AUTO-CHROMA: AUTOMATING THE DESIGN OF TEST-TIME TIME-SERIES MODEL PORTFOLIOS WITH MEMORY-AUGMENTED LLM AGENTS.

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series foundation models (TSFMs) deliver accurate zero-shot forecasts across domains, but their accuracy is tied to model sizes that make inference costly at scale. Chroma offers an alternative paradigm: a portfolio of small specialists, each post-trained from a small TSFM on a subset of the post-training data, matches models orders of magnitude larger with only a few specialists active during inference. However, the portfolio is designed heuristically: the data is partitioned once by sampling frequency or domain, although any subset of it could define a specialist. We present **Auto-Chroma**, an agentic framework that casts portfolio design as a search over specialist configurations and runs the search end to end with memory-augmented large language model (LLM) agents. Rather than guessing from a fixed prompt, the agents reason over a memory of every trial, diagnosing each evaluated portfolio and proposing the next as a hypothesis grounded in the full history of the search. On `CB-II` and `fev-bench-mini`, with three univariate and multivariate generalists from 4M to 28M parameters, the portfolios found outperform the frequency heuristic portfolio on test in five of the six settings and achieve competitive performance with respect to large TSFM baselines, and under the same budget Auto-Chroma leads greedy and random search throughout, surpassing the final performance of random search within a fraction of the budget while post-training far fewer specialists. Beyond performance, we analyze both the portfolios the search produces and the search process itself: how fragile the heuristic partitions are, whether the portfolios transfer across benchmarks, whether the validation objective faithfully reflects test performance, how similar the portfolios are across repeated runs and orchestrator LLMs, and how the agents use the memory of past rounds. To our knowledge, this is the first work to investigate portfolio construction as an autonomous process, and the first to treat the search and the portfolios it finds as a subject of study in their own right.

1 INTRODUCTION

Pretrained time series models have become a practical tool for forecasting at scale: a single model produces zero-shot forecasts for thousands of heterogeneous series, as in business and IT observability (Palaskar et al., 2024; Woo et al., 2023). The pretraining paradigm also extends beyond forecasting, with dedicated pretrained models for anomaly detection (Shentu et al., 2025), compatibility with multimodal retrieval (Chen et al., 2025), and empirical evidence of task transferability (Auer et al., 2025a). This breadth across domains and tasks has motivated scaling-law studies that report zero-shot accuracy improving with parameters, data, and compute (Yao et al., 2025; Edwards et al., 2024; Shi et al., 2024). The field has largely answered this evidence with a rapid succession of time series foundation models (TSFMs): one general model for all series (Ansari et al., 2024; Das et al., 2024; Woo et al., 2024; Auer et al., 2025b; Ansari et al., 2025; Khwaja et al., 2026).

However, a growing body of evidence indicates that TSFMs use only part of its capacity. Leading TSFMs leave up to half of their parameters inactive on a downstream task, and the active subnetwork varies across datasets (Zhao et al., 2025); a further share of heads and layers is redundant regardless of dataset (Bao et al., 2026); and the low-rank structure of time series embeddings makes attention

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.

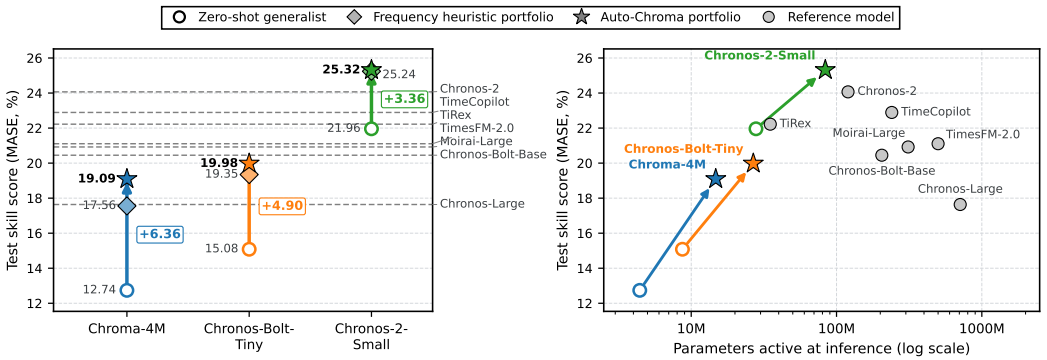

Figure 1: Auto-Chroma on CB-II. Colors mark the forecasting model: Chroma-4M, Chronos-Bolt-Tiny, and Chronos-2-Small. Left: Test skill score (MASE, %, vs. Seasonal Naive) per model: zero-shot generalist, frequency heuristic portfolio, and Auto-Chroma portfolio; boxed numbers give the gain over zero-shot, and dashed lines mark reference models. Right: Test skill score vs. parameters active at inference (log scale).

layers compressible while maintaining full accuracy at a fraction of the memory (Yu et al., 2026). A TSFM therefore pays, at every forecast, for capacity that the series at hand does not use.

Chroma (Kayaalp et al., 2026) turns this observation into a design principle: if a task exercises only part of a large model, then it should be served by several small models, and only by those it needs. Instead of one large *generalist* TSFM, Chroma builds a *portfolio* of small TSFMs called *specialists*, each post-trained from a smaller generalist on a subset of the post-training corpus. During inference, greedy ensemble selection on validation windows activates only a few specialists per task, so the portfolio matches or surpasses TSFMs that are orders of magnitude larger at a fraction of the cost. This raises a natural question: **how should the portfolio be designed?** Kayaalp et al. (2026) partition the corpus by a dataset-level heuristic, such as domain or sampling frequency. This design has two limitations. First, the heuristic restricts the design space: one specialist per frequency or domain fixes both the number of specialists and the grouping of datasets, so no specialist can combine datasets across groups. Second, it never evaluates the contribution of a dataset, so a dataset that matches the rule can still degrade its specialist, and adding data to the corpus does not necessarily improve the portfolio. A procedure that designs the portfolio from evaluation feedback is therefore desired, yet the space it must search is combinatorial, since any subset of the corpus can define a specialist and any set of specialists a portfolio. LLM-free search strategies such as random or greedy search are ill-suited to this space: they use at most the score of each candidate portfolio, so they either explore blindly or refine only locally.

In this work, we present **Auto-Chroma**, which addresses these limitations by treating the partition of the post-training corpus as a search problem and handing the search to large language model (LLM) agents. LLMs can read the per-task results and training logs of each candidate and reason about which datasets a specialist needs. Figure 2 illustrates the process: in each round, an *analyst* agent diagnoses the latest portfolio against past ones, a *strategist* agent proposes the next portfolio with its rationale and per-task predictions. A persistent database keeps every portfolio, score, and report, so each proposal is a hypothesis checked against all past outcomes rather than a guess from a fixed prompt. Auto-Chroma is compatible with univariate and multivariate data and, in principle, with any generalist and any LLM. More specifically, we make the following contributions.

Memory-augmented agentic portfolio search. We build a memory-augmented agentic system that searches over specialist configurations end-to-end, with validation performance as the objective; to our knowledge, this is the first work to cast the construction of a model portfolio as an agentic search. Without any manual design, the portfolios found outperform the frequency heuristic portfolio on the test split in five of the six settings and achieve competitive performance with respect to large TSFM baselines (see Figure 1). Under an identical budget, Auto-Chroma also leads greedy and random

search from the first rounds, surpassing the final performance of random search within a quarter of the budget while post-training $5\times$ fewer specialists (see Figure 4).

Insights on portfolios, evaluation, and agents. We study both the portfolios the search produces and the trajectory that leads to them, and draw three conclusions, none of which is specific to Chroma. For portfolio design, we examine how fragile the heuristic partitions are and whether the portfolios found transfer across benchmarks. For evaluation, a validation–test scatter of every candidate portfolio evaluated during the search exposes a pitfall in the standard holdout protocol that any holdout-based model selection inherits. For LLM agents, we compare the portfolios found within and across orchestrator LLMs and trace how the agents use the memory of past rounds, findings that apply to any LLM agent that searches a configuration space of specialists.

2 Background and Related Work

Problem formulation. Formally, let $\boldsymbol{Y}_{1:T} \in \mathbb{R}^{T\times D}$ denote a time series with context length T and D target variates. A forecaster f_θ predicts the future $\boldsymbol{Y}_{T+1:T+H}$ over a horizon H and returns a forecast $\hat{\boldsymbol{Y}}_{T+1:T+H}$, given as a set of quantiles with the median as point forecast. A task τ fixes a dataset, a horizon H_τ, a set of series, and a metric m, and a benchmark $\mathcal{B}$ is a finite set of tasks. Each task holds out the last H_τ observations of every series as the test split. Following Chroma (Kayaalp et al., 2026) and `AutoGluon` (Shchur et al., 2023), we use the H_τ observations before the test split as the validation split, which is the standard temporal holdout of time series modeling (Hyndman & Athanasopoulos, 2021). We write $\mathcal{B}^{\mathrm{val}}$ and $\mathcal{B}^{\mathrm{test}}$ for the benchmark under the two splits. Following `fev-bench` (Shchur et al., 2025), $e_m(f,\tau)$ is the error of f on τ under m, and $S_m\big(f \,\big|\, b, \mathcal{B}\big)$ is its skill score over $\mathcal{B}$ against a reference b, in percent, higher is better.

Time series foundation models. TSFMs are trained once on a corpus of many datasets and forecast unseen series zero-shot. Representative families include Chronos (Ansari et al., 2024; 2025), TimesFM (Das et al., 2024), Moirai (Woo et al., 2024; Liu et al., 2025a), MOMENT (Goswami et al., 2024), TiRex (Auer et al., 2025b), Sundial (Liu et al., 2025b), and TOTO (Cohen et al., 2025; Khwaja et al., 2026). However, in many deployments, inference time and cost take higher priority, so a paradigm built on small models is desired. Tiny Time Mixers (Ekambaram et al., 2024) pretrain a mixer of about 1M parameters with adaptive patching so that one small model serves many resolutions. Chroma (Kayaalp et al., 2026) goes one step further: it replaces one large model with a portfolio of small specialists, of which only a few are active per task, so it is at once a family of small TSFMs and an ensemble of them.

Mixtures and ensembles of forecasters. The ensemble side of Chroma follows a long line of forecast combination (Wang et al., 2023). Two recent designs bring it to pretrained forecasters. Time-MoE (Shi et al., 2025) builds the mixture into one network, where a learned router activates a few experts per token. Synapse (Das et al., 2026b) combines independently trained TSFMs at inference, with weights set by their recent errors. Auto-Chroma does not weight a fixed set of experts but searches over the configurations of the specialists, so the portfolio can reach beyond the capacity of any fixed set of forecasters.

Chroma. A pretrained generalist f_θ is post-trained into specialists, each on a slice of a post-training corpus $\mathcal{D}$. Specialist k is defined by a configuration $c_k = (\mathcal{D}_k, \eta_k, s_k)$ with dataset subset $\mathcal{D}_k \subseteq \mathcal{D}$, learning rate η_k, and step count s_k, and its parameters are $\theta_k = \mathrm{FT}(\theta; c_k)$, where FT runs s_k post-training steps on $\mathcal{D}_k$ at rate η_k from θ. A portfolio $P = \{c_1, \ldots, c_K\}$ is a set of K specialist configurations, and the generalist is always member 0 with $\theta_0 = \theta$. Chroma partitions the corpus by a dataset-level heuristic, one specialist per sampling frequency ($P^{\mathrm{h}}_{\mathrm{freq}}$) or per domain ($P^{\mathrm{h}}_{\mathrm{dom}}$). On each task τ, greedy ensemble selection (Caruana et al., 2004) fits weights $\boldsymbol{w}^\tau$ on the validation split. Only members with $w^\tau_k > 0$ run at inference, and the rows $\boldsymbol{w}^\tau$ over $\tau \in \mathcal{B}$ form the routing matrix $\boldsymbol{W} \in \mathbb{R}^{|\mathcal{B}|\times(K+1)}$. Further details on ensemble and test-time forecast are given in Appendix B.

Agentic time series forecasting. LLMs enter forecasting in two roles. As forecasters, the LLM produces the numbers itself, either directly from the series (Gruver et al., 2023; Jin et al., 2024) or through a team of agents that reason over textual context, as in Nexus (Das et al., 2026a). As routers, the LLM acts at inference over fixed pretrained forecasters: TimeCopilot (Garza & Rosillo, 2025) selects and ensembles foundation models, and MoiraiAgent (Yan et al., 2026) picks one TSFM per query and adjusts its forecast from textual context. Auto-Chroma acts at training time: the agent

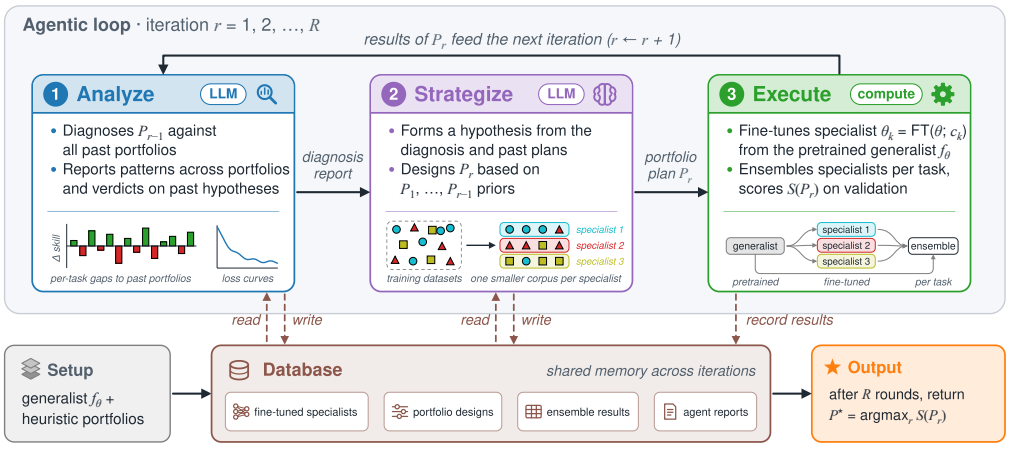

Figure 2: The Auto-Chroma iteration. **Setup**: the two heuristic portfolios are post-trained from the frozen generalist f_θ, scored, and written to the database. **Round** r: the analyst call π_{A} diagnoses the latest portfolio P_{r-1} against the past portfolios; the strategist call π_{S} forms a hypothesis from the diagnosis and the history and designs P_r; the execute step post-trains each specialist $\theta_k = \mathrm{FT}(\theta;\, c_k)$ from f_θ, ensembles the members per task, and scores $S(P_r)$. **Database**: an immutable memory of specialists, portfolio designs, ensemble results, and agent reports, read by both agents and written by every phase. **Output**: after $r_{\max}$ rounds the run returns $P^\star = \arg\max_r S(P_r)$.

decides which specialists to post-train and on which slice of the corpus. Similar propose-evaluate-revise loops drive architecture and algorithm search in other fields (Chen et al., 2023; Novikov et al., 2025); to our knowledge, we are the first to study this design for time series forecasting, where the search object is the partition of the training corpus rather than the model.

3 Auto-Chroma

In this section, we introduce Auto-Chroma, an autonomous agentic framework that constructs a portfolio of specialists. We formalize the search problem in Section 3.1, and describe the memory of the search (Section 3.2) and the search loop with its two agents (Section 3.3).

3.1 Problem Statement

Section 2 defines a specialist by its configuration $c_k = (\mathcal{D}_k, \eta_k, s_k)$ and a portfolio P by its set of configurations. Auto-Chroma treats P as the decision variable. Formally, the portfolio space is

$$\mathcal{P} = \Big\{ P = \{c_1, \ldots, c_K\} \;:\; c_k = (\mathcal{D}_k, \eta_k, s_k),\ \emptyset \neq \mathcal{D}_k \subseteq \mathcal{D},\ 1 \leq K \leq K_{\max} \Big\}, \tag{1}$$

where we empirically cap the portfolio size to $K_{\max}$ to limit the search space, and the generalist f_θ is member 0 of every portfolio. Exhaustive search over $\mathcal{P}$ is infeasible, since a single specialist already ranges over $2^{|\mathcal{D}|}$ dataset subsets before η_k and s_k are chosen, and evaluating one candidate post-trains up to $K_{\max}$ specialists. An optimization procedure is therefore necessary. Thus, we propose to optimize the portfolio via its validation skill score. We formalize the optimization process as

$$P^\star = \arg\max_{1 \leq r \leq r_{\max}} S(P_r), \qquad S(P) = \sum_{m \in \mathcal{M}} S_m\big(P \,\big|\, P_{\mathrm{ref}}, \mathcal{B}^{\mathrm{val}}\big), \tag{2}$$

where a run proposes $r_{\max}$ candidate portfolios and returns the best, and S sums the skill score of `fev-bench` (Shchur et al., 2025) over the leaderboard metrics $\mathcal{M}$. Under metric m, the skill score measures the average relative error reduction of P over a fixed reference portfolio P_{ref} on the validation tasks $\mathcal{B}^{\mathrm{val}}$. Here, P is scored through the ensemble it induces: on each validation task

τ, greedy ensemble selection fits $\boldsymbol{w}^\tau$ over the $K+1$ members, and the error of P on τ is that of the weighted forecast $\sum_{k=0}^{K} w_k^\tau \hat{\boldsymbol{Y}}_{k,\tau}$, where $\hat{\boldsymbol{Y}}_{k,\tau}$ is the forecast of member k on τ. Skill scores normalize all metrics to the same scale, so their sum over $\mathcal{M}$ is a valid multi-metric objective.

3.2 MEMORY

An LLM that proposes portfolios from a fixed prompt has no access to past outcomes: it may retry a design that already failed, and it cannot check whether its expectations for a design held. Auto-Chroma therefore records every artifact of the search in an append-only database. Each candidate portfolio P_i contributes five artifacts. The first is the portfolio itself, given by its specialist configurations. The second is its evaluation, namely the score $S(P_i)$ and the routing matrix $\boldsymbol{W}_i$. The third is the report R_i^{S} of the strategist π_{S} that proposed P_i, which states the design rationale and the per-task predictions. The fourth is the report R_i^{A} of the analyst π_{A} that evaluated P_i in the following round, which states the diagnosis. The fifth is the configuration and training log of every specialist that P_i introduced. When the r-th candidate is proposed, the database holds

$$\mathcal{H}_r = \left\{(P_i, S(P_i), \boldsymbol{W}_i, R_i^{\mathrm{S}})\right\}_{i<r} \cup \left\{R_i^{\mathrm{A}}\right\}_{i<r-1}, \tag{3}$$

where R_{r-1}^{A} is written during round r. This design turns each proposal into a testable hypothesis rather than an unconditioned guess, where the predictions recorded in R_i^{S} are evaluated against the outcome of P_i in R_i^{A}. Because both reports persist, every later proposal is conditioned on the verified outcomes of earlier designs, and a failed design is marked with its cause instead of being retried.

