}}_c = \text{Top}_k \{g \in \mathcal{G} : \widehat{\epsilon}(g) > 0\} \quad \text{ranked by } \widehat{\epsilon}(g). \tag{5}$$

If $m < m_{\min} = 8$, or no view has positive improvement, $\widehat{\mathcal{A}}_c = \varnothing$ and the base forecast is returned. We also evaluate a margin rule that requires a minimum mean improvement and a minimum fraction of wins over $\mathcal{J}$; its full definition is in Section A.1.

Algorithm 1 BGAA with deployed top-k selection

Require: frozen forecaster f, views $\mathcal{G}$, horizon H

Selection, once per config

for sampled historical gate windows $w_j \in \mathcal{W}_c^{\text{gate}}$ **do**
  evaluate $\widehat{L}_j(g)$ for $g \in \mathcal{G} \cup \{\text{id}\}$ on n_f rolling origins
end for
discard windows with $\widehat{L}_j(\text{id}) = 0$
if $m < m_{\min}$ **then**
  $\widehat{\mathcal{A}}_c \leftarrow \varnothing$
else
  compute $\widehat{\epsilon}(g)$ and retain up to k highest-scoring views with $\widehat{\epsilon}(g) > 0$
end if

Application, per forecast window

$\widehat{Q}^{\text{id}} \leftarrow f(x_{1:T}, H)$
if $\widehat{\mathcal{A}}_c = \varnothing$ **then**
  return $\widehat{Q}^{\text{id}}$
end if
return $\text{Aggregate}\big(\widehat{Q}^{\text{id}}, \{\widehat{Q}^g : g \in \widehat{\mathcal{A}}_c\}\big)$

Forecast aggregation. Aggregation combines the base forecast and the candidate forecasts of $\widehat{\mathcal{A}}_c$, each with equal weight $\lambda_0 = 1/(|\widehat{\mathcal{A}}_c| + 1)$, into the *aggregate forecast*

$$\widehat{Q}^{\text{agg}} = \text{Aggregate}\big(\widehat{Q}^{\text{id}}, \{\widehat{Q}^g : g \in \widehat{\mathcal{A}}_c\}\big) \in \mathbb{R}^{H \times |\mathcal{L}|}.$$

Vincent averaging averages corresponding quantiles. **CDF mixture** converts each finite quantile fan to an interpolated CDF, averages the resulting distributions, and inverts the mixture at the nine levels in $\mathcal{L}$. We interpolate linearly between reported quantiles and extrapolate linearly in the tails using the slope of the nearest end segment, truncating the CDF at 0 and 1. Our deployed policy, `mixtop3-mase`, combines the MASE gate, top-3 admission, and CDF mixing; `top3-mase` uses the same selected set with Vincent averaging.

Properties of the aggregation rules. For an aggregation rule convex in the gate's loss, Theorem 2 shows that admitted historical improvements carry over to the aggregate on the same backtest windows. Let $f_{\text{agg}} \colon (x_{1:T}, H) \mapsto \widehat{Q}^{\text{agg}}$ denote the aggregate forecaster (BGAA's $\tilde{f}$), $\widehat{L}_j(\text{agg})$ its backtest loss, given by Equation (3) with f_{agg} in place of f_g, and

$$\widehat{L}_{\text{rel}}(\text{agg}) = \frac{1}{m} \sum_{j \in \mathcal{J}} \frac{\widehat{L}_j(\text{agg})}{\widehat{L}_j(\text{id})}$$

its mean backtest loss relative to the base. Under uniform weights, if $\epsilon' = \min_{g \in \widehat{\mathcal{A}}_c} \widehat{\epsilon}(g) > 0$ is the smallest admitted relative improvement, then $\widehat{L}_{\text{rel}}(\text{agg}) \le 1 - (1 - \lambda_0)\epsilon' \le 1$. This covers Vincent averaging under the MASE or CRPS gate, but not the deployed `mixtop3-mase` policy, because CDF-mixture quantiles are not convex combinations of member quantiles. The result bounds $\widehat{L}_{\text{rel}}(\text{agg})$ only; it does not bound $\mathcal{R}_c(f_{\text{agg}})$ or a benchmark's aggregation rule.

CDF mixing has a separate exact-CRPS property: for member CDFs F_i with weights $\lambda_i \ge 0$, $\sum_i \lambda_i = 1$, and an outcome $y \in \mathbb{R}$, $\text{CRPS}(\sum_i \lambda_i F_i, y)$ is at most the weighted average of the members' CRPS values (Theorem 3). This does not imply the mixture beats the base or the best member, and the identity applies to the full CDF before conversion back to nine quantiles. Full assumptions and proofs are in Section A.

Computational cost. BGAA separates selection from application. Selection evaluates the view pool on historical backtests once per config; the resulting $\widehat{\mathcal{A}}_c$ is reused across its forecast windows. Application must still recompute the base and admitted candidate forecasts for each new context–horizon pair. Individual views can require zero additional calls when they post-process a cached base forecast, one call for a transformed context, or multiple sequential or seeded calls for recursive or stochastic views. A cached base forecast is reusable only for the same input, not across forecast windows. The four *thinking modes* vary the number of candidate views, gate windows and rolling origins, and admitted views. Across the four models they span roughly 1.9–$9.4\times$ the cost of a single pass on GIFT-Eval (Table 5); exact forward-pass accounting is in Section C.1.

Table 1: **BGAA improves performance across all 12 model×benchmark combinations.** Frozen base versus the deployed `mixtop3-mase` policy, with no per-model or per-benchmark retuning. Boldface marks the lower score within each matched base–BGAA comparison; it does not indicate a ranking across models. Δ% reports relative change in MASE and the probabilistic metric, respectively; negative values indicate improvement.

benchmark	model	base (MASE / CRPS†)	BGAA (MASE / CRPS†)	p (MASE / CRPS†)	Δ% (MASE / CRPS†)	win/loss
GIFT-Eval	TiRex-2	0.6972 / 0.4791	**0.6934 / 0.4778**	<0.001 / 0.004	−0.54/ − 0.27	68/22
	Chronos-2	0.7043 / 0.4830	**0.6963 / 0.4780**	<0.001 / <0.001	−1.14/ − 1.04	69/20
	PatchTST-FM-R1	0.7052 / 0.4820	**0.7017 / 0.4812**	<0.001 / <0.001	−0.50/ − 0.17	70/18
	Toto-2-313M	0.7032 / 0.4838	**0.6968 / 0.4788**	<0.001 / <0.001	−0.91/ − 1.03	68/22
BOOM	TiRex-2	0.6418 / 0.3855	**0.6205 / 0.3772**	<0.001 / <0.001	−3.33/ − 2.15	3209/584
	Chronos-2	0.6547 / 0.3935	**0.6290 / 0.3821**	<0.001 / <0.001	−3.92/ − 2.90	3366/441
	PatchTST-FM-R1	0.6812 / 0.4071	**0.6463 / 0.3915**	<0.001 / <0.001	−5.12/ − 3.83	3389/412
	Toto-2-313M	0.5853 / 0.3532	**0.5740 / 0.3492**	<0.001 / <0.001	−1.94/ − 1.13	2942/790
fev-bench (MASE / SQL†)	TiRex-2	1.6192 / 1.3653	**1.5315 / 1.2908**	<0.001 / <0.001	−5.42/ − 5.46	70/23
	Chronos-2	1.6332 / 1.3416	**1.5373 / 1.2719**	<0.001 / <0.001	−5.87/ − 5.20	76/18
	PatchTST-FM-R1	1.5852 / 1.3150	**1.5114 / 1.2571**	<0.001 / <0.001	−4.65/ − 4.40	70/24
	Toto-2-313M	1.6652 / 1.3881	**1.6127 / 1.3319**	<0.001 / <0.001	−3.15/ − 4.05	68/23

4 Experimental setup

Models. We evaluate four public frozen TSFMs spanning different architectures and pretraining corpora: Chronos-2 (Ansari et al., 2025), TiRex-2 (Podest et al., 2026), PatchTST-FM-R1 (Wen et al., 2026), and Toto-2-313M (Khwaja et al., 2026), all evaluated in their univariate settings. Toto-2-313M is the 313M family member used in our model-scaling comparisons.

Benchmarks. GIFT-Eval (Aksu et al., 2024) contains 97 configs. BOOM (Cohen et al., 2025) contains 7413 dataset×term observability tasks, of which the official leaderboard evaluates a 6750-task scaled subset. fev-bench (Shchur et al., 2026) contains 100 forecasting tasks.

Metrics and statistics. We report MASE for point accuracy and CRPS on GIFT-Eval and BOOM, with SQL on fev-bench. GIFT-Eval uses the geometric mean of metric-to-seasonal-naive ratios, BOOM uses shifted geometric mean normalized by DataDog's seasonal-naive run, and fev-bench uses raw metric means and skill against its published seasonal-naive baseline. We report relative changes, paired win/loss counts, and paired Wilcoxon tests over matched benchmark configs/tasks.

Evaluated method. All main results use `mixtop3-mase`, with $k = 3$ and $m_{\min} = 8$. The candidate pool and global design choices were chosen once using Chronos-2 on a labelled 15-config subset of the GIFT-Eval validation split and then frozen. No test, BOOM, or fev-bench data were used for this choice, and nothing is retuned by model, benchmark, or configuration. This global design choice is distinct from BGAA's inference-time selection, which selects views from each configuration's observed history.

Baselines. The primary baseline is each model's unchanged-input forecast. We also compare against alternative BGAA selection and aggregation policies, compute-matched random, noise, and identity controls, and published larger-model results used only for the model-scaling comparisons.

5 Main results

BGAA improves all evaluated model–benchmark combinations. Table 1 shows that BGAA improves both point and probabilistic forecasting across all 12 model×benchmark combinations without per-model or per-benchmark retuning. MASE improves by 0.5–1.1% on GIFT-Eval, 1.9–5.1% on BOOM, and 3.2–5.9% on fev-bench. These are benchmark-level aggregate improvements; individual configurations can regress, as analyzed below.

Test-time compute can dominate a larger model under the same compute–accuracy comparison. On GIFT-Eval, the `high` TiRex-2 + BGAA configuration achieves lower MASE than Toto-2-2.5B with approximately 21% less estimated inference compute: 0.6937 versus 0.6956 MASE at approximately 96 versus 121 PFLOPs (Figure 2). This is a pairwise dominance result under our estimated inference-FLOPs accounting; it does not imply the same advantage in latency or memory. Separately, on BOOM, BGAA on Toto-2-313M achieves lower MASE than the published 1B and 2.5B members of the same family (Table 2).

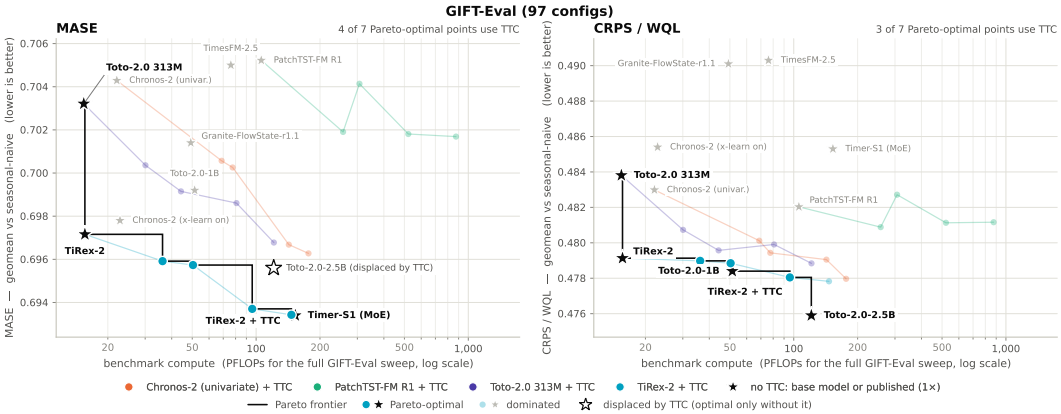


Figure 2: **Test-time compute improves MASE under estimated inference-FLOPs scaling.** On GIFT-Eval (97 configurations), lines connect four test-time compute budgets for each frozen model; stars denote published models at $1\times$ compute. Each rung uses the model's measured benchmark PFLOPs scaled by its measured forward-pass ratio $(\text{base} + \text{gate} + \text{apply})/\text{base}$. The CRPS ladders also improve over their corresponding bases.

Figure 2 compares the four *thinking modes*, which vary test-time compute while keeping the BGAA procedure fixed (Table 5). Returns diminish with budget: `high` captures 71–96% of the `xhigh` MASE gain while using 60–81% of its compute. The CRPS ladders improve over their corresponding bases but do not establish the same dominance result. Cross-learning-*on* CHRONOS-2 also improves over its univariate counterpart at similar base compute, illustrating that test-time compute is not the only useful scaling direction. The FLOPs axis ignores batching and should be interpreted as estimated inference-compute accounting rather than latency. Full accounting is given in Section C.1.

Table 2: **BGAA on TOTO-2-313M (313M) exceeds the published larger members of the same family on BOOM MASE.** Excerpt of the full recomputed leaderboard (Table 13). This table compares accuracy; model size alone does not establish inference-compute efficiency after adding BGAA.

system	MASE	CRPS	note
TOTO-2-313M + BGAA	**0.574**	**0.349**	**ours**
TOTO-2-2.5B	0.581	0.349	published
TOTO-2-1B	0.582	0.349	published
TOTO-2-313M	0.585	0.351	published
TOTO-2-313M base	0.585	0.353	our reproduction
TOTO (V1)	0.617	0.375	published
TIREX-2+ BGAA	**0.621**	**0.377**	**our run**
TIMESFM-2.5	0.632	0.382	published
TIREX-2 base	0.642	0.386	our run
MOIRAI-2-SMALL	0.672	0.427	published

Matched controls isolate the role of gating and structured views. Averaging the full candidate pool without gating worsens MASE by 7.66%, whereas BGAA improves it by 1.58% on the same pool (Figure 4a). On BOOM, the gate is inactive when fewer than $m_{\min} = 8$ backtest windows are available; median improvement is therefore zero for the 2929 such tasks, rises to 0.84% around $N = 11$, and remains positive for larger buckets (Figure 4c).

Compute-matched controls show that extra inference alone is insufficient. With $k = 2$ and the pool and aggregation fixed, random selection worsens MASE by 4.82% and CRPS by 10.73%, while averaging jittered copies recovers only 3% and 10% of the corresponding BGAA gains (Table 9). Lookback-only and offset-only pools recover 38% and 6% of the full-pool MASE gain, while cleaning-only views recover 46% (Table 8). Finally, using gate evidence at least $4H$ before the forecast still improves MASE by 1.15%, versus

Table 3: **Regressions relative to the base are uncommon under both policies.** Per-configuration MASE improvement for CHRONOS-2 under `mixtop3-mase` and `routemix05-crps`, relative to the frozen base. Improvement is defined so that positive values are better; therefore, negative values in the "worst improvement" columns denote degradation. Full CDFs are in Figure 4b.

		deployed (`mixtop3-mase`)		conservative (`routemix05-crps`)	
benchmark	configs	% no worse than base	worst improvement	% no worse than base	worst improvement
GIFT-Eval	97	79%	−8%	85%	−6%
BOOM	6750	93%	−93%	92%	−53%
fev-bench	100	91%	−2%	90%	−1%

1.40% at lag zero, despite only 0.53 Jaccard overlap in admitted sets (Table 10). Further pool and aggregation ablations are reported in Table 7.

Most configurations do not regress. Table 3 shows that 79%, 93%, and 91% of configurations on GIFT-Eval, BOOM, and fev-bench, respectively, are no worse than their frozen base under the deployed policy. The worst regressions are 8% on GIFT-Eval and 2% on fev-bench. On BOOM, the larger negative tail is driven by a small number of near-constant series for which the scaled metric is unstable. These are empirical observations, not consequences of the backtest guarantee, which does not extend to unseen forecast horizons or benchmark-level metrics (Section 3).

6 Interpretation and supporting evidence

The results above establish that BGAA improves frozen forecasters, but do not explain why *representing* the same observed history can produce useful alternatives. We therefore ask: *what kind of model behavior would make structured views useful rather than merely different?* The argument below is not required by BGAA and is not an accuracy theorem. Instead, a restricted transport result provides a concrete interpretation that yields empirical predictions we test across the four models.

