FACTORIZED INITIALIZATION FOR FEW-STEP ADAPTATION IN TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Few-step adaptation in time-series forecasting leaves little optimization time to repair a poor start. We study how that start should combine a reusable direction with variation induced by a sequence descriptor. Statistical Sequence-Conditioned Initialization (SSCI) represents these roles explicitly: a learned shared displacement provides the reusable start, and a bounded descriptor-conditioned residual modifies it before a fixed adaptation procedure. Shared-5, our reference, adapts the pretrained shared base for five support updates. With equal complete-update projections, factorization improves Shared-5-relative query gain over a complete conditional generator by $+3.88$ percentage points (pp) on External-7 and $+4.59$ pp on a temporal-split Monash replication. Using one jointly trained checkpoint, enabling the residual improves all six held-out real sources, with a source-equal mean of $+3.44$ pp. In a five-seed descriptor-replacement experiment, substituting a descriptor from another dynamical regime raises five-step query loss from 0.0869 to 0.1904. Statistics alone provide a competitive conditioning signal, and the residual effect persists across update budgets and host architectures. These results support an explicit shared-plus-residual parameterization for the starting point supplied to a constrained optimizer.

1 INTRODUCTION

Few-step adaptation in time-series forecasting gives a forecaster a short support history and only a small number of gradient updates. The starting parameters must carry structure that transfers across tasks while admitting variation derived from the available sequence. When the update budget is small, the parameterization of these two roles becomes part of the learning problem.

Shared-start meta-learning learns one initialization and lets support evidence enter through later gradients (Finn et al., 2017; Nichol et al., 2018). Conditional parameter generation instead maps a task representation directly to weights or updates (Ha et al., 2017; Przewiezlikowski et al., 2024). The latter is expressive enough to contain a constant component, but it must discover which parameter changes should be shared and which should vary with support inside one generated output. This creates the central question of our study: under a fixed few-step optimizer, does making that decomposition explicit change adaptation?

We test the hypothesis that an explicit decomposition provides a useful inductive bias for finite-budget optimization. An unconditional displacement supplies the reusable warm start, while a bounded residual is generated from support-only sequence descriptors (Talagala et al., 2023). We compare this decomposition with a complete generator of equal capacity and update norm. A paired branch comparison measures the contribution of the residual within one checkpoint, and independently trained shared starts place the complete initializer in context.

SSCI is an implementation of this initialization principle. It materializes a norm-bounded, low-rank parameter correction before adaptation. Figure 1 shows the shared and conditional paths and their paired comparison. Meta-training uses 21 synthetic tasks; the real-source suites and seven synthetic OOD families are held out from model selection. The evaluation asks whether factorization improves the learned start, how much the conditional path contributes beyond the shared displacement, and whether descriptor content can steer that contribution under known dynamics.

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.

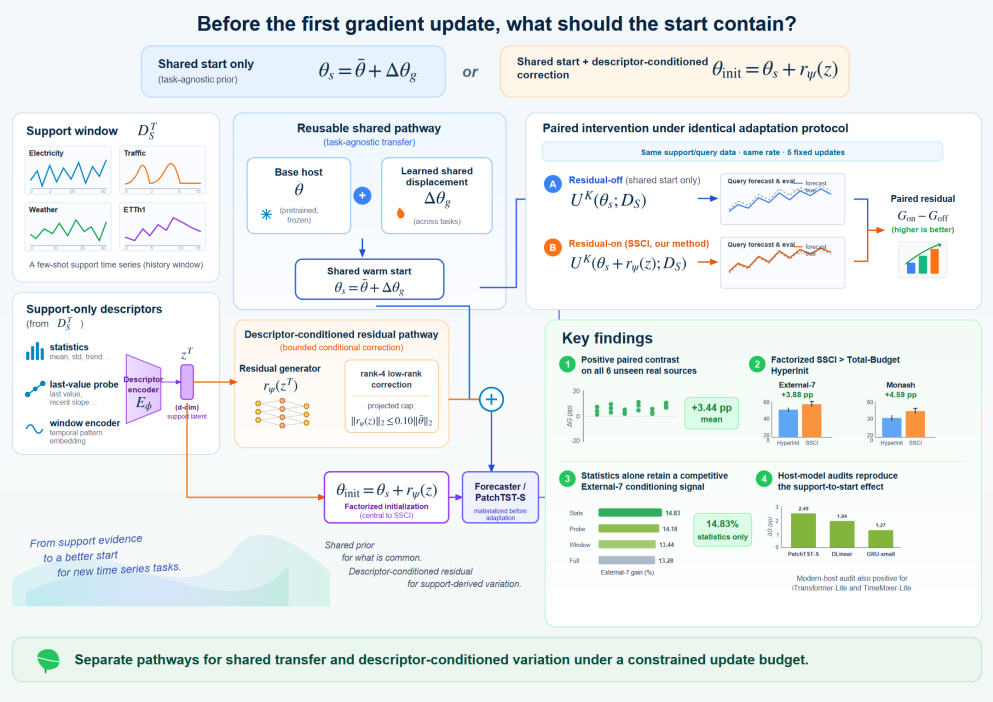

Figure 1: SSCI as a support-to-start interface. A reusable shared warm start is refined by a descriptor-conditioned residual before adaptation. The paired branches measure the residual contribution within one checkpoint; the lower panels preview the principal factorization, descriptor, and host-model results.