3.3 SEARCH LOOP

Figure 2 illustrates the general mechanism of Auto-Chroma. A run is initialized by post-training and scoring the two heuristic portfolios $P_{\mathrm{freq}}^{\mathrm{h}}$ and $P_{\mathrm{dom}}^{\mathrm{h}}$. Each round r then consists of three steps. The analyst π_{A} evaluates the latest portfolio P_{r-1}: it compares it per task with the strongest competing portfolio, inspects the training logs of its specialists, extracts win and loss patterns across the leaderboard, and verifies hypotheses made by the strategist in the previous round. The strategist π_{S} reads this diagnosis and the analyst reports of the strongest past portfolios, and proposes a candidate P_r together with a rationale and per-task predictions. The orchestrator post-trains the specialists of P_r from f_θ, fits the ensemble weights on the validation split, and scores $S(P_r)$. Formally, with the state $\mathcal{H}_r$ of Eq. 3, one round is

$$R_{r-1}^{\mathrm{A}} = \pi_{\mathrm{A}}(\mathcal{H}_r), \qquad (P_r, R_r^{\mathrm{S}}) = \pi_{\mathrm{S}}\big(R_{r-1}^{\mathrm{A}}, \mathcal{H}_r\big), \tag{4}$$

Two properties of the loop are central to the design. First, the LLM enters only through π_{A} and π_{S}; post-training, ensemble selection, and scoring are deterministic procedures that the LLM neither implements nor influences. Second, post-trained specialists are cached by their configuration c_k. A specialist proposed in an earlier round is therefore loaded from the cache, so the cost of evaluating P_r is proportional to the number of new specialists, not to its size K. The number of unique specialists trained is thus a natural measure of search cost, and under this measure Auto-Chroma explores more broadly than greedy search while training far fewer new specialists than random search (see Figure 4). The loop terminates after $r_{\max}$ rounds and returns the best accepted candidate as $P^\star$.

The objective of Eq. 2, the database of Eq. 3, and the loop of Eq. 4 together define a single optimization for portfolio construction. Auto-Chroma thereby replaces the manually designed partition of the post-training corpus with a search over partitions, in which the LLM selects the specialists to post-train, deterministic evaluation scores the resulting portfolio, and the database conditions each selection on the outcomes of all previous ones.

4 EXPERIMENTS

In this section, we evaluate Auto-Chroma through five research questions (RQs): RQ1 and RQ2 assess it as an optimizer of its validation objective, and RQ3 to RQ5 study the portfolios it finds. Since our contribution is the search, RQ1 to RQ3 report validation performance, the objective that the search optimizes (Eq. 2). RQ4 and RQ5 then study how validation performance carries over to test performance. Appendix E further reports the stability of the search across repeated runs, the use

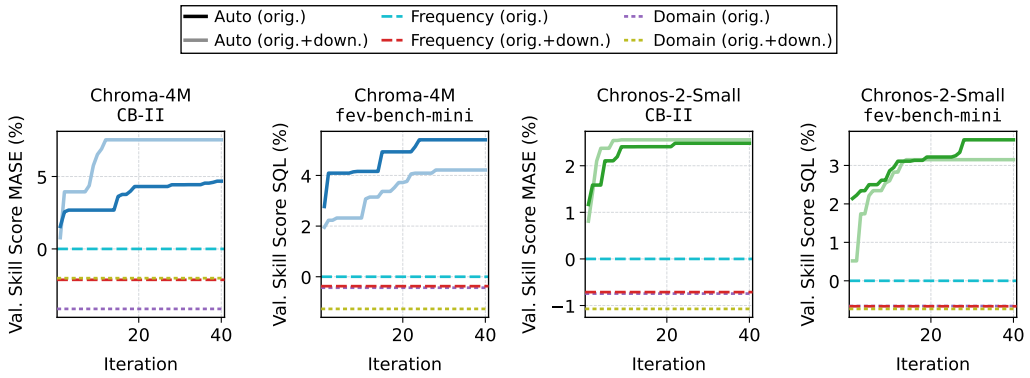

Figure 3: Best validation performance found up to each round, as the skill score against $P^{\mathrm{h}}_{\mathrm{freq}}$ on the original corpus. Solid lines: Auto-Chroma with Chroma-4M or Chronos-2-Small on the original corpus, lighter (Chroma-4M, Chronos-2-Small) on the original + downsampled corpus. Dashed lines: the frequency and domain heuristic portfolios on the original corpus, and the frequency and domain ones on the original + downsampled corpus.

of the memory by the agents, an ablation of the analyst, the transfer of portfolios across benchmarks, and the computational cost.

Setup. We consider three generalists: Chroma-4M (Kayaalp et al., 2026) (4M parameters), Chronos-Bolt-Tiny (Ansari et al., 2024) (9M), and the multivariate Chronos-2-Small (Ansari et al., 2025) (28M). Specialists are post-trained on one of two corpora: the *original* corpus $\mathcal{D}$ of 44 datasets (see Appendix F), and the *original* + *downsampled* corpus, which adds 168 downsampled datasets derived from 36 of them by mean aggregation to coarser frequencies. Each run starts from an empty database and proposes $r_{\max} = 40$ portfolios of at most $K_{\max} = 10$ specialists, with $P_{\mathrm{ref}} = P^{\mathrm{h}}_{\mathrm{freq}}$ built on the original corpus and DeepSeek V4 Flash Preview (DeepSeek-AI, 2026) as the orchestrator LLM unless stated otherwise. We evaluate on Chronos Benchmark II (`CB-II`) (Ansari et al., 2024) and `fev-bench-mini` (Shchur et al., 2025). Performance is measured by the skill score (Shchur et al., 2025), computed against $P^{\mathrm{h}}_{\mathrm{freq}}$ on the validation split and against Seasonal Naive (Hyndman & Athanasopoulos, 2021) on the test split. Each run returns the portfolio that maximizes Eq. 2, the sum of the validation skill scores over metrics. Further details are given in Appendix C.1.

4.1 RQ1: Does Auto-Chroma improve on heuristic portfolios?

Heuristic portfolios have an inherent pitfall: they partition the corpus by a fixed prior, such as frequency or domain, without accounting for the contribution of each post-training dataset, so a dataset that is consistent with the prior of a specialist can still degrade it. To examine this pitfall, we add downsampled data to the original corpus, which both priors assign without ambiguity, since every downsampled dataset has a frequency and inherits the domain of its source. Figure 3 shows that the additional data degrades the validation performance of the frequency heuristic in all four panels, by 0.4 to 2.1 points of skill score, and that of the domain heuristic in three of four. More data thus does not reliably help a fixed partition rule, which confirms the pitfall for the portfolio design of Chroma (Kayaalp et al., 2026). In contrast, Auto-Chroma avoids this pitfall by design: it assigns datasets to specialists from validation feedback rather than from a fixed prior, so it can exclude or reassign a dataset that degrades performance. Accordingly, it improves with the number of rounds on either corpus and ends above all four heuristic portfolios in every panel. Appendix D.1 extends this comparison to both metrics of each benchmark, to the test split, and to Chronos-Bolt-Tiny.

4.2 RQ2: Is Auto-Chroma a better search strategy?

We compare Auto-Chroma with two LLM-free strategies. Random search samples a new portfolio in every round, and greedy search applies one random mutation to its current leader, the best portfolio

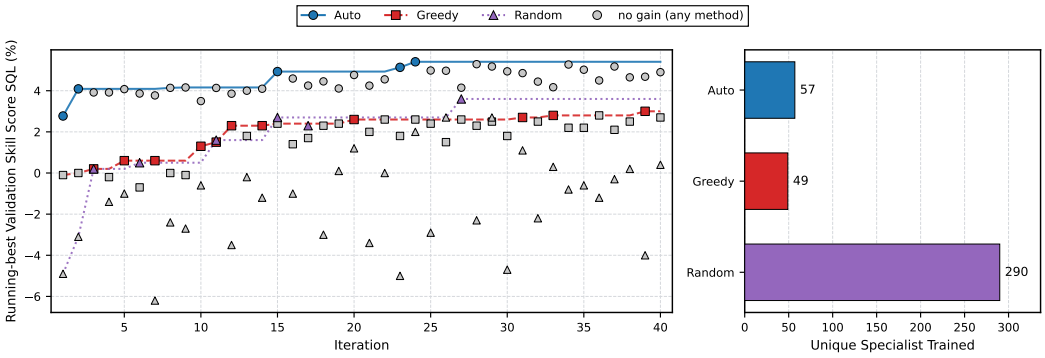

Figure 4: Auto-Chroma against greedy and random search for Chroma-4M on `fev-bench-mini`. Left: best validation skill score found up to each round (SQL, vs. $P^{\mathrm{h}}_{\mathrm{freq}}$); colored markers are running-best portfolios, and gray markers are portfolios with no gain. Right: search cost as the number of unique specialist trained.

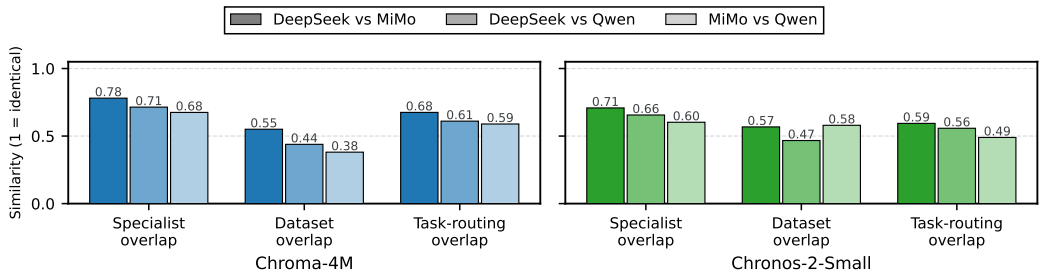

Figure 5: Pairwise similarity of the best portfolios of three orchestrator LLMs on `fev-bench-mini`, for Chroma-4M (left) and Chronos-2-Small (right), in specialist ($\Omega_{\mathrm{specialist}}$), dataset ($\Omega_{\mathrm{dataset}}$), and task-routing ($\Omega_{\mathrm{task}}$) overlap.

found so far, and keeps the mutant only if it improves. Figure 4 shows the result for Chroma-4M. We observe that the two LLM-free strategies fail in opposite ways. Random search explores without memory: it post-trains 290 unique specialists, yet the skill scores of its candidates spread from about -6% to 4%, and most candidates fall below $P^{\mathrm{h}}_{\mathrm{freq}}$. In contrast, the candidates of greedy search stay in a narrow band, and it post-trains only 49 specialists. However, its moves are conservative, since every candidate changes one specialist of a single past portfolio, and the search settles on a local optimum that it does not leave by round 40. Auto-Chroma balances the two. The strategist favors portfolios that differ structurally, which provides the exploration of random search, and it conditions every proposal on the verified outcomes of all past portfolios, which provides the refinement of greedy search. Consequently, Auto-Chroma post-trains a moderately small number of specialists, while its candidates stay in a band as narrow as that of greedy search and all improve on $P^{\mathrm{h}}_{\mathrm{freq}}$, and it leads both strategies from the first candidate to the last. Further details on the LLM-free strategies and an additional experiment with Chronos-2-Small are given in Appendix D.2.

4.3 RQ3: Do different orchestrator LLMs find similar portfolios?

We use the portfolio that a run returns as a proxy for the behavior of its orchestrator LLM, and we measure how similar the portfolios found by different LLMs are. On `fev-bench-mini`, for Chroma-4M and Chronos-2-Small, we replace the default orchestrator LLM by MiMo v2.5 (Xiaomi MiMo Team et al., 2026) or by a locally served Qwen3.5-27B-FP8 (Qwen Team, 2026) under an otherwise identical configuration, and we compare the best portfolios of each pair of LLMs on three axes. *Specialist overlap* asks whether two portfolios group the datasets into the same specialists,

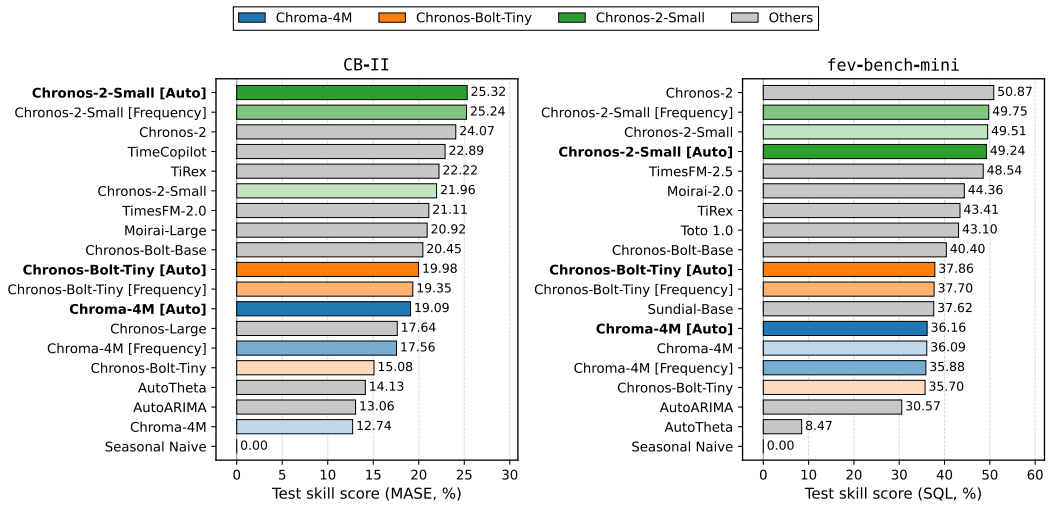

Figure 6: Test leaderboards on `CB-II` and `fev-bench-mini` (higher is better). For each model, [Auto] is the Auto-Chroma portfolio, [Frequency] is the frequency heuristic portfolio, and the bar without a tag is the zero-shot generalist.

weighted by the ensemble weights. *Dataset overlap* asks whether they place their ensemble weight on the same datasets, regardless of the grouping. *Task-routing overlap* asks whether they deploy the same weighted specialists on each task, rather than only on average over the benchmark. Their formal definitions are given in Appendix C.2. We observe that every LLM ends above both heuristic portfolios in validation, yet through different portfolios: the similarity of two LLMs is only 0.38 to 0.78 on the three axes (see Figure 5). Notably, the similarity follows the size of the LLMs. DeepSeek V4 Flash and MiMo v2.5, which both have about 10x more total parameters than Qwen3.5-27B, form the most similar pair in five of the six comparisons, whereas MiMo v2.5 and Qwen3.5-27B form the least similar pair in five of the six. This suggests that the portfolio a run returns reflects properties of its orchestrator LLM, such as its size, so that portfolio similarity offers a proxy for comparing orchestrator LLMs. Details of this experiment are given in Appendix D.3.

4.4 RQ4: Do Auto-Chroma portfolios improve zero-shot performance?

We now compare the performance of Auto-Chroma on the test split across two zero-shot benchmarks, `CB-II` and `fev-bench-mini`. Figure 6 places the Auto-Chroma portfolios of the three models among their frequency heuristic portfolios, their generalists, and the reference models. We observe that, out of the six comparisons (three models on two benchmarks), the Auto-Chroma portfolio achieves a modest improvement over both its frequency heuristic portfolio and its generalist in five. The most prominent gains are those of Chroma-4M and Chronos-Bolt-Tiny on `CB-II`, about 6 and 5 points of MASE skill score over their generalists, while Chronos-2-Small with Auto-Chroma tops the leaderboard of this benchmark, above the larger Chronos-2. The exception is Chronos-2-Small on `fev-bench-mini`, where the Auto-Chroma portfolio falls slightly below both. Results on both metrics of each benchmark are given in Appendix D.4.

4.5 RQ5: When does the validation gain carry over to test?

RQ4 leaves one exception: on `fev-bench-mini`, the Chronos-2-Small portfolio of Auto-Chroma gains about 3.7 points of skill score over $P^{\mathrm{h}}_{\mathrm{freq}}$ in validation (see Figure 3), yet it falls slightly below $P^{\mathrm{h}}_{\mathrm{freq}}$ on test. To understand when a validation gain carries over, we evaluate every candidate portfolio of the two Chronos-2-Small runs of both benchmarks on the test split (see Figure 7). Each panel plots the validation skill score of a candidate against its test skill score, so a candidate in the upper-right quadrant is above $P^{\mathrm{h}}_{\mathrm{freq}}$ on both splits, and its validation gain carries over. For the

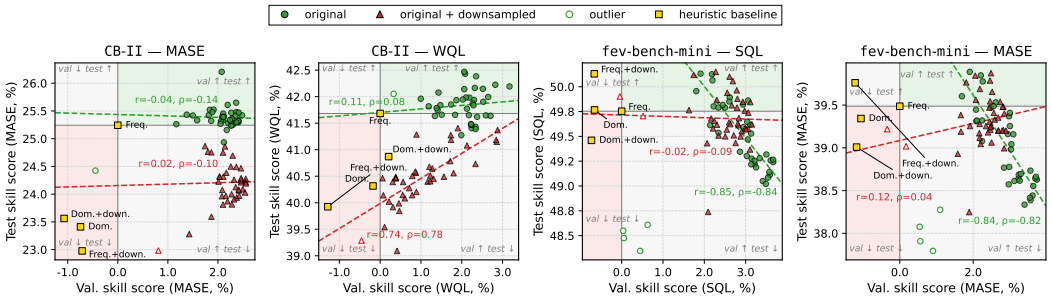

Figure 7: Validation performance against test performance for every candidate portfolio of the two Chronos-2-Small runs, with Pearson r and Spearman ρ per run. Outliers are excluded from the fits.

run on the original corpus, every candidate is above $P^{\mathrm{h}}_{\mathrm{freq}}$ in validation. On `CB-II`, almost every candidate is in the upper-right quadrant, so the improvement of RQ4 holds for almost any candidate that the search could return. On `fev-bench-mini`, in contrast, only 12 of the 36 candidates are in the upper-right quadrant on SQL, and the correlation between validation performance and test performance is negative, which accounts for the exception of RQ4. The difference between the two benchmarks lies in the number of forecasts that a task scores. When we recompute the skill scores of Figure 7 on groups of tasks of increasing size (see Appendix D.5), the negative correlation appears on the tasks with few forecasts of both benchmarks, and it fades as the number of forecasts grows. `CB-II` is prone to the same problem, but to a lesser degree, so that the validation gain carries over on the full benchmark. These findings are consistent with prior work: a selection signal computed from a small sample is known to be unreliable (Cawley & Talbot, 2010), in particular for ensemble selection (Caruana et al., 2006), whose weights are fit on the same holdout window on which the portfolio is scored. The limit thus lies in the selection signal that a benchmark provides, which any model selection on this holdout inherits, rather than in the search. Tasks with more series already supply a stable signal (Tashman, 2000), and designing validation tasks whose signal generalizes reliably to a wide range of test tasks is an important direction for future work.