A deterministic view adds no observations. Nevertheless, a frozen forecaster can return a different predictive distribution when the same history is presented in another representation. Let Π be an idealized distribution over complete trajectories and let $Q_P(\tau \mid x_{1:T}) \in \mathbb{R}^H$ denote the conditional future τ-quantiles under trajectory distribution P, and write $f(x_{1:T}, H)_{\cdot,\tau} \in \mathbb{R}^H$ for the column of the frozen forecast at level τ.

Assumption 1 (PFN idealization)**.** For an observed context $x_{1:T}$, the frozen model returns the conditional quantiles under Π: $f(x_{1:T}, H)_{\cdot,\tau} = Q_\Pi(\tau \mid x_{1:T}), \qquad \tau \in \mathcal{L}$.

Now let $\psi\colon \mathbb{R} \to \mathbb{R}$ be a fixed strictly increasing bijection applied pointwise, and let Π_ψ be the trajectory distribution obtained by applying ψ^{-1} pointwise to trajectories drawn from Π. Let $(\Pi_\psi)_{1:T}$ denote its marginal over contexts.

Proposition 1 (Transport identity)**.** *Under Assumption 1,* $\psi^{-1}\big(f(\psi(x_{1:T}), H)_{\cdot,\tau}\big) = Q_{\Pi_\psi}(\tau \mid x_{1:T})$ *for* $(\Pi_\psi)_{1:T}$*-almost every context* $x_{1:T}$ *and* $\tau \in \mathcal{L}$*.*

For this restricted class of transformations, forecasting a transformed history and mapping its quantiles back is equivalent to prediction under a transported trajectory distribution. This does not imply greater accuracy, nor do we claim that released TSFMs satisfy Assumption 1 exactly. None of our non-identity deployed views satisfies all of the proposition's conditions: Rank-to-Gaussian is context-fitted and clamped, crops and offsets alter the observations presented to the model, and differencing is not pointwise and uses an approximate quantile-path reconstruction. The full statement, proof, and scope are in Section A.3. Proposition 1 instead provides a tractable case where re-presenting the same observations changes the effective predictive distribution without adding information, motivating us to ask whether BGAA's broader views induce similarly structured changes.

This yields a testable hypothesis: if structured views systematically change a frozen model's forecasting behavior rather than acting as arbitrary ensemble noise, then *which* views help should depend on the model, individual views should induce repeatable changes in representation space, and these changes should produce recognizable forecast-level signatures. We test these predictions below; full protocols and results are in Section E.

Useful view families are model-specific. Across the six model pairs, Kendall's τ_{K} between per-family solo-gated benefit vectors ranges from 0.01 to 0.35, indicating weak agreement on which families help (Table 19). Differencing is strongest for CHRONOS-2 (-0.85% MASE) but slightly hurts TIREX-2 ($+0.01\%$), while `lookback` is strongest for PATCHTST-FM-R1 and TOTO-2-313M (Table 18). Admitted families also vary across domains for CHRONOS-2 (Figure 10a), suggesting that useful views depend on both the model and configuration.

Views induce systematic, model-specific changes. Across 626 GIFT-Eval contexts, we compare view-induced displacements with rollout samples from each model, used as a proxy for its learned forecasting behavior. Some views move contexts consistently: `diff` and `smooth` have directional consistency of 0.75–0.78, versus 0.04–0.07 for crops and offsets (Table 21). Different views are also non-redundant, with mean absolute pairwise displacement cosine of 0.36–0.48 (Table 22). Finally, `diff` aligns with a mode of each model's rollout distribution (cosine $+0.92$ to $+0.97$), but with

different modes across models (Table 23 and Figure 9). These patterns indicate repeatable, model-specific effects rather than arbitrary perturbations.

The structure persists in forecast space. The final prediction is that a view's effect should remain recognizable in the forecast itself. On a 15-configuration subset, `diff` widens predictive intervals by a log-ratio of $+2.10$ to $+2.25$, with the same sign on 14–15 of 15 configurations (Table 24). This is a property of the complete forecasting view, including its cumulative-sum pull-back, rather than only of the transformed input. The broader signature analysis in Section E.3 shows the same family-level structure across the other forecast statistics and models.

Taken together, this evidence forms a consistent chain: useful views differ by model, individual views move contexts in repeatable and distinct directions, and those changes remain visible in the resulting predictive distributions. This evidence strengthens the interpretation that structured views expose different behaviors of the same frozen TSFM rather than merely being generic ensembling. It does not establish causal prior transport: the rollout distributions are proxies for learned forecasting behavior, and we do not intervene on the models' pretraining distributions.

7 Limitations and future directions

This section focuses on the main limitation revealed by our diagnostics: historical backtesting supports reliable config-level selection, but provides much weaker guidance for individual forecast windows. We also discuss what this means for future test-time compute methods.

The backtest guarantee in Section A.1 is deliberately limited: it controls loss on the historical gate windows, not on the future forecast window. Thus, the empirical question is whether a ranking learned from the past remains informative at deployment. At the config level, it often does; at the individual-window level, the signal is much weaker. On Chronos-2, on GIFT-Eval the median Spearman correlation between backtest and realized view rankings is 0.81 per configuration but only 0.27 per forecast window (Figure 5). Unlike config-level selection, which can pool evidence across many historical forecasting tasks, window-level selection must infer from the current context which view will be useful for the next particular horizon.

Hindsight selection confirms that the best-performing view can vary substantially across windows. The backtest's top-1 contains the realized best view for 24% of configurations but only about 8% of individual windows, and its per-window top-1 recovers only about 25% of the hindsight-available MASE gain (Figure 6). Cross-fitted analyses likewise indicate that better fixed per-configuration choices remain possible (Table 16). These quantities are diagnostic rather than deployable targets: hindsight uses the realized future, and the cross-fitted splits remain temporally dependent (Section D.2). They therefore show where the current verifier loses information, not how much improvement a future context-only selector could necessarily recover.

Our evaluation is also limited to univariate forecasting, and BGAA incurs additional inference cost, primarily from evaluating candidate views on historical windows (Table 6). The main open question is whether information available before the forecast can support more accurate window-specific selection without making verification prohibitively expensive. This could require better window-level signals, better candidate views, or both. Other promising directions include: *learned verifiers* that predict useful views from context and view features; *verification-set design* that selects informative past windows; and *adaptive budgets* that spend more compute when views disagree.

8 Conclusion

In this paper, we show that test-time compute can improve frozen time series foundation models without retraining or future labels. BGAA generates forecasts from multiple views of the same context and uses a leak-free rolling-origin backtest to select which views to aggregate. With one fixed pool and policy, BGAA improves all 12 model–benchmark combinations, with MASE gains of 0.5–1.1% on GIFT-Eval, 1.9–5.1% on BOOM, and 3.2–5.9% on fev-bench. Mechanism experiments show structured, model-dependent view effects and consistent forecast signatures, supporting our prior-transport interpretation. The main remaining limitation is window-level selection: backtest rankings are strong at the configuration level but much weaker per window, with median Spearman correlation 0.81 versus 0.27 for Chronos-2 on GIFT-Eval. This motivates better window-specific selection alongside improved candidate views.

AI USE STATEMENT

We used large language models as coding and writing assistants during the development of BGAA and while writing the paper. The initial base implementation of BGAA was developed primarily by the authors. Claude Opus 4.8 and Opus 5 were later used to scale the implementation, reproduce benchmark pipelines, configure environments, refactor code, and support model- and benchmark-specific integration. Opus 5 was also used for TIREX-2-specific kernel modifications to improve evaluation efficiency on H100 GPUs. Every code file was reviewed by at least one author, and the full codebase was checked at least three times for data leakage. After the final refactor, we reran the relevant experiments and verified that the earlier results remained reproducible.

The first draft of the paper was primarily written by the authors. Claude Opus 5 was used to review the text and check consistency with the final implementation. GPT-5.6 Sol was later used to compress the manuscript from more than 14 pages to 9 pages. This was done line-by-line or paragraph-by-paragraph rather than by regenerating the manuscript as a whole. Every resulting line was reviewed by at least two authors before inclusion, and any incorrect, ambiguous, or overstated language introduced during compression was revised manually.

The authors developed the main theory, its interpretation, and the central arguments. GPT-5.6 Sol and Claude Fable 5 were used to refine notation, compress the language, check the presentation of mathematical statements, and improve consistency between the theory in the main text and appendix. Fable 5 was used to add relevant cross-references to the theory where needed. All theoretical claims and proofs were subsequently reviewed and validated by the authors. We take full responsibility for the implementation, experiments, analysis, theoretical claims, and final text of the paper.

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. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024. URL https://openreview.net/forum?id=Z2cMOOANFX.

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

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting, 2025. URL https://arxiv.org/abs/2510.15821.

J. M. Bates and C. W. J. Granger. The combination of forecasts. *OR*, 20(4):451–468, 1969. ISSN 14732858. URL http://www.jstor.org/stable/3008764.

Shuxiao Chen, Edgar Dobriban, and Jane H. Lee. A group-theoretic framework for data augmentation. *Journal of Machine Learning Research*, 21(245):1–71, 2020. URL http://jmlr.org/papers/v21/20-163.html.

Robert T. Clemen. Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting*, 5(4):559–583, 1989. ISSN 0169-2070. doi: https://doi.org/10.1016/0169-2070(89)90012-5. URL https://www.sciencedirect.com/science/article/pii/0169207089900125.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean Ogier du Terrail, Anna-Monica Toon, Kan Wang, Stephan Xie, Zongzhe Xu, Viktoriya Zhukova, David Asker, Ameet Talwalkar, and Othmane Abou-Amal. This time is different: An observability perspective on time series foundation models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=1jDAYXfcS2.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10148–10167. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/das24c.html.

Isaac Gibbs and Emmanuel J. Candès. Adaptive conformal inference under distribution shift. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

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.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16115–16152. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/goswami24a.html.

Ruijin Hua, Zichuan Liu, Kun Zhang, and Yiyuan Yang. Diversified scaling inference in time series foundation models, 2026. URL https://arxiv.org/abs/2601.17376.

Harshavardhan Kamarthi and B Aditya Prakash. Large pre-trained time series models for cross-domain time series analysis tasks. *NeurIPS*, 2024.

Emaad Khwaja, Chris Lettieri, Gerald Woo, Eden Belouadah, Marc Cenac, Guillaume Jarry, Enguerrand Paquin, Xunyi Zhao, Viktoriya Zhukov, Othmane Abou-Amal, Chenghao Liu, Ameet Talwalkar, and David Asker. Toto 2.0: Time series forecasting enters the scaling era, 2026. URL https://arxiv.org/abs/2605.20119.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, December 2022. ISSN 1095-9203. doi: 10.1126/science.abq1158. URL http://dx.doi.org/10.1126/science.abq1158.

Kenneth C. Lichtendahl, Yael Grushka-Cockayne, and Robert L. Winkler. Is it better to average probabilities or quantiles? *Management Science*, 59(7):1594–1611, 2013. ISSN 00251909, 15265501. URL http://www.jstor.org/stable/23443872.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

Clare Lyle, Mark van der Wilk, Marta Kwiatkowska, Yarin Gal, and Benjamin Bloem-Reddy. On the benefits of invariance in neural networks, 2020. URL https://arxiv.org/abs/2005.00178.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models, 2021. URL https://arxiv.org/abs/2112.00114.

Manya Pandey, Dhruv Kumar, Murari Mandal, and Saurabh Deshpande. GITCO: Gated inference-time context optimization in TSFMs. In *2nd ICML Workshop on Foundation Models for Structured Data*, 2026. URL https://openreview.net/forum?id=AKHqrrUdWD.

Patrick Podest, Marco Pichler, Elias Bürger, Levente Zólyomi, Bernhard Voggenberger, Wilhelm Berghammer, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. Tirex-2: Generalizing tirex to multivariate data and streaming, 2026. URL https://arxiv.org/abs/2607.01204.

Divya Shanmugam, Davis Blalock, Guha Balakrishnan, and John Guttag. Better aggregation in test-time augmentation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1194–1203, 2021. doi: 10.1109/ICCV48922.2021.00125.

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

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Kamilė Stankevičiūtė, Ahmed Alaa, and Mihaela van der Schaar. Conformal time-series forecasting. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Rx9dBZaV_IP.

Allan Timmermann. Forecast combinations. volume 1, chapter 04, pages 135–196. Elsevier, 1 edition, 2006. URL https://EconPapers.repec.org/RePEc:eee:ecofch:1-04.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Yunshi Wen, Wesley M. Gifford, Chandra Reddy, Lam M. Nguyen, Jayant Kalagnanam, and Anak Agung Julius. Revisiting the generic transformer: Deconstructing a strong baseline for time series foundation models, 2026. URL https://arxiv.org/abs/2602.06909.

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*, ICML'24. JMLR.org, 2024.

A THEORY AND PROOFS

Section A.1 proves the backtest-dominance property used in Section 3. Section A.2 gives the aggregation results behind Vincent and CDF-mixture aggregation and states which aggregator each score covers. Section A.3 formalizes the transported-prior interpretation tested in Section 6.

A.1 THE BACKTEST REFEREE

This section gives the formal statement and proof of the gate guarantee summarized in Section 3.

Fix a dataset configuration c. Let id denote the identity view, which returns the base forecast, and $g \in \mathcal{G}$ a candidate view. On gate window w_j, let $\widehat{L}_j(g) \geq 0$ be the loss of view $g \in \mathcal{G} \cup \{\mathrm{id}\}$, averaged over the n_f rolling forecast origins in that window. The gate uses absolute error of the median forecast for `*-mase` policies and the mean pinball loss over $\mathcal{L}$ for `*-crps` policies. Windows with $\widehat{L}_j(\mathrm{id}) = 0$ are excluded; let $\mathcal{J}$ be the set of usable windows and $m = |\mathcal{J}|$ their number. Define the mean per-window loss relative to the base as

$$\widehat{L}_{\mathrm{rel}}(g) = \frac{1}{m} \sum_{j \in \mathcal{J}} \frac{\widehat{L}_j(g)}{\widehat{L}_j(\mathrm{id})}, \qquad \widehat{L}_{\mathrm{rel}}(\mathrm{id}) = 1.$$

The relative improvement of candidate g is $\widehat{\epsilon}(g) = 1 - \widehat{L}_{\mathrm{rel}}(g)$. The margin rule requires $\widehat{\epsilon}(g) \geq \epsilon_{\min}$, where $\epsilon_{\min} > 0$ is the minimum improvement, and a win on at least a fraction ϕ of usable windows. Top-k keeps the k candidates with largest positive $\widehat{\epsilon}(g)$. Both rules require at least $m_{\min}$ usable windows.

Let $\widehat{\mathcal{A}}_c$ denote the admitted candidate set and f_{agg} the resulting aggregate, with loss $\widehat{L}_j(\mathrm{agg})$. The aggregate assigns weight $\lambda_0 \geq 0$ to the base and nonnegative weights λ_g to admitted candidates, with

$$\lambda_0 + \sum_{g \in \widehat{\mathcal{A}}_c} \lambda_g = 1.$$

If $\widehat{\mathcal{A}}_c = \varnothing$, the policy returns the base forecast. The result below depends only on the gate losses and aggregation rule, not on the underlying forecaster.

Theorem 2 (Backtest dominance)**.** *Suppose that, on every usable gate window, the aggregate loss satisfies*

$$\widehat{L}_j(\mathrm{agg}) \leq \lambda_0 \widehat{L}_j(\mathrm{id}) + \sum_{g \in \widehat{\mathcal{A}}_c} \lambda_g \widehat{L}_j(g).$$

This holds for Vincent averaging under pinball loss or absolute error of the median forecast. More generally, if ℓ is exact CRPS, CDF mixing also satisfies the required inequality. If $\widehat{\mathcal{A}}_c \neq \varnothing$, let

$$\epsilon' = \min_{g \in \widehat{\mathcal{A}}_c} \widehat{\epsilon}(g)$$

be the smallest improvement among admitted candidates. Then

$$\widehat{L}_{\mathrm{rel}}(\mathrm{agg}) \leq 1 - (1 - \lambda_0)\epsilon' \leq 1 = \widehat{L}_{\mathrm{rel}}(\mathrm{id}).$$

Under the margin rule, $\epsilon' \geq \epsilon_{\min}$; under top-$k$, $\epsilon' > 0$. If $\widehat{\mathcal{A}}_c = \varnothing$, the aggregate equals the base.