Contributions. Our contributions are threefold:

- **Initialization principle.** We formulate a few-step start as a reusable displacement plus a support-conditioned correction, making shared transfer and sequence-specific variation explicit before adaptation.
- **Factorized implementation.** We introduce SSCI, which generates a bounded low-rank residual from support-only sequence descriptors and adds it to a learned shared warm start while leaving the host forecaster and inner optimizer unchanged.
- **Empirical evidence.** Against an equal-capacity complete generator under the same norm budget, factorization improves real-source gain by $+3.88$ and $+4.59$ pp. The residual helps all six held-out real sources in a paired checkpoint comparison. Separately, descriptor replacement shows that regime content steers the correction in synthetic data with known regimes.

2 Related Work

2.1 Shared and Task-Conditioned Meta-Learning

MAML and Reptile learn a shared initialization for gradient-based adaptation (Finn et al., 2017; Nichol et al., 2018). CAVIA adapts context parameters, CNAPs conditions adaptation on a task representation, and task-aware modulation and conditional meta-learning provide further ways to combine shared models with task evidence (Zintgraf et al., 2019; Requeima et al., 2019; Vuorio et al., 2019; Denevi et al., 2020). LEO performs adaptation in a learned latent parameter space (Rusu et al., 2019), whereas HyperMAML generates updates from support-set evidence (Przewiezlikowski et al., 2024). Two recent ICLR neighbors clarify where our method enters the adaptation process. Source-free time-series adaptation factorizes source backbones into a tunable core and frozen factors to regularize gradient-based target adaptation (Patel et al., 2025). MotherNet instead maps an entire

training set to child-network weights in one forward pass, replacing per-dataset gradient descent with parameter generation (Müller et al., 2025). SSCI materializes a bounded additive residual in the host weights before a fixed five-step inner loop, allowing subsequent gradients to refine a start that already separates shared transfer from support-conditioned variation.

2.2 Task representations and parameter generation

Hypernetworks map conditioning inputs to network parameters (Ha et al., 2017). Task2Vec uses probe-network Fisher information to represent tasks, while Dataset2Vec learns set-level features (Achille et al., 2019; Jomaa et al., 2021). Statistical time-series features provide another representation for model selection and transfer (Talagala et al., 2023). Low-rank updates constrain the parameter footprint of adaptation (Hu et al., 2022); Meta-UCF extends task-conditioned LoRA generation to continual learning in language models (Xiao et al., 2026). These precedents motivate our implementation. We study the value of separating an unconditional start from its conditional correction. Equal-capacity generation and descriptor-replacement experiments distinguish the parameterization from the support representation.

2.3 Sequence context in forecasting

Forecasting methods incorporate sequence context at several locations. Time-LLM reprograms inputs and TEMPO uses prompts (Jin et al., 2024; Cao et al., 2024). ICTSP constructs in-context examples from lookback and future segments (Lu et al., 2025). COSA adapts forecasts through a context-aware output-space correction around a frozen forecaster (Im & Kwon, 2026). Corpus-scale and zero-shot models offer complementary routes to cross-series transfer (Oreshkin et al., 2021; Das et al., 2024; Ansari et al., 2024; Woo et al., 2024; Ekambaram et al., 2024). These approaches differ in where sequence evidence enters the prediction process. Our study places that evidence in the initial parameters and evaluates its contribution after the same few support updates. This placement enables residual-on/off comparisons without changing the forecasting architecture or the inner-loop optimizer.

3 Method

3.1 Factorized starting parameters

Each episode $\tau = (D_S^\tau, D_Q^\tau)$ contains support input–target pairs and a temporally disjoint query set. Let f_θ be a forecaster and let $U_K(\theta, D_S; \eta_0)$ denote K fixed-rate support updates with scalar rate η_0. Stage 1 learns base parameters $\bar{\theta}$ across the meta-training sources. Stage 2 freezes $\bar{\theta}$ and learns a displacement $\Delta\theta_g$ shared by all inputs. Their sum is the reusable start,

$$\theta_s = \bar{\theta} + \Delta\theta_g, \qquad \theta_{\text{init}}^\tau = \theta_s + r_\psi(z^\tau), \quad z^\tau = E_\phi(\text{Desc}(D_S^\tau)) \,. \tag{1}$$

The first expression separates the fixed host from the learned shift reused across inputs. The second adds a residual generated from a support-derived descriptor. The shared term θ_s is invariant to the descriptor, whereas $r_\psi(z^\tau)$ is its conditional component. Both terms are materialized in the forecaster weights before adaptation.