5 Conclusion, Limitations, and Future Work

The contribution of Auto-Chroma is not simply that LLM agents can design portfolios that outperform the heuristic portfolios of Chroma, but that it establishes, to our knowledge, the first treatment of portfolio construction as an autonomous search, and the first to treat the portfolios it finds and the trajectory that leads to them as a subject of study in their own right. Auto-Chroma retires the fixed heuristic-based partition rule of Chroma in favor of a search over partitions of the post-training corpus, in which memory-augmented LLM agents propose, diagnose, and revise portfolios without any human intervention. We believe that Auto-Chroma marks a first step toward the fully automated design of model portfolios, and that its findings reach well beyond Chroma, to holdout-based model selection at large and to any LLM agent that searches a configuration space.

Limitations and future work. Auto-Chroma inherits the holdout protocol of Chroma, whose validation signal is most stable on tasks that score many forecasts (Section 4.5); designing validation tasks whose signal generalizes reliably to a wide range of test tasks is an important direction. In this paper, every run starts from an empty database to isolate its outcome for analysis, although the memory of Section 3.2 natively allows a new search to ground its proposals in the specialists and reports of earlier runs. Persisting this memory across models and benchmarks, in the manner of coding-agent harnesses such as Claude Code[1] that retain project rules and memory across sessions, is a promising direction. Our generalists also come from the Chronos family, and extending the post-training pipeline beyond the models that share the `AutoGluon` interface (Shchur et al., 2023), to portfolios that mix specialists of several generalists remains future work.

[1] `https://code.claude.com/docs/en/memory`

AI USE STATEMENT

In this work, we used generative AI tools to partially design experiments, to implement code for the methods, and to partially interpret results; the research ideas and the method are ours. We have not used generative AI tools for ideation, proposing hypotheses, executing experiments (pre-training, post-training, or evaluation), or the bibliography; synthetic data generation, mathematical claims and proofs, translation, and qualitative analysis are not applicable. Additionally, we used generative AI tools to aid and polish the writing under the authors' high-level supervision, to create result visualizations, to find and curate public datasets, and to assist in finding related work, although the literature survey was largely performed by the authors. We have reviewed all AI-assisted work: code was tested by the authors, all reported numbers come from our own runs, and all text was revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix C.1 specifies the generalists and their checkpoints, the post-training hyperparameters of the specialists, the two corpora, the acceptance rule of the search, the orchestrator LLMs and how they are served, and the evaluation protocol with the sources of all reference results. The heuristic portfolios follow the construction of Kayaalp et al. (2026) on the corpus of Table 6. Appendix B details ensemble selection under rolling-window multivariate evaluation, and Appendix G gives the agent harness, excerpts of the analyst and strategist prompts, and example reports. Algorithms 4, 5, and 6 in Appendix D.2 define the LLM-free baselines, and Appendix C.2 the similarity measures. All benchmarks and training datasets are publicly available, and the Chronos-Bolt-Tiny and Chronos-2-Small checkpoints are available from the AutoGluon organization on the Hugging Face Hub. The code to run the main experiments, together with the weights of our Chroma-4M generalist, is publicly available[2].

REFERENCES

Nans Addor, Andrew J. Newman, Naoki Mizukami, and Martyn P. Clark. The CAMELS data set: catchment attributes and meteorology for large-sample studies. *Hydrology and Earth System Sciences*, 21(10):5293–5313, 2017. doi: 10.5194/hess-21-5293-2017.

Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Türkmen, 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 Bernie Wang. Chronos: Learning the language of time series. *TMLR*, 2024, 2024.

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

Andreas Auer, Daniel Klotz, Sebastian Böck, and Sepp Hochreiter. Pre-trained forecasting models: Strong zero-shot feature extractors for time series classification. In *Recent Advances in Time Series Foundation Models Have We Reached the 'BERT Moment'?*, 2025a.

Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. TiRex: Zero-shot forecasting across long and short horizons with enhanced in-context learning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Proc. of NeurIPS*, 2025b.

Anthony Bao, Venkata Hasith Vattikuti, Jeffrey B. Lai, and William Gilpin. Universal redundancies in time series foundation models. In *Proc. of ICML*, 2026.

[2] https://anonymous.4open.science/r/auto-chroma-main-B73E/

A Bashir, C Leap, A Blumenthal, T Estrada, A Bidram, M Martinez-Ramon, and M Abdullah. Power, voltage, frequency and temperature dataset from Mesa Del Sol microgrid. Dryad, 2023. URL https://doi.org/10.5061/dryad.fqz612jzb.

Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. Ensemble selection from libraries of models. In Carla E. Brodley (ed.), *Proc. of ICML*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004. doi: 10.1145/1015330.1015432.

Rich Caruana, Art Munson, and Alexandru Niculescu-Mizil. Getting the most out of ensemble selection. In *Sixth International Conference on Data Mining (ICDM'06)*, pp. 828–833, 2006. doi: 10.1109/ICDM.2006.76.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *JMLR*, 11(70):2079–2107, 2010.

Angelica Chen, David Dohan, and David R. So. EvoPrompting: Language models for code-level neural architecture search. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Proc. of NeurIPS*, 2023.

Jialin Chen, Ziyu Zhao, Gaukhar Nurbek, Aosong Feng, Ali Maatouk, Leandros Tassiulas, Yifeng Gao, and Rex Ying. TRACE: grounding time series in context for multimodal embedding and retrieval. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Proc. of NeurIPS*, 2025.

Alex Clerc and Elizabeth Lingkan. Hill of Towie wind farm open dataset. Zenodo, 2026. URL https://doi.org/10.5281/zenodo.20204946.

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 Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Proc. of NeurIPS*, 2025.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proc. of ICML*. OpenReview.net, 2024.

Sarkar Snigdha Sarathi Das, Palash Goyal, Mihir Parmar, Nanyun Peng, Vishy Tirumalashetty, Chun-Liang Li, Rui Zhang, Jinsung Yoon, and Tomas Pfister. Nexus : An agentic framework for time series forecasting, 2026a.

Sarkar Snigdha Sarathi Das, Palash Goyal, Mihir Parmar, Yiwen Song, Long T. Le, Lesly Miculicich, Jinsung Yoon, Rui Zhang, Hamid Palangi, and Tomas Pfister. Synapse: Adaptive arbitration of complementary expertise in time series foundational models. *TMLR*, 2026, 2026b.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence, 2026.

Thomas Duc and Eric Simley. SMARTEOLE wind farm control open dataset. Zenodo, 2022. URL https://doi.org/10.5281/zenodo.7342466.

Thomas D. P. Edwards, James Alvey, Justin Alsing, Nam H. Nguyen, and Benjamin D. Wandelt. Scaling-laws for large time-series models, 2024.

Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam Nguyen, Wesley M. Gifford, Chandra Reddy, and Jayant Kalagnanam. Tiny time mixers (TTMs): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Proc. of NeurIPS*, 2024.

Jose A. Fiorucci, Tiago R. Pellegrini, Francisco Louzada, Fotios Petropoulos, and Anne B. Koehler. Models for optimising the theta method and their relationship to state space models. *International Journal of Forecasting*, 32(4):1151–1161, 2016. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2016.02.005.

Azul Garza and Renée Rosillo. TimeCopilot, 2025. URL https://arxiv.org/abs/2509.00616.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive. In *Proc. of NeurIPS*, 2021.

Ary L. Goldberger and David R. Rigney. *Nonlinear Dynamics at the Bedside*, pp. 583–605. Springer New York, New York, NY, 1991. ISBN 978-1-4612-3118-9. doi: 10.1007/978-1-4612-3118-9_22.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In *Proc. of ICML*. OpenReview.net, 2024.

J. C. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, 27(4): 857–871, 1971. ISSN 0006341X, 15410420.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Proc. of NeurIPS*, 2023.

R.J. Hyndman and G. Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, 2021. ISBN 9780987507136.

Rob J. Hyndman and Yeasmin Khandakar. Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, 27(3):1–22, 2008. doi: 10.18637/jss.v027.i03.

Sergey Ioffe. Improved consistent sampling, weighted minhash and L1 sketching. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pp. 246–255, USA, 2010. IEEE Computer Society. ISBN 9780769542560. doi: 10.1109/ICDM.2010.80.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *Proc. of ICLR*. OpenReview.net, 2024.

Mert Kayaalp, Ali Caner Turkmen, Oleksandr Shchur, Pedro Mercado, Abdul Fatir Ansari, Michael Bohlke-Schneider, and Bernie Wang. Test-time efficient pretrained model portfolios for time series forecasting. In *Proc. of ICLR*, 2026.

Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The UCI Machine Learning Repository. https://archive.ics.uci.edu, 2023.

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.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Chenghao Liu, Taha Aksu, Juncheng Liu, Xu Liu, Hanshu Yan, Quang Pham, Silvio Savarese, Doyen Sahoo, Caiming Xiong, and Junnan Li. Moirai 2.0: When Less Is More for Time Series Forecasting, 2025a.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proc. of ICML*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025b.

Michael Franklin Mbouopda, Thomas Guyet, Nicolas Labroche, and Abel Henriot. FrenchPiezo: the French mainland groundwater level multivariate time series. Zenodo, 2022. URL https://doi.org/10.5281/zenodo.7193812.

Clayton Miller, Anjukan Kathirgamanathan, Bianca Picchetti, Pandarasamy Arjunan, June Young Park, Zoltan Nagy, Paul Raftery, Brodie W. Hobson, Zixiao Shi, and Forrest Meggers. The building data genome project 2, energy meter data from the ASHRAE great energy predictor III competition. *Scientific Data*, 7(1), 2020. ISSN 2052-4463. doi: 10.1038/s41597-020-00712-x.

Andrew J. Newman, Kevin Sampson, Martyn Clark, A. Bock, Roland Viger, D. Blodgett, Nans Addor, and M. Mizukami. CAMELS: Catchment Attributes and MEteorology for Large-sample Studies. Zenodo, 2022. URL https://doi.org/10.5065/D6MW2F4D.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery, 2025.

Aristotelis Ntafalias, Sofia Visvardi, and Axel Weißenfeld. Six-month monitoring dataset from a 10-turbine onshore wind farm in Greece. Zenodo, 2024. URL https://doi.org/10.5281/zenodo.14546480.

Santosh Palaskar, Vijay Ekambaram, Arindam Jati, Neelamadhav Gantayat, Avirup Saha, Seema Nagar, Nam H. Nguyen, Pankaj Dayama, Renuka Sindhgatta, Prateeti Mohapatra, Harshit Kumar, Jayant Kalagnanam, Nandyala Hemachandra, and Narayan Rangaraj. AutoMixer for improved multivariate time-series forecasting on business and IT observability data. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 22962–22968. AAAI Press, 2024. doi: 10.1609/AAAI.V38I21.30336.

Qwen Team. Qwen3.5: Towards native multimodal agents, 2026. URL https://qwen.ai/blog?id=qwen3.5.

Cory A. Rieth, Ben D. Amsel, Randy Tran, and Maia B. Cook. Additional Tennessee Eastman process simulation data for anomaly detection evaluation. Harvard Dataverse, 2017. URL https://doi.org/10.7910/DVN/6C3JR1.

Oleksandr Shchur, Caner Turkmen, Nick Erickson, Huibin Shen, Alexander Shirkov, Tony Hu, and Yuyang Wang. AutoGluon-TimeSeries: AutoML for probabilistic time series forecasting. In *Proc. of AutoML*, 2023.

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, 2025.

Qichao Shentu, Beibu Li, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series anomaly detector with adaptive bottlenecks and dual adversarial decoders. In *Proc. of ICLR*. OpenReview.net, 2025.

Jingzhe Shi, Qinwei Ma, Huan Ma, and Lei Li. Scaling law for time series forecasting. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Proc. of NeurIPS*, 2024.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-MoE: Billion-scale time series foundation models with mixture of experts. In *Proc. of ICLR*. OpenReview.net, 2025.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (eds.), *Proceedings*

of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019, pp. 2828–2837. ACM, 2019. doi: 10.1145/3292500.3330672.

Leonard J. Tashman. Out-of-sample tests of forecasting accuracy: An analysis and review. *International Journal of Forecasting*, 16(4):437–450, 2000. ISSN 0169-2070. doi: https://doi.org/10.1016/S0169-2070(00)00065-0. The M3- Competition.

Xiaoqian Wang, Rob J Hyndman, Feng Li, and Yanfei Kang. Forecast combinations: An over 50-year review. *International Journal of Forecasting*, 39(4):1518–1547, 2023.

Gerald Woo, Chenghao Liu, Akshat Kumar, and Doyen Sahoo. Pushing the limits of pre-training for time series forecasting in the CloudOps domain, 2023.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Proc. of ICML*. OpenReview.net, 2024.

Xiaomi MiMo Team, Anqi Liu, Aoxin Ma, Bo Chen, Bo Yang, Chen Wang, Chen Zhang, Chengda Tang, Chengwei Wang, Chiheng Lou, Depeng Yan, Fuli Luo, Gang Wang, Hailin Zhang, Jiale Sun, Kang Zhou, Rui Huang, Shaohui Liu, Shen Huang, Shijie Cao, Shuaishuai Fan, Tianling Zhou, Xiangwei Deng, Xueyang Xie, Xuli Wang, Yingchun Lai, Yu Yang, Yuan Zhang, Zhen Tang, Zhonghua Deng, and Zihan Jiang. Full-pipeline inference optimization for MiMo-V2.5 series: Pushing hybrid SWA efficiency to the limit, 2026.

Hanshu Yan, Hong-Quang Pham, Ibrahim Taha Aksu, Jun Hao Liew, Caiming Xiong, Silvio Savarese, Doyen Sahoo, and Junnan Li. MoiraiAgent: An agentic framework for context-aware time-series forecasting. Salesforce AI Research Blog, 2026. URL `https://www.salesforce.com/blog/moiraiagent/`. Blog post, January 14, 2026.

Qingren Yao, Chao-Han Huck Yang, Renhe Jiang, Yuxuan Liang, Ming Jin, and Shirui Pan. Towards neural scaling laws for time series foundation models. In *Proc. of ICLR*. OpenReview.net, 2025.

Annan Yu, Danielle C. Maddix, Boran Han, Xiyuan Zhang, Abdul Fatir Ansari, Oleksandr Shchur, Christos Faloutsos, Andrew Gordon Wilson, Michael W. Mahoney, and Bernie Wang. Understanding transformers for time series: Rank structure, flow-of-ranks, and compressibility. In *Proc. of ICLR*, 2026.

Lifan Zhao, Yanyan Shen, Zhaoyang Liu, Xue Wang, and Jiaji Deng. Less is more: Unlocking specialization of time series foundation models via structured pruning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Proc. of NeurIPS*, 2025.

APPENDIX

A Notation **16**

B Ensemble selection under rolling-window multivariate evaluation **18**

C Experiment details **19**

C.1 Experiment setup 19

C.2 Portfolio similarity 20

D Additional results for the research questions **21**

D.1 Auto-Chroma against heuristic portfolios on both corpora 21

D.2 Random and greedy search 23

D.3 Sensitivity to the orchestrator LLM 26

D.4 Comparison with reference models on both metrics 26

D.5 Task size and the reliability of the validation signal 27

E Supplementary analyses: search stability, memory utilization, analyst ablation, transfer, and cost **30**

E.1 Search stability across repeated runs 30

E.2 Memory utilization 30

E.3 Analyst ablation 30

E.4 Transfer across benchmarks 32

E.5 Computational cost and token usage 32

F Training datasets **32**

G Implementation details **34**

A NOTATION

Table 1 lists every symbol used in the paper, grouped by the object it describes. Scalars and indices are italic, vectors are bold lowercase, matrices and sequences are bold uppercase, and sets are calligraphic. A hat marks a forecast, a bar a task average, and a star an optimum.

Table 1: Notation used in the paper.