Proof. For a non-empty admitted set, the assumed aggregation inequality holds on each usable gate window. For Vincent averaging, it follows from convexity of pinball loss; absolute error of the median forecast is proportional to the $\tau = \frac{1}{2}$ pinball loss. For CDF mixing under exact CRPS, it follows from Theorem 3. Since $\widehat{L}_j(\mathrm{id}) > 0$ on every usable window, dividing by $\widehat{L}_j(\mathrm{id})$ and averaging gives:

$$\widehat{L}_{\mathrm{rel}}(\mathrm{agg}) \leq \lambda_0 + \sum_{g \in \widehat{\mathcal{A}}_c} \lambda_g \widehat{L}_{\mathrm{rel}}(g) = 1 - \sum_{g \in \widehat{\mathcal{A}}_c} \lambda_g \widehat{\epsilon}(g) \leq 1 - (1 - \lambda_0)\epsilon'.$$

The empty-set case follows directly from returning the base forecast. □

Theorem 2 applies to the gate windows and to the mean per-window relative loss $\widehat{L}_{\mathrm{rel}}$ used by the gate. It does not guarantee improvement under the benchmarks' reported aggregation rules, which generally differ from $\widehat{L}_{\mathrm{rel}}$ and can weight configurations or forecast errors differently. It also does not guarantee improvement on the future forecast horizon without assumptions linking the backtest to that horizon. Thus, we measure horizon performance directly in Tables 1 and 3. Table 10 probes the backtest-to-horizon link itself: when the gate's evidence is restricted to observations at least $\mathrm{lag} \times H$ before the forecast start ($\mathrm{lag} \leq 4$), the horizon MASE gain of the Vincent top-2 policy stays at 1.1–1.6% ($p \leq 0.015$) even though the admitted set changes substantially (Jaccard 0.50–0.58 against lag 0), on a 15-config GIFT-Eval test subset for CHRONOS-2.

Note: The theorem covers the Vincent `top3-mase` policy. It does not cover the deployed `mixtop3-mase`, since CDF-mixture quantiles are not convex combinations of member quantiles. Similarly, the `*-crps` gate uses mean pinball loss over $\mathcal{L}$ rather than exact CRPS, so CDF mixing under that gate does not inherit the exact-CRPS guarantee. The positive margin $\epsilon_{\min}$, win-fraction threshold ϕ, and minimum-window requirement $m_{\min}$ are robustness checks against small or noisy backtests, not finite-sample guarantees on the forecast horizon.

A.2 AGGREGATION

This section gives the full aggregation result summarized in Section 3.

Let $F_1, \ldots, F_n$ be the CDFs of n forecast distributions, with weights $\lambda_i \geq 0$ and $\sum_{i=1}^n \lambda_i = 1$, where the weights are determined independently of the realized outcome $y \in \mathbb{R}$. Multi-horizon scores average the corresponding one-step scores, so the result below applies at each horizon. In our implementation, each F_i is the interpolated CDF constructed from the model's nine-quantile forecast (Section B). Vincent averaging combines quantiles level by level, while CDF mixing averages the predictive distributions.

Theorem 3 (CRPS decomposition for CDF mixtures)**.** *Let*

$$\bar{F} = \sum_{i=1}^{n} \lambda_i F_i$$

be the mixture CDF, and assume the member forecasts have finite CRPS. Then, for every realized outcome y,

$$\mathrm{CRPS}(\bar{F}, y) = \sum_{i=1}^{n} \lambda_i \, \mathrm{CRPS}(F_i, y) - \underbrace{\sum_{i=1}^{n} \lambda_i \int_{\mathbb{R}} \big(F_i(z) - \bar{F}(z)\big)^2 \, \mathrm{d}z}_{\textit{diversity term} \geq 0},$$

where z indexes the forecast value. The diversity term depends only on the forecast distributions, not on y.

Proof. For outcome y, define

$$a_i(z) = F_i(z) - \mathbf{1}\{y \leq z\}, \qquad \bar{a}(z) = \sum_{i=1}^{n} \lambda_i a_i(z),$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. Since $\bar{a}(z) = \bar{F}(z) - \mathbf{1}\{y \leq z\}$, the weighted-mean identity gives

$$\bar{a}(z)^2 = \sum_{i=1}^{n} \lambda_i a_i(z)^2 - \sum_{i=1}^{n} \lambda_i \big(a_i(z) - \bar{a}(z)\big)^2.$$

Moreover, $a_i(z) - \bar{a}(z) = F_i(z) - \bar{F}(z)$. Integrating both sides over $z \in \mathbb{R}$ and using the definition of CRPS gives the result. □

Since the diversity term is non-negative, the CRPS of a CDF mixture is no worse than the weighted average CRPS of its members for every outcome, with equality when all positive-weight member CDFs agree almost everywhere. The term can also be computed without labels, giving a label-free measure of forecast disagreement and motivating the spread-routed policy in Section B.

This identity applies to the mixture $\bar{F}$ itself, before we read it back as a nine-quantile forecast for deployment. It is specific to exact CRPS.

The guarantee in Theorem 2 requires the aggregation rule to be convex in the score used by the gate. Vincent averaging satisfies this condition for the absolute-error score used by `*-mase` and for pinball loss. CDF mixing satisfies it for exact CRPS. Our `*-crps` policies instead rank views using mean pinball loss over $\mathcal{L}$, so their CDF-mixture variants do not inherit the exact-CRPS guarantee.

The deployed `mixtop3-mase` policy similarly combines a MASE gate with CDF mixing, and thus does not inherit formal backtest dominance on either MASE or CRPS. We use it because it gives the best mean MASE in the policy ablation. The corresponding Vincent policy, `top3-mase`, uses the same gate and admitted set and is covered by the backtest guarantee in Theorem 2.

A.3 Restricted transport interpretation

A trajectory is $z = (z_1, \ldots, z_{T+H}) \in \mathbb{R}^{T+H}$, with context $z_{1:T}$ and future $z_{T+1:T+H}$. Let Π denote an idealized trajectory distribution associated with the frozen model, $x_{1:T} \in \mathbb{R}^T$ a context, and $\mathcal{L}$ the reported quantile levels. We write $Q_P(\tau \mid x_{1:T})$ for the vector of conditional τ-quantiles of the future given $z_{1:T} = x_{1:T}$ under trajectory distribution P, and $f(x_{1:T}, H)_{\cdot,\tau}$ for the corresponding model forecast, the column of $f(x_{1:T}, H)$ at level τ.

Assumption 1 (PFN idealization). Given a context $x_{1:T}$, the frozen model returns the conditional future quantiles under Π:

$$f(x_{1:T}, H)_{\cdot,\tau} = Q_\Pi(\tau \mid x_{1:T}), \qquad \tau \in \mathcal{L}.$$

The PFN interpretation motivates this idealization (Müller et al., 2022). Pretrained probabilistic forecasters such as Chronos motivate asking whether an analogous interpretation is useful for TSFMs (Ansari et al., 2024), but we do not assume that it holds exactly. We use it only to motivate a mechanism with observable predictions, which we test in Section 6.

Proposition 1 (Transport identity). Let $\psi\colon \mathbb{R} \to \mathbb{R}$ be a fixed strictly increasing bijection applied pointwise, and define its trajectory-level extension

$$\Psi(z) = (\psi(z_1), \ldots, \psi(z_{T+H})).$$

Let Π_ψ be the distribution of $\Psi^{-1}(z)$ for $z \sim \Pi$, and let $(\Pi_\psi)_{1:T}$ denote its marginal distribution over contexts. Here, $\psi(x_{1:T})$ and $\psi^{-1}(\cdot)$ are also applied coordinatewise. Under Assumption 1, for $(\Pi_\psi)_{1:T}$-almost every context $x_{1:T}$ and every $\tau \in \mathcal{L}$,

$$\psi^{-1}\big(f(\psi(x_{1:T}), H)_{\cdot,\tau}\big) = Q_{\Pi_\psi}(\tau \mid x_{1:T}).$$

Thus, forecasting the transformed context and mapping its quantiles back is equivalent to prediction under the transported trajectory distribution Π_ψ.

Proof. Let $z \sim \Pi_\psi$ and define $\tilde{z} = \Psi(z)$. By the definition of Π_ψ,

$$\tilde{z} \sim \Pi.$$

Since Ψ applies ψ coordinatewise,

$$\tilde{z}_{1:T} = \psi(z_{1:T}), \qquad \tilde{z}_{T+1:T+H} = \psi(z_{T+1:T+H}).$$

Because Euclidean trajectory space is standard Borel, regular conditional distributions exist. Now fix a context $x_{1:T}$. Since ψ is a bijection, $z_{1:T} = x_{1:T}$ corresponds exactly to $\tilde{z}_{1:T} = \psi(x_{1:T})$. Also,

$$z_{T+1:T+H} = \psi^{-1}(\tilde{z}_{T+1:T+H})$$

coordinatewise. Therefore, for $(\Pi_\psi)_{1:T}$-almost every $x_{1:T}$, the conditional distribution of $z_{T+1:T+H}$ given $z_{1:T} = x_{1:T}$ is obtained by applying ψ^{-1} coordinatewise to the conditional distribution of $\tilde{z}_{T+1:T+H}$ given $\tilde{z}_{1:T} = \psi(x_{1:T})$.

Because ψ is strictly increasing, conditional quantiles transform coordinatewise:

$$Q_{\Pi_\psi}(\tau \mid x_{1:T}) = \psi^{-1}(Q_\Pi(\tau \mid \psi(x_{1:T}))).$$

Finally, Assumption 1 gives

$$f(\psi(x_{1:T}), H)_{\cdot,\tau} = Q_{\Pi}(\tau \mid \psi(x_{1:T})).$$

Substituting this into the previous display yields

$$Q_{\Pi_\psi}(\tau \mid x_{1:T}) = \psi^{-1}\big(f(\psi(x_{1:T}), H)_{\cdot,\tau}\big),$$

which is the claimed identity. □

Observation-map views. Lookback crops and patch-phase offsets do not satisfy the transport construction. Instead, they change the information presented to the model without changing the frozen forecaster. The gate treats them like any other candidate, and Theorem 2 does not require a transport interpretation.

Scope of the transport interpretation. Proposition 1 applies only to fixed pointwise strictly increasing bijections. The identity view satisfies these conditions trivially, but none of our non-identity deployed views satisfies them all exactly. Rank-to-Gaussian is the closest case, but it is fitted to the current context and clamps forecasts to the observed range. Lookback and offset views instead change the observation map. The remaining families are structured perturbations motivated by the same picture rather than exact transported trajectory distributions.

Differencing requires one additional caveat. Its pull-back constructs a level-τ path by cumulatively summing level-τ increment forecasts. Since the quantile of a sum is not generally the sum of its marginal quantiles, the resulting fan need not correspond to a transported posterior predictive. This can widen intervals with horizon, so part of the `diff` effect in Table 24 comes from the pull-back itself. This does not affect Theorem 2, which treats each view as a forecast-generating policy.

Selection over transported views. For views satisfying Proposition 1, the backtest can be interpreted as selecting among forecasts induced by different transported trajectory distributions. We use this exact case to motivate the broader structured-view interpretation. The gate ranks views by predictive loss; its weights are not Bayesian model probabilities.

B THE VIEW CATALOGUE

What makes a legal view. A view must (i) use no information beyond the available context, (ii) return a forecast fan on the original data scale, through pull-back, restriction, or output post-processing, and (iii) change something that is not removed by the model's per-instance normalization. Requirement (iii) excludes affine warps: with RevIN-style normalization, rescaling and shifting the context are exact no-ops, which we verified empirically before constructing the pool.

Table 4: **The deployed pool of views**. Tiers are strictly nested, crop $\subset$ micro $\subset$ ideas $\subset$ ideas+decomp, which makes Table 7's left column a decomposition rather than a list. "#" counts *views*; "passes" counts forward passes per window, which differs when a view is seeded (`jitter`/`jitternorm` average 3 seeded draws) or sequential (`recursive_f0.5` needs 2 calls, `recursive_f0.25` needs 4). Thus, the gate uses up to 58 passes for the identity plus 51 non-identity candidates (52 total views); where the decomposition family deduplicates to a single view, e.g. period-1 configs, this becomes 56 passes and 50 total views.

tier	family	#	passes	degrees, and how the fan is inverted
crop	identity	1	1	– (cached base forecast)
crop	lookback	18	18	keep the last αT samples, $\alpha \in \{0.98, 0.95, 0.925, 0.9, 0.875, 0.85, 0.8, 0.75, 0.7, 0.625, 0.55, 0.5, 0.4375, 0.375, 0.3125, 0.25, 0.1875, 0.125\}$; restriction map, no fan inversion
crop	offset	14	14	shift the patch grid by $s \in \{1, 2, 3, 4, 6, 8, 10, 12, 14, 16, 20, 24, 32, 48\}$ samples ($\texttt{patch_len} = 16$); restriction map
micro	jitter	1	3	additive σ=0.002·std noise, 3 seeded draws averaged in quantile space; median-neutral
micro	jitternorm	1	3	as above, rescaled to preserve the context's scale
micro	winsor	1	1	clip at the 0.5% tails
ideas	diff	1	1	first differencing; cumulative-sum inversion along each quantile path (comonotone pull-back, an approximation; Section A.3)
ideas	sdiff	1	1	seasonal differencing at the config's period m; seasonal cumulative-sum inversion
ideas	multideseason	1	1	remove the multiplicative seasonal profile at period m; multiply it back onto every quantile
ideas	seasontile	2	2	tile the last seasonal period, $K \in \{1, 2\}$
ideas	fft	3	3	keep the top-k frequencies, $k \in \{4, 8, 12\}$; low-pass of the context only
ideas	rankgauss	1	1	empirical ranks $\to$ Gaussian order statistics; invert through the context's empirical quantile function, clamped to its observed range; context-fitted, degree 0.0[1]
ideas	nonneg	1	0	project the fan onto $[0, \infty)$; reuses the cached identity pass
ideas	supportfloor	1	0	project the fan onto the context's observed support; reuses the identity pass
ideas	recursive	2	6	short-step rollout composed to H, step fraction $f \in \{0.5, 0.25\}$ (2 and 4 sequential calls)
decomp	decomp	≤ 3	≤ 3	seasonal decomposition at the base period and its sub/super harmonics $\{m, m/2, 2m\}$; deduplicated, so 1–3 views per config

Two views are free. `nonneg` and `supportfloor` post-process the identity fan rather than re-running the model, so they require *zero* additional forward passes in both the gate and apply steps once the identity forecast is memoized. Together they account for 16% of rank-weighted admits, giving them the largest admit share per additional pass in the pool and motivating their inclusion in the cheapest thinking mode in Table 6.

Families that were tried and rejected. For completeness, we also report families that were evaluated but not included in the deployed candidate set. These include detrending, STL decomposition, coarse resampling, motif retrieval (using either raw or model-embedding distance), regime/changepoint crops, anomaly cleaning, ARIMA-residual views, log-difference and reflected log-difference, moving-average and EWMA smoothing, despiking, and power transforms. Smoothing is informative: Section E.2 finds that its context displacement points away from the measured modes of the models' rollout distributions, a pattern consistent with moving contexts away from regions favored by the frozen model.

How rejection was decided, and why it is not post-hoc. Every family, whether deployed or rejected, was evaluated using the same procedure before deployment decisions were made. We constructed the family's full degree grid, gated it *solo* (that family plus the identity) using the leak-free backtest, and scored the resulting policy against the CHRONOS-2 base on a labelled 15-config subset of the GIFT-Eval *validation* split. A family was admitted to the deployed tier only if its solo-gated ΔMASE was negative and its worst-config regression was bounded on this CHRONOS-2 *validation* subset. This is the only stage at which labels informed a design choice: the pool was then frozen before any test-split evaluation or evaluation with the other models, and neither BOOM nor fev-bench was used for an admit/reject decision. Rejected families were evaluated under the same protocol and continue to be measured by it; harmful variants (`smooth`, `ewma`, and `winsor` above the smallest degree) remain in the *ranking* pool but are excluded at deployment. Their reported results are therefore regenerated under the same protocol rather than taken from a one-off sweep. Section 6 reports the resulting per-family fingerprints across all four models.

Two families appear only in the fingerprint. Table 18 also gates two fixed-pool families that are not in the deployed tier. `changepoint` is the regime crop listed above: it finds the strongest single mean shift in the most recent $1 - \rho$ of the context, $\rho \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$, and starts the context at that break if the shift exceeds 0.5 standard deviations. `warp` resamples the context by linear interpolation to $\mathrm{round}(wT)$ points and back to T, with $w \sim \mathcal{U}[1 - f, 1 + f]$, $f \in \{0.01, 0.02, 0.035, 0.05, 0.075, 0.1, 0.15\}$, and averages 3 seeded draws. Both resamplings span the whole context, so the round trip keeps every time index in place.

Aggregation details. Let $\widehat{\mathcal{A}}$ be the admitted set, $\widehat{Q}_0$ the identity fan, and w_0 the weight on the base. The first three rules assign uniform weight to the base and admitted members, giving $w_0 = 1/(|\widehat{\mathcal{A}}| + 1)$; the soft-gate instead reweights the admitted set.

- **Vincent (quantile averaging).** $\bar{q}(\tau) = \sum_i w_i q_i(\tau)$ at each level τ. Since each member is monotone in τ, their weighted average is also a valid quantile fan.
- **CDF mixture.** $\bar{F} = \sum_i w_i F_i$, where each F_i is the interpolated CDF of its nine-quantile fan. We invert the mixture on a shared grid by pooling and sorting the members' quantiles, evaluating $\bar{F}$ on this grid, and reading back the 9 deployed levels by linear interpolation. Theorem 3 applies exactly to $\bar{F}$ before this nine-quantile read-back. Its quantiles are not convex combinations of member quantiles, so the MASE-gated mixture is not covered by Theorem 2 on MASE or CRPS (Section A.2); the Vincent policy over the same admitted set, `top3-mase`, is the variant covered by the historical-backtest guarantee in Theorem 2.
- **Spread-routed mixture** (`routemix` θ). For each window, compute the pairwise-CRPS spread $s = \frac{2}{k(k-1)} \sum_{i<j} \mathrm{CRPS}(F_i, F_j)$ among admitted members, normalized by the window's own label scale. Use the CDF mixture when $s > \theta$ and Vincent otherwise. We report $\theta \in \{0.3, 0.5, 1.0\}$, with $\theta = 0.5$ as the reported safe alternative. The statistic is a label-free disagreement measure

[1] `rankgauss` falls back to the identity forecast when the context has fewer than 16 observations, contains non-finite values, or has zero range.

closely related to the diversity term in Theorem 3, but differs in both the routing set and scale normalization.

- **Temperature soft-gate.** $w_i \propto \exp(-\widehat{L}(g_i)/T)$ over the admitted set, with T set to the spread of admitted backtest scores. We report this policy for completeness but do not deploy it: it gives the best CRPS and the worst MASE in Table 7.

Policies are named `<rule><k>-<objective>`, where the suffix denotes the backtest ranking objective: `mase` uses median absolute error, while the legacy `crps` suffix uses the nine-level pinball sum rather than exact CRPS. The deployed policy is `mixtop3-mase`.

C COMPUTE LADDER AND ADDITIONAL CONTROLS

We provide the additional details supporting Section 5: the thinking-mode ladder and compute accounting used for the compute frontier, together with the pool, policy, rival, compute-matched, and temporal-robustness controls from Section 5. The BGAA procedure itself is defined in Section 3.

C.1 THINKING MODES AND THE COMPUTE FRONTIER

We use the four *thinking modes* in Table 5 to trace the compute–performance frontier in Figure 2. Across modes, we vary only the test-time compute budget: the number of candidates scored by the gate, the number of gate windows and folds, and the number of views retained for aggregation. All modes draw from the same fixed candidate catalogue and use the same selection procedure and frozen model; lower-compute modes evaluate nested subsets of that catalogue.

Table 5: **Thinking modes define a compute ladder over a fixed candidate catalogue** (GIFT-Eval, 97 configs). Across modes, we vary only the test-time compute budget: the number of candidates scored by the gate, the number of gate windows and folds, and the number of views retained for aggregation. Each mode evaluates a nested subset of the same candidate catalogue, and the selection procedure is fixed. Cost is reported in forward passes relative to the no-test-time compute baseline. The left block gives the full CHRONOS-2 results; the right block reports cost / ΔMASE% for the other three models. `low` and `medium` use `mixtop2-mase`, while `high` and `xhigh` use `mixtop3-mase`; these are compute-scaling variants for the frontier rather than changes to the deployed primary policy.

		CHRONOS-2				TIREX-2	PATCHTST-FM-R1	TOTO-2-313M
mode	cand	cost	ΔMASE%	ΔCRPS%	worst%	cost / ΔMASE%		
base	0	1.00×	—	—	—	1.00×	1.00×	1.00×
low	9	3.12×	−0.53	−0.59	+3.42	2.32× / −0.18	2.43× / −0.47	1.95× / −0.41
medium	19	3.51×	−0.57	−0.74	+6.38	3.23× / −0.20	2.90× / −0.15	2.87× / −0.58
high	31	6.47×	−1.08	−0.81	+7.92	6.15× / −0.50	4.93× / −0.48	5.25× / −0.65
xhigh	51	8.00×	**−1.14**	**−1.04**	+7.92	9.40× / −0.54	8.28× / −0.50	7.87× / −0.91

Compute accounting. Figures 2 and 3 and Table 5 place the test-time compute ladders and published systems on a common compute axis. We use a conservative accounting scheme so that the comparison is consistent across models and compute budgets.

Compute axis. For each model, the $1\times$ point is its measured benchmark PFLOPs for a full GIFT-Eval sweep, using the same profiling table for our models and the published baselines. We place each test-time compute rung at $1 \times \text{PFLOPs} \times c$, where $c = (\text{base} + \text{gate} + \text{apply})/\text{base}$ is the forward-pass ratio recorded during evaluation. We charge every additional pass at the measured cost of a base pass. Since some views use shorter contexts, this slightly overestimates their cost, but gives a simple and comparable accounting across models and rungs.

Interpretation of the compute estimate. The axis measures estimated FLOPs rather than wall-clock time. We do not account for parallel batching across independent view calls, and shortened-context passes are charged at the full base-pass cost. In deployment, the base forecast can also be cached, and the gate cost amortized across forecast windows from the same config. Thus, we treat the reported values as conservative compute estimates.

Paired and published baselines. The ladder values come from our test-time compute runs over the 97 GIFT-Eval configs, while the published stars use the reported benchmark results from the compute table. We keep these sources separate. For example, our paired TOTO-2-313M base is $0.7032/0.4838$, compared with $0.7028/0.4814$ in the published table, while our PATCHTST-FM-R1 base is $0.7052/0.4820$, compared with $0.7069/0.4829$. We use the base from each test-time compute run to anchor its corresponding ladder, so the hollow base marker does not necessarily coincide exactly with the published star.

Table 6: **Thinking modes provide a controlled test-time compute ladder.** All four modes draw from the same fixed augmentation catalogue and use the same BGAA procedure; only the test-time compute budget changes. Lower-compute modes evaluate nested subsets of the catalogue. **(a) Budget parameters:** P denotes the number of forward passes per backtest window, and $W = \mathrm{clip}(\alpha N/(PF), W_{\min}, W_{\max})$ is the median number of gate windows. Gate and apply costs are reported relative to the base run, with gate% denoting the fraction of total cost spent on gating. **(b) Candidate families:** entries show the number of degrees included at each mode. The sets are nested. The resulting total cost and performance for each mode are reported in Table 5.

(a) Budget knobs

	candidates			gate				aggregate	cost split, CHRONOS-2		
mode	families	views	P	α	$W_{\min}/W_{\max}$	W	F	top-k	gate	apply	gate%
`low`	7	9	9	1	24/24	24	1	2	0.05	2.07	1.7
`medium`	11	19	19	6	32/48	48	2	2	0.37	2.15	10.4
`high`	13	31	36	16	64/96	64	3	3	1.81	3.66	28.0
`xhigh`	15	51	58	∞	16/128	128	3	3	4.14	2.87	51.7

(b) Families admitted

family	passes/view	`low`	`medium`	`high`	`xhigh`
`supportfloor`	0	1	1	1	1
`nonneg`	0	1	1	1	1
`rankgauss`	1	1	1	1	1
`winsor`	1	1	1	1	1
`multideseason`	1	1	1	1	1
`recursive`	2–4	1	1	2	2
`lookback`	1	3	6	10	18
`diff`	1	—	1	1	1
`jitter`	n_s	—	1	1	1
`jitternorm`	n_s	—	1	1	1
`offset`	1	—	4	7	14
`sdiff`	1	—	—	1	1
`decomp`	1	—	—	3	3
`fft`	1	—	—	—	3
`seasontile`	1	—	—	—	2
jitter seeds n_s		—	1	2	3

On yearly configs (period = 1), `decomp` contributes 1 degree, so `high`/`xhigh` present 30/50 views there.

Chronos-2 configurations. Our test-time compute results use Chronos-2 with cross-learning disabled, corresponding to the univariate 22.05 PFLOPs configuration. Since that row does not have a published benchmark score, we use our base evaluation to anchor the ladder. We show the standard cross-learning-enabled Chronos-2 separately at 22.83 PFLOPs. The two configurations have nearly identical compute, so this comparison separates gains from cross-series conditioning from gains obtained by spending the same budget on test-time compute. On GIFT-Eval, the cross-learning-enabled model performs better.

Compute scaling on fev-bench. Figure 3 shows the same compute–accuracy analysis on fev-bench. Across all four frozen models, increasing the test-time compute budget improves over the corresponding single-pass baseline on both MASE and SQL. The size of the gain varies across models and metrics. The larger-budget CHRONOS-2 and TIREX-2 settings give the strongest MASE improvements, while the other two models still improve consistently over their own bases.

Thus, we use fev-bench as a second check that the compute–accuracy trade-off extends beyond GIFT-Eval, rather than as evidence that a single model defines the Pareto frontier in every setting. As in Figure 2, each rung is placed using the model's measured $1\times$ benchmark PFLOPs multiplied by its measured forward-pass ratio, $(\text{base} + \text{gate} + \text{apply})/\text{base}$, following the conservative accounting described above.

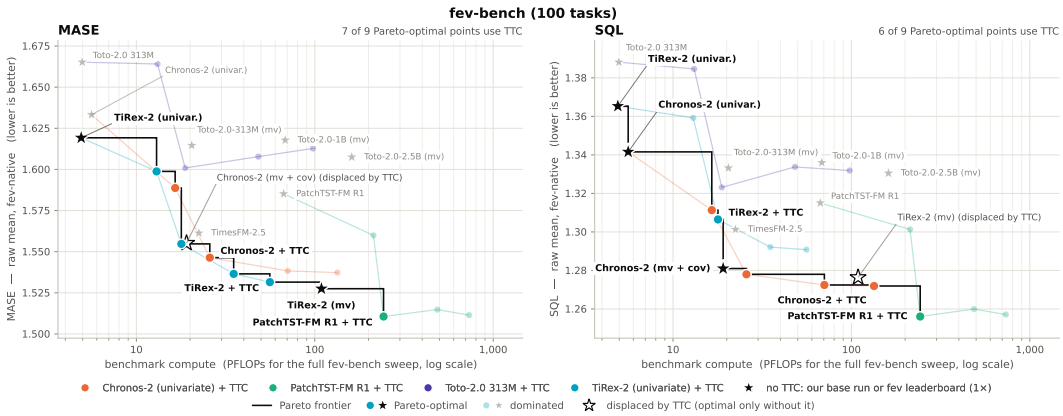


Figure 3: **Pareto curves for accuracy vs computational cost.** (fev-bench, 100 tasks). Lines are our test-time compute ladders over a frozen model, cheapest rung first; stars are published models on fev-bench. A rung sits at its model's measured $1\times$ benchmark PFLOPs times the measured forward-pass ratio $(\text{base} + \text{gate} + \text{apply})/\text{base}$. As in Section C.1, charging each extra pass at the base-pass cost is a conservative accounting convention because some views shorten the context.

C.2 Controls and ablations

We use the following controls to test alternative explanations for the observed gains. In Table 9, we compare the gate against random-k, noise-k, and identity-k baselines at matched $k = 2$ using paired tests. Table 8 restricts the candidate pool to specific view families, while Table 10 progressively ages the backtest windows to test whether the gate mainly exploits the most recent regime.

Table 7 separates the effects of the candidate pool and the aggregation rule. The pool ablation is nested: crops and offsets recover 26% (Chronos-2) and 44% (Toto-2-313M) of the final MASE gain, adding the ideas-tier families improves further, and the decomposition family contributes an additional 0.41% relative MASE improvement for Chronos-2 and 0.14% for Toto-2-313M. The aggregation ablation shows a trade-off between average improvement and regression frequency. Among the fixed top-k policies, the CDF mixture gives the best mean MASE. The spread-routed variant gives up some of this gain but improves the win/loss count $(74/15)$ and reduces the worst regression relative to the deployed mixture. Theorem 3 motivates the label-free spread statistic used by this policy, but its exact CRPS identity applies to the mixture CDF before the nine-quantile read-back, not necessarily to the reported CRPS after read-back. Finally, the temperature soft-gate gives the worst MASE but the best reported CRPS, consistent with the verification limitation studied in Section 7.

C.2.1 Rival controls

We restrict the candidate pool to lookback-only, offset-only, and cleaning-only views, while keeping the same gated top-2 Vincent policy on Chronos-2. We use the same evaluation subset and report paired comparisons between each restricted pool and the full candidate pool.

C.2.2 Compute-matched controls

This control tests whether the gain comes simply from spending $k\times$ more inference compute. We keep the candidate catalogue fixed to the deployed tier (the identity plus 51 non-identity candidates, 52 total views at period 24), use the same $k = 2$ members per window, the same equal-weight Vincent aggregation with the base, and the same `gluonts` scoring. The only change is how the k members are selected. For `random_k`, we average over three seeds to reduce variance from a single random draw.

First, **noise-k gives a small but measurable gain**. Averaging k tiny-jitter copies of the identity ($\sigma = 0.002$) improves MASE by 0.03% ($p = 2 \times 10^{-4}$). This suggests that generic variance reduction contributes to the overall effect, but only modestly: noise-k recovers about 3% of the

Table 7: **Pool and aggregation ablations separate the sources of improvement. Left:** nested pool ablations for CHRONOS-2 (top) and TOTO-2-313M (bottom) on GIFT-Eval, using top-2 Vincent unless otherwise noted. **Right:** aggregation ablations using the deployed pool on GIFT-Eval. Expanding the candidate pool generally improves MASE, while the aggregation rule determines the trade-off between MASE and CRPS. For CHRONOS-2, the crop tiers provide essentially no CRPS improvement (both $p > 0.05$), and switching to the mixture is approximately MASE-neutral. The decomposition family provides the final MASE gain of the deployed configuration over the preceding pool.

pool tier	MASE	CRPS	policy	MASE	CRPS
base (no TTC)	0.7043	0.4830	margin (conservative)	0.7011	0.4818
crop core	0.7022	0.4829	top-2 Vincent	0.6971	0.4792
+ micro views	0.7020	0.4825	top-3 Vincent	0.6968	0.4788
+ 9 ideas families	0.6991	0.4807	CDF mixture top-3	**0.6963**	0.4780
+ mixture aggregation	0.6992	0.4801	spread-routed mixture	0.6988	0.4794
+ decomposition family	**0.6963**	**0.4780**	tempvinc-mase	0.7042	**0.4766**

pool tier	MASE	CRPS	policy	MASE	CRPS
base (no TTC)	0.7032	0.4838	margin (conservative)	0.6997	0.4808
crop core	0.7004	0.4817	top-2 Vincent	**0.6968**	0.4795
+ micro views	0.7000	0.4812	top-3 Vincent	0.6975	0.4793
+ 9 ideas families	0.6980	0.4808	CDF mixture top-3	**0.6968**	0.4788
+ mixture aggregation	0.6978	0.4800	spread-routed mixture	0.6974	0.4789
+ decomposition family	**0.6968**	**0.4788**	tempvinc-mase	0.7002	**0.4713**

Table 8: **Restricted-pool controls** (CHRONOS-2, gated top-2 Vincent, subset scope). "share" denotes the fraction of the full-pool gain recovered by each restricted pool. The right block reports paired full-vs.-control comparisons on the same configs. We use these paired tests for the main comparison, since a restricted pool can recover part of the aggregate gain while still underperforming the full pool on the same examples.

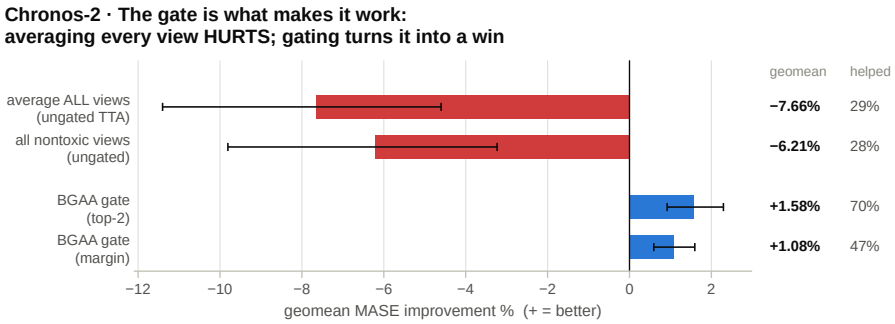
	control alone				full pool vs control		
pool	ΔMASE%	w/l	p	share	Δ%	w/l	p
full pool	-1.40	11/2	0.013	100%	—	—	—
lookback-only	-0.52	10/1	0.016	38%	-0.88	7/4	0.248
offset-only	-0.09	5/2	0.237	6%	-1.31	10/3	0.033
cleaning-only	-0.65	8/2	0.022	46%	-0.75	9/5	0.470

MASE gain and 10% of the CRPS gain. The gated policy still performs substantially better head to head, by 0.99% MASE ($p = 2 \times 10^{-7}$) and 0.71% CRPS ($p = 4 \times 10^{-4}$).

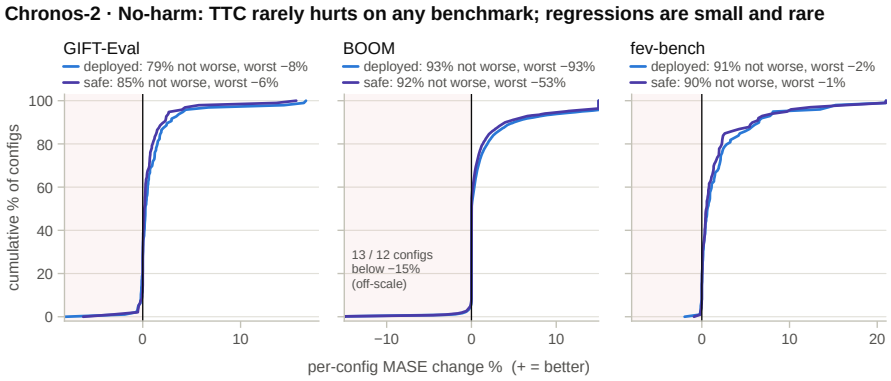
Second, **random-k performs worse than the base forecast**, by 4.82% MASE and 10.73% CRPS. This is expected for a broad candidate pool that contains views which can be poorly aligned with a given series (Section E.2): uniformly sampling from the pool often includes harmful candidates. We therefore do not report a "share of gain recovered" for this arm, since the corresponding ratio would be negative. The result also shows that additional test-time compute is not useful by itself; without selection, it can degrade performance.

Third, **identity-k is an exact no-op**. Its change is 0.00 on all 97 configs at full score precision. We use this as a sanity check on the evaluation harness: equal-weight Vincent aggregation of duplicate forecasts should reproduce the original forecast exactly. A non-zero change here would indicate that the aggregation implementation itself is altering the output, for example through quantile reordering or interpolation.

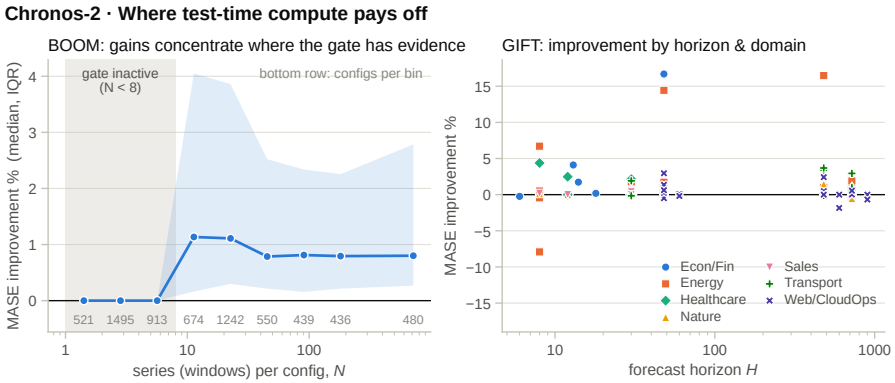
Note: the reference arm in this experiment is gated top-2 Vincent rather than the deployed `mixtop3-mase` policy. This gives a MASE improvement of -1.02%, compared with -1.14% for the deployed policy in Table 1 on the same 97 configs. We use the simpler top-2 Vincent policy for all arms so that the selection rule is the only factor that changes across the comparison.



(a) Ungated versus gated aggregation.



(b) CDF of per-config MASE changes.



(c) Gain as a function of backtest windows N and horizon H.

Figure 4: **The gate drives the gains and benefits from sufficient backtest evidence.** **(a)** Ungated aggregation over the full candidate pool compared with the gated policy. **(b)** CDFs of per-config MASE changes for the policies in Table 3. **(c)** MASE improvement as a function of the number of available backtest windows N and forecast horizon H. When $N < 8$, the gate is disabled and BGAA returns the base forecast, as described in Section 3.

C.2.3 Temporal robustness

We test how sensitive the gate is to the recency of its backtest evidence. For a given lag, we restrict the backtest to observations at least lag $\times H$ before the forecast start, while the final forecast still uses the full context. If the gate mainly relies on the most recent regime, we would expect its performance to deteriorate quickly as the lag increases.

Table 9: **Compute-matched controls** (CHRONOS-2, GIFT-Eval, $n = 97$, $k = 2$). The left block compares each control with the frozen base. The right block reports the paired comparison between BGAA and that control on the same configs. We use the latter as the main test, since a control may improve over the base while still underperforming the gated selection rule. All significance tests use paired Wilcoxon tests across configs.

	vs base (MASE)			vs base (CRPS)			BGAA vs arm (MASE)		
arm	$\Delta\%$	w/l	p	$\Delta\%$	w/l	p	$\Delta\%$	w/l	p
BGAA top-2	-1.02	70/19	1×10^{-8}	-0.79	67/22	5×10^{-5}	—	—	—
random-k	$+4.82$	35/60	4×10^{-6}	$+10.73$	31/63	8×10^{-8}	-5.57	79/18	1×10^{-12}
noise-k	-0.03	69/28	2×10^{-4}	-0.08	71/26	5×10^{-6}	-0.99	67/30	2×10^{-7}
identity-k	$+0.00$	0/0	1	$+0.00$	0/0	1	-1.02	70/19	1×10^{-8}

Table 10: **Temporal robustness of the gate** (CHRONOS-2, top-2 Vincent, subset scope, $n = 15$). We progressively restrict the backtest to older windows. *Jaccard* reports the overlap between the admitted view set at each lag and the set selected at lag 0.

lag ($\times H$)	ΔMASE%	ΔCRPS%	p (MASE)	Jaccard vs lag 0
0	-1.40	-1.07	0.013	1.00
1	-1.11	-1.19	0.011	0.58
2	-1.59	-0.78	0.005	0.50
4	-1.15	-1.11	0.015	0.53

The gain remains stable across lags, ranging from -1.1% to -1.6%, and is significant in every case, even though the admitted set changes substantially. This suggests that the gate is exploiting a signal that is not limited to the most recent regime. We do not interpret the non-monotonic pattern across lags as evidence that older backtest windows are better; with only $n = 15$ configs, the difference at lag 2 is within the noise of this subset. CRPS follows a similar trend to MASE, while MASE is the metric used to rank the policy.

D FULL RESULTS AND VERIFICATION DIAGNOSTICS

This section collects the full results underlying Section 5 and the verification analyses from Section 7. We include the cross-model policy grid for each benchmark, the complete official BOOM leaderboard corresponding to the excerpt in Table 2, the per-domain breakdown, the cross-fitted headroom and ranking results, and a small set of qualitative forecast examples illustrating how BGAA changes the prediction on selected successful windows. Compute-ladder, gate-ablation, and matched-control analyses are kept with their underlying experiments in Sections C.1 and C.2 rather than duplicated here.

D.1 BENCHMARK-LEVEL RESULTS

Table 11: **GIFT-Eval results across four models and four policies** (97 configs; geomean of metric/seasonal-naive). We report $\Delta\%$ and MASE win/loss counts relative to each model's own base; all MASE comparisons have $p < 0.001$. `mixtop3-mase` is the deployed policy. `routemix05-crps` gives the best win/loss ratio for PATCHTST-FM-R1 and ties for the best ratio on CHRONOS-2, but with a smaller mean gain. `top2-mase` has the best win/loss ratio for TIREX-2 and TOTO-2-313M. On MASE, PATCHTST-FM-R1 is the only model for which `top2-mase` strictly outperforms the deployed policy; TOTO-2-313M ties it at 0.6968.

model	policy	MASE	CRPS	ΔMASE%	ΔCRPS%	w/l
CHRONOS-2	base	0.7043	0.4830	—	—	—
	`top2-mase`	0.6971	0.4792	−1.02	−0.79	70/19
	`top3-mase`	0.6968	0.4788	−1.06	−0.87	74/15
	`routemix05-crps`	0.6988	0.4794	−0.79	−0.74	74/15
	`mixtop3-mase`	**0.6963**	**0.4780**	**−1.14**	−1.04	69/20
TIREX-2	base	0.6972	0.4791	—	—	—
	`top2-mase`	0.6938	0.4777	−0.49	−0.30	73/17
	`top3-mase`	0.6940	0.4779	−0.45	−0.26	71/19
	`routemix05-crps`	0.6948	0.4781	−0.33	−0.21	67/22
	`mixtop3-mase`	**0.6934**	**0.4778**	**−0.54**	−0.27	68/22
PATCHTST-FM-R1	base	0.7052	0.4820	—	—	—
	`top2-mase`	0.7009	0.4810	−0.62	−0.22	72/16
	`top3-mase`	0.7022	0.4818	−0.43	−0.05	69/19
	`routemix05-crps`	0.7038	0.4818	−0.21	−0.05	75/13
	`mixtop3-mase`	**0.7017**	**0.4812**	−0.50	−0.18	70/18
TOTO-2-313M	base	0.7032	0.4838	—	—	—
	`top2-mase`	0.6968	0.4795	−0.91	−0.90	73/17
	`top3-mase`	0.6975	0.4793	−0.81	−0.93	72/18
	`routemix05-crps`	0.6974	0.4789	−0.82	−1.01	69/20
	`mixtop3-mase`	**0.6968**	**0.4788**	**−0.91**	−1.03	68/22

Energy contains the largest number of configs and improves for all four models, while Healthcare is the most consistent domain by win/loss count ($4/0$ for every model). Web/CloudOps is flat or harmful on GIFT-Eval (except TOTO-2-313M), but the same pattern does not appear on BOOM, which consists entirely of observability data and shows large overall gains in Table 1. One relevant difference is the amount of available backtest evidence: the Web/CloudOps configs in GIFT-Eval are relatively short, whereas the BOOM tasks provide substantially more windows for the gate. This comparison is descriptive and does not isolate history length from domain effects.

Benchmark comparability notes.

- **BOOM.** Our BOOM results are directly comparable to the public leaderboard. We evaluate the full official universe of 7413 dataset×term tasks and use DataDog's evaluation code, which produces the 6750-task scaled set, replaces non-finite raw metrics by column means, normalizes by the published seasonal-naive run, and then computes the shifted geometric mean with $\varepsilon = 10^{-5}$. We also recompute all 23 published systems through the same pipeline and verify the resulting

Table 12: **fev-bench results across all four models** (all 100 tasks; `fev`-native raw metric means and skill relative to the published seasonal-naive baseline, where higher is better). SQL is the primary `fev` metric and is used for the paired win/loss counts; every test-time compute row has $p < 0.001$ against its corresponding base. `routemix05-crps` again trades some mean improvement for a substantially better win/loss ratio, e.g., $85/8$ versus $70/23$ for TiRex-2.

model	policy	SQL	WQL	MASE	SQL skill	MASE skill	SQL w/l
Chronos-2	base	1.3416	0.2193	1.6332	0.438	0.312	—
	routemix05-crps	1.2733	0.2142	1.5431	0.450	0.328	90/4
	mixtop3-mase	1.2719	0.2151	1.5373	0.450	0.329	76/18
TiRex-2	base	1.3653	0.2275	1.6192	0.426	0.305	—
	routemix05-crps	1.2942	0.2229	1.5492	0.436	0.314	85/8
	mixtop3-mase	1.2908	0.2223	1.5315	0.435	0.317	70/23
PatchTST-FM-R1	base	1.3150	0.1956	1.5852	0.446	0.328	—
	routemix05-crps	1.2584	0.1943	1.5149	0.459	0.343	81/10
	mixtop3-mase	1.2571	0.1939	1.5114	0.460	0.345	70/24
Toto-2-313M	base	1.3881	0.2151	1.6652	0.433	0.316	—
	routemix05-crps	1.3380	0.2121	1.6103	0.440	0.323	84/8
	mixtop3-mase	1.3319	0.2107	1.6127	0.444	0.325	68/23

Table 13: **Full official BOOM leaderboard** (6750 scaled tasks from the 7413-task benchmark; DataDog seasonal-naive baseline; shifted geometric mean with $\varepsilon = 10^{-5}$). We recompute all 23 published systems using DataDog's evaluation pipeline and verify the reproduced scores against the published leaderboard. Bold rows are our runs. Table 2 shows the corresponding excerpt used in the main text.

system	MASE	CRPS	system	MASE	CRPS
Toto-2-313M+BGAA	**0.574**	**0.349**	`moirai_2_small`	0.672	0.427
`toto_2.0_2.5b`	0.581	0.349	`moirai_1.1_base`	0.710	0.428
`toto_2.0_1b`	0.582	0.349	`moirai_1.1_large`	0.720	0.436
`toto_2.0_313m`	0.585	0.351	`moirai_1.1_small`	0.729	0.442
Toto-2-313M base	**0.585**	**0.353**	`timesfm_2_0_500m`	0.725	0.447
`toto_2.0_22m`	0.601	0.363	`chronos_bolt_base`	0.726	0.451
`toto` (v1)	0.617	0.375	`chronos_bolt_small`	0.733	0.455
TiRex-2+BGAA	**0.621**	**0.377**	`timer`	0.796	0.639
`toto_2.0_4m`	0.624	0.377	`time_moe_200M`	0.881	0.643
Chronos-2+BGAA	**0.629**	**0.382**	`time_moe_50M`	0.806	0.649
`timesfm_2_5_200m`	0.632	0.382	`visionts`	0.988	0.673
TiRex-2 base	**0.642**	**0.386**	`autoarima`	0.824	0.736
`chronos_2`	0.641	0.392	`seasonalnaive`	1.000	1.000
PatchTST-FM-R1+BGAA	**0.646**	**0.392**	`autotheta`	1.123	1.018
Chronos-2 base	**0.655**	**0.394**	`autoets`	0.842	1.975
PatchTST-FM-R1 base	**0.681**	**0.407**			

Ordered by CRPS, as the leaderboard is. BGAA moves every one of our four models up the table. Toto-2-313M with test-time compute (0.574) is the best MASE of any *zero-shot* row here.

scores against `boom_leaderboard.csv`; for example, Toto-2-313M approximately reproduces at $0.585/0.351$.

Note: our evaluations are univariate, whereas published leaderboard entries use each model's native configuration, although our Toto-2-313M base approximately reproduces the published Toto-2-313M result for the same checkpoint. Also, the 663 tasks with zero seasonal-naive score, which the leaderboard handles separately, are not part of the scaled table reported here.

- **fev-bench.** We report fev-bench using its native raw metric means together with skill relative to the published seasonal-naive baseline. Temperature-family policies achieve higher skill but worse raw-mean CRPS and do not produce results for 8 short-horizon tasks. The deployed mix, route, and top-k policies improve both raw means and skill across all 100 tasks.

Table 14: GIFT-Eval results for CHRONOS-2 across the deployed policy grid. Changes are relative to the base score of 0.7043 MASE / 0.4830 CRPS; "worst" denotes the largest regression on any single config. **Bold** marks the final deployed policy.

policy	MASE	Δ%	p	w/l	worst%	CRPS	Δ%	p	w/l
base	0.7043	—	—	—	—	0.4830	—	—	—
margin-mase	0.7011	−0.46	<0.001	34/6	+12.07	0.4818	−0.24	0.064	29/11
top2-mase	0.6971	−1.02	<0.001	70/19	+6.08	0.4792	−0.79	<0.001	67/22
top3-mase	0.6968	−1.06	<0.001	74/15	+8.39	0.4788	−0.87	0.001	66/23
mixtop2-mase	0.6966	−1.09	<0.001	68/21	+5.67	0.4783	−0.97	0.002	61/28
mixtop3-mase	**0.6963**	−1.14	<0.001	69/20	+7.92	0.4780	−1.04	<0.001	64/25
routemix05-crps	0.6988	−0.79	<0.001	74/15	+6.08	0.4794	−0.74	<0.001	75/15
tempvinc-crps	0.7069	+0.37	<0.001	77/12	+7.22	0.4785	−0.92	<0.001	74/16
tempvinc-mase	0.7042	−0.01	<0.001	74/15	+13.31	0.4766	−1.32	<0.001	68/21
tempmix-mase	0.7031	−0.17	<0.001	69/20	+9.61	0.4747	−1.71	0.001	64/25

Table 15: **Per-domain CRPS change on GIFT-Eval** under the deployed `mixtop3-mase` policy. We report the mean Δ% across configs in each domain, with win/loss counts in parentheses. This is the domain-level counterpart of the fingerprint analysis in Figure 10a and Section 6: the pattern varies substantially across models. For example, Econ/Fin ranges from −4.00% for CHRONOS-2 to +1.53% for TIREX-2, while Web/CloudOps ranges from −1.47% for TOTO-2-313M to +2.46% for PATCHTST-FM-R1. Healthcare is the only domain where all four models improve (4/0 each), with the largest gain for TOTO-2-313M at −4.42%.

model	Econ/Fin	Energy	Health	Nature	Sales	Transport	Web/CloudOps
CHRONOS-2	−4.00 (4/2)	−0.93 (24/8)	−3.43 (4/0)	−0.25 (10/4)	−0.25 (2/1)	−0.75 (11/3)	−0.19 (9/7)
TIREX-2	+1.53 (1/5)	−1.09 (27/5)	−3.54 (4/0)	−0.08 (10/4)	−0.04 (2/1)	−0.42 (11/4)	+1.60 (9/7)
PATCHTST-FM-R1	−0.75 (3/1)	−0.74 (26/6)	−1.64 (4/0)	−0.34 (10/4)	−0.17 (3/0)	−0.63 (12/3)	+2.46 (11/5)
TOTO-2-313M	−0.87 (2/4)	−1.07 (28/4)	−4.42 (4/0)	−0.34 (9/5)	−0.43 (3/0)	+0.06 (9/6)	−1.47 (13/3)

D.2 VERIFICATION HEADROOM AND LABEL-BASED DIAGNOSTICS (TABLE 16)

We retain the per-window pinball loss on the realized test horizon for every view and use it to construct three label-based diagnostics. The *cross-fitted best-view gain* selects one fixed view per config, using disjoint halves of the windows for selection and evaluation. The *label-ranked gate* keeps the deployed policy unchanged but replaces the backtest ranking with a ranking computed from labeled held-out windows. The *per-window hindsight selection* selects the lowest-loss view separately for each forecast window using that window's labels. These diagnostics use future labels only for analysis; the deployed policy does not.

Evaluation protocol. We score all diagnostics with the same gluonts MASE and CRPS metrics used in Table 1 and apply the same aggregation procedure. Each reported value is the relative change in the aggregate metric with respect to the base forecast, computed over the same set of windows.

- *Windows.* For each config, we use $W = \min(1024, N)$ evenly spaced test windows from the evaluation sampler used for the main results. We evaluate the identity forecast together with every available non-identity view in the deployed catalogue on these windows and retain the per-window losses. For MASE, we drop windows with zero seasonal error to match gluonts.
- *Deployed policy.* We evaluate the view set admitted by the deployed run using the deployed CDF mixture. This reproduces the per-config results underlying Table 1 to within 10^{-7} relative error.
- *Cross-fitting split.* We divide the W windows into two contiguous halves and remove the $\lceil H/\text{stride} \rceil$ windows closest to the split boundary so that selection and evaluation labels

Table 16: **Verification headroom is much larger at the forecast-window level.** CRPS improvement over the frozen base in %. The first "realized" column uses all 97 configs; the others use the 53 configs that can be cross-fit. The *label-ranked gate* keeps the deployed policy fixed but ranks views with held-out labels. The reported realized/label-ranked ratio is the realized improvement divided by the label-ranked improvement. The *best-view gain (x-fit)* selects one view per config on one half of its windows and evaluates it on the other. The *per-window hindsight selection* selects the best view per window using its realized labels. These label-based quantities are diagnostics, not deployable verifier targets.

	all 97 configs	53 configs long enough to cross-fit				
model	realized (Table 1) %	realized %	label-ranked gate %	realized / label-ranked ratio	best-view gain (x-fit) %	per-window hindsight %
CHRONOS-2	1.04	1.54	1.84	0.84	4.10	19.4
TIREX-2	0.27	0.54	0.89	0.61	1.48	16.6
PATCHTST-FM-R1	0.18	0.39	1.17	0.33	1.42	17.5
TOTO-2-313M	1.03	0.87	0.99	0.88	1.38	16.6

Table 17: **Supporting diagnostics for Table 16.** Results are reported on the same 53 cross-fittable configs as improvement over the base in percent (positive is better). *In* selects and evaluates on the same windows, while *x-fit* selects on one half and evaluates on the other. The MASE block reports the same diagnostics using the point-forecast metric.

	CRPS				MASE					
	best-view gain		label-ranked gate		realized	label-ranked	realized/	best-view gain		per-window
model	in	x-fit	in	x-fit		gate (x-fit)	label-ranked	in	x-fit	hindsight
TIREX-2	3.43	1.48	1.06	0.89	0.58	0.57	1.02	1.79	1.09	19.0
CHRONOS-2	5.36	4.10	1.79	1.84	1.00	1.18	0.85	2.44	0.76	20.7
PATCHTST-FM-R1	3.54	1.42	1.09	1.17	0.35	1.23	0.28	3.46	2.36	21.3
TOTO-2-313M	2.94	1.38	1.14	0.99	1.11	1.25	0.88	1.74	0.72	19.4

do not overlap. We then swap the two halves and pool the held-out evaluations. We require at least 8 retained windows per half, which leaves 53 of the 97 configs. On the remaining 44 shorter configs, the deployed policy improves by -0.1 to 1.2% across models.

- *Label-ranked gate.* We keep the deployed top-3 selection rule but rank views using labeled test-window losses instead of backtest losses. The *in* variant ranks on the same half used for evaluation; the reported *x-fit* variant ranks on the opposite half.
- *Best-view gain.* We select a single view, including identity, with the lowest CRPS on the selection half and evaluate it on the other half.
- *Per-window hindsight selection.* For each evaluation window, we select the view with the lowest summed pinball loss using that window's realized labels.

On the 53 cross-fittable configs, the deployed policy's all-window and held-out improvements differ by less than 0.1 percentage points for every model.

Table 16 summarizes these diagnostics for each model. The realized/label-ranked ratio is the realized improvement divided by the improvement from the label-ranked gate on the same 53 configs. This measures how much of the gain available to the fixed policy is recovered by the leak-free backtest ranking. In-sample best-view selection is optimistic: it exceeds the cross-fitted gain by 1.3–2.5$\times$ (Table 17), which is why we use the split above. The label-ranked gate is less sensitive, with in-sample and x-fit results differing by at most 0.2 percentage points.

For MASE, the realized/label-ranked ratio exceeds one for TIREX-2. We therefore interpret this quantity as a ranking diagnostic, not as a ceiling: under distribution shift, labels from the opposite half can rank views worse than the recent backtest. The two halves are also contiguous and autocorrelated, and their contexts may overlap near the split boundary, so cross-fitting reduces label reuse without making the samples independent. The per-window hindsight best-view diagnostic should be read similarly. It uses the realized outcome and selects only among individual views; for CRPS, an aggregate can outperform every member (Theorem 3), so this hindsight best-view diagnostic is not an upper bound over aggregation policies.

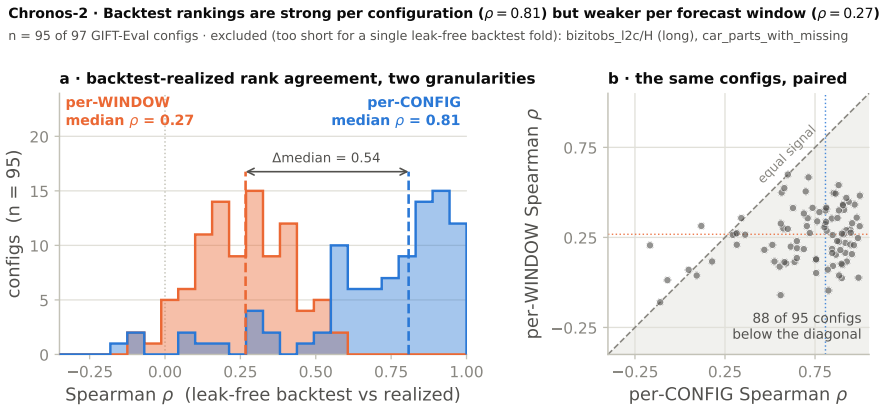


Figure 5: **Backtest rankings are strong per configuration but much weaker per forecast window.** For CHRONOS-2 on 95 GIFT-Eval configurations, median Spearman ρ between leak-free backtest and realized view rankings falls from 0.81 per configuration to 0.27 per window. Results use 8,313 sampled windows and MASE.

Configuration-level versus window-level ranking. The ranking diagnostics show the same separation. On CHRONOS-2, the median Spearman correlation between backtest and realized view rankings is 0.81 when rankings are aggregated per configuration, but only 0.27 when computed per forecast window (Figure 5). The top-k hit-rate and gain-recovery diagnostics show the same degradation at the window level (Figure 6). Because the per-window targets use the realized outcome, they quantify ranking difficulty and heterogeneity rather than an achievable target for a context-only verifier.

Why averaging more backtest windows does not solve window-level selection. The gate statistic averages relative loss over m historical windows. Under the usual independent-noise heuristic, the sampling component of its standard error decreases on the order of $1/\sqrt{m}$. The configuration-level gate uses $m = 8$–128 windows, whereas a window-specific comparison based on a single realized window has no analogous averaging. This heuristic helps explain why configuration-level rankings are more stable, but it is not a finite-sample guarantee: the gate windows are dependent time-series samples, and the realized future window may come from a different local regime. More historical windows can stabilize a configuration-level average; they do not by themselves identify which view will be best on a particular future window.

Fine-grained weighting is not better supported by the current signal. The policy ablation is consistent with this limitation. The temperature-weighted gate uses more of the relative backtest-score information: it achieves the best CRPS among the tested policies but the worst MASE (Table 7). Fixed top-k instead uses the backtest primarily to identify a small set of promising views and is less sensitive to errors in their fine ordering. This is evidence about the policies tested here, not a general claim that weighting cannot help.

D.3 QUALITATIVE EXAMPLES

We additionally inspect a small set of forecast windows where the deployed BGAA policy improves over the frozen base model. These examples illustrate the behavior of BGAA, and each plot follows the same format. The left panel shows the base context together with the inputs or forecast-side transformations associated with the views admitted by the gate. The middle panel compares the forecasts produced by these views before aggregation, together with the resulting BGAA mixture. The right panel compares the base and final predictive distributions against the realized forecast horizon. This makes it possible to inspect how the admitted views alter the frozen model's prediction and how their combination changes the final forecast in windows where BGAA is successful.

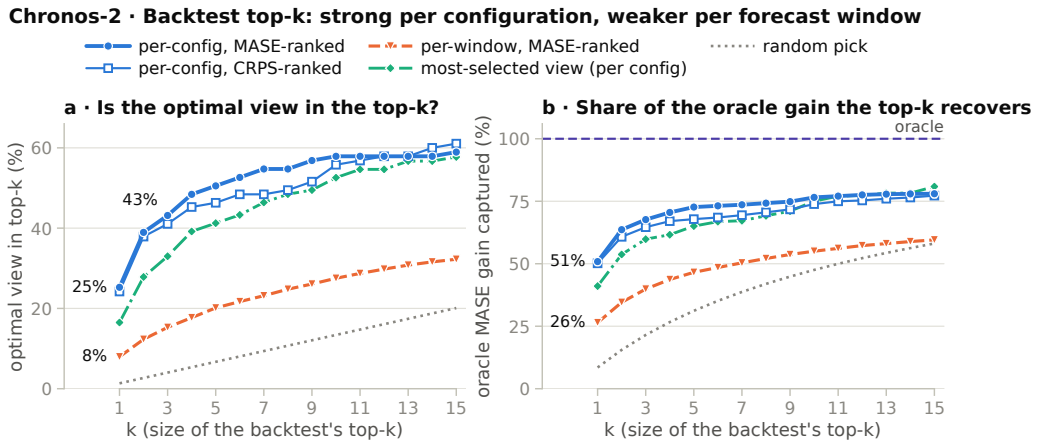


Figure 6: **Agreement between backtest and hindsight view rankings for CHRONOS-2 on GIFT-Eval.** (a) Fraction of configs whose realized best view lies in the backtest's top-k. Ranked per config, the backtest's top-1 contains the best view for 24–25% of configs and its top-3 for 41–43%, well above both the most-selected-view heuristic (16% at k=1) and random selection (1.3% at k=1). The top-1 hit rate is therefore about 19× chance. Ranked per window, the top-1 hit rate falls to about 8%. (b) Share of the hindsight MASE gain recovered by the top-k: the per-config top-1 recovers 50–51%, while the per-window top-1 recovers about 26%. Because the per-window target depends on the realized outcome, we use it only as a diagnostic rather than as an achievable target for a verifier that observes only x.

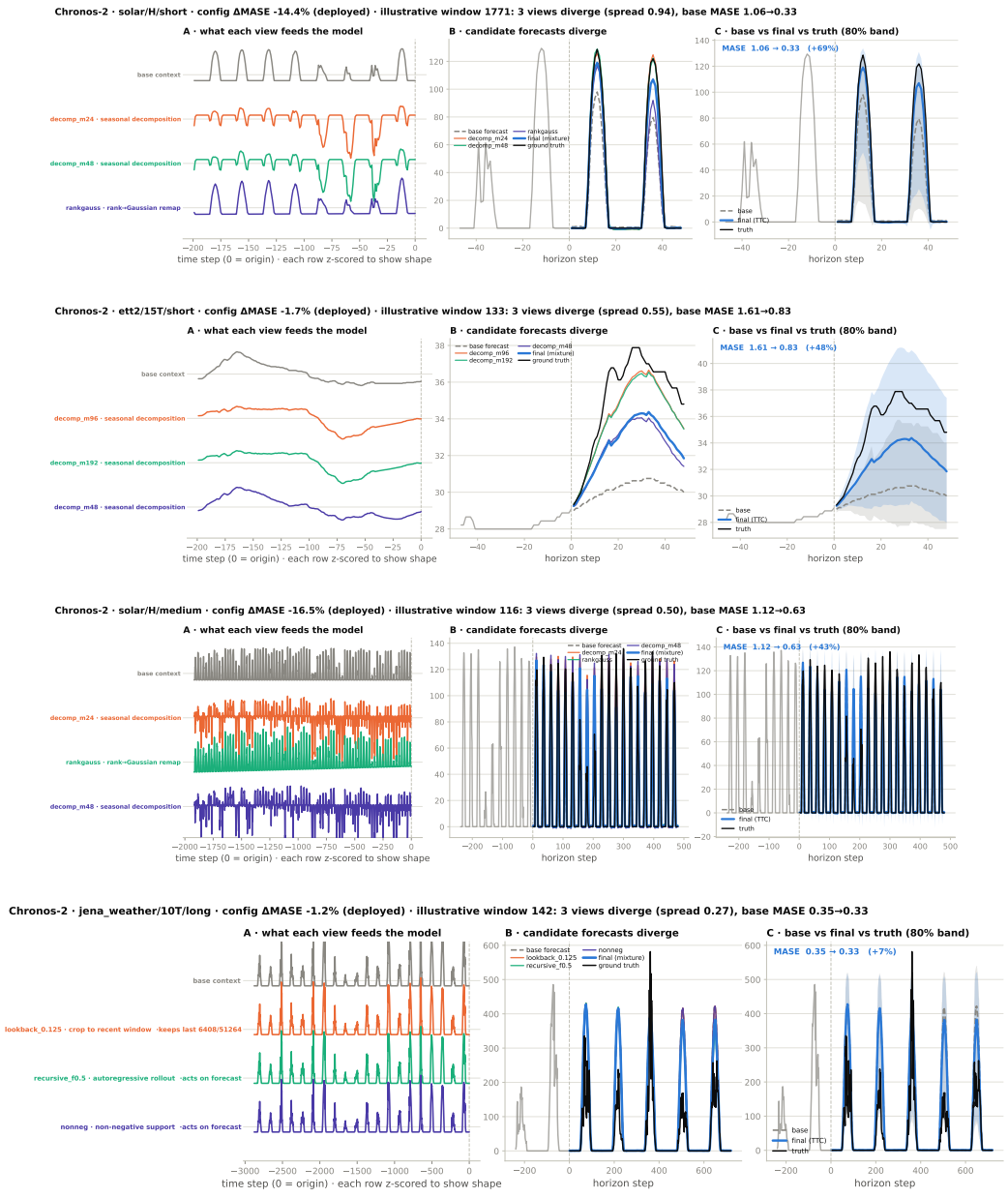


Figure 7: **Qualitative examples for BGAA improvements.** Selected forecast windows for which BGAA improves over the frozen base model. Each subfigure shows the admitted views, their candidate forecasts, and the resulting BGAA prediction relative to the realized horizon.

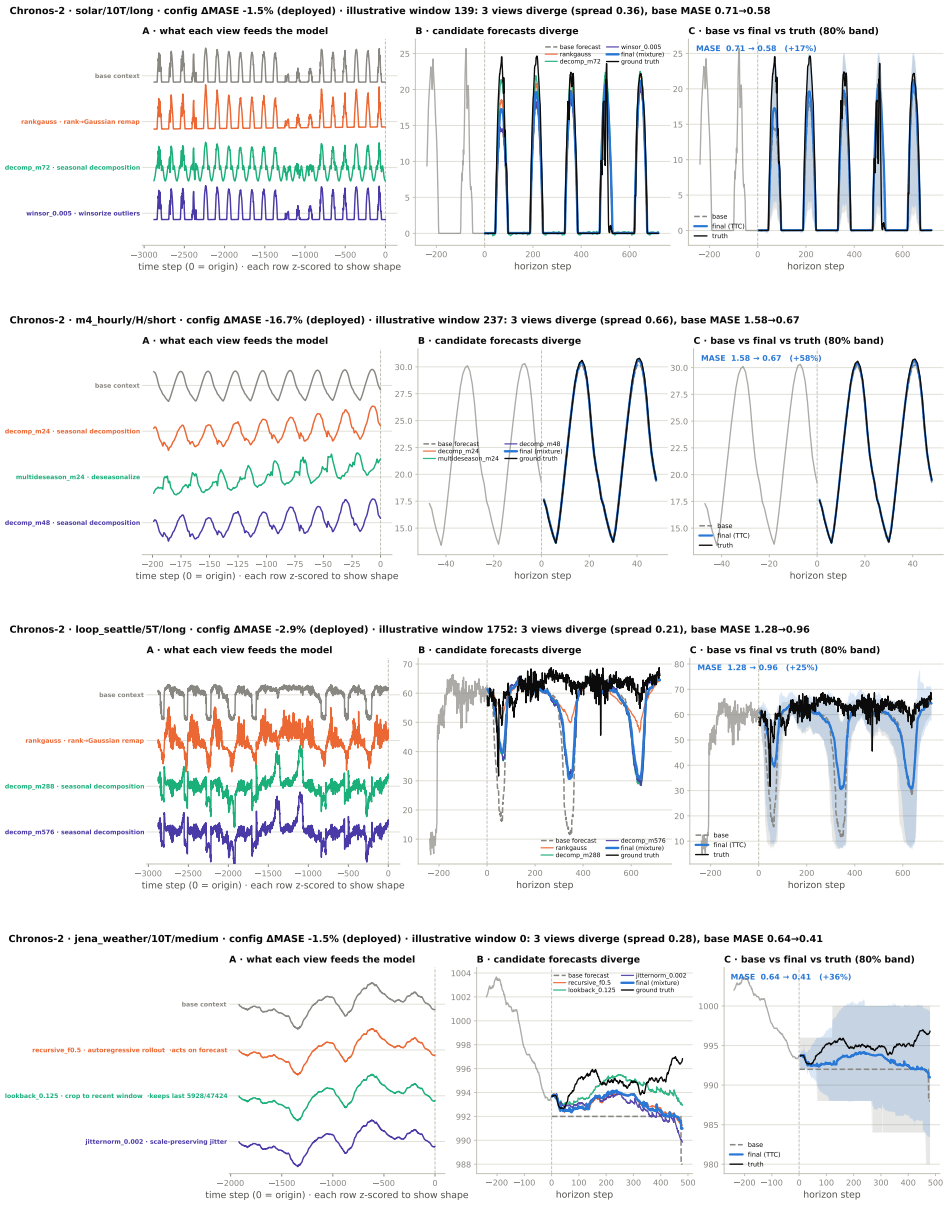


Figure 8: **Qualitative examples for BGAA improvements.** Additional successful forecast windows under the deployed policy.

E MECHANISM EXPERIMENTS: PROTOCOLS AND FULL RESULTS

This section gives the full protocols and results for the three mechanism analyses in Section 6: fingerprinting, prior geometry, and forecast signatures. When an experiment reuses forecast evaluation, we run it through the same evaluation pipeline used for the main results, with the same candidate pool, leak-free backtest, admission rules, aggregation, and scoring. We do not modify the deployed outputs for these analyses.

Experiments that restrict the candidate pool or change the selection rule are reported separately with the controls in Section C.2. Unless noted otherwise, the experiments in this section use a 15-config GIFT-Eval test *subset* used for selected mechanism and forecast-signature analyses rather than the full set of 97 configs.

E.1 FINGERPRINTING (SECTION 6)

We evaluate each view family separately for each model by restricting the candidate pool to that family plus the identity and applying the deployed backtest gate unchanged. We then measure the resulting gain relative to that model's own base forecast. Families are drawn from the full fixed pool, which overlaps the deployed tier in Table 4 but differs from it in four ways: it adds `changepoint` and `warp`, gates `jitter` and `jitternorm` over their full 7-degree grids ($\sigma \in [0.002, 0.03]$) rather than the single deployed $\sigma = 0.002$, merges `sdiff` into `diff` and `supportfloor` into `nonneg`, and omits the cleaning views (`smooth`, `ewma`, `winsor`). To compare family preferences across models, we compute Kendall's τ between the resulting 14-family benefit vectors. Permutation p-values are estimated from 10^4 random label shuffles.

Why the fingerprint uses the full pool. We fingerprint the full fixed pool rather than the deployed tier, for three reasons. First, the deployed tier is itself the output of a label-informed admit decision (Section B). Fingerprinting only its survivors would select the family set on the quantity being compared. Second, that selection keeps the families that helped. It truncates the benefit vectors, and truncation typically pulls rank correlation toward zero, the direction our claim predicts. The full pool removes the truncation and makes the test harder. Restricted to the 12 families it shares with the deployed tier, mean cross-model τ falls from 0.19 to 0.15 (range -0.09 to 0.33), and removing any single family keeps every pair at $\tau \leq 0.45$. Third, gating each family over its full degree grid measures it at the degree the gate prefers for each model, not at the one fixed for deployment (`jitter`'s single σ=0.002). The grids are identical across models, so any advantage a larger grid gives is shared by all four columns and would tend to add agreement, not remove it.

The differences are also qualitative. TIREX-2's strongest family is `jitter` (-0.36%). `lookback` is strongest for PATCHTST-FM-R1 (-0.65%) and TOTO-2-313M (-1.39%), but gives little gain for TIREX-2 (-0.05%). Within CHRONOS-2, admitted families also vary across domains (Figure 10a), supporting the use of config-specific selection.

We distinguish two notions of cross-model agreement. The *solo-gated* benefit vectors above have Kendall τ in the range 0.01–0.35. In contrast, when we compute the same comparison using per-config *admission* patterns over full GIFT-Eval, the agreement is much higher (0.50–0.72). This is largely because admission is dominated by the two crop families, which are used by all four models. For the mechanism analysis, we therefore focus on agreement in realized family benefit rather than agreement in whether a family is admitted.

E.2 PRIOR GEOMETRY (SECTION 6)

We use ancestral rollouts as a proxy for each model's learned forecasting behavior, with short-step samples drawn by inverse-CDF sampling from the model's own 9-quantile head. We study the resulting geometry by embedding both contexts and rollout samples in two spaces: a 20-dimensional shape-feature space and the model's own embedding space. For each model, we standardize both using its rollout samples. We identify modes by applying k-means to the rollout distribution and name each mode by its most distinctive features. For context i and view g, we define the displacement as $d_i = z(g(x_i)) - z(x_i)$, using $n = 626$ GIFT-Eval contexts. Figure 9 shows the joint t-SNE projection; we also compute PCA, embedding-space, and three-dimensional projections.

Table 18: **Solo-gated family benefit in ΔMASE%** (15 config GIFT-Eval test subset; more negative is better; 0 indicates that the family was never admitted). Each column gives the family-level fingerprint for one model; pairwise agreement between these fingerprints is reported in Table 19. **Bold** marks the best-performing family for each model. The preferred family differs across models, and TIREX-2 has the flattest profile: its best family yields about a quarter of the gain of TOTO-2-313M's best family. Notably, `diff` is the strongest family for CHRONOS-2 but slightly hurts TIREX-2.

family	CHRONOS-2	TIREX-2	PATCHTST-FM-R1	TOTO-2-313M
changepoint	−0.034	0	0	0
decomp	−0.814	−0.084	−0.109	−0.214
diff	**−0.849**	+0.012	−0.537	−0.264
fft	−0.030	0	−0.094	0
jitter	−0.370	**−0.362**	−0.130	−0.105
jitternorm	−0.362	−0.357	−0.127	−0.104
lookback	−0.480	−0.047	**−0.649**	**−1.390**
multideseason	0	0	−0.065	−0.035
nonneg	+0.002	−0.068	−0.075	−0.003
offset	−0.080	−0.016	+0.002	−0.980
rankgauss	−0.317	−0.180	−0.529	0
recursive	+0.015	−0.213	−0.347	−0.207
seasontile	−0.182	−0.082	−0.149	0
warp	−0.397	−0.113	−0.104	−0.270

Table 19: **Cross-model agreement between family fingerprints.** We report Kendall's τ between the fingerprints in Table 18, with permutation p-values. Agreement is well below 1 for every model pair, indicating that useful families are not ranked consistently across models. The two pairs involving TIREX-2 and a model whose profile is dominated by `lookback` or `diff` are not significantly different from zero agreement. This is consistent with a model-specific ranking of view families, with some shared signal from families that are broadly useful.

model A	model B	τ	perm. p
CHRONOS-2	TIREX-2	0.089	0.350
CHRONOS-2	PATCHTST-FM-R1	0.297	0.078
CHRONOS-2	TOTO-2-313M	0.352	0.049
TIREX-2	PATCHTST-FM-R1	0.223	0.148
TIREX-2	TOTO-2-313M	0.012	0.480
PATCHTST-FM-R1	TOTO-2-313M	0.148	0.253

Shape features. The 20 shape features capture temporal dependence, trend, spectral structure, distributional shape, and local variation (Table 20). We compute them from the last 512 points of each context or rollout after z-normalizing the series, making the representation invariant to level and scale. For each model, we standardize the features using the mean and standard deviation of its own rollout samples. We then fit PCA and the k-means mode centroids in this model-specific standardized space. Thus, the coordinates and modes in each panel are defined relative to that model's own rollout distribution.

Directionality. `diff` and `smooth` move contexts in similar directions across examples, with mean cosine 0.75–0.78, compared with 0.04–0.07 for crops and offsets (Table 21).

Interpreting the directionality results. We treat these results as a mechanism diagnostic rather than direct evidence of prior transport. The $\sigma = 0.002$ jitter control still shows non-trivial directionality (≈ 0.50), because even small perturbations systematically change noise-sensitive features such as `diff_std_ratio` and `turning_rate`. We therefore also look at displacement magnitude and mode alignment. Within the same model, `diff` moves contexts 3.0–5.5× farther than jitter, while jitter is among the smallest non-crop movements in feature space and is the smallest displacement in the model embedding space for three of the four models. We also test whether each view moves toward a consistent rollout mode.

Table 20: **Shape features used for contexts and model rollouts.** We compute all features on z-normalized series using at most the last 512 observations. Bounded statistics are clipped and rescaled where needed so that no single coordinate dominates the representation.

group	features	description
Autocorrelation and memory	`acf1`, `acf2`, `acf5`, `acf10`, `acf_first_zero`, `pacf2`	Autocorrelation at lags $1, 2, 5, 10$; first non-positive ACF lag up to 99 (scaled by 100); and lag-2 partial autocorrelation.
Trend and oscillation	`trend_r2`, `turning_rate`, `crossing_rate`	Linear-trend R^2, frequency of sign changes in first differences, and frequency of crossings of the series mean.
Spectrum	`spectral_entropy`, `seasonal_peak_power`, `seasonal_peak_freq`	Normalized spectral entropy, power share of the strongest non-DC Fourier component, and its normalized frequency. The spectrum uses a Hann window.
Distribution shape	`skew`, `kurtosis`, `iqr_over_std`, `mad_over_std`, `tail_frac_2sig`	Skewness, excess kurtosis, normalized IQR and MAD, and the fraction of observations with $\lvert z\rvert > 2$.
Smoothness, jumps, and persistence	`diff_std_ratio`, `step_max`, `runlen_mean`	Relative first-difference variability, largest one-step change, and mean run length above or below the median.

Table 21: **Directionality** in feature space: does a view induce a consistent displacement across contexts? *magnitude* is $\|\bar{d}\|$, *consistency* is $\mathrm{mean}_i \cos(d_i, \bar{d})$ with a 95% bootstrap CI. Read the two columns together; see the caveat below.

	CHRONOS-2			TIREX-2		
view	magnitude	consistency	95% CI	magnitude	consistency	95% CI
`diff`	6.61	0.773	[0.761, 0.785]	13.29	0.781	[0.768, 0.793]
`smooth_ma7`	2.86	0.768	[0.753, 0.781]	7.27	0.752	[0.734, 0.770]
`jitter_0.002`	1.21	0.504	[0.457, 0.548]	4.43	0.456	[0.413, 0.499]
`sdiff`	2.19	0.417	[0.391, 0.442]	5.43	0.490	[0.463, 0.517]
`rankgauss`	1.23	0.341	[0.310, 0.371]	2.22	0.348	[0.319, 0.376]
`winsor_0.05`	0.54	0.337	[0.312, 0.362]	0.93	0.388	[0.364, 0.411]
`deseason`	1.07	0.269	[0.238, 0.299]	2.47	0.293	[0.256, 0.326]
`lookback_0.5`	0.35	0.067	[0.050, 0.084]	0.73	0.065	[0.049, 0.082]
`offset_8`	0.04	0.045	[0.032, 0.058]	0.05	0.040	[0.029, 0.052]
`seasontile`	0.07	0.045	[0.031, 0.059]	0.13	0.048	[0.035, 0.063]

As we have clarified previously, these results should not be read as a literal application of Proposition 1. In particular, `rankgauss` is fitted to the context and `diff` acts across time. Crops and offsets behave differently as well: they induce only small shape-space displacements (magnitude 0.04–0.05 for `offset_8` and 0.35–0.73 for `lookback_0.5`, versus 6.61–13.29 for `diff` (Table 21). This is expected because they change the observation map rather than the shape of the series, and this case is covered separately by Section A.3. Overall, we use the geometry to test whether the views induce structured, model-specific changes, not to establish causality.

Table 22: **Non-redundancy of view displacements.** We compute the cosine similarity between the mean displacement directions induced by different views. Values near 0 indicate that two views move contexts in different directions, while values near ± 1 indicate similar or opposing directions. For each model and representation space, we report the mean and maximum absolute off-diagonal cosine. For CHRONOS-2 in feature space, the maximum is the `diff`–`sdiff` pair (0.81).

	feature space		embedding space	
model	mean |off-diag|	max	mean |off-diag|	max
TIREX-2	0.42	0.90	0.54	0.85
CHRONOS-2	0.36	0.81	0.55	0.97
PATCHTST-FM-R1	0.48	0.96	0.44	0.97
TOTO-2-313M	0.40	0.84	0.33	1.00

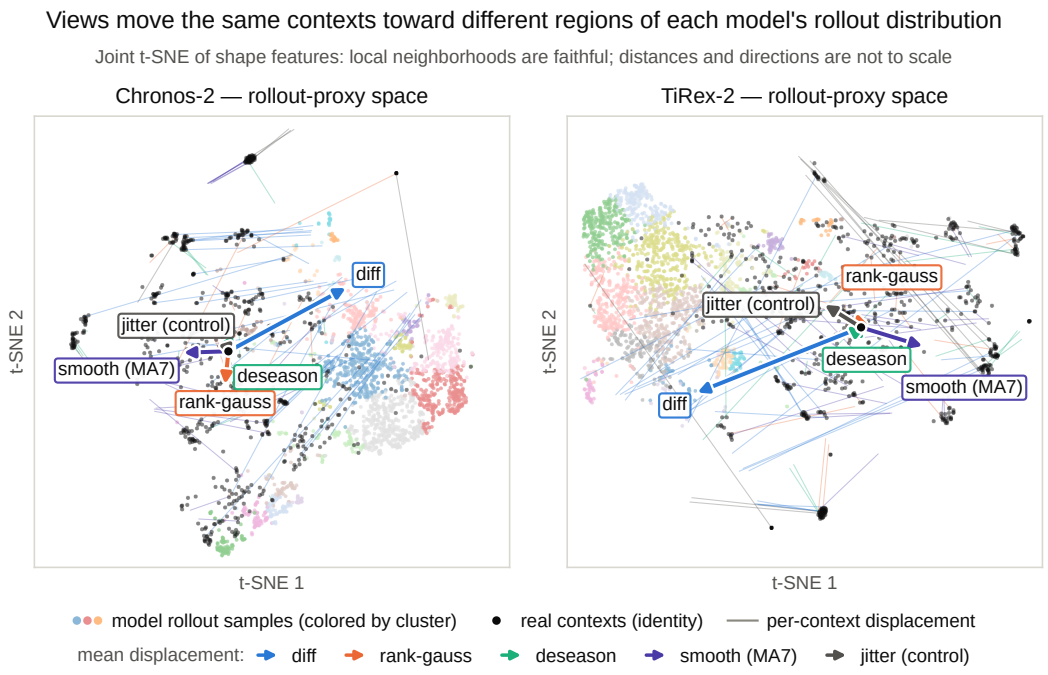


Figure 9: **Views move the same contexts toward different regions of each model's rollout distribution.** We use ancestral rollouts from each model's quantile head as a proxy for its learned forecasting behavior and visualize the resulting 20-dimensional shape-feature space with joint t-SNE. Colored points are rollout samples, grouped by k-means mode; black points are real GIFT-Eval contexts; arrows show the mean displacement induced by each view. We use the same contexts and views for both models, but `diff` moves toward different rollout modes: for CHRONOS-2, a low-`iqr_over_std`, high-`step_max` mode with mass 0.07; for TIREX-2, a high-`kurtosis`, high-`skew` mode with mass 0.02. This provides the geometric counterpart to the fingerprint result. t-SNE preserves local neighborhoods, but distances should not be interpreted metrically.

Non-redundancy. Displacement directions from different views have mean absolute cosine 0.36–0.48 in feature space, indicating that the views make distinct changes (Table 22).

The extreme pairs are also consistent with the intended semantics of the views. For CHRONOS-2, the two differencing families are strongly aligned, $\cos(\texttt{diff}, \texttt{sdiff}) = +0.81$, while the intentionally opposing `deseason` and `smooth` views have cosine -0.62. In contrast, `jitter` and `lookback`, which do not share a common mechanism, have only weak alignment, with cosine $+0.12$.

Table 23: **Destination of the `diff` displacement.** For each model, we report the rollout mode most aligned with the mean `diff` displacement, together with the cosine between that displacement and the direction from the real-context centroid to the mode centroid. The alignment is strong for all four models, but the target mode differs across models, as also shown in Figure 9. Mode names are based on the cluster's most distinctive features, and "mass" denotes the fraction of rollout samples assigned to that mode.

model	top mode for `diff`	cosine	mass
TIREX-2	high `kurtosis` / high `skew`	+0.97	0.02
CHRONOS-2	low `iqr_over_std` / high `step_max`	+0.97	0.07
PATCHTST-FM-R1	high `tail_frac_2sig` / high `seasonal_peak_freq`	+0.96	0.05
TOTO-2-313M	low `iqr_over_std` / low `mad_over_std`	+0.92	0.08

Mode movement and poorly aligned views. We also measure how strongly each view changes mode assignments, independent of its displacement direction. For each view, we compute the Jensen–Shannon divergence between its distribution over rollout modes and the corresponding identity distribution. On CHRONOS-2, with jitter at 0.019, we obtain `diff` 0.269, `sdiff` 0.106, `rankgauss` 0.054, `smooth_ma7` 0.050, `deseason` 0.042, `winsor` 0.022, `lookback` 0.019, and 0.001 for both `offset` and `seasontile`. TIREX-2 shows the same ordering at the top: `diff` 0.402, `sdiff` 0.157, and `smooth_ma7` 0.144, with crop-based views again near the bottom. `smooth_ma7` is a useful counterexample. It moves contexts by an amount comparable to several useful views, but Figure 10c shows little alignment with any rollout mode. Its largest mode cosine is only $+0.03$ over 20 modes for CHRONOS-2 and $+0.13$ for TIREX-2, with the remaining modes negatively aligned. Thus, large movement alone is not sufficient: `smooth_ma7` changes the representation substantially without moving toward the rollout modes targeted by the useful views.

E.3 FORECAST SIGNATURES (SECTION 6)

We characterize the forecast-level effect of each view family using three statistics: the log-ratio of predictive interval width, the median shift normalized by the label scale of each window, and the change in coverage. We compute these quantities over the test windows where the view is applied, first per family and config, and then aggregate using the median across windows and configs.

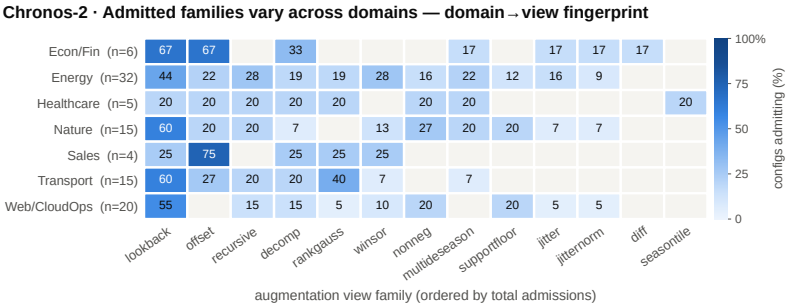
Table 24: **Forecast signatures by view family** for CHRONOS-2 and TIREX-2 (15 configs each). *width* is the log-ratio of predictive interval width relative to the base, *sign* is the fraction of configs with the reported sign, *shift* is the scale-normalized median shift, and *cov* is the change in coverage. Families that materially change the forecast distribution do so consistently, while the remaining families have much smaller effects.

	CHRONOS-2				TIREX-2			
family	width	sign	shift	cov	width	sign	shift	cov
`diff`	+2.249	0.93	0.555	+0.185	+2.103	1.00	0.516	+0.144
`sdiff`	+0.398	0.87	0.578	+0.034	+0.375	0.93	0.507	+0.015
`recursive`	−0.461	0.93	0.068	−0.178	−0.548	1.00	0.050	−0.202
`fft`	−1.328	0.93	0.868	−0.676	−1.541	0.93	0.753	−0.714
`rankgauss`	+0.004	0.60	0.144	−0.029	−0.002	0.53	0.125	−0.008
`jitter`	0.000	0.80	0.004	+0.001	0.000	0.80	0.004	0.000
`lookback`	0.000	—	0.036	+0.003	−0.002	0.60	0.045	0.000
`offset`	0.000	—	0.002	0.000	0.000	—	0.000	0.000

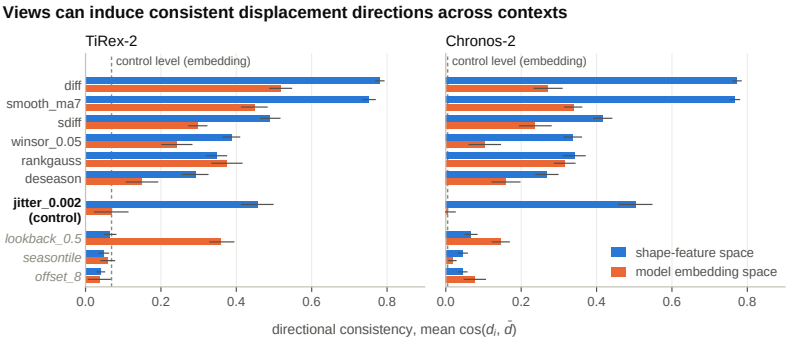
`fft` consistently narrows predictive intervals by -1.33 to -1.54, while `lookback`, `offset`, and `jitter` have little effect on interval width (Table 24). Part of `diff`'s interval widening comes from the cumulative-sum pull-back, which sums marginal quantiles across the horizon (Section A.3). Thus, the signature reflects the full view, not just the input transform.

We summarize the signatures in two ways. First, families are clearly distinct within each model. The between/within-family variance ratios for (width, shift, coverage) are $[32.0, 10.4, 71.6]$ for CHRONOS-2, $[25.9, 9.6, 76.0]$ for TIREX-2, $[34.2, 10.0, 56.7]$ for PATCHTST-FM-R1, and $[35.5, 9.9, 86.9]$ for TOTO-2-313M. Thus, across all four models and all three statistics, between-family variation is roughly 10–87$\times$ larger than within-family variation.

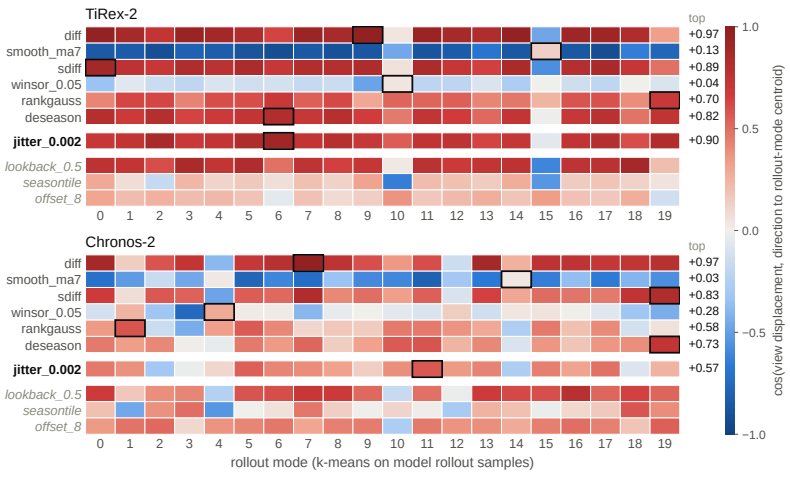
Second, the non-degenerate signatures are stable across models. Comparing CHRONOS-2 and TIREX-2, the per-family signature cosines range from $+0.95$ to $+1.00$ for the ten families with nonzero signatures (`diff` $+1.00$, `sdiff` $+1.00$, `recursive` $+1.00$, `fft` $+0.99$, `jitternorm` $+0.97$). The remaining four families: `nonneg`, `offset`, `seasontile`, and `supportfloor`, have a zero median signature on TIREX-2. They leave most windows unchanged, although they alter enough windows to be admitted occasionally (Table 18). Thus, their cosines are undefined rather than small. Since this cosine is computed from only three statistics, we use it as a consistency check rather than as strong evidence on its own. We also report the fraction of dropped windows due to non-finite fp16 values or zero label scale. The worst case is 0.47 for `diff` on CHRONOS-2; the median is 0.0, and $9/15$ configs drop no windows.



(a) domain → admitted family



(b) views induce consistent displacements



(c) view → rollout mode, per model

Figure 10: **Summary of the mechanism analyses in Section 6. (a)** Admitted families vary across domains. **(b)** Individual views induce consistent displacement directions. **(c)** These displacements can align with different modes of each model's rollout distribution. Figure 9 visualizes the same effect in a rollout-based proxy for each model's learned forecasting behavior.

F NEGATIVE RESULTS

We report several negative results that influenced the final design of BGAA.

N1. Per-series test-time training does not help on average. We first tried updating the model separately for each series at test time. This changes aggregate performance by only -0.10%, which is not statistically significant. A leak-free per-series gate reduces the worst regression from $+8.87\%$ to $+1.99\%$, but still does not improve the mean. We therefore keep the base model frozen.

N2. Several plausible views are neutral or harmful. Log-difference, reflected log-difference, moving-average smoothing, despiking, and power transforms do not improve performance in our experiments. Smoothing is a useful example: it moves the context substantially, but does not align with a rollout mode for either CHRONOS-2 or TIREX-2 (Section 6). This suggests that a large change to the input is not enough by itself; the transformed context must also induce a useful response from the frozen model.

N3. Continued pretraining on narrow data slices degrades performance. We also continued pretraining CHRONOS-2 on distributions close to the evaluation data using several objectives: MAE on the median, pinball+MSE, center-weighted pinball, and expectile loss. All of these variants performed worse than the frozen base on the configurations we tested. The only improvement came from a domain-specific perturbation schedule that used an oracle domain label, which would not be available at deployment.

G REPRODUCIBILITY

Unified evaluation harness. All results in this paper are produced by a single evaluation driver parameterized by model, benchmark, and policy. We verified this implementation against the four research codebases that preceded it and obtained a maximum absolute difference of 0.0 on all shared policy rows (25 rows on a smoke configuration and 367 rows over the 15-config subset). We repeated this check after the multi-model refactor. Adding BOOM and fev-bench also leaves the GIFT-Eval evaluation unchanged, with identical results for all $97/97$ configs.

Submitted code archive. We include an anonymized code archive with the submission. It contains the core BGAA implementation together with scripts for setting up the environment, loading the evaluated models and benchmarks, and reproducing the main experiments. The archive is intended to make the main results reproducible during review without requiring the full internal research codebase.

Release artifacts. After the double-blind review period, we will release the full evaluation harness, including the thinking-mode ladder; the complete view catalogue and pool builder; per-config CSVs underlying all 12 model–benchmark cells and appendix tables and the figure-generation scripts. We will also share the BOOM leaderboard replication code, including the check against DataDog's published table; and the mechanism experiments from Section 6 together with the compute-matched controls in Section C.2.2. All four model checkpoints are public, so the experiments can be reproduced from the released code and public weights. We use no proprietary datasets or checkpoints.