More explicitly, $U_0(\theta, D_S; \eta_0) = \theta$ and $U_{k+1} = U_k - \eta_0 \nabla_\theta L_S(U_k; D_S)$ for $k = 0, \ldots, K-1$. We use first-order updates: the support gradient is not differentiated through in the outer objective. The two paths are

$$\theta_{\text{off}}^\tau(K) = U_K(\theta_s, D_S^\tau; \eta_0) \,, \tag{2}$$

$$\theta_{\text{on}}^\tau(K) = U_K(\theta_s + r_\psi(z^\tau), D_S^\tau; \eta_0) \,. \tag{3}$$

The two paths enable a paired measurement of the residual within one trained initializer. Section 5 also compares the full initializer with a separately trained Global-only start.

3.2 Descriptor representation and correction

The descriptor uses support data only and combines fixed sequence statistics, deterministic probe responses, and a learned support-window representation. Their concatenation is mapped to $z^\tau =$

$E_\phi(\mathrm{Desc}(D_S^\tau)) \in \mathbb{R}^{64}$. The 130 statistics summarize level, scale, trend, temporal dependence, spectrum, and variability. The exact 26 per-series quantities, five cross-series aggregations, and encoder architecture are provided in the appendix. All normalizers and temporal indices are fitted on support data.

The three views separate interpretable sequence summaries from learned representations. We retrain statistics-only, probe-only, window-only, and full variants under the same support/query protocol to identify which source of information sustains the initializer.

For a targeted tensor $\bar{\theta}_\ell \in \mathbb{R}^{p_\ell \times q_\ell}$, the correction uses rank-four learned bases and a descriptor-conditioned core,

$$r_\psi(z^\tau)_\ell = \Pi_\ell(U_\ell C_\ell(z^\tau) V_\ell)\,, \tag{4}$$

where Π_ℓ enforces $\|r_\psi(z^\tau)_\ell\|_2 \le 0.10\|\bar{\theta}_\ell\|_2$. The default target set contains 30 tensors spanning the patch projection, both Transformer layers, final normalization, and forecasting head. Matrix-valued weights use rank-four factors; bias vectors and the convolution kernel use four learned flattened bases. The appendix gives the exact parameter-name patterns, shapes, and counts. This low-rank parameterization with a norm-bounded correction leaves the host architecture and inner optimizer unchanged.

3.3 Learning the start

We use two fixed-budget stages. Both stages use the same forecasting loss on sampled episodes. For any support or query batch $B = (X, Y)$,

$$L(\theta; B) = \tfrac{1}{2}\,\mathrm{SmoothL1}(f_\theta(X), Y) + \tfrac{1}{2}\,\mathrm{MSE}(f_\theta(X), Y)\,. \tag{5}$$

We write $L_S(\theta) = L(\theta; B_S)$ and $L_Q(\theta) = L(\theta; B_Q)$ for the support and query batches. The Stage 1 objective is

$$\mathcal{L}_{\mathrm{base}}(\bar{\theta}) = \mathbb{E}_{\tau,\, B_S \subset D_S^\tau}\left[L_S(\bar{\theta})\right]. \tag{6}$$

Here B_S is a minibatch from the support window. Stage 1 uses AdamW with learning rate 3×10^{-4}, weight decay 10^{-4}, and gradient-norm clipping at 1.0 for $E_1 = 50$ sampled episodes per seed. We then freeze $\bar{\theta}$ and jointly optimize $\Delta\theta_g$, E_ϕ, and r_ψ with the Stage 2 objective

$$\mathcal{L}_{\mathrm{init}} = \mathbb{E}_\tau\left[L_Q(\theta^\tau_{\mathrm{on}}(5)) + 0.2 L_Q(\theta^\tau_{\mathrm{on}}(0)) + 10^{-4}\mathcal{R}_g(\Delta\theta_g; \bar{\theta})\right]. \tag{7}$$

For target-tensor index set $\mathcal{I}$ and element count n_ℓ, the global penalty is

$$\mathcal{R}_g(\Delta\theta_g; \bar{\theta}) = \frac{1}{|\mathcal{I}|}\sum_{\ell \in \mathcal{I}} \frac{1}{n_\ell}\left\| \frac{\Delta\theta_{g,\ell}}{\|\bar{\theta}_\ell\|_2 + \epsilon} \right\|_F^2, \qquad \epsilon = 10^{-6}. \tag{8}$$

Here $\|\cdot\|_2$ is the Euclidean norm after flattening (the Frobenius norm for matrices). Stage 2 uses the same AdamW settings for $E_2 = 50$ initializer episodes per seed. Query loss enters only this second stage. Training uses the residual-on path; residual-off is evaluated afterward through Eq. 3. The independent Global-only model is trained separately. Both stages run for their fixed episode budgets without validation-based early stopping, and the inner loop uses one scalar rate throughout.

4 Experimental Setup

Data and episodes. The primary paired evaluation contains Bitcoin, COVID Deaths, US Births, Australian Electricity Demand, FRED-MD, and Weather, following the source provenance catalogued by the Monash Time Series Forecasting Archive (Godahewa et al., 2021). These six real sources are excluded from meta-training and model selection. Meta-training uses 21 synthetic in-distribution tasks. Synthetic OOD contains seven held-out generator families. External-7 contains four ETT sources plus Bitcoin, COVID Deaths, and US Births. Monash test is a protocol replication over the latter three identifiers, using a separate temporal split and normalization rule.

Unless noted otherwise, the canonical protocol uses PatchTST-S, five inner updates at $\eta_0 = 10^{-3}$, 50 episodes in each training stage, three training seeds, and eight repetitions per source and seed. Each episode samples 16 support windows. A sample maps a 48-step history to the next 12 targets; one descriptor aggregates the episode's support windows. Training uses 16 query windows and

held-out evaluation uses 32. Support and query windows come from separate temporal partitions with a one-horizon embargo, so their raw time intervals do not overlap. Query values are excluded from descriptors, normalization, and support-side selection. Episodes and seeds are averaged within source before sources receive equal weight; paired 95% bootstrap intervals resample source identifiers.

The experiments address three questions: whether explicit factorization improves on complete conditional generation under the same update budget, how much the conditional branch contributes within one learned start, and whether that contribution follows descriptor content. Additional studies vary the descriptor, update budget, and host architecture. A complete map of checkpoints and aggregation protocols appears in the appendix.

Comparisons and metrics. All comparisons reuse the canonical episodes unless stated otherwise. The paired residual comparison evaluates residual-off and residual-on from one jointly trained checkpoint with identical support/query episodes, host, optimizer, scalar rate, and update budget. Independent Global-only retrains the warm start without the descriptor path. For the Total-Budget comparison, let $P_\ell(\delta) = \min\{1, 0.10\|\bar{\theta}_\ell\|_2/(\|\delta\|_2+\epsilon)\}\delta$. Factorized SSCI uses $P_\ell(\Delta\theta_{g,\ell}+r_\psi(z)_\ell)$, while HYPERINIT uses $P_\ell(h(z)_\ell)$ for a complete generated displacement. Both use the same descriptor, host, target tensors, optimizer, five-step budget, and complete-update cap. Their conditioner counts are 205,077 and 204,529, respectively, with rank/basis settings 4/4 and 6/8. First-order MAML (FOMAML) supplies a shared-start reference on the canonical held-out episodes (Finn et al., 2017). PatchTST-S is the primary host (Nie et al., 2023); DLinear (Zeng et al., 2023) and GRU-small test transfer across host models.

Our reference, *Shared-5*, applies five updates directly to $\bar{\theta}$. For method m,

$$\theta^\tau_{\mathrm{S5}} = U_5(\bar{\theta}, D^\tau_S; \eta_0), \qquad G^\tau_m = \frac{L_Q(\theta^\tau_{\mathrm{S5}}) - L_Q(\theta^\tau_m)}{|L_Q(\theta^\tau_{\mathrm{S5}})| + \epsilon}, \qquad \Delta_{\mathrm{res}} = \overline{G_{\mathrm{on}}} - \overline{G_{\mathrm{off}}}. \tag{9}$$

We report $100G^\tau_m$ as *Shared-5-relative gain*: the fraction of Shared-5 query error removed on the same episode. Shared-5 is zero by definition, and positive values mean lower query loss. The source-equal $100\Delta_{\mathrm{res}}$ is the *paired residual* in pp, using the same Shared-5 denominator for both branches. An *independent-start contrast* instead compares separately trained systems. To anchor this relative measure, the canonical comparison reports support-normalized MSE and MAE; the corresponding Shared-5 anchors are $1.765/0.947$ on Synthetic OOD, $32.571/2.670$ on External-7, and $75.487/5.047$ on Monash. The appendix contains negative-transfer rates, rank/cap and layer-placement sweeps, source-level results, and additional protocol details.

5 RESULTS

5.1 EXPLICIT FACTORIZATION VERSUS COMPLETE CONDITIONAL GENERATION

Under the Total-Budget comparison defined in Section 4, HYPERINIT emits the entire displacement through one map, whereas SSCI assigns shared and conditional changes to separate paths. Figure 2 (right) shows higher source-equal gain for the factorized start on External-7 ($+3.88$ pp $[1.17, 6.87]$) and the Monash replication ($+4.59$ pp $[1.29, 11.06]$). Synthetic OOD gives $+1.80$ pp with a wider interval $[-5.42, 10.18]$. The real-source results favor an explicit shared-plus-residual start under the same complete-update budget.

The ordering recurs with an independently trained anchored complete generator: the factorized start gains $+7.14$ pp $[3.36, 11.20]$ on External-7 and $+5.70$ pp $[2.59, 11.79]$ on Monash (Table 1). Its Synthetic OOD gap is $+0.83$ pp $[-8.00, 10.52]$. The Factorized value is 19.67% in this anchored protocol; the canonical protocol uses different checkpoints and projection and yields 20.43%.

5.2 SHARED TRANSFER AND CONDITIONAL CORRECTION

We next toggle r_ψ inside one trained initializer using the paired comparison from Section 4. Figure 2 (left) shows a positive contrast on all six held-out real sources, with a source-equal mean of $+3.44$ pp $[1.74, 5.33]$. The residual branch improves post-adaptation query loss within this jointly trained start.

An independently trained Global-only model reaches 14.38%, compared with 15.36% for Factorized SSCI; the difference is $+0.98$ pp with interval $[-1.33, 3.31]$ (Table 2). Most of the learned warm-

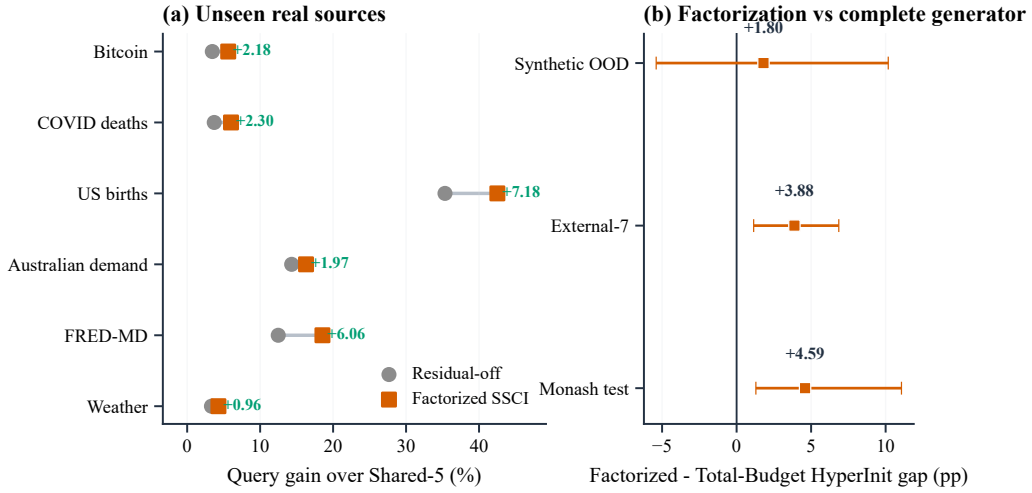

Figure 2: Factorization and residual effects. **Left:** six held-out real sources under one checkpoint; gray circles are residual-off and orange squares residual-on query gains. Every paired difference is positive. **Right:** Factorized SSCI minus Total-Budget HYPERINIT with equal complete-update projections. The real-source gaps are $+3.88$ pp $[1.17, 6.87]$ on External-7 and $+4.59$ pp $[1.29, 11.06]$ on Monash; Synthetic OOD is $+1.80$ pp $[-5.42, 10.18]$. Error bars are paired 95% source-bootstrap intervals.

start gain therefore comes from the shared displacement, with the conditional residual supplying a smaller correction.

The canonical comparison with FOMAML provides a strong shared meta-initialization reference (Table 3). FOMAML has higher source-equal gain on External-7 (22.03 vs. 20.43) and Monash (14.42 vs. 14.03); Factorized SSCI has higher gain and lower MSE/MAE on Synthetic OOD, and lower MAE on Monash. A source-family analysis in the appendix localizes the largest Factorized-minus-FOMAML gaps to trend ($+12.13$ pp), seasonal ($+3.60$ pp), and long-memory ($+2.64$ pp).

Table 1: Factorized versus anchored complete generation. Values are source-equal Shared-5-relative gains (%). Both methods share the descriptor, backbone, target tensors, optimizer, five-step budget, and per-tensor update projection. Gaps are paired contrasts in percentage points.

Suite	Factorized SSCI	Anchored HyperInit	Gap (pp)
External-7	26.62	19.48	**+7.14 [3.36, 11.20]**
Monash replication	19.43	13.73	**+5.70 [2.59, 11.79]**
Synthetic OOD	12.13	11.30	+0.83 [-8.00, 10.52]

Table 2: Factorized SSCI versus an independently optimized Global-only start. Values are source-level Shared-5-relative gains (%); the final column is Factorized SSCI minus Global-only in pp. The interval is a paired 95% source-bootstrap interval.

Source	Independent Global-only	Factorized SSCI	Contrast (pp)
Australian demand	16.69	12.80	-3.89
Bitcoin	4.12	6.23	$+2.10$
COVID deaths	4.30	6.56	$+2.26$
FRED-MD	15.23	20.70	$+5.47$
US births	41.92	42.50	$+0.58$
Weather	4.04	3.35	-0.69
Source-equal mean	14.38	15.36	$+0.98$ $[-1.33, +3.31]$

Nonlinear, regime, and AR families favor FOMAML. This profile identifies the families for which fixed sequence summaries provide the most useful pre-gradient signal.

Table 3: Factorized SSCI and FOMAML on canonical held-out episodes. Gain is Shared-5-relative query gain (higher is better); MSE and MAE use support-fitted normalization (lower is better). Values are source-level means on the same episodes. Bold marks the better value within each metric.

Suite	Gain ↑		MSE ↓		MAE ↓	
	FOMAML	Factorized	FOMAML	Factorized	FOMAML	Factorized
Synthetic OOD	11.39	**12.04**	1.489	**1.441**	0.838	**0.817**
External-7	**22.03**	20.43	**30.581**	30.593	**2.404**	2.409
Monash test	**14.42**	14.03	**71.523**	71.545	4.718	**4.690**

5.3 Sequence statistics as conditioning signals

The descriptor ablation compares independently trained initializers (Table 5). No-Temporal has the highest mean on External-7 (15.61%), while Full has the highest on Synthetic OOD (9.51%). Statistics-only reaches 14.83% on External-7 without the probe or learned window view. **Simple sequence summaries can supply a competitive conditioning signal, while the useful descriptor composition varies by suite.** The remaining comparisons use Full as the common descriptor.

The real-source comparison establishes branch utility within a learned initializer. A separate five-seed synthetic experiment tests content sensitivity by fixing the target support/query episode and replacing only its descriptor. At $K = 5$, a wrong-regime descriptor increases mean query loss from 0.0869 to 0.1904 (paired seed-level gap 0.1035, 95% CI $[0.0443, 0.1627]$). Layerwise residual-norm matching increases it to 0.2184, and the true descriptor remains best from $K = 0$ through 20 (Table 4). **For known synthetic regimes, descriptor content steers the conditional correction toward a better start.**

These results distinguish two roles of the descriptor. On the real suites, the residual branch improves the jointly trained initializer, and fixed statistics retain competitive gain. In the synthetic regime experiment, changing descriptor content changes the quality of the selected start. The central design choice is therefore the factorized start rather than a particular multi-stream encoder.

5.4 Update budgets, host architectures, and cost

With checkpoints fixed across K, the real-suite paired residual is positive from initialization through 20 updates (Table 6A). It moves from $+2.70$ to $+2.72$ pp on External-7 and from $+1.50$ to $+1.59$ pp on Monash. Synthetic OOD is near zero at initialization ($+0.07$ pp) and rises to $+0.71$ pp at $K = 20$. The zero-update point is directly trained through the $0.2L_Q(\theta_{\text{on}}(0))$ term. Shared-5-relative gains decrease with K on both real suites and increase on Synthetic OOD, revealing a suite-dependent interaction between the start and subsequent gradients.

Positive within-checkpoint branch effects recur across the three evaluated host architectures. Under the shared 50/25/25 schedule, the Synthetic OOD effects are $+1.94$, $+1.27$, and $+2.49$ pp for DLinear, GRU-small, and PatchTST-S (Table 6B). A separate rank/cap sweep in the appendix reports positive residual means across all nine settings ($r \in \{2, 4, 8\}, c \in \{0.05, 0.10, 0.20\}$).

Conditioning adds a fixed support-side cost. Descriptor construction, encoding, and residual generation take 52.0 ms and 24.16 MiB peak memory in this implementation. The five adaptation updates then take 106.35 ms from the factorized start and 105.64 ms from the shared start (Table 6B). A FOMAML outer training step, including query backpropagation, takes 139.10 ms.

Table 4: Descriptor replacement on fixed target episodes. Mean query loss over five seeds (lower is better), with 90 paired episodes per seed. "Matched wrong" preserves each residual's layerwise norm.

K	True	Wrong	Matched wrong	Residual off
0	**0.0937**	0.1988	0.2268	0.1212
1	**0.0919**	0.1966	0.2247	0.1197
2	**0.0904**	0.1948	0.2228	0.1184
5	**0.0869**	0.1904	0.2184	0.1153
10	**0.0835**	0.1858	0.2137	0.1119
20	**0.0797**	0.1808	0.2085	0.1079

Table 5: Sequence representations for the conditional residual. Values are three-seed Shared-5-relative gains (%) with 95% dataset-bootstrap intervals; each variant is trained separately.

Descriptor	Synthetic OOD	External-7
Statistics only	8.45 $[-1.71, 19.45]$	**14.83** $[7.85, 21.37]$
Probe only	9.42 $[-0.26, 20.25]$	14.18 $[7.55, 20.64]$
Window only	9.41 $[-0.64, 20.05]$	13.44 $[6.33, 21.01]$
No temporal	9.25 $[-1.72, 20.52]$	**15.61** $[8.56, 21.86]$
Full	**9.51** $[-0.29, 19.88]$	13.28 $[7.79, 18.77]$

Table 6: Operating range. **A:** source-level Shared-5-relative gains (%) across adaptation budgets; residuals are on-minus-off differences in pp. **B:** within-checkpoint host contrasts on Synthetic OOD and PatchTST-S timing and memory.

A. Adaptation budget

Suite	K	Off	Factorized	Resid.
Synthetic OOD	0	12.16	12.23	+0.07
	5	12.32	12.58	+0.26
	20	12.73	13.44	+0.71
External-7	0	18.67	21.37	+2.70
	5	18.40	21.11	+2.71
	20	17.55	20.27	+2.72
Monash test	0	13.17	14.67	+1.50
	5	13.00	14.53	+1.53
	20	12.48	14.07	+1.59

B. Host transfer and cost

Host	Off	On	Resid.
DLinear	10.21	12.15	+1.94
GRU-small	1.52	2.79	+1.27
PatchTST-S	12.60	15.09	+2.49

Component	Med. ms	P90 ms
Descriptor (CPU)	32.56	34.04
Encoder (GPU)	1.91	1.93
Generator (GPU)	17.53	17.63
Shared adapt.	105.64	106.52
Fact. adapt.	106.35	106.96
Cond. peak	24.16 MiB	
Adapt. peak	23.78 MiB	

6 Discussion

The main result concerns how the start is parameterized. Under an equal complete-update budget, separating a shared displacement from a conditional residual improves on complete generation across both real-source protocols. The decomposition assigns source-invariant transfer to one path and descriptor-dependent variation to another, with each path learning a distinct role before adaptation.

The complete generator is expressive enough to represent both terms, so the comparison isolates how those roles are organized in the parameterization. The structural difference is that SSCI makes the reusable displacement independent of the descriptor and restricts task-dependent variation to a bounded residual. Its advantage on the two real-source protocols points to the value of encoding these invariances directly when the optimizer receives only a few subsequent updates.

The two paths contribute at different scales. Retrained Global-only recovers most of the complete initializer's gain, which identifies the shared displacement as the primary transferable component. Within the joint checkpoint, however, enabling the residual improves every source in the six-source test. The synthetic descriptor-replacement experiment adds a separate result: when the dynamical regime is known, the correct descriptor selects a better correction than a descriptor from another regime, even after matching residual norms. Thus the real suites establish branch utility, while the synthetic study establishes content sensitivity across known regime changes.

The comparison with FOMAML locates the benefit of pre-gradient conditioning more precisely. Fixed sequence summaries are most effective for trend, seasonal, and long-memory families in our source-level analysis; FOMAML is stronger for nonlinear, regime, and AR families. The methods emphasize different information routes: one shapes a shared start for later gradients, while the other can also alter the start from sequence descriptors before the first update.

The budget results further separate initialization from subsequent optimization. On the real suites, the paired residual remains nearly constant through 20 updates while both branches' Shared-5-relative gains decline along the fixed-rate trajectory. Synthetic OOD follows a different trajectory. Thus the residual contribution persists as the adaptation budget extends, while total query gain remains source-dependent.

Stage 2 optimizes the five-update endpoint, whereas the budget sweep follows the same fixed-rate trajectory out to 20 updates. On the real suites, residual-on and residual-off decline together beyond the trained endpoint while maintaining nearly the same gap. This shared decline indicates that the extra-step behavior belongs to the inner-loop trajectory, while the conditional gap remains stable.

6.1 Implications for few-step adaptation

The experiments suggest a practical division of labor for few-step adaptation. A learned shared displacement captures the direction that transfers across tasks. A bounded conditional path can then adjust that start when its descriptor resolves dynamics relevant to the target family. Trend, seasonal, and long-memory signals favor this pre-gradient route in our source-family analysis, whereas nonlinear and regime-driven families favor a gradient-shaped shared start.

Factorization also exposes independent design levers. The descriptor determines which sequence properties reach the conditional path, while rank and norm caps bound how far that path can move from the reusable start. These choices leave the host forecaster and inner optimizer unchanged. The resulting interface first learns a shared direction and then allocates bounded conditional capacity before adaptation begins.

7 Conclusion

SSCI gives few-step forecasting a factorized start: a learned shared displacement plus a bounded descriptor-conditioned residual. It improves over equal-budget complete generation on two held-out real-source protocols. The residual contributes on all six real sources, and descriptor replacement shows content sensitivity across known synthetic regimes.

Together, the results identify a useful division of labor at initialization. The shared path carries the primary transferable direction, while the bounded conditional path adjusts that direction before gradient adaptation. This factorization persists across adaptation budgets and host architectures, providing a general interface for making the first few optimization steps more effective without changing the forecaster or inner optimizer.

AI USE STATEMENT

Generative AI tools assisted with drafting portions of the manuscript, polishing its language, and generating the synthetic datasets used in the experiments. The authors reviewed the AI-assisted material and take responsibility for the final manuscript and datasets.

REPRODUCIBILITY STATEMENT

Section 3 specifies the initialization and training objectives, and Section 4 defines the support-only evaluation and aggregation protocol. The supplementary material provides architecture and descriptor specifications, additional comparisons, source-level results, and protocol maps. The accompanying files include the core `ssci` package, configurations, smoke tests, principal experiment scripts, and JSON summaries used for the reported tables. A clean CPU run passes 15 smoke tests; one GPU-only test is skipped. Independently trained baselines and unavailable-source records are identified separately in the summaries. The anonymous code mirror, including the pinned release commit, is available at `https://anonymous.4open.science/r/ssci-initialization-priors-03C3/`.

REFERENCES

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2Vec: Task embedding for meta-learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6429–6438, 2019. doi: 10.1109/ICCV.2019.00653.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. URL `https://openreview.net/forum?id=gerNCVqqtR`.

Defu Cao, Furong Jia, Sercan O. Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. TEMPO: Prompt-based generative pre-trained transformer for time series forecasting. In *International Conference on Learning Representations*, 2024.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 10148–10167. PMLR, 2024.

Giulia Denevi, Massimiliano Pontil, and Carlo Ciliberto. The advantage of conditional meta-learning for biased regularization and fine tuning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 964–974. Curran Associates, Inc., 2020.

Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam H. 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. *arXiv preprint arXiv:2401.03955*, 2024.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1126–1135, 2017.

Rakshitha W. Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob J. Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive. In *Proceedings of the 35th Conference on Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 1, 2021.

David Ha, Andrew Dai, and Quoc V. Le. HyperNetworks. In *International Conference on Learning Representations*, 2017.

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

Jeonghwan Im and Hyuk-Yoon Kwon. COSA: Context-aware output-space adapter for test-time adaptation in time series forecasting. In *International Conference on Learning Representations*, 2026.

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 *International Conference on Learning Representations*, 2024.

Hadi S. Jomaa, Lars Schmidt-Thieme, and Josif Grabocka. Dataset2Vec: Learning dataset meta-features. *Data Mining and Knowledge Discovery*, 35:964–985, 2021. doi: 10.1007/s10618-021-00737-9.

Jiecheng Lu, Yan Sun, and Shihao Yang. In-context time series predictor. In *International Conference on Learning Representations*, 2025.

Andreas C. Müller, Carlo Curino, and Raghu Ramakrishnan. MotherNet: Fast training and inference via hyper-network transformers. In *International Conference on Learning Representations*, 2025.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

YuQi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. Meta-learning framework with applications to zero-shot time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(10):9242–9250, 2021. doi: 10.1609/aaai.v35i10.17115.

Gaurav Patel, Chris Sandino, Behrooz Mahasseni, Ellen Zippi, Erdrin Azemi, Ali Moin, and Juri Minxha. Efficient source-free time-series adaptation via parameter subspace disentanglement. In *International Conference on Learning Representations*, 2025.

Marcin Przewiezlikowski, Przemyslaw Przybysz, Jacek Tabor, Maciej Zieba, and Przemyslaw Spurek. HyperMAML: Few-shot adaptation of deep models with hypernetworks. *Neurocomputing*, 598:128179, 2024. doi: 10.1016/j.neucom.2024.128179.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*, 2019.

Thiyanga S. Talagala, Rob J. Hyndman, and George Athanasopoulos. Meta-learning how to forecast time series. *Journal of Forecasting*, 42(6):1476–1501, 2023. doi: 10.1002/for.2963.

Risto Vuorio, Shao-Hua Sun, Hexiang Hu, and Joseph J. Lim. Multimodal model-agnostic meta-learning via task-aware modulation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53140–53164. PMLR, 2024.

ShiLin Xiao, Tianxiang Xu, Canran Xiao, Weihao Luo, Liwei Hou, and Chuangxin Zhao. Meta-UCF: Unified task-conditioned LoRA generation for continual learning in Large Language Models. In *International Conference on Learning Representations*, 2026.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*, 37:11121–11128, 2023.

Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 7693–7702. PMLR, 2019.