Symbol	Meaning
	Time series
$\boldsymbol{Y}_{1:T} \in \mathbb{R}^{T \times D}$	History with context length T and D target variates; $D = 1$ is a univariate series
$\boldsymbol{Y}_{T+1:T+H}$	Future to predict over the horizon H
$\hat{\boldsymbol{Y}}_{T+1:T+H}$	Forecast, a set of quantiles with the median as point forecast
$\boldsymbol{Y}^{(i)} \in \mathbb{R}^{L \times D}$, $\boldsymbol{Y}^{(i,d)}$	Full history of item i with last observation L; its variate d
$\hat{\boldsymbol{Y}}^{(i)}$, $\hat{\boldsymbol{Y}}^{(i,d)}$	Forecast of item i by one member; its variate d
I, $\mathcal{S}$	Items per task; the series set of a task after expansion, $\lvert\mathcal{S}\rvert = I \cdot D$
	Tasks and benchmarks
τ, H_τ	Task (dataset, horizon, series, metric); its horizon
$\mathcal{B}$, $\lvert\mathcal{B}\rvert$	Benchmark, a finite set of tasks; number of tasks
$\mathcal{B}^{\mathrm{val}}$, $\mathcal{B}^{\mathrm{test}}$	Benchmark under the validation split and under the test split
	Generalist and specialists
f_θ, θ	Generalist and its frozen pretrained parameters, from which every specialist is post-trained
$\mathcal{D}$, $\mathcal{D}_k \subseteq \mathcal{D}$	Post-training corpus of datasets; dataset subset of specialist k
η_k, s_k	Learning rate and number of post-training steps of specialist k
$c_k = (\mathcal{D}_k, \eta_k, s_k)$	Configuration of specialist k
$\theta_k = \mathrm{FT}(\theta;\, c_k)$, f_{θ_k}	Parameters and model of specialist k
	Portfolios and ensembles
K, $K_{\max}$	Number of specialists in a portfolio; cap on K
$P = \{c_1, \ldots, c_K\}$, $\mathcal{P}$	Portfolio (generalist implicit); portfolio space
$P^{\mathrm{h}}_{\mathrm{freq}}$, $P^{\mathrm{h}}_{\mathrm{dom}}$	Heuristic portfolios, one specialist per sampling frequency or per domain
$\boldsymbol{w}^\tau$, w^τ_k	Ensemble weights on task τ, nonnegative and summing to one over the $K+1$ members; weight of member k, which runs at inference only if $w^\tau_k > 0$
$\hat{\boldsymbol{Y}}_{k,\tau}$, $\hat{\boldsymbol{Y}}_{\mathrm{ens}} = \sum_{k=0}^{K} w^\tau_k \hat{\boldsymbol{Y}}_{k,\tau}$	Forecast of member k on task τ; ensemble forecast
$\bar{w}_k = \frac{1}{\lvert\mathcal{B}\rvert} \sum_\tau w^\tau_k$	Task-average weight of member k
$\boldsymbol{W} \in \mathbb{R}^{\lvert\mathcal{B}\rvert \times (K+1)}$	Routing matrix; row τ is $\boldsymbol{w}^\tau$
	Rolling windows inside a task
n, N	Window index; number of windows
T_n	Cutoff of window n; weights are fit on $\boldsymbol{Y}_{T_n-H+1:T_n}$ and tested on $\boldsymbol{Y}_{T_n+1:T_n+H}$
$\boldsymbol{w}^{\tau,n}$, $\hat{\boldsymbol{Y}}_{k,\tau,n}$	Weights and forecast of member k on window n, where the weight of member k is the fraction of the first $J^\star$ greedy picks that chose it; task values are means over n
$e^{\tau,n}_{\mathrm{val}}$, $e^{\tau,n}_{\mathrm{test}}$, e^{τ}_{val}, e^{τ}_{test}	Validation and test error of window n; their means over the N windows
j, J, $J^\star$, k_j	Greedy step, number of steps, step with the lowest running error, member picked at step j
ε_j	Running error of the greedy average after j picks
	Metrics and skill
m, $\mathcal{M}$, $m^\star$	Metric; all metrics of a benchmark; primary metric

Symbol	Meaning
$e_m(f, \tau)$	Error of forecaster f on task τ under m
b, P_{ref}	Reference forecaster; reference portfolio of the optimization
$S_m\big(f \,\big|\, b, \mathcal{B}\big)$	Skill score of f over $\mathcal{B}$ against b under m, in percent, higher is better
	Search loop
r, $r_{\max}$	Round; budget in rounds
P_r	Portfolio proposed in round r
$\mathcal{H}_r$	State of the search held in the database
R_i^{A}, R_i^{S}	Analyst diagnosis of P_i; strategist rationale and predictions for P_i
π_{A}, π_{S}	Analyst and strategist LLM calls
β_m, ξ	Best accepted skill score under m and acceptance margin
π_{greedy}, π_{rand}	LLM-free greedy and random search baselines
$\mathcal{V}$, P_{lead}, Q	Set of portfolios already evaluated; accepted candidate with the highest $S(P)$, which greedy search mutates; proposed portfolio before the check that it is not in $\mathcal{V}$
$\mathcal{G}_s$, $\mathcal{G}_\eta$, u, γ	step grid, learning-rate grid, size of a drawn dataset subset, scaling factor
$P^\star = \arg\max_{r \le r_{\max}} S(P_r)$	Output of the search
	Portfolio similarity between runs A and B
$\mathcal{D}_i^A$, K^A, $\boldsymbol{w}^{A,\tau}$, $\bar{w}_i^A$	Dataset subset, specialist count, per-task weights, and task-average weight of member i in run A
$J(\mathcal{D}_i, \mathcal{D}_j)$	Jaccard similarity of two dataset subsets
$\kappa(i, j)$	Member kernel
$\langle \bar{w}^A, \bar{w}^B \rangle_\kappa$, $\|\bar{w}^A\|_\kappa$	Kernel-weighted inner product and norm of weight vectors
$v_d^A = \sum_{i : d \in \mathcal{D}_i^A} \bar{w}_i^A$	Dataset mass of run A on dataset d
$\Omega_{\text{specialist}}(A, B)$	Specialist overlap (cosine of the task-average weights under κ)
$\Omega_{\text{dataset}}(A, B)$	Dataset overlap (shared mass divided by combined mass)
$\Omega_{\text{task}}(A, B)$	Task-routing overlap (mean over tasks of the cosine of the per-task weights under κ)

Algorithm 1 FORECAST: forecast of one item by one member

procedure FORECAST(f_θ, $\boldsymbol{Y}^{(i)}_{1:T}$) ▷ $\boldsymbol{Y}^{(i)}_{1:T} \in \mathbb{R}^{T\times D}$: all D columns of item i
  if f_θ is multivariate **then**
    $\hat{\boldsymbol{Y}}^{(i)} \leftarrow f_\theta(\boldsymbol{Y}^{(i)}_{1:T})$ ▷ one call reads all D columns and returns all D forecasts
  else
    $\hat{\boldsymbol{Y}}^{(i,d)} \leftarrow f_\theta(\boldsymbol{Y}^{(i,d)}_{1:T})$ for $d = 1, \dots, D$ ▷ D calls, each reads and returns one column
  end if
  return $\hat{\boldsymbol{Y}}^{(i)} = [\hat{\boldsymbol{Y}}^{(i,1)}, \dots, \hat{\boldsymbol{Y}}^{(i,D)}]$
end procedure

Algorithm 2 Greedy ensemble selection on the validation windows of task τ

Require: members $f_{\theta_0}, \dots, f_{\theta_K}$; task τ with I items, D variates, horizon H, last observation L, N windows, primary metric $m^\star$; number of greedy steps J

$\mathcal{S} \leftarrow \{\boldsymbol{Y}^{(i,d)} : i = 1, \dots, I,\ d = 1, \dots, D\}$ ▷ one series per item and variate; $|\mathcal{S}| = I \cdot D$
for $n = 1, \dots, N$ **do**
  $T_n \leftarrow L - (N - n + 1)H$ ▷ cutoff of window n
  for $k = 0, \dots, K$ **do**
    for $i = 1, \dots, I$ **do**
      $\hat{\boldsymbol{Y}}^{(i)}_{k,\tau,n} \leftarrow$ FORECAST(f_{θ_k}, $\boldsymbol{Y}^{(i)}_{1:T_n-H}$)
    end for
  end for
  $\hat{\boldsymbol{Y}}_{\text{ens}} \leftarrow \boldsymbol{0}$
  for $j = 1, \dots, J$ **do** ▷ greedy selection with replacement
    $k_j \leftarrow \arg\min_k e_{m^\star}\big(\frac{j-1}{j}\hat{\boldsymbol{Y}}_{\text{ens}} + \frac{1}{j}\hat{\boldsymbol{Y}}_{k,\tau,n},\ \boldsymbol{Y}_{T_n-H+1:T_n}\big)$ ▷ pooled over series
    $\hat{\boldsymbol{Y}}_{\text{ens}} \leftarrow \frac{j-1}{j}\hat{\boldsymbol{Y}}_{\text{ens}} + \frac{1}{j}\hat{\boldsymbol{Y}}_{k_j,\tau,n}$ ▷ running average of the forecasts
    $\varepsilon_j \leftarrow e_{m^\star}(\hat{\boldsymbol{Y}}_{\text{ens}}, \boldsymbol{Y}_{T_n-H+1:T_n})$ ▷ running error after j picks
  end for
  $J^\star \leftarrow \arg\min_{j \le J} \varepsilon_j$ ▷ stop at the lowest running error
  $e^{\tau,n}_{\text{val}} \leftarrow \varepsilon_{J^\star}$ ▷ validation error of window n
  $w^{\tau,n}_k \leftarrow |\{j \le J^\star : k_j = k\}| \,/\, J^\star$ for $k = 0, \dots, K$ ▷ fraction of picks
end for
return $\{\boldsymbol{w}^{\tau,n}\}_{n=1}^N,\ \boldsymbol{w}^\tau = \frac{1}{N}\sum_n \boldsymbol{w}^{\tau,n},\ e^\tau_{\text{val}} = \frac{1}{N}\sum_n e^{\tau,n}_{\text{val}}$

B ENSEMBLE SELECTION UNDER ROLLING-WINDOW MULTIVARIATE EVALUATION

This appendix specifies how the ensemble weights are fit (Algorithm 2) and how a portfolio forecasts with fixed weights (Algorithm 3). Chroma (Kayaalp et al., 2026) takes both from the `GreedyEnsemble` of `AutoGluon` (Shchur et al., 2023), which handles one target series per item and one evaluation window. We re-implement the same greedy selection (Caruana et al., 2004) for the multivariate rolling-window tasks of our benchmarks, with the selection rule and scorers of `AutoGluon` unchanged, so that it coincides with the stock `GreedyEnsemble` on a univariate task with one window.

Setup. Each item of a task is expanded into its D variates, and Algorithm 1 states how a multivariate and a univariate member read an item. The validation block of a window is the temporal holdout of Section 2. Every `CB-II` task has $N = 1$; `fev-bench-mini` declares N per task.

Fitting the weights. Each window receives its own greedy run on the pooled $I \cdot D$ series of its validation block (Algorithm 2). Every step adds, with replacement, the member whose inclusion in the running average most reduces the primary error $e_{m^\star}$. We use $J = 100$ steps and stop at the step $J^\star$ with the lowest running error. The running average after $J^\star$ picks equals the weighted sum $\sum_k w^{\tau,n}_k \hat{\boldsymbol{Y}}_{k,\tau,n}$ with $w^{\tau,n}_k$ the fraction of picks of member k, so the weights $\boldsymbol{w}^{\tau,n}$ over the $K+1$ members are nonnegative, sum to one, and are sparse. The validation error $e^{\tau,n}_{\text{val}}$ scores this sum on the block the weights were fit on.

Algorithm 3 Weighted ensemble forecast on the test windows of task τ

Require: members $f_{\theta_0}, \ldots, f_{\theta_K}$; task τ as in Algorithm 2; metric m; weights $\{\boldsymbol{w}^{\tau,n}\}_{n=1}^{N}$ from Algorithm 2

$\mathcal{S} \leftarrow \{\boldsymbol{Y}^{(i,d)} : i = 1, \ldots, I,\ d = 1, \ldots, D\}$
for $n = 1, \ldots, N$ **do**
  $T_n \leftarrow L - (N - n + 1)H$ ▷ cutoff of window n
  for k with $w_k^{\tau,n} > 0$ **do** ▷ only weighted members run
    for $i = 1, \ldots, I$ **do**
      $\hat{\boldsymbol{Y}}_{k,\tau,n}^{(i)} \leftarrow \textsc{Forecast}(f_{\theta_k}, \boldsymbol{Y}_{1:T_n}^{(i)})$
    end for
  end for
  $e_{\text{test}}^{\tau,n} \leftarrow e_m\big(\sum_k w_k^{\tau,n} \hat{\boldsymbol{Y}}_{k,\tau,n},\ \boldsymbol{Y}_{T_n+1:T_n+H}\big)$ ▷ weights reused
end for
return $e_{\text{test}}^{\tau} = \frac{1}{N}\sum_n e_{\text{test}}^{\tau,n}$

What the search reads. During the search, Auto-Chroma evaluates each proposed portfolio P_r with Algorithm 2 on $\mathcal{B}^{\text{val}}$. The task errors e_{val}^{τ} give the objective $S(P_r) = \sum_{m \in \mathcal{M}} S_m\big(P_r \,\big|\, P_{\text{ref}}, \mathcal{B}^{\text{val}}\big)$ (Equation 2). The task weights $\boldsymbol{w}^{\tau}$ form the rows of the routing matrix $\boldsymbol{W}_r$, which enters the history $\mathcal{H}_r$ read by the analyst and the strategist. Under multi-window evaluation, $\boldsymbol{w}^{\tau}$ is the mean of the per-window weight vectors and is a descriptive summary of the member contributions on task τ; under single-window evaluation, it coincides with the fitted weights of the task.

Forecasting. Every portfolio forecasts each test task with Algorithm 3: window n of each test task is forecast by the members with $w_k^{\tau,n} > 0$ from the full context $\boldsymbol{Y}_{1:T_n}$ and combined with the stored $\boldsymbol{w}^{\tau,n}$ of the same window. The weights are never refit on test data, and only weighted members run, which is what makes the portfolio test-time efficient.

C Experiment details

This appendix gives the details that the experiments of Section 4 share: the experiment setup (Appendix C.1) and the formal definitions of the portfolio similarities (Appendix C.2).

C.1 Experiment setup

Chroma-4M and Chronos-Bolt-Tiny. Both univariate models instantiate the Chronos-Bolt architecture (Ansari et al., 2024) at 4.4M (Kayaalp et al., 2026) and 8.7M parameters, respectively. We use the public checkpoint of Chronos-Bolt-Tiny, whereas no pretrained checkpoint of Chroma-4M is publicly available, so we pretrain it from scratch following Ansari et al. (2024). Post-training follows the protocol of Kayaalp et al. (2026): each specialist is initialized from its generalist and differs from the other specialists only in its dataset subset $\mathcal{D}_k$, learning rate η_k, and step count s_k. We optimize with AdamW (weight decay 0.1, $\beta = (0.9, 0.98)$) and gradient clipping at 1.0, with a batch of 256 series, a context length of 2048, and a prediction length of 64; the learning rate decays linearly from η_k to zero over s_k steps without warmup. Each variate of a multivariate item is treated as an independent univariate series, so that no covariates reach the model. The specialists of the heuristic portfolios use $s_k = 1000$ and the default learning rate of the model, $\eta_k = 10^{-3}$ for Chroma-4M and 10^{-4} for Chronos-Bolt-Tiny.

Chronos-2-Small. We use the public checkpoint of the multivariate model (27.9M parameters) of Ansari et al. (2025). Its specialists share the optimizer, learning-rate schedule, context length, prediction length, and heuristic defaults above ($s_k = 1000$, $\eta_k = 10^{-5}$), but differ in the construction of training samples. A multivariate item forms one sample with all D variates, and the covariate regime of each sample follows the pretraining recipe of Chronos-2 in two draws: with probability 0.5, all variates remain targets; otherwise, each variate is designated a known-future covariate with probability 0.5, subject to at most $\lfloor D/2 \rfloor$ covariates, and the remaining variates are the targets of the loss. A batch holds 256 target series.

Corpora. The original corpus consists of the 44 datasets listed in Appendix F. It closely follows the training corpus of Kayaalp et al. (2026), with three modifications: we exclude the datasets suspected of overlap with the evaluation benchmarks and those not publicly available, and we add nine multivariate datasets so that the corpus also covers the covariate regime of Chronos-2-Small. The original + downsampled corpus retains all 44 datasets and adds 168 downsampled datasets, derived from 36 of the 44 datasets with 1 to 12 per source dataset, for a total of 212.

Heuristic portfolios. Following Chroma (Kayaalp et al., 2026), the frequency heuristic $P^{\mathrm{h}}_{\mathrm{freq}}$ assigns one specialist per frequency band (sub-hourly, hourly, daily, weekly, and above weekly), and the domain heuristic $P^{\mathrm{h}}_{\mathrm{dom}}$ assigns one specialist per domain (energy, retail, transport, weather, web, and various). On the original + downsampled corpus, we construct $P^{\mathrm{h}}_{\mathrm{freq}}$ and $P^{\mathrm{h}}_{\mathrm{dom}}$ over all 212 datasets under the same partition rule, where a downsampled dataset is assigned to the frequency band of its new frequency and to the domain of its source dataset. The validation reference P_{ref} is always $P^{\mathrm{h}}_{\mathrm{freq}}$ on the original corpus, so that skill scores remain comparable across the two corpora.

Orchestrator. Both agents run in the harness of Appendix G, with each call in a new session. The default orchestrator LLM is DeepSeek V4 Flash, and the LLM ablation uses MiMo v2.5 and Qwen3.5-27B. The former two are served through an API with a context of 1M tokens, whereas Qwen3.5-27B is quantized to FP8 and served locally with vLLM (Kwon et al., 2023) with a context of 180k tokens. A run proposes $r_{\max} = 40$ portfolios of at most $K_{\max} = 10$ specialists. A candidate is accepted if its skill score on at least one metric of $\mathcal{M}$ exceeds the best accepted value β_m of that metric by more than a margin $\xi = 0.1$ percentage points, which filters improvements within evaluation noise. The leader is the accepted candidate with the highest $S(P)$ and serves as the reference of the next analyst report.

Benchmarks and reference models. Each benchmark defines two leaderboard metrics $\mathcal{M}$, the first of which is the primary metric $m^\star$. `CB-II` comprises 27 tasks scored by the mean absolute scaled error (MASE) for point forecasts and the weighted quantile loss (WQL) for probabilistic forecasts, with one evaluation window per task. `fev-bench-mini` is the official subset of `fev-bench` provided by `fev` and comprises 20 tasks scored by the scaled quantile loss (SQL) and MASE, with several rolling windows per task. We follow the evaluation protocol of Shchur et al. (2025) and take the results of the reference models from the public repository of `fev`[3], except for Chronos-2, which we re-evaluate with cross-learning disabled for a fair comparison with Chronos-2-Small.

C.2 Portfolio similarity

This subsection defines the three similarities that compare the portfolios of different orchestrator LLMs in RQ3 (Figure 5) and the portfolios of repeated runs in Appendix E.1 (Figure 15).

Setup. Let A and B be the best portfolios of two runs on the same benchmark $\mathcal{B}$ and the same forecasting model. Portfolio A has members $i = 0, \ldots, K^A$, where member 0 is the generalist and member $i \geq 1$ is a specialist with dataset subset $\mathcal{D}^A_i \subseteq \mathcal{D}$. The validation split gives A a routing matrix with rows $\boldsymbol{w}^{A,\tau}$, and we define the task-average weight of member i as $\bar{w}^A_i = \frac{1}{|\mathcal{B}|}\sum_{\tau\in\mathcal{B}} w^{A,\tau}_i$, the mean of its ensemble weight over the tasks. Because the weights of every row $\boldsymbol{w}^{A,\tau}$ are nonnegative and sum to 1, so are the task-average weights: $\bar{w}^A_i \geq 0$ and $\sum_i \bar{w}^A_i = 1$. All similarities see a specialist only through its dataset subset, and not through its learning rate or its step count.

Member kernel. We compare two members with the kernel

$$\kappa(i,j) = \begin{cases} J(\mathcal{D}_i, \mathcal{D}_j) = |\mathcal{D}_i \cap \mathcal{D}_j| \,/\, |\mathcal{D}_i \cup \mathcal{D}_j| & \text{if } i \text{ and } j \text{ are specialists,} \\ 1 & \text{if } i \text{ and } j \text{ are the generalist,} \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where J is the Jaccard index. The Jaccard index is a positive semidefinite kernel on finite sets (Gower, 1971), and thus so is κ.

[3] `https://github.com/autogluon/fev`

Specialist overlap. We compare two portfolios through the kernel-weighted product of their weight vectors,

$$\langle \bar{w}^A, \bar{w}^B \rangle_\kappa = \sum_i \sum_j \bar{w}_i^A \, \bar{w}_j^B \, \kappa(i,j), \tag{6}$$

which is the expected kernel value of two members, one drawn from each portfolio with probabilities given by the weights. Because κ is positive semidefinite, Eq. 6 is an inner product, and $\|\bar{w}^A\|_\kappa = \langle \bar{w}^A, \bar{w}^A \rangle_\kappa^{1/2}$ is its norm. Because $\|\bar{w}^A\|_\kappa < 1$ for any portfolio that spreads its weight over members with distinct dataset subsets, we report the cosine normalization, the *specialist overlap*

$$\Omega_{\text{specialist}}(A,B) = \frac{\langle \bar{w}^A, \bar{w}^B \rangle_\kappa}{\|\bar{w}^A\|_\kappa \, \|\bar{w}^B\|_\kappa} \in [0,1]. \tag{7}$$

The Cauchy–Schwarz inequality bounds it by 1, and the nonnegative kernel values and weights bound it by 0 from below. It equals 1 when the two portfolios are the same weighted grouping of the datasets, and 0 when every pair of weighted specialists has disjoint data and at most one portfolio weights the generalist.

Dataset overlap. The dataset mass of A on a dataset d is the total weight of the specialists that were post-trained on it,

$$v_d^A = \sum_{i \,:\, d \in \mathcal{D}_i^A} \bar{w}_i^A, \tag{8}$$

and the generalist adds no mass. The *dataset overlap* is the weighted Jaccard index (Ioffe, 2010) of the two mass vectors, the mass that the two portfolios share divided by the mass that they cover together,

$$\Omega_{\text{dataset}}(A,B) = \frac{\sum_{d \in \mathcal{D}} \min\left(v_d^A, \, v_d^B\right)}{\sum_{d \in \mathcal{D}} \max\left(v_d^A, \, v_d^B\right)} \in [0,1], \tag{9}$$

which reduces to the Jaccard index for binary vectors. The two measures are complementary. The dataset overlap depends on a portfolio only through its mass vector, so it does not see how the datasets are grouped into specialists: two portfolios that cover the same datasets with different partitions have $\Omega_{\text{dataset}} = 1$. The specialist overlap sees the grouping, but it is less sensitive to a dataset that only one portfolio covers.

Task-routing overlap. Both measures above use the task-average weights $\bar{w}^A$, which are the mean over tasks of the per-task weights $\boldsymbol{w}^{A,\tau}$. The *task-routing overlap* applies Eq. 7 to the per-task weights instead and averages the cosine over tasks,

$$\Omega_{\text{task}}(A,B) = \frac{1}{|\mathcal{B}|} \sum_{\tau \in \mathcal{B}} \frac{\langle \boldsymbol{w}^{A,\tau}, \boldsymbol{w}^{B,\tau} \rangle_\kappa}{\|\boldsymbol{w}^{A,\tau}\|_\kappa \, \|\boldsymbol{w}^{B,\tau}\|_\kappa} \in [0,1], \tag{10}$$

and it equals 1 only if the two portfolios deploy the same weighted members on every task.

D Additional results for the research questions

This appendix gives the additional results behind the five research questions of Section 4, with one subsection per question.

D.1 Auto-Chroma against heuristic portfolios on both corpora

This subsection complements RQ1 (Section 4.1) with the sensitivity of the heuristic portfolios and of the Auto-Chroma portfolio to the post-training corpus, in validation and on test. Specifically, for Chroma-4M and Chronos-2-Small, we construct the heuristic portfolios and run Auto-Chroma on the original + downsampled corpus as described in Appendix C.1, and we measure how the performance of each portfolio changes when the downsampled data enter the post-training corpus. The goal is to examine how sensitive each kind of portfolio is to the composition of the corpus: a heuristic portfolio keeps its fixed partition rule, so the added data are assigned to its specialists whether or not they help, whereas Auto-Chroma adaptively partitions the post-training data thanks to the evaluation feedback of each round. Figure 8 reports, for each portfolio and metric, the change

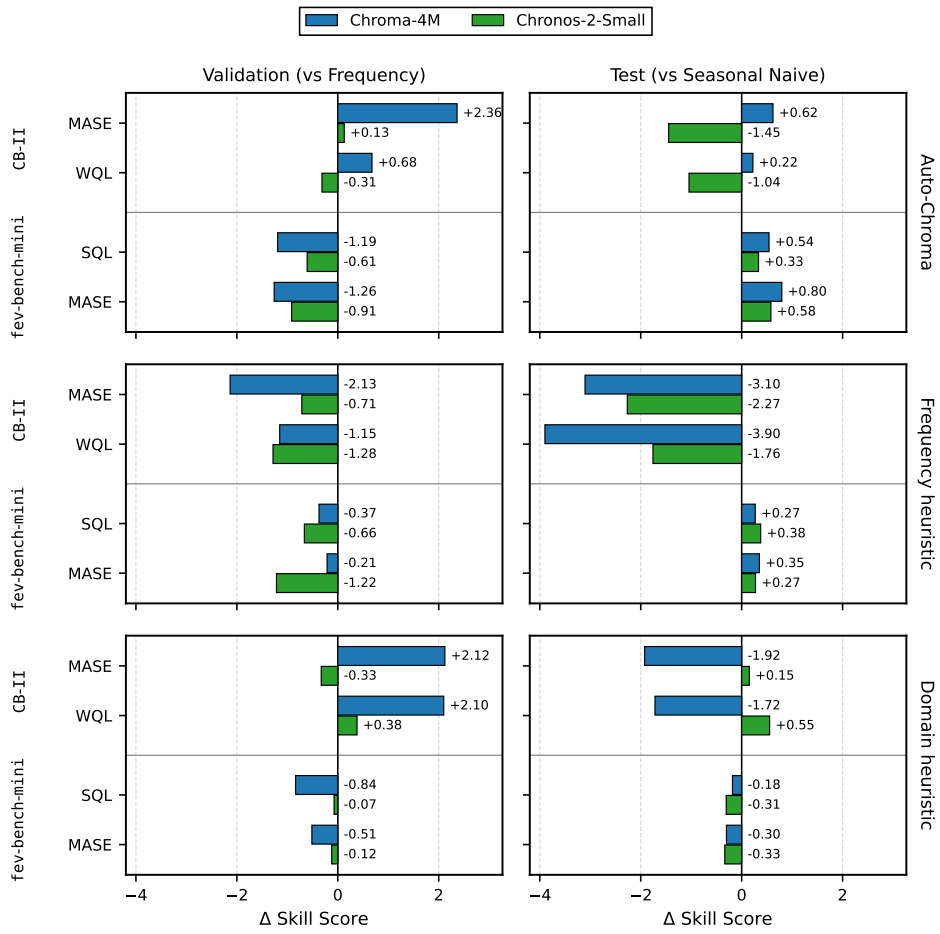

Figure 8: Skill score of a portfolio built on the original + downsampled corpus minus that of the same portfolio built on the original corpus, in validation (left) and on test (right), in percentage points, for the best Auto-Chroma portfolio (top), the frequency heuristic portfolio (middle), and the domain heuristic portfolio (bottom). A positive value means that the added data improved the portfolio.

in skill score from the original corpus to the original + downsampled corpus, so that a positive value means that the added data improved the portfolio.

Heuristic portfolios. On the original corpus, the domain heuristic is below the frequency heuristic in validation on every metric of both benchmarks, which is why the frequency heuristic serves as the reference portfolio. Adding the downsampled data mostly degrades the heuristic portfolios (Figure 8, middle and bottom). In validation, the frequency heuristic loses in all eight cases and the domain heuristic in five of eight, and no heuristic portfolio on the original + downsampled corpus ends more than a fraction of a point above the reference. On test, the same picture holds, and the benchmark sets the size of the drop (Figure 8, right). On `CB-II`, the frequency heuristic drops by up to 3.90 points of skill score and the domain heuristic of Chroma-4M by about 2, whereas on `fev-bench-mini` no heuristic portfolio moves by more than half a point. Figure 9 adds Chronos-Bolt-Tiny to the test comparison. Its heuristic portfolios follow the same trend: the frequency heuristic loses on `CB-II`, and on `fev-bench-mini` both heuristics lose about 1.3 points on both metrics. The few gains in either figure are of two kinds, and neither favors the larger corpus: the frequency heuristic gains under half a point on the test split of `fev-bench-mini`, and the domain heuristic gains up to about 2 points on `CB-II`, which only closes part of its gap to the frequency heuristic. In short, a fixed partition rule loses up to 3.90 points of skill score from the added data and never gains more than a fraction of a point over the frequency heuristic on the original corpus.

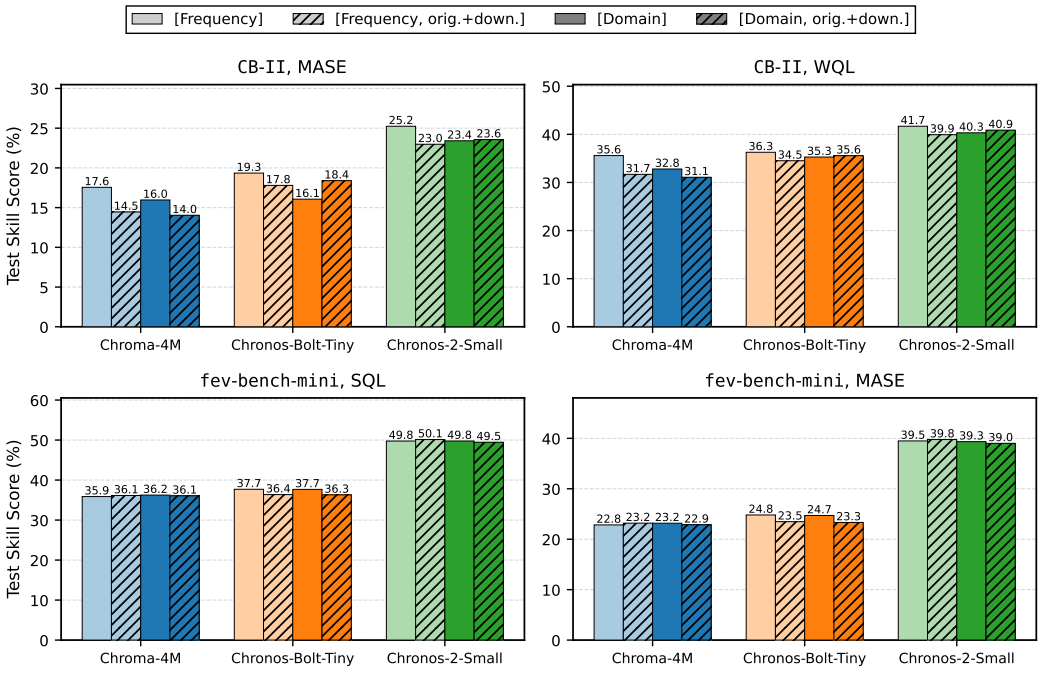

Figure 9: Test skill score against Seasonal Naive, in percent, of the frequency and domain heuristic portfolios of the three forecasting models, on both metrics of each benchmark. Colors mark the forecasting model (Chroma-4M, Chronos-Bolt-Tiny, Chronos-2-Small); within a model, light bars are the frequency heuristic and dark bars the domain heuristic. Hatched bars, tagged orig.+down., are the portfolios built on the original + downsampled corpus; the others are built on the original corpus.

Auto-Chroma. On either corpus, the Auto-Chroma portfolio ends above all four heuristic portfolios in validation (Figure 3), which supports RQ1. The added data changes its validation performance in both directions, by up to 2.36 points (Figure 8, top left), so the search does not consistently gain from the larger corpus. On test, the portfolios of the original corpus are the Auto-Chroma bars of RQ4 (Figure 6). With the added data, test performance improves in six of the eight comparisons, by up to 0.80 points, and drops in the two comparisons of Chronos-2-Small on `CB-II`, by up to 1.45 points (Figure 8, top right). Therefore, Auto-Chroma is far less sensitive to the added data than the heuristic portfolios.

D.2 Random and greedy search

This subsection describes the two LLM-free search strategies of RQ2 (Section 4.2) and repeats the comparison of Figure 4 for Chronos-2-Small (Figure 10), which shows the same contrast with random search, whereas greedy search matches Auto-Chroma late in the budget.

LLM-free search strategies. Random and greedy search replace the two agents of an Auto-Chroma round by a proposal rule, π_{rand} or π_{greedy} (Algorithms 5 and 6), and leave every other step of the round unchanged: the budget of $r_{\max} = 40$ candidates, the post-training and evaluation code, the objective, and the acceptance rule of Appendix C.1 with margin ξ and reference $P_{\text{ref}} = P^{\text{h}}_{\text{freq}}$ (Algorithm 4). The hyperparameter grids are $\mathcal{G}_s = \{250, 500, 1000, 2000\}$ and $\mathcal{G}_\eta = \{0.25, 0.5, 1, 2, 4\}\times$ the default learning rate of the model. Random search samples a new portfolio in every round: 4 to 10 specialists with distinct dataset subsets of 1 to 6 datasets, each with a step count from $\mathcal{G}_s$ and a learning rate from $\mathcal{G}_\eta$. Greedy search is a hill climb that starts from P_{ref} with the default step count and learning rate. In every round it applies one random mutation to the accepted candidate with the highest $S(P)$: move, add, or remove a dataset; split or merge specialists; add or remove a specialist; or halve or double the step count or the learning rate of a specialist.

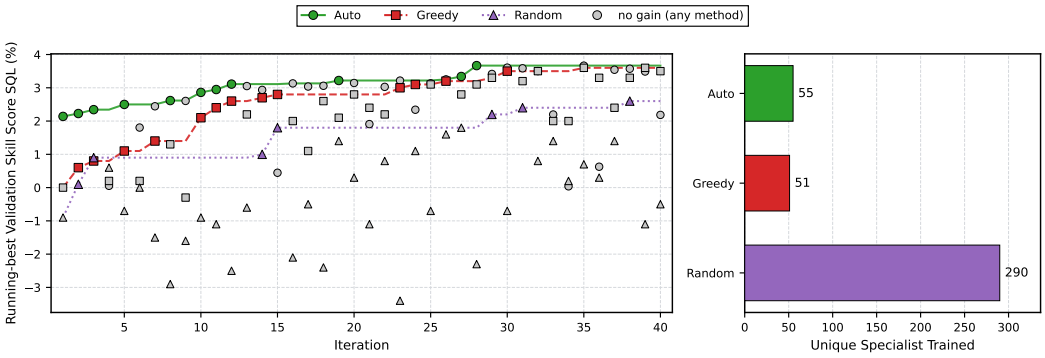

Figure 10: Auto-Chroma against greedy and random search for Chronos-2-Small on `fev-bench-mini`. Panels and markers follow Figure 4.

Algorithm 4 LLM-free search: the loop shared by random and greedy search

Require: proposal rule $\pi \in \{\pi_{\text{rand}}, \pi_{\text{greedy}}\}$; corpus $\mathcal{D}$; budget $r_{\max}$; margin ξ; reference portfolio P_{ref}; set $\mathcal{V}_0$ of the portfolios already evaluated for the forecasting model, with $P_{\text{ref}} \in \mathcal{V}_0$

$\mathcal{V} \leftarrow \mathcal{V}_0$
$P_{\text{lead}} \leftarrow P_{\text{ref}}$
$\beta_m \leftarrow 0$ for $m \in \mathcal{M}$ ▷ greedy search starts from P_{ref}
for $r = 1, \ldots, r_{\max}$ **do**
  $P_r \leftarrow \perp$ ▷ $\perp$: no candidate yet
  for up to 200 attempts **do** ▷ propose a portfolio that was never evaluated
    $Q \leftarrow \pi_{\text{rand}}()$ or $Q \leftarrow \pi_{\text{greedy}}(P_{\text{lead}})$ ▷ Algorithm 5 or 6
    if $Q \notin \mathcal{V}$ **then**
      $P_r \leftarrow Q$
      break
    end if
  end for
  if $P_r = \perp$ **then stop the search**
  end if
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{P_r\}$
  $\theta_k \leftarrow \text{FT}(\theta;\, c_k)$ for every $c_k \in P_r$
  $S_m\big(P_r \,\big|\, P_{\text{ref}}, \mathcal{B}^{\text{val}}\big)$ for $m \in \mathcal{M} \leftarrow$ Algorithm 2 on every task of $\mathcal{B}^{\text{val}}$
  if $S_m\big(P_r \,\big|\, P_{\text{ref}}, \mathcal{B}^{\text{val}}\big) > \beta_m + \xi$ for at least one $m \in \mathcal{M}$ **then** ▷ acceptance rule
    $\beta_m \leftarrow \max\big(\beta_m,\, S_m\big(P_r \,\big|\, P_{\text{ref}}, \mathcal{B}^{\text{val}}\big)\big)$ for every $m \in \mathcal{M}$
    if $S(P_r) > S(P_{\text{lead}})$ **then** ▷ used by greedy search only
      $P_{\text{lead}} \leftarrow P_r$
    end if
  end if
  record P_r, its skill scores, and whether it is accepted
end for
return the record of all candidates

Algorithm 5 π_{rand}: uniform sample of a portfolio

Require: corpus $\mathcal{D}$; step grid $\mathcal{G}_s$; learning-rate grid $\mathcal{G}_\eta$; cap $K_{\max}$

procedure DRAWSPECIALIST( )
  $u \sim \mathcal{U}\{1, \dots, \min(6, |\mathcal{D}|)\}$ ▷ size of the dataset subset
  $\mathcal{D}' \leftarrow$ uniform subset of $\mathcal{D}$ of size u
  $s \sim \mathcal{U}(\mathcal{G}_s)$; $\eta \sim \mathcal{U}(\mathcal{G}_\eta)$
  return $(\mathcal{D}', \eta, s)$
end procedure
procedure π_{rand}( )
  $K \sim \mathcal{U}\{4, \dots, K_{\max}\}$; $P \leftarrow \emptyset$
  while $|P| < K$ **do**
    $(\mathcal{D}', \eta, s) \leftarrow$ DRAWSPECIALIST( )
    if $\mathcal{D}' \neq \mathcal{D}_k$ for every $c_k \in P$ **then** ▷ dataset subsets are pairwise distinct
      $P \leftarrow P \cup \{(\mathcal{D}', \eta, s)\}$
    end if
  end while
  return P
end procedure

Algorithm 6 π_{greedy}: one random mutation of the leader

Require: leader P; corpus $\mathcal{D}$; step grid $\mathcal{G}_s$; learning-rate grid $\mathcal{G}_\eta$; cap $K_{\max}$

procedure $\pi_{\mathrm{greedy}}(P)$
  for each of the nine operators o, in a uniformly random order **do**
    $Q \leftarrow P$; draw $c_k = (\mathcal{D}_k, \eta_k, s_k)$ uniformly from Q
    if $o =$ MOVEDATASET **and** $|Q| \geq 2$ **and** $|\mathcal{D}_k| \geq 2$ **then**
      draw $c_{k'} \in Q \setminus \{c_k\}$ and $d \in \mathcal{D}_k$ uniformly
      $\mathcal{D}_k \leftarrow \mathcal{D}_k \setminus \{d\}$; $\mathcal{D}_{k'} \leftarrow \mathcal{D}_{k'} \cup \{d\}$
    else if $o =$ ADDDATASET **and** $\mathcal{D} \setminus \mathcal{D}_k \neq \emptyset$ **then**
      draw $d \in \mathcal{D} \setminus \mathcal{D}_k$ uniformly; $\mathcal{D}_k \leftarrow \mathcal{D}_k \cup \{d\}$
    else if $o =$ REMOVEDATASET **and** $|\mathcal{D}_k| \geq 2$ **then**
      draw $d \in \mathcal{D}_k$ uniformly; $\mathcal{D}_k \leftarrow \mathcal{D}_k \setminus \{d\}$
    else if $o =$ SPLIT **and** $|\mathcal{D}_k| \geq 2$ **and** $|Q| < K_{\max}$ **then**
      $u \sim \mathcal{U}\{1, \dots, |\mathcal{D}_k| - 1\}$; $\mathcal{D}' \leftarrow$ uniform subset of $\mathcal{D}_k$ of size u
      $\mathcal{D}_k \leftarrow \mathcal{D}_k \setminus \mathcal{D}'$; $Q \leftarrow Q \cup \{(\mathcal{D}', \eta_k, s_k)\}$ ▷ the new specialist inherits η_k, s_k
    else if $o =$ MERGE **and** $|Q| \geq 2$ **then**
      draw $c_{k'} \in Q \setminus \{c_k\}$ uniformly
      $\mathcal{D}_k \leftarrow \mathcal{D}_k \cup \mathcal{D}_{k'}$; $Q \leftarrow Q \setminus \{c_{k'}\}$ ▷ c_k keeps η_k, s_k
    else if $o =$ ADDSPECIALIST **and** $|Q| < K_{\max}$ **then**
      $Q \leftarrow Q \cup \{$DRAWSPECIALIST( )$\}$ ▷ Algorithm 5
    else if $o =$ REMOVESPECIALIST **and** $|Q| \geq 2$ **then**
      $Q \leftarrow Q \setminus \{c_k\}$
    else if $o =$ STEPS **then**
      $\gamma \sim \mathcal{U}\{\frac{1}{2}, 2\}$; $s_k \leftarrow \max(50, \lfloor \gamma s_k \rfloor)$
    else if $o =$ LEARNINGRATE **then**
      $\gamma \sim \mathcal{U}\{\frac{1}{2}, 2\}$; $\eta_k \leftarrow \min\big(\max \mathcal{G}_\eta,\ \max(\min \mathcal{G}_\eta,\ \gamma\eta_k)\big)$ ▷ clip to grid
    else
      continue ▷ the precondition of o fails
    end if
    if $Q \in \mathcal{P}$ **and** the dataset subsets of Q are pairwise distinct **and** $Q \neq P$ **then**
      return Q ▷ the first operator that yields a valid new portfolio
    end if
  end for
end procedure

D.3 SENSITIVITY TO THE ORCHESTRATOR LLM

Table 2 reports the best portfolio of a full run of 40 rounds with each of the three orchestrator LLMs of RQ3 (Section 4.3), under the same setting. The three LLMs differ in size and in architecture. DeepSeek V4 Flash is a mixture-of-experts (MoE) model with 284B total parameters, of which 13B are active per token (DeepSeek-AI, 2026). MiMo v2.5 is an MoE model with 310B total parameters, of which 15B are active per token (Xiaomi MiMo Team et al., 2026). Qwen3.5-27B is a dense model with 27B parameters, all of which are active (Qwen Team, 2026), so the two MoE models have about 10x more total parameters and about half as many active parameters. We observe that all LLMs reach approximately equal performance, except for MiMo v2.5 with Chroma-4M, whose validation skill score is about 1.5 points below the other two LLMs. Importantly, the partitions of these portfolios differ as discussed in Section 4.3.

Table 2: Best portfolios of three orchestrator LLMs on `fev-bench-mini`. Validation skill score against $P^{\mathrm{h}}_{\mathrm{freq}}$ and test skill score against Seasonal Naive, in percent.

		Validation (%)		Test (%)	
		SQL	MASE	SQL	MASE
Chroma-4M	DeepSeek V4 Flash (Preview)	5.41	5.24	36.16	23.16
	MiMo v2.5	3.87	3.53	37.00	24.25
	Qwen3.5-27B-FP8	5.36	5.17	36.44	23.60
Chronos-2-Small	DeepSeek V4 Flash (Preview)	3.67	3.74	49.24	38.63
	MiMo v2.5	3.47	3.34	49.37	38.89
	Qwen3.5-27B-FP8	4.20	4.09	49.41	38.85

Table 3: Reference models of the test leaderboards on `CB-II` and `fev-bench-mini`. The results of the reference models are taken from the public repository of `fev` (Shchur et al., 2025), with the exception of Chronos-2, which we re-evaluate with cross-learning disabled for a fair comparison with Chronos-2-Small. Seasonal Naive is the reference of the skill score on both benchmarks.

Reference model	Type	`CB-II`	`fev-bench-mini`
Chronos-2 (Ansari et al., 2025)	TSFM	✓	✓
Chronos-Large (Ansari et al., 2024)	TSFM	✓	
Chronos-Bolt-Base (Ansari et al., 2024)	TSFM	✓	✓
TimesFM-2.0 (Das et al., 2024)	TSFM	✓	
TimesFM-2.5 (Das et al., 2024)	TSFM		✓
Moirai-Large (Woo et al., 2024)	TSFM	✓	
Moirai-2.0 (Liu et al., 2025a)	TSFM		✓
TiRex (Auer et al., 2025b)	TSFM	✓	✓
Toto 1.0 (Cohen et al., 2025)	TSFM		✓
Sundial-Base (Liu et al., 2025b)	TSFM		✓
TimeCopilot (Garza & Rosillo, 2025)	LLM router	✓	
AutoARIMA (Hyndman & Khandakar, 2008)	Statistical	✓	✓
AutoTheta (Fiorucci et al., 2016)	Statistical	✓	✓
Seasonal Naive (Hyndman & Athanasopoulos, 2021)	Statistical	✓	✓

D.4 COMPARISON WITH REFERENCE MODELS ON BOTH METRICS

This subsection completes RQ4 (Section 4.4), which shows one metric per benchmark in Figure 6. Figures 11 and 12 give the test leaderboards on both metrics of each benchmark, for the three forecasting models, their frequency heuristic portfolios, their zero-shot generalists, and the reference models. Table 3 lists the reference models of each benchmark. We observe that Auto-Chroma can work universally well across settings, regardless of models, benchmarks, or metrics. On `CB-II` (Figure 11), every Auto-Chroma portfolio improves on its generalist on both metrics, and the gain grows as the model shrinks: about 3 points of MASE skill score for Chronos-2-Small,

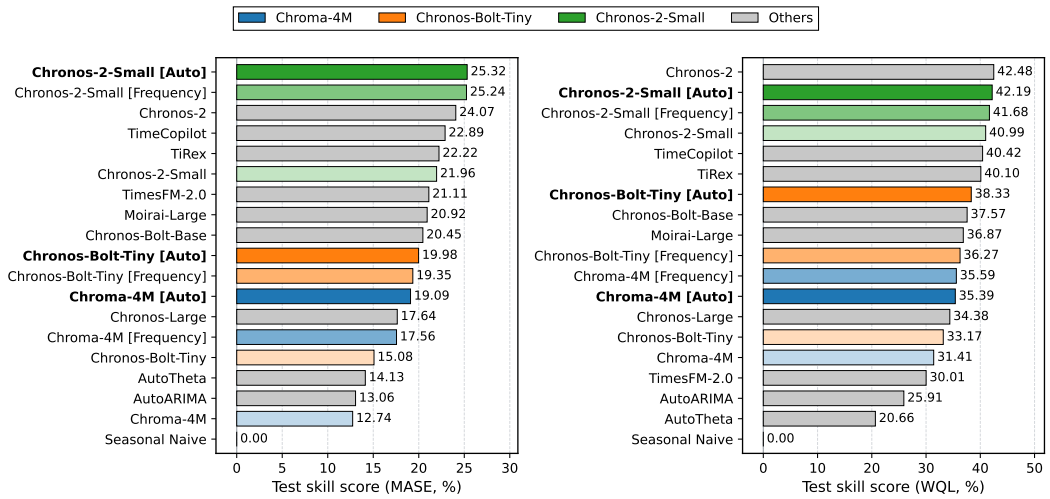

Figure 11: Test leaderboards on `CB-II` on both metrics (higher is better). For each model, [Auto] is the Auto-Chroma portfolio, [Frequency] is the frequency heuristic portfolio, and the bar without a tag is the zero-shot generalist.

5 for Chronos-Bolt-Tiny, and 6 for Chroma-4M. Each portfolio also passes reference models that are much larger than its members. Chronos-2-Small with Auto-Chroma is first on MASE, above Chronos-2, and second on WQL. On both metrics, Chronos-Bolt-Tiny with Auto-Chroma is above Chronos-Bolt-Base and Moirai-Large, and Chroma-4M with Auto-Chroma is above Chronos-Large. On `fev-bench-mini` (Figure 12), the differences are smaller. Chronos-Bolt-Tiny with Auto-Chroma gains about 2 points of SQL and 3 points of MASE skill score over its generalist, the largest gain on this benchmark, and it is the best of its three bars on both metrics. The three bars of Chroma-4M lie within one point of one another, and Chroma-4M with Auto-Chroma is still the best of them on both metrics. The three bars of Chronos-2-Small also lie within one point of one another, and together with Chronos-2 they form the top group of the leaderboard on both metrics, above TimesFM-2.5. Within this group, however, the Auto-Chroma portfolio is the lowest of the three Chronos-2-Small bars: it falls below both its frequency heuristic portfolio and its generalist, by less than one point of skill score on SQL as well as on MASE, which makes Chronos-2-Small on `fev-bench-mini` the one exception of RQ4. This case is examined in RQ5 (Section 4.5), which explains why the validation gain of this run does not carry over to the test split.

D.5 Task size and the reliability of the validation signal

This subsection completes RQ5 (Section 4.5), which reads Figure 7 on the full benchmarks. Here we break both benchmarks down by task size to locate the tasks on which the validation gain of a candidate portfolio carries over to the test split.

Setup. A task scores one forecast per series and per evaluation window, so we measure its size by its number of scored forecasts, the number of series times the number of windows. On `CB-II`, which has one window per task, this count ranges from 5 to about 100,000; on `fev-bench-mini`, which has several rolling windows per task, from 20 to about 11,000. We sort the tasks of each benchmark by this count, split them into three groups, and recompute both skill scores on each group, so that each panel of Figures 13 and 14 is restricted to one group. As in Figure 7, the validation skill score is measured against $P^{\mathrm{h}}_{\mathrm{freq}}$ and the horizontal line marks the test skill score of $P^{\mathrm{h}}_{\mathrm{freq}}$, so a candidate in the upper-right quadrant (shaded green) beats $P^{\mathrm{h}}_{\mathrm{freq}}$ on both splits. A group on which the validation gain carries over therefore places its candidates in this quadrant, whereas a group on which the validation signal misleads the search places them below the horizontal line, with a negative correlation.

`CB-II`. Figure 13 shows that the picture changes with the number of forecasts. On the 9 tasks with the fewest forecasts (5 to 111), almost every candidate of the run on the original corpus falls

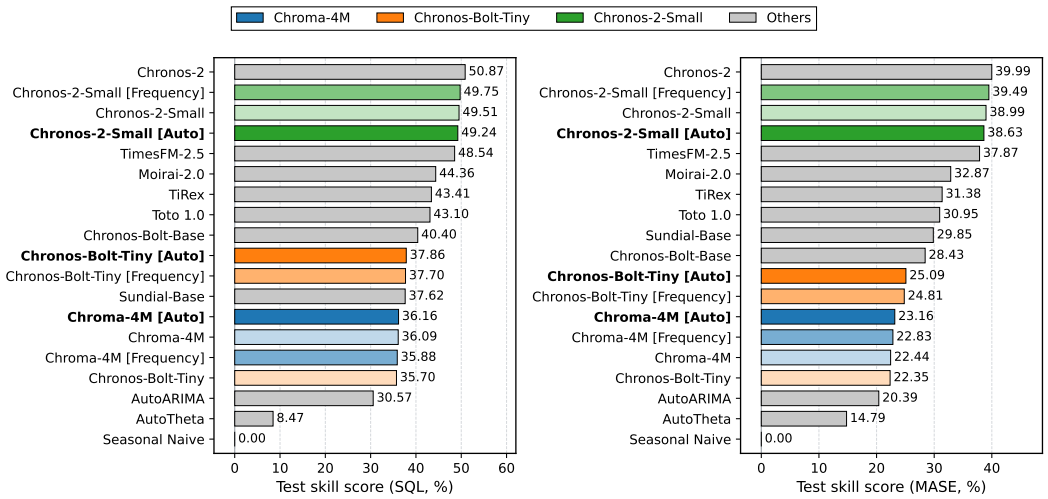

Figure 12: Test leaderboards on `fev-bench-mini` on both metrics (higher is better). For each model, [Auto] is the Auto-Chroma portfolio, [Frequency] is the frequency heuristic portfolio, and the bar without a tag is the zero-shot generalist.

below $P^{\mathrm{h}}_{\mathrm{freq}}$ on test on both metrics. On the 14 tasks of the middle group (181 to 3,010) and the 4 largest tasks (23,000 to about 100,000), every candidate but one is in the upper-right quadrant on both metrics, with correlations from about zero to moderately positive. The run on the original + downsampled corpus behaves the same way, but its loss on the 9 smallest tasks, 2 to 5 points below $P^{\mathrm{h}}_{\mathrm{freq}}$ on both metrics, is large enough to drag every candidate below $P^{\mathrm{h}}_{\mathrm{freq}}$ on the full test split in Figure 7.

`fev-bench-mini`. Figure 14 shows the same dependence, with both metrics in agreement, and what changes from left to right is the spread of the test skill scores. On the 8 tasks with the fewest forecasts (20 to 160), the candidates scatter over more than 3 points on both sides of $P^{\mathrm{h}}_{\mathrm{freq}}$, with the strongest negative correlation of the breakdown ($r = -0.75$ on both metrics for the run on the original corpus), so whether a candidate gains on this group is inconclusive. On the 6 tasks of the middle group (240 to 420), the spread is smaller but the candidates of both runs settle about one point below $P^{\mathrm{h}}_{\mathrm{freq}}$, so this group is still far from carrying the validation gain. On the 6 largest tasks (700 to about 11,000), each run forms a tight cluster with a span of less than one point and a correlation about zero, the most stable signal of the breakdown: the run on the original corpus gains less than one point in validation and lands within half a point of $P^{\mathrm{h}}_{\mathrm{freq}}$ on test, whereas the run on the original + downsampled corpus gains more in validation and keeps part of it, with almost every candidate in the upper-right quadrant. The validation gain itself shrinks with the number of forecasts, from a median of about 4.6 points over $P^{\mathrm{h}}_{\mathrm{freq}}$ on the smallest tasks to about 1 point on the largest.

Taken together, on both benchmarks, tasks with few scored forecasts produce widely spread test skill scores and a negative correlation between validation performance and test performance, whereas tasks with many scored forecasts produce tightly clustered test skill scores that preserve the validation gain. This is consistent with the view that a single-window evaluation needs enough forecasts to give a stable score, which more series or more rolling windows supply (Tashman, 2000). This limitation is thus a property of the benchmark rather than of the search: the tasks in question score at most a few hundred forecasts, whereas a practical deployment routinely evaluates a portfolio on thousands of series over many historical windows, which gives a stable holdout score of the kind that the largest tasks already show, and under which the negative correlation reported here would not be expected to arise.

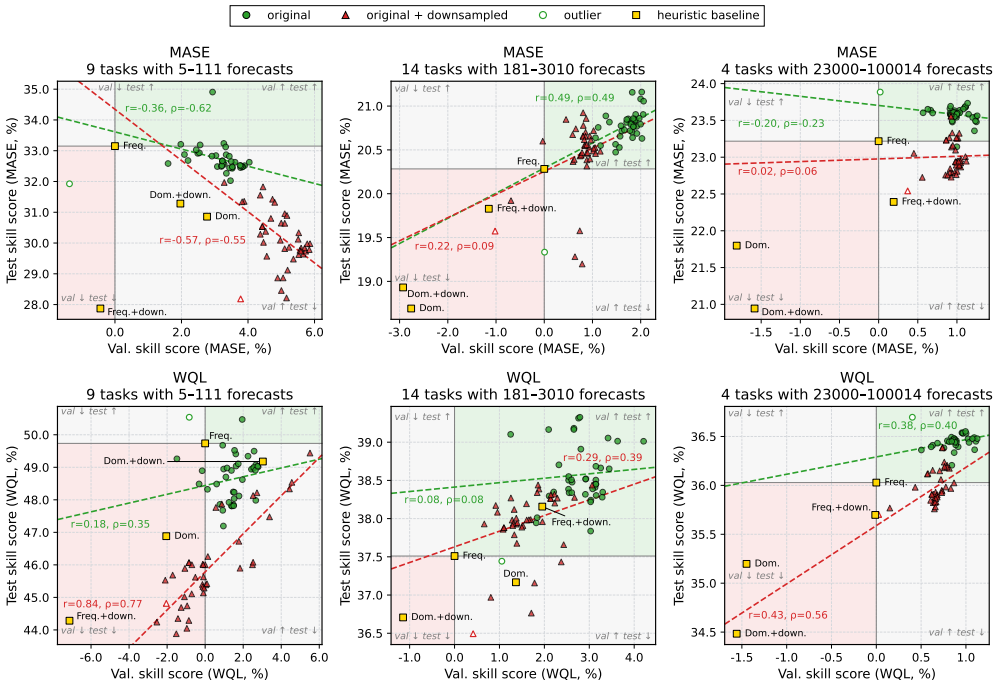

Figure 13: Validation performance against test performance on `CB-II` by task size, for MASE (top) and WQL (bottom). Columns are groups of tasks, from the fewest scored forecasts (left) to the most (right). Markers and fits follow Figure 7.

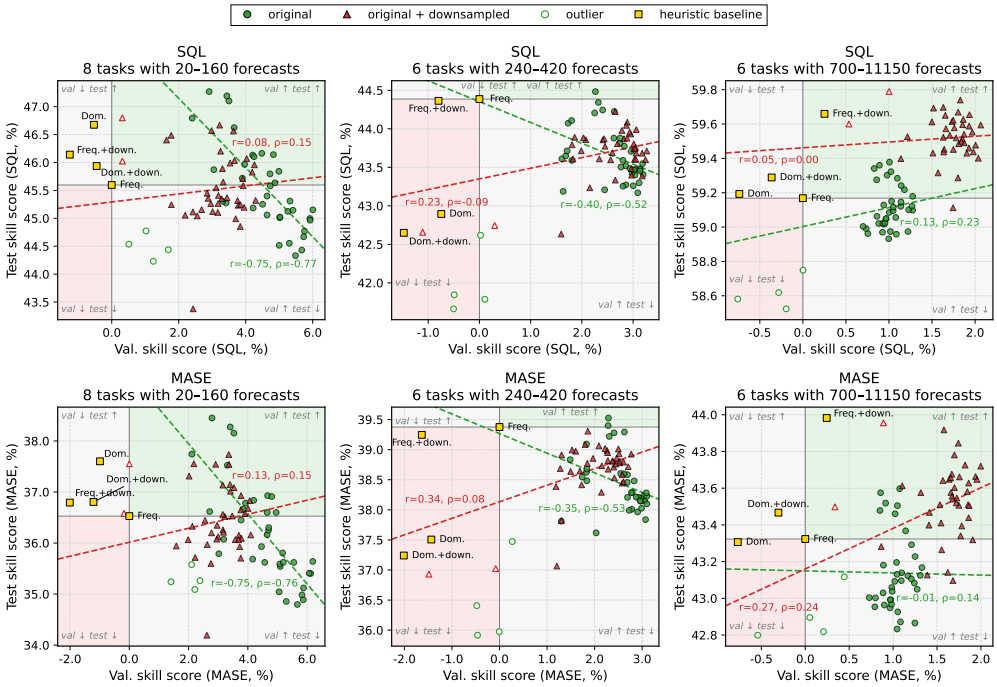

Figure 14: Same as Figure 13 for `fev-bench-mini`, for SQL (top) and MASE (bottom).

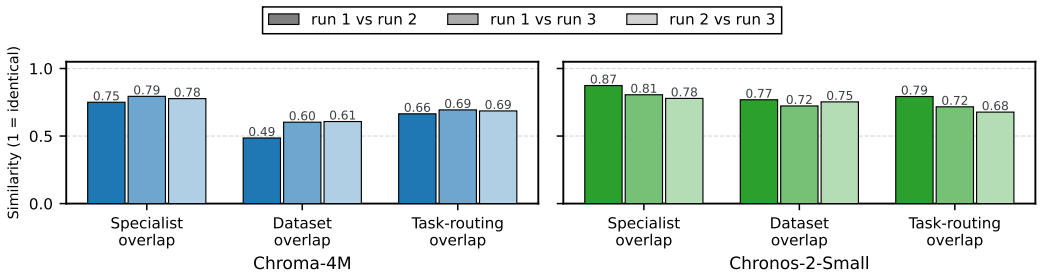

Figure 15: Pairwise similarity of the best portfolios of three repeated runs with the same orchestrator LLM on `fev-bench-mini`, for Chroma-4M (left) and Chronos-2-Small (right), in specialist ($\Omega_{\text{specialist}}$), dataset ($\Omega_{\text{dataset}}$), and task-routing ($\Omega_{\text{task}}$) overlap.

E SUPPLEMENTARY ANALYSES: SEARCH STABILITY, MEMORY UTILIZATION, ANALYST ABLATION, TRANSFER, AND COST

This appendix presents five supplementary analyses: (i) the stability of the search, assessed by the agreement among the portfolios discovered in independent runs, (ii) how the agents use the memory of past rounds, (iii) an ablation of the analyst stage, (iv) the transferability of a run's discoveries to benchmarks other than the one it was optimized on, and (v) the computational cost of the search.

E.1 SEARCH STABILITY ACROSS REPEATED RUNS

RQ3 (Section 4.3) varies the orchestrator LLM; here we hold the LLM fixed and ask how similar the portfolios of repeated runs are. We repeat the default run three times on `fev-bench-mini`, for Chroma-4M and for Chronos-2-Small, with DeepSeek V4 Flash Preview as the orchestrator LLM and an identical configuration, and we compare the best portfolios of each pair of runs on the three similarity metrics in Figure 15. Repeated runs find related but distinct portfolios: no pair is identical on any axis, and the overlaps range from 0.49 to 0.87. Portfolios found under the same orchestrator LLM tend, however, to be more similar to one another than those found under different orchestrator LLMs, on all three axes and for both models: the specialist overlap between repeated runs is 0.75 to 0.87, whereas across LLMs it is 0.60 to 0.78 (Figure 5). The separation is clear for Chronos-2-Small, where every pair of repeated runs exceeds every cross-LLM pair on all three axes, whereas for Chroma-4M the two closest LLMs yield portfolios about as similar as a repetition and the gap is driven by the third LLM. The choice of the orchestrator LLM therefore shifts the discovered portfolio at least as much as the stochasticity of a single run, but neither source of variation dominates the other.

E.2 MEMORY UTILIZATION

We examine whether the agents draw on the memory of past rounds (Section 3.2) or only on the latest outcome. Figure 16 traces every mention of an earlier candidate in the reports of the Chroma-4M run on `CB-II`. Both agents draw on the reports of earlier rounds throughout the run, and accepted candidates are cited far more often than rejected ones: they receive about 70% of all strategist mentions of earlier candidates although they account for 13 of the 40 rounds. This suggests that the agents use the memory selectively and build their proposals on the portfolios that improved the objective.

E.3 ANALYST ABLATION

Table 4 compares the loop with and without the analyst stage, with the default orchestrator LLM in both variants. The variant without the analyst omits the call π_{A} in every round, so that no analyst report is ever written. The strategist instead operates under a prompt adapted to this setting, which directs it to the raw files and its own reports of previous rounds. The two variants perform com-

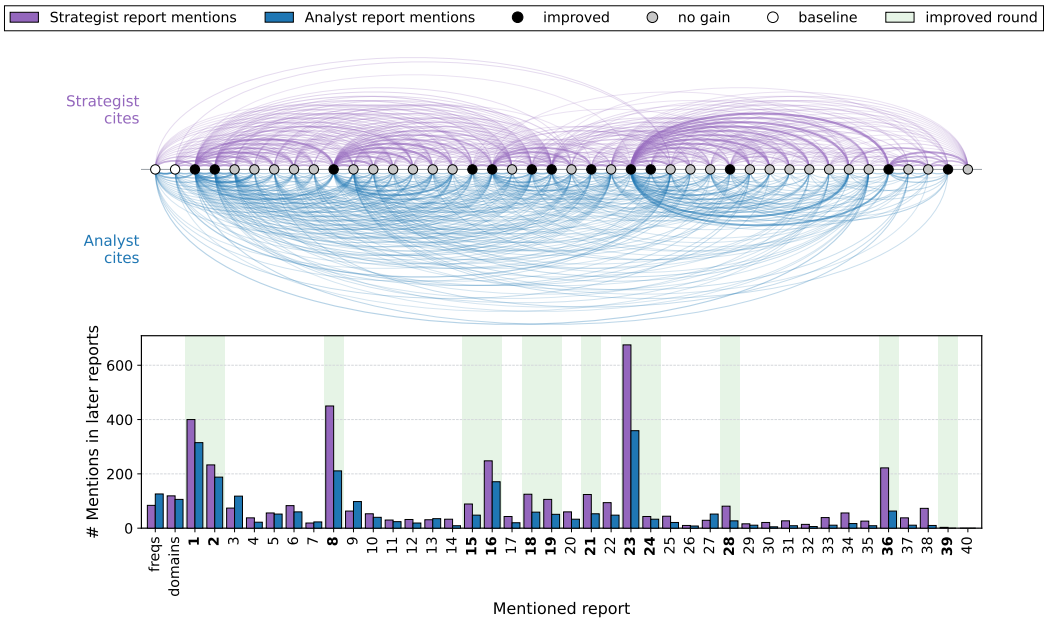

Figure 16: Mentions of earlier candidates in the reports of the Chroma-4M run on `CB-II`. Top: candidates in round order, with an arc from a candidate to every later report that mentions it, by the strategist (above) or the analyst (below); arc width and opacity grow with the number of mentions. Bottom: the number of mentions of each candidate in all later reports, by agent. Filled markers and shaded columns mark accepted candidates.

Table 4: Auto-Chroma with and without the analyst stage. Validation skill score against $P^{\mathrm{h}}_{\mathrm{freq}}$ and test skill score against Seasonal Naive, in percent.

		Validation (%)		Test (%)	
`CB-II`		MASE	WQL	MASE	WQL
Chroma-4M	With analyst	4.54	5.34	19.09	35.39
	Without analyst	5.99	5.31	20.06	37.40
Chronos-2-Small	With analyst	2.06	3.17	25.32	42.19
	Without analyst	2.33	3.61	24.95	42.74
`fev-bench-mini`		SQL	MASE	SQL	MASE
Chroma-4M	With analyst	5.41	5.24	36.16	23.16
	Without analyst	4.29	4.12	35.85	22.64
Chronos-2-Small	With analyst	3.67	3.74	49.24	38.63
	Without analyst	3.86	3.85	49.04	38.36

parably: their validation and test skill scores differ by about one point or less in every cell except Chroma-4M on `CB-II`, where the variant without the analyst leads by about 1.5 points in validation MASE and about 2 points in test WQL, and neither variant leads consistently across the remaining cells. We nevertheless adopt the variant with the analyst throughout the paper, for two reasons. First, the analyst reports make each round interpretable: every report verifies the hypotheses of the preceding strategist report against the measured outcome and records the result in the database, where every later round can read it (Box G.2). Second, these reports become part of the memory that both agents cite throughout the run (Appendix E.2). Retaining the analyst does not make the loop expensive: Appendix E.5 shows that the LLM cost of a run remains low, since the reports are served largely from cached input tokens.

Table 5: Test skill score against Seasonal Naive, in percent, on each benchmark, of the portfolio searched on that benchmark and of the portfolio transferred from the other benchmark, with its specialists fixed and its ensemble weights refitted on the target validation split.

	Evaluated on	CB-II		fev-bench-mini	
	Portfolio	MASE	WQL	SQL	MASE
Chroma-4M	Searched on target	19.09	35.39	36.16	23.16
	Transferred from other	18.55	36.04	36.62	23.66
Chronos-2-Small	Searched on target	25.32	42.19	49.24	38.63
	Transferred from other	25.16	42.09	49.39	38.78

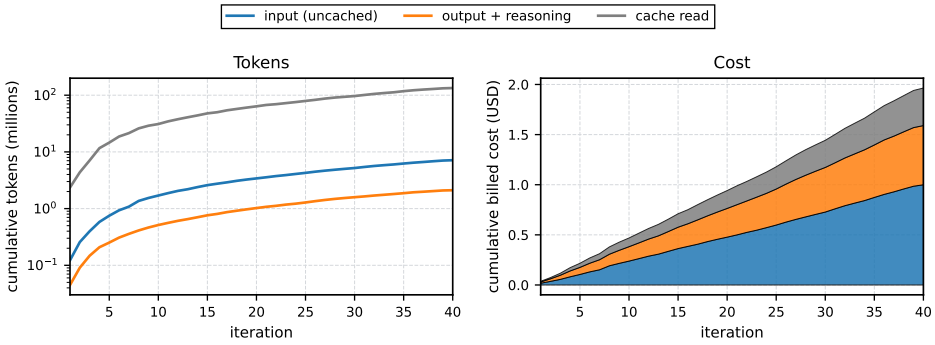

Figure 17: Cumulative LLM tokens by field (left, in millions, log scale) and cumulative billed cost in USD (right) of the Chroma-4M run on CB-II.

E.4 Transfer across benchmarks

We evaluate the best portfolio of each run on the benchmark it was not searched on. Its specialists are kept fixed, and only the ensemble weights are refitted on the validation split of the target benchmark. Table 5 reports, for each model and each benchmark, the test skill scores of the portfolio searched on that benchmark and of the portfolio transferred from the other benchmark. The two rows of each block are on par: they differ by less than one point on both metrics, and the transferred portfolio scores higher in five of the eight comparisons. Hence, a portfolio found on one benchmark transfers to another with no loss, although its specialists were never selected for the target tasks. This suggests that the benchmark-specific part of a portfolio lies in the ensemble weights alone, which are fitted on the validation split of the target at no training cost, whereas the specialists themselves carry over to new settings.

E.5 Computational cost and token usage

Figure 17 shows how the LLM tokens and the billed cost accumulate over the 40 rounds of the Chroma-4M run on CB-II, where output tokens include reasoning tokens. The run uses 7.1M input, 2.1M output, and 133.8M cache-read tokens, and it costs USD 1.96 at the list rate of DeepSeek V4 Flash Preview: $0.14 per million uncached input tokens, $0.28 per million output tokens, and $0.0028 per million cache-read tokens. The cost grows almost linearly with the number of rounds, which indicates that the agents read a bounded set of reports in each round rather than the full history.

F Training datasets

Table 6 lists the 44 datasets of the original corpus $\mathcal{D}$. Chroma (Kayaalp et al., 2026, Appendix C) post-trains its specialists on the Chronos training corpus (Ansari et al., 2024) together with additional datasets from the Monash Time Series Forecasting Archive (Godahewa et al., 2021) and the UCI Machine Learning Repository (Kelly et al., 2023). The first three groups of the table reuse these

sources, after we exclude the datasets suspected of overlap with the evaluation benchmarks and those not publicly available. We download the Monash datasets from their original Zenodo records and re-derive the two UCI retail datasets from the original files. We additionally add the nine multivariate datasets that we introduce for the covariate regime of Chronos-2-Small.

Table 6: Training datasets of the original corpus, grouped by the repository from which we obtain them.

Dataset	Domain	Frequency
Chronos training corpus (Ansari et al., 2024)		
m4_daily	various	1D
ushcn_daily	weather	1D
weatherbench_daily	weather	1D
wiki_daily_100k	web	1D
m4_hourly	various	1H
mexico_city_bikes	transport	1H
solar_1h	energy	1H
taxi_1h	transport	1H
uber_tlc_hourly	transport	1H
electricity_15min	energy	15min
solar	energy	5min
taxi_30min	transport	30min
m4_monthly	various	1M
m4_weekly	various	1W
Monash Time Series Forecasting Archive (Godahewa et al., 2021)		
bitcoin	various	1D
bitcoin_without_missing_values	various	1D
covid_mobility	transport	1D
covid_mobility_without_missing_values	transport	1D
sunspot	weather	1D
sunspot_without_missing_values	weather	1D
temperature_rain	weather	1D
us_births	various	1D
electricity_hourly	energy	1H
kdd_cup_2018	weather	1H
kdd_cup_2018_without_missing_values	weather	1H
pedestrian_counts	transport	1H
rideshare	transport	1H
aus_wind_1min	energy	1min
london_smart_meters	energy	30min
solar_10_minutes	energy	10min
solar_4_seconds	energy	4s
wind_4_seconds	weather	4s
UCI Machine Learning Repository (Kelly et al., 2023)		
online_retail_I	retail	1D
online_retail_II	retail	1D
Kaggle		
heart_rate (Goldberger & Rigney, 1991)	various	0.5s
bdg2 (Miller et al., 2020)	energy	1H
Other public repositories (Zenodo, GitHub, Harvard Dataverse, Dryad)		
smd (Su et al., 2019)	various	1min
camels (Newman et al., 2022; Addor et al., 2017)	weather	1D

Dataset	Domain	Frequency
`tep` (Rieth et al., 2017)	various	3min
`french_piezo` (Mbouopda et al., 2022)	weather	1D
`hill_of_towie` (Clerc & Lingkan, 2026)	energy	10min
`greek_wind_farm` (Ntafalias et al., 2024)	energy	10min
`smarteole` (Duc & Simley, 2022)	energy	1min
`mesa_del_sol` (Bashir et al., 2023)	energy	10s

G IMPLEMENTATION DETAILS

This appendix gives further details on the implementation of Auto-Chroma: the harness that runs the loop of Section 3.3, the report scaffolds that the harness fills before each agent call, and excerpts of the agent instructions and of the reports they produce.

Harness. The objective in Eq. 2 is a search over dataset partitions, and every object the search manipulates is a file: a specialist is a configuration file, a portfolio is a directory of such files, and the database of Section 3.2 is a directory of reports and result tables. Consequently, an agent needs no capability beyond reading, writing, and creating files on persistent storage. We therefore build on OpenCode[4], an open-source agent harness that already provides file tools, per-path permissions, and isolated multi-agent sessions, and we add only a thin layer on top of it, which is the orchestrator of Section 3. The orchestrator is a small deterministic Python script that drives the three phases of each round: it prepares the scaffold of a report, starts a fresh OpenCode session with the corresponding instruction, and, once the session returns, checks the output before the next phase may start. For the analyst, the check verifies that every required section of the report is filled. For the strategist, it verifies that exactly one portfolio was written, that its size respects $K_{\max}$, that every dataset name exists in $\mathcal{D}$ and does not appear in the benchmark, that the training command is well formed, and that the hyperparameters in the command match those stated in the report. A failed check re-prompts the same session with the specific violation, up to a fixed number of attempts, after which the round is aborted. Write access is scoped per phase: the analyst may edit only its own report, and the strategist only its own report and the configuration of its new portfolio. When a phase succeeds, the orchestrator sets these files to read-only, so that past entries of the database can be read but never modified, as stated in Section 3.2; the specialist checkpoints, the evaluation caches, the raw data, and the source code are inaccessible to both agents throughout.

Report scaffolds. In addition to validation, the orchestrator pre-fills the report of each agent with every quantity that can be computed from the database. Specifically, before the analyst call, it assembles the leaderboard, the per-task differences of the latest portfolio against the strongest competing portfolio on both metrics, the worst and best tasks with the competing portfolios and ensemble weights on those tasks, and a summary of every specialist of the latest portfolio; before the strategist call, it assembles the leaderboard. The scaffold thus fixes a uniform report structure across rounds and runs, and it separates the two sources of content: the orchestrator supplies every number, and the LLM supplies only the words, namely the annotations of the pre-filled tables, the diagnosis, the hypotheses, and the design rationale. Consequently, every number in a report is computed by the evaluation code rather than transcribed by the LLM, and the agent devotes its context to reasoning over the evidence and to writing the report. Table 7 lists the sections of the two reports and who fills each one.

Excerpts. Boxes G.1 to G.4 give excerpts of the two instruction files and of the two reports of round 3 of the Chroma-4M run on `CB-II`, each report placed after the instruction that produced it. The text is verbatim, with omissions marked [. . .]. The complete instructions are included in the code release.

[4] `https://opencode.ai`

Table 7: Sections of the analyst and strategist reports and who fills each one. The leaderboard and the task diff cover both metrics; the LLM adds a `notes` column to every pre-filled table of a numbered section. A dash marks no contribution.

Section	Pre-filled by the orchestrator	Written by the LLM
	Analyst report	
1. Leaderboard	top-10 portfolios: win rate, skill score	notes
2. Task diff	per-task deltas vs. the strongest competing portfolio, split by sign	notes
3. Worst tasks	best 5 portfolios on each of the 5 largest regressions	notes
4. Best tasks	routing row $\boldsymbol{w}^\tau$ and average $\bar{w}_k$ on the 5 largest improvements	which member carries the gain
Specialist summary	hash, $\mathcal{D}_k$, s_k, η_k per member	—
5. Training diagnostics	—	final loss and convergence of each specialist
6. Pattern analysis	—	win/loss patterns by task frequency, domain, and partitioning
7. Opportunity map	—	which portfolios solve each regression; which member carries each gain
8. Hypothesis validation	—	each strategist claim: confirmed, partial, or refuted
Conclusion	—	strengths, weaknesses, key specialists, hypothesis outcomes
	Strategist report	
Leaderboard	top-10 portfolios: win rate, skill score	notes
1. Lessons	—	one bullet per analyst report read
2. Design rationale	—	one design change with cited evidence; target tasks; specialist outline
3. Specialist definitions	specialist hashes	$\mathcal{D}_k$, s_k, η_k with reasons; training command
4. Predicted outcome	—	per-task gains tied to specialists

Box G.1: Analyst instruction (π_A) INSTRUCTION

You are a diagnostic analyst for a time series forecasting portfolio optimization project.

Your job is to examine evaluation results and produce a factual diagnosis: what happened, why, and where the opportunities are. You do NOT create portfolios, run training, or prescribe what to build next – you only diagnose.

[...]

Your primary reference point is the reference portfolio (A) – the best-scoring portfolio excluding the one being analyzed [...]

Mandatory procedure

[...]

Step 1: Annotate the pre-filled tables

[...] fill in the `notes` column for every row in the **Leaderboard**, **Task Diff**, **Worst Tasks**, and **Best Tasks – Specialist Importance** sections [...]

Step 2: Inspect training logs → write Section 5

[...] **Immediately edit Section 5 (Training Diagnostics)** in the report with per-specialist loss curve observations. For each specialist, record [...]

- the final loss value;
- the loss trajectory at the end of training: *plateaued*, *still decreasing*, *unstable*, or *spiked / NaN*;
- a rough **headroom estimate** – the relative loss drop over the last ˜10 % of training [...]

Step 3: Synthesize fleet-wide patterns → write Section 6

[...] **Scope discipline for this section: fleet-wide regularities only – what the leaderboard as a whole says about how to win.** [...]

Frequency-based patterns: Which dataset frequencies correlate with wins or losses *across the fleet*? Cite portfolios and frequency groups from the Leaderboard and Worst Tasks tables.

Domain-based patterns: Which domains show consistent wins or losses? Cross-reference with specialist training data via the Specialist Summary. Again, aggregate across tasks – do not list each task individually.

Grouping approach comparison: How do the grouping strategies visible on the Leaderboard (e.g. frequency-grouped, domain-grouped, mixed, baseline) compare? Which strategy is currently winning overall, and on what kinds of tasks does each strategy tend to do better or worse? [...]

Step 4: Map per-task opportunities and strengths → write Section 7

[...] Scope discipline: this is the analyst's *map of possibilities*, not a plan. Every bullet is an observation, not a recommendation. Do not write "should", "try", "add a specialist with ...", or rank candidate designs – only lay out opportunities and constraints that are visible in the data already in this report. The strategist decides what to do about them. [...]

Step 5: Validate strategist hypotheses → write Section 8

[...] you **must** read that file. Focus on two sections:

- **Section 2 (Design Rationale)** – the hypotheses motivating the design.
- **Section 4 (Predicted Outcome)** – the per-task / per-specialist predictions made before training.

Compare each claim against the actual results you have already written. [...] Then write **Hypothesis Validation** with the following sub-sections:

- **Per-specialist outcomes.** [...]
- **Design Rationale and Predicted Outcome.** [...]

Step 6: Synthesize → write Conclusion

Write the **Conclusion** section at the very end of the report. This section is the single most-consumed piece of the analyst's output – future iterations may see *only* this section (rather than the full report) when reviewing past experiments. Write it with that in mind. [...]

Constraints

- [...] **Never speculate about data you haven't read. Every factual claim must point to a specific source you actually read** – a file path, table, or row reference. [...]

Box G.2: Analyst report excerpt REPORT

1. Leaderboard Summary

MASE

model_name	win_rate	skill_score	notes
chroma_4m/exp_003	0.750	0.027	current best on MASE – new leader (+0.001 vs exp_002)
chroma_4m/exp_002	0.685	0.026	close second, +0.001 behind leader

[...]

2. Task Diff

[...]

Regressions

task_name	MASE_delta	WQL_delta	notes
ercot	0.028	0.002	biggest regression – hourly task; same hourly specialist as exp_001, ensemble shift
monash_cif_2016	0.015	0.011	monthly task; also WQL regressed; monthly specialist weight may have shifted

[...]

4. Best Tasks – Specialist Importance

[...]

task	generalist	d9540512	0f24241b	31050a4d	[...]	notes
[...]						
exchange_rate	0.000	0.660	0.010	0.330	[...]	d9540512 (weather) 0.660 + 31050a4d (web) 0.330 – matches predicted split exactly
monash_m1_yearly	0.000	0.000	1.000	0.000	[...]	0f24241b (various) carries solo at 1.000
average	0.139	0.051	0.163	0.026	[...]	highest avg: 625322f7 (monthly) 0.247, 0e53cf87 (energy) 0.178, 0f24241b (various) 0.163

[...]

Conclusion

- **Strengths:** exp_003 achieves the highest MASE skill in the fleet (0.027, +0.001 over exp_002, +0.011 over exp_001, §1 Leaderboard). The largest improvements over exp_001 are on yearly tasks: monash_m1_yearly (−0.414 MASE), exchange_rate (−0.396), and monash_m3_yearly (−0.103) – driven by the various (0f24241b) and weather+web (d9540512+31050a4d) specialists (§2 Improvements, §7.2). MASE win rate reaches 0.750, also the fleet's highest.
- **Weaknesses:** WQL skill (0.015) partially recovers from exp_002's 0.011 but remains 0.013 below exp_001's 0.028, failing the strategist's primary target (§1). The 1500-step monthly specialist change recovered only 0.004 of the 0.017 WQL gap. 6 tasks regressed from exp_001 (§2 Regressions), with ercot (−0.028) and monash_cif_2016 (−0.015) the largest – same specialists as exp_001 but ensemble composition shifted. The various specialist (0f24241b) shows catastrophic training instability (318.95 spike, §5) vs the stable version in exp_002 – the data refinement may have backfired.
- **Key specialists:**
  - 0f24241b (06_various.yaml) – strong on monash_m1_yearly (1.000), monash_m3_yearly (0.762), m4_yearly (0.519), monash_m1_quarterly (0.500); training: 1000 steps @ 0.001; loss catastrophically unstable – spike to 318.95 at step 925, final 28.83.
  - 625322f7 (05_monthly.yaml) – strong on monash_tourism_monthly (0.953), monash_tourism_quarterly (0.826), monash_m1_monthly (0.950); training: 1500 steps @ 0.001 on m4_monthly; loss noisy plateaued at 35.31 no headroom.
  - 0e53cf87 (07_energy_crossfreq.yaml) – strong on monash_australian_electricity (0.989), ETTm (0.616), monash_cif_2016 (0.529); training: 1500 steps @ 0.0005; loss unstable – spike to 120.29 at step 1450, final 24.35.
  - d9540512 (09_weather.yaml) + 31050a4d (10_web.yaml) – strong on exchange_rate (0.660 + 0.330); training: both 1000 steps @ 0.001; both fully plateaued (25.63 and 27.85).
- **Hypothesis outcomes:** confirmed: daily, weekly, retail, weather, web specialists preserved performance, various specialist met task-level predictions for yearly tasks, MASE skill within predicted range; partially confirmed: hourly specialist mixed (ETTh improved, ercot regressed), energy specialist lost some yearly proxy role, various specialist refinement may have caused training instability; refuted: WQL skill below predicted range (0.015 vs 0.020–0.032), 1500-step monthly failed to recover predicted tourism_yearly gain (−0.004 actual vs −0.05 to −0.10 predicted), monthly specialist weight diluted (0.247 vs exp_001's 0.310), tourism_quarterly lost exp_002's gain, ercot cif_2016 m3_monthly hospital regressed instead of improving, "low risk of regression" assumption refuted for third consecutive portfolio.

Box G.3: Strategist instruction (π_S) INSTRUCTION

You are a portfolio designer for a time series forecasting portfolio optimization project.

Objective

Design a portfolio that **improves either skill score by more than 0.001** over the current leader (highest sum of the two skill scores [...]). Either axis is a valid win. Gains below 0.001 on **both** axes are evaluation noise (`no_gain`). Design ambitiously: broad coverage across all benchmark tasks, not narrow wins. [...]

Your role

Translate the analyst's diagnostic findings into a concrete portfolio design: dataset groupings, YAML files, and hyperparameters. You own all design decisions. [...]

Mandatory procedure

[...] **CRITICAL: Think before you build.** Steps 1–4 are the *design phase* – you must complete them (read all data, review experiments, write Sections 1 and 2 of the report) **before** creating any YAML files or portfolio artifacts. [...]

Step 1: Read foundational state

[...]

Step 2: Review past experiments + form design hypothesis

- [...] Read the analyst reports for the strongest portfolios you noted in step 1. [...]

Step 3: Annotate leaderboard + write Section 1 (Lessons)

- [...] **Write Section 1 (Lessons from Past Experiments).** Write **one bullet per analyst report you read** in Step 2 – not a curated subset. [...] Each bullet must contain: portfolio name (backticked), what it tried (one phrase), the outcome (`improved` / `no_gain` / failed status [...]), and the design takeaway you are carrying into this iteration. [...]

Step 4: Write Section 2 (Design Rationale) – the portfolio blueprint

[...] **Primary axis decision.** Before writing, choose **one** primary axis for this iteration. This is the single most impactful lever given the current state of evidence. [...]

- **First line** immediately after `## 2. Design Rationale` MUST be a single-sentence summary prefixed with `Rationale:` [...]. This line is parsed automatically [...]

Step 5: Create portfolio, validate

Implement the blueprint you wrote in Section 2.

[...]

- Ensure the portfolio has **no more than 10 YAML files** (10 specialists). [...]
- Choose `--max-steps` and `--learning-rate` for each specialist. [...] Consider: smaller datasets may need fewer steps to avoid overfitting, while larger or more complex datasets may benefit from more steps. [...]

Step 6: Write Section 3 (Specialist Definitions)

A specialist is uniquely defined by the triple (datasets, `learning_rate`, `max_steps`). [...]

Step 7: Write Section 4 (Predicted Outcome)

Which specific tasks should improve and by roughly how much. Tie each prediction to a specialist from Section 3 [...]. Both evaluation metrics are valid win axes. [...]

Constraints

- [...] **Maximum 10 specialists per portfolio.** Each portfolio may contain at most 10 YAML files (i.e. 10 specialists). [...]
- **Never modify existing portfolios.** Only create and edit the new portfolio you are proposing. [...]

Design moves

Pick **one** primary move per iteration. [...]

Group composition

Restructure which datasets are grouped together into specialists. Cherry-pick strong specialists from *any* past portfolio (not just the leader) and combine them with new groupings targeting weak tasks. [...]

Per-specialist hyperparameter sweep

Vary `max_steps and/or learning_rate` on specialists from a chosen base portfolio (any `improved` portfolio, not necessarily the leader). [...]

Specialist refinement

Take a load-bearing specialist [...] and either (a) extend its training, (b) narrow its dataset mix to drop the noisiest contributors, or (c) split it into two more focused specialists. [...]

Creative grouping seeds

Unconventional groupings that break away from the patterns established by prior experiments. [...]

Box G.4: Strategist report excerpt REPORT

1. Lessons from Past Experiments

- **`base_freqs`** (baseline): Frequency-grouped specialists (daily, hourly, subhourly, weekly, monthly) beat domain grouping by +0.041 MASE / +0.040 WQL (§1). **Takeaway:** Frequency alignment is the dominant axis; carry forward all frequency specialists unchanged.
- **`base_domains`** (`no_gain`): Domain grouping lost broadly (−0.041 MASE, −0.040 WQL) but won selectively on `exchange_rate` (weather+web combo at 1.0 weight, §4) and `m1_yearly` (various specialist at 1.0 weight, §4). **Takeaway:** These two specialist designs (weather+web, various) are proven winners for specific tasks – borrow them.
- **`exp_001`** (improved, sum=0.044, leader): Added energy cross-freq (+0.028 WQL, strongest single-task gain on `australian_electricity` −0.251), multi-freq (failed – 0.000 weight on most tasks due to training spike to 96.6, §5), monthly at 1500 steps (carried `tourism_yearly` at 0.830 weight, §4). Regressed on `exchange_rate` (+0.271), `tourism_quarterly` (+0.214), `tourism_monthly` (+0.071), `m1_yearly` (+0.070). **Takeaway:** Energy cross-freq and 1500-step monthly are key to WQL; multi-freq design is stable only at 1000 steps (`5bb7d3c3` config). [...]

2. Design Rationale

Rationale: Combine `exp_001`'s 1500-step monthly (key WQL driver for `tourism_yearly`) with `exp_002`'s various+weather+web fixes (key MASE drivers) – the single change is restoring the 1500-step monthly to recover WQL while preserving all `exp_002` MASE gains.

Primary axis and evidence: Specialist design – specifically, restoring the 1500-step monthly specialist (`625322f7`) that was the main WQL carrier in `exp_001`. Evidence: (a) `exp_002` §2 Regressions shows `monash_tourism_yearly` regressed +0.155 MASE after reversion to 1000 steps; (b) `exp_001` §4 shows monthly specialist carried `tourism_yearly` at 0.830 weight and `tourism_quarterly` at 0.228; (c) `exp_002` §7.1 confirms "the winning portfolios (`exp_001`) share the trait of higher-step monthly specialist (1500) + energy cross-freq combination." All other specialists from `exp_002` are proven winners – daily, hourly, subhourly, weekly, various, energy, retail, weather, web – and are carried forward unchanged. This is a targeted recovery of the single design variable that caused `exp_002`'s WQL regression, not an incremental tweak. [...]

3. Specialist Definitions

[...]

`05_monthly.yaml` (`625322f7`)

- **Datasets:** `m4_monthly`
- **`max_steps`:** 1500 – **KEY CHANGE from `exp_002` (1000 steps).** Restored to `exp_001`'s config per analyst finding: 1500-step monthly (`625322f7`) carried `tourism_yearly` at 0.830 weight (`exp_001` §4) vs 1000-step version (`8d98d6ae`) which lost −0.155 MASE on the same task (`exp_002` §2 Regressions).
- **`learning_rate`:** 0.001 – `exp_001`'s proven config
- **Rationale:** The single highest-impact portfolio-wide specialist (avg weight 0.310 in `exp_001` §4, 0.284 in `exp_002` §4). The 1500-step version is critical for `tourism_yearly`, `m4_yearly`, and as quarterly proxy. This is the targeted fix to recover `exp_002`'s WQL regression.

[...]

```
chroma train-portfolio config/portfolios/exp_003 --model-id
    pretrained_generalists/chroma_4m/checkpoint-final --max-steps 1000 --max-steps 1000
    --max-steps 1500 --max-steps 1000 --max-steps 1500 --max-steps 1000 --max-steps 1500
    --max-steps 1000 --max-steps 1000 --max-steps 1000 --learning-rate 0.001
    --learning-rate 0.001 --learning-rate 0.0005 --learning-rate 0.001 --learning-rate
    0.001 --learning-rate 0.001 --learning-rate 0.0005 --learning-rate 0.001
    --learning-rate 0.001 --learning-rate 0.001
```

4. Predicted Outcome

[...] **Predicted WQL skill: 0.020–0.032** (recovery of the −0.017 WQL loss from `exp_002`, potentially matching `exp_001`'s 0.028). The 1500-step monthly specialist is the proven carrier of `tourism_yearly` at 0.830 weight per `exp_001` §4 – restoring it should recover the +0.155 MASE regression on this task and improve WQL distributional forecasts.

[...]

Task	Current `exp_002` `MASE_delta` vs `exp_001`	Predicted `exp_003` `MASE_delta`	Rationale
`monash_tourism_yearly`	+0.155	**−0.05 to −0.10**	1500-step monthly (`625322f7`) carried at 0.830 weight in `exp_001` §4; restoring it directly recovers this task's carry
[...]			
`monash_cif_2016`	+0.016	**−0.01 to −0.02**	Same dynamic – 1500-step monthly had 0.228 weight on this task in `exp_001` §4