EVOFIF: EVOLUTIONARY DISCOVERY OF INDUCTIVE BIASES FOR FORECAST-ORIENTED TIME SERIES IMPUTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series imputation is widely used as a preprocessing step for forecasting, yet minimizing imputation error does not necessarily improve forecasting performance. Forecast-oriented time series imputation (FOTSI) therefore optimizes imputation for the downstream task, and existing methods are model-centric, since each of them fits a model to one dataset, missingness pattern or forecasting model and has to be retrained whenever the scenario changes. We propose EvoFIF (Evolutionary Forecast-oriented Imputation Framework), which instead discovers reusable configurations of imputation inductive biases by evolutionary search. EvoFIF extracts an initial pool of explicit biases from classical imputation algorithms with a large language model, evolves it by utility-weighted Thompson Sampling with LLM-based mutation and crossover, and verifies every candidate by downstream forecasting utility, objective alignment and computational cost. On nine multivariate datasets, four missingness patterns and four forecasting models, EvoFIF attains the lowest downstream loss, imputes a scenario orders of magnitude faster than every baseline, and its reusability dominates that of the existing frameworks, since one configuration transfers across datasets, patterns, forecasters and horizons without retraining.

1 INTRODUCTION

Time series imputation is usually posed as a data recovery problem, whose objective is to reconstruct missing observations faithfully. In practice imputed series feed downstream tasks such as forecasting, classification and anomaly detection, and reconstruction accuracy is a poor proxy for the utility they need, since it may even hinder the task it serves (Le Morvan & Varoquaux, 2025; Wang et al., 2024). Research has therefore moved towards **task-oriented imputation**, which judges an imputation strategy by the **utility it delivers to the downstream task**.

Forecasting is the most sensitive of these tasks. A forecaster consumes the series both as input X and as label Y, whereas classification and anomaly detection use it only as X. Label-side noise distorts the learned mapping far more than input-side noise (Cheng et al., 2024), and missing values are a structured instance of such noise. They also break the autocorrelation and the regular spacing that forecasting relies on, interrupt the state transition of the underlying dynamical system, and hide seasonality, trend and phase (Chen et al., 2018; Che et al., 2018; Tashiro et al., 2021). **Forecast-oriented time series imputation (FOTSI)** is therefore the central setting of task-oriented imputation.

Existing FOTSI methods fall into three categories, summarized in Table 1 and illustrated in Figure 1, and all three are **model-centric**. Each is designed for a particular task and reports strong results in the setting it targets, yet none of them generalizes, since the strategy they deliver is part of a model fitted to one scenario and the training procedure stays coupled to the scenario and to the forecaster that consume it, so adapting to a new dataset, missingness pattern, forecaster or horizon means retraining at full cost (Zhang et al., 2025; Xue & Zhang, 2026). FOTSI therefore needs a strategy that is decoupled from the scenario and stays effective and cheap when the scenario changes.

Therefore, **balancing effectiveness, reusability and computational efficiency is the central objective of FOTSI**, and Figure 1 shows why the model-centric route struggles to reach it. Adapting

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.

Table 1: Categories of methods for forecast-oriented time series imputation. Reusability asks whether the strategy can be applied to a new scenario without retraining.

Category	Representative method	Optimization target	Reusability	Comp. cost
Evaluation-guided	Generalized Representers (Wang et al., 2024)	Gain estimation of imputation	✗ Retraining required	Medium
Joint optimization	TOI-VSF (Hao et al., 2025b)	Prediction loss minimization	✗ Retraining required	High
Joint optimization	VIDA (Liang et al., 2025)	Prediction loss + distribution alignment	✗ Retraining required	High
Joint optimization	GIMCC (Hao et al., 2025a)	Prediction loss + causal consistency	✗ Retraining required	High
Representation learning	TimeCIB (Choi & Lee, 2024)	Information preservation + reconstruction loss	✗ Retraining required	Medium

to a dataset means training or fine-tuning the model, and deep imputers carry high training and inference costs, while end-to-end imputers learn their patterns implicitly from a single dataset (Wang et al., 2025b). Since FOTSI spans those axes at once, a strategy tailored to one of them starts from scratch on the next.

To meet this objective, **explicit inductive biases** provide a structural representation that is decoupled from any particular model. They state which assumptions govern the missing values, such as local continuity, periodicity, non-stationarity and cross-variable dependence, and each stays available as a component that different models and tasks can invoke (You et al., 2025). A single bias expresses one aspect only, whereas real series combine several, so the question is how to discover complementary biases that match the downstream task. The space is discrete and combinatorial, and the choice of biases, their composition and the concrete rules all determine the utility, so it cannot be enumerated by hand. This obstacle has recently become surmountable, because **LLM-guided evolutionary search** lets a language model propose and modify candidate programs while an automatic evaluator keeps the valuable ones. FunSearch and AlphaEvolve demonstrate this mechanism in mathematics and in computing systems (Romera-Paredes et al., 2024; Novikov et al., 2025), and EVIL applies it to executable algorithms on event sequences and time series (Berghaus, 2026). Moving the search object from the model to structural components turns an intractable combinatorial problem into a controlled discovery process.

We therefore propose **EvoFIF** (Evolutionary Forecast-oriented Imputation Framework), which turns imputation inductive biases from structural assumptions implicit in a trained model into explicit, composable search objects with executable instantiations, and shifts the optimization object of FOTSI from a scenario-specific model to a transferable configuration of inductive biases. EvoFIF

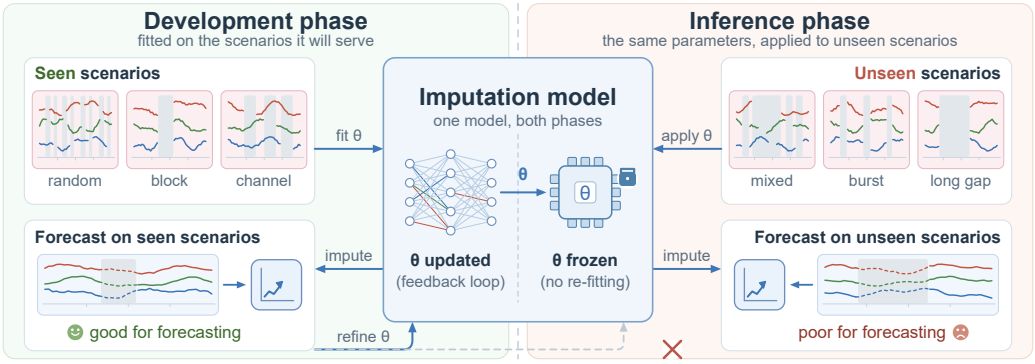


Figure 1: Scenario dependence of model-centric imputation.

extracts an initial set of biases from classical imputation algorithms, expands it with an LLM within a **utility-weighted Thompson Sampling** search, and retains a candidate only if it improves the downstream utility at an acceptable cost. Three challenges follow. (1) The initial biases set the starting point and must be theoretically grounded, verifiable and complementary, since time series are structurally heterogeneous and a single bias cannot cover diverse scenarios (Kim et al., 2025). (2) The space explodes combinatorially, so random search spends the budget on low-value candidates while greedy exploitation converges prematurely. (3) The value of a discovered configuration is hard to verify, because imputation quality and forecasting utility are not always consistent and an LLM-generated function may fail to execute or cost too much.

Our main contributions are summarized as follows.

(1) We propose EvoFIF, and with it we move the study of forecast-oriented imputation from the design of a **model-centric** framework to the **discovery of inductive biases**, which turns the optimization object of FOTSI into a transferable configuration of inductive biases instead of a scenario-specific model.

(2) We realize the search with **utility-weighted Thompson Sampling**, which accounts for the uncertainty of each bias, balances exploitation against exploration, and extends the credit of a crossover to both parents, so the search avoids premature convergence.

(3) Experiments on nine datasets, four missingness patterns and four forecasters show that EvoFIF satisfies the three requirements of FOTSI at once. It attains the lowest downstream loss on 21 of the 36 dataset–pattern combinations, with an average rank of 1.44–1.89 against 3.69 for the strongest baseline, it imputes a scenario orders of magnitude faster than every baseline, and a single configuration transfers across datasets, patterns, forecasters and horizons without retraining, which is what effectiveness, efficiency and reuse mean together.

2 Basic Definitions and Problem Formulation

Definition 2.1 (Forecast-oriented time series imputation (FOTSI)). A FOTSI scenario is a tuple $\tau = (D, R, g, H)$, with dataset D, missingness pattern R, downstream forecasting model g and prediction-task configuration H such as the prediction horizon. Admissible scenarios form the scenario space $\mathcal{T}$. For a given τ, an imputation function $f \in \mathcal{F}$ maps the observed series X and its mask M to $\tilde{X} = f(X, M)$, from which the forecaster produces $\hat{Y} = g(\tilde{X}; H)$. FOTSI seeks

$$f^* = \arg\max_{f \in \mathcal{F}} U(f), \tag{1}$$

where $U(f)$ is the utility that f delivers to the downstream forecasting task.

Definition 2.2 (Downstream utility). To meet the FOTSI objective, which asks for effectiveness, computational efficiency and reusability at the same time, the downstream utility of an imputation function assesses those three properties,

$$U(f) = \alpha\, U_{\text{eff}}(f) + \beta\, U_{\text{effi}}(f) + \gamma\, U_{\text{reuse}}(f), \qquad \alpha, \beta, \gamma \geq 0, \quad \alpha + \beta + \gamma = 1, \tag{2}$$

where, with $L_{\text{pred}}(f; \tau) = L\big(g(f(X, M); H), Y\big)$ the downstream prediction loss on scenario τ and $C_{\text{imp}}(f; \tau)$ the execution cost of f on τ, the two are

$$U_{\text{eff}}(f) = -\,\mathbb{E}_{\tau \sim \mathcal{P}_{\mathcal{T}}}\, L_{\text{pred}}(f; \tau), \qquad U_{\text{effi}}(f) = -\,\mathbb{E}_{\tau \sim \mathcal{P}_{\mathcal{T}}}\, C_{\text{imp}}(f; \tau), \tag{3}$$

and, for target scenarios $\tau' \sim \mathcal{P}_{\mathcal{T}'}$ that lie outside the optimization distribution,

$$U_{\text{reuse}}(f) = -\,\mathbb{E}_{\tau' \sim \mathcal{P}_{\mathcal{T}'}} \Big[\, L_{\text{pred}}(f; \tau') - L_{\text{pred}}(f^*_{\tau'}; \tau') \,\Big], \qquad f^*_{\tau'} = \arg\min_{f \in \mathcal{F}} L_{\text{pred}}(f; \tau'), \tag{4}$$

the bracket being the transfer regret of f on τ'. The first two terms can be estimated during optimization, while the third cannot, because it involves scenarios that the optimization never sees.

Definition 2.3 (Imputation inductive biases). **Imputation inductive biases** are structural assumptions about how the missing values of a series relate to its observed part that hold independently of the scenario. A configuration of complementary inductive biases, together with the rules that decide

which of them applies to which part of a series, is written $b \in \mathcal{B}$, where $\mathcal{B}$ is the biases space[1]. Under this definition the reusability requirement of Definition 2.2 is attainable, because such assumptions hold independently of the scenario, and Lemma B.2 in Appendix B.2 establishes that the functions they realize are exactly the imputation functions that can be executed across scenarios without retraining,

$$\mathrm{Inst}(\mathcal{B}) = \mathcal{F}_{\mathrm{reuse}}. \tag{5}$$

A configuration is realized by an executable imputation function $f_b = \mathrm{Inst}(b)$, where $\mathrm{Inst}(\cdot)$ is the instantiation operator. We write it rather than a term implying execution, because b denotes the assumptions themselves and $\mathrm{Inst}(b)$ is the concrete rule that instantiates them, which is what is executed and evaluated downstream. A scenario-bound parameterization cannot meet it, since its parameters are a function of the scenario (Sener & Koltun, 2018; Song et al., 2026), as Appendix B.1 develops.

Proposition 2.1 (Optimization target of FOTSI). If the utility of equation 2 requires cross-scenario reuse without retraining, then the FOTSI problem of equation 1 is equivalent to

$$\max_{f \in \mathcal{F}_{\mathrm{reuse}}} U(f) \;=\; \max_{b \in \mathcal{B}} U(f_b). \tag{6}$$

The equivalence follows from equation 5, since the two spaces contain the same functions (Appendix B), and it isolates a configuration of biases as the object to optimize, which Definition 2.4 names.

Definition 2.4 (Discovery of inductive biases). The optimization object of Proposition 2.1 is therefore a configuration of inductive biases rather than an individually optimized bias, and the discovery of inductive biases is the search for the configuration that maximizes the downstream utility,

$$b^* = \arg\max_{b \in \mathcal{B}} U(f_b), \tag{7}$$

which by Proposition 2.1 is the same as maximizing the FOTSI utility of equation 2 over the reusable imputation functions, and we call b^* the discovered configuration.

Definition 2.4 closes the reformulation, and Section 3 presents the framework that performs the discovery.

3 Methodology

3.1 Overview of EvoFIF

To meet the requirements that Section 2 places on a FOTSI strategy, we propose **EvoFIF** (Evolutionary Forecast-oriented Imputation Framework), which models FOTSI as a closed discovery loop of three components (Figure 2). As the biases space is open-ended, the **extraction of inductive biases** distils the assumptions of classical imputation algorithms into an initial pool of explicit configurations, so the search starts from assumptions that are theoretically grounded and complementary. As the space grows combinatorially under a finite budget, the **evolution of inductive biases** selects parents by utility-weighted Thompson Sampling and produces offspring by LLM-based mutation and crossover, so the budget follows the posterior probability of usefulness while uncertain candidates keep a chance of being explored. As the value of a candidate is not directly observable, the **verification of inductive biases** scores it by the downstream forecasting loss, an alignment term and a cost term, admits it only if it improves the pool, and returns the evidence to the sampling of the next round, which closes the loop. Once the budget is exhausted, the loop returns the best configuration of the pool, which is the discovered configuration b^* of Definition 2.4; the three components are detailed in Sections 3.2–3.4 and the sampling strategy in Appendix C.

3.2 Extraction of Inductive Biases

As the starting point of the evolution, the quality of the initial pool decides what the search can reach, so its candidates have to be **theoretically grounded, verifiable and complementary**. We

[1] Such a scenario-independent regularity is what domain generalization relies on when structure rather than parameters has to transfer (Zhong et al., 2025).

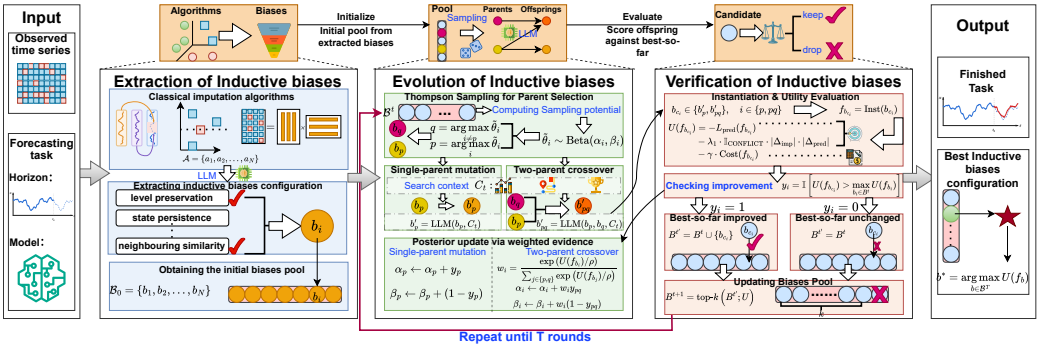


Figure 2: Overview of EvoFIF.

therefore build the pool from classical statistical and machine-learning imputation algorithms $\mathcal{A} = \{a_1, \ldots, a_N\}$, because their assumptions are explicit, since each algorithm states the structure it expects, such as local continuity for forward fill or cross-variable dependence for KNN, so an LLM can read a bias off instead of inferring it, and complementary, since each of them covers a different structural aspect of a series, which is what a scenario that mixes several structures requires. They are also cheap and well understood, and structurally simple methods remain competitive at lower cost (Afkanpour et al., 2024), whereas the biases inside a deep network are distributed over its parameters and are hard to convert into configurations that are explicit and verifiable (Mino Calero et al., 2026). An LLM turns these biases into explicit, structured configurations that $\mathrm{Inst}(\cdot)$ maps to executable functions, so the initial pool is $b_i = \mathrm{Extract}_{\mathrm{LLM}}(a_i)$, with $\mathcal{B}_0 = \{b_1, b_2, \ldots, b_N\}$. The seven seeds are mean, forward fill (LOCF), KNN, MissForest, IIM, TRMF and XGBoost, and each of them carries a distinct and explicit inductive bias that the search can later recombine (Appendix A.2), so they are the starting point of the evolution rather than baselines.

3.3 Evolution of Inductive Biases

The combinatorial growth of the biases space makes the evolution budget-bound. A purely random search would spend most of the LLM budget on low-value candidates, and a greedy strategy concentrates on a few existing ones and converges prematurely (Squillero & Tonda, 2016). To address this, we select parents by **utility-weighted Thompson Sampling**, which draws one sample from the posterior of every candidate and takes the two largest draws as the parents (Russo et al., 2018), and we assign the evidence of one crossover to its two parents in proportion to their utility, which is the weighting the name refers to. The two operators therefore attain **local refinement and structural innovation** at the same time, since mutation refines one configuration of inductive biases in place while crossover merges two of them. Appendix C derives the rule from the bandit model and shows that the weighted update keeps its guarantees.

3.3.1 Thompson Sampling for Parent Selection

The budget goes to the candidates whose posterior makes them most promising. For every b_i in the pool we maintain the latent probability $\theta_i = P(y_i = 1)$ that it produces a successful offspring, where success means that the new configuration beats the current pool, that is, that $U(f_{b'_i})$ exceeds the best utility of the pool. Since y_i is Bernoulli, we model θ_i by a Beta posterior that starts from the uninformative $\mathrm{Beta}(1, 1)$ and is updated by the evidence collected so far. No algorithm is universally better than another (Wolpert & Macready, 1997) and none stays best across datasets and missingness settings (Du et al., 2024), so an uninformative prior is the honest choice, since no algorithm is universally better than another (Wolpert & Macready, 1997) and none stays best across datasets and missingness settings (Du et al., 2024). At the beginning of every round we draw one sample per candidate, a single realization of its latent potential, and take the two largest draws as the parents. A strong record is thus exploited more often, while a wide posterior keeps a chance of being explored.

3.3.2 LLM-BASED MUTATION AND CROSSOVER

The LLM turns the selected parents into new configurations through two complementary operators, since it searches the heuristic space rather than the solution space (Chauhan et al., 2026), while a single operator can settle in a restricted region of it. Single-parent mutation provides **local refinement** of one configuration of inductive biases, $b'_p = \mathrm{LLM}(b_p, C_t)$, where C_t denotes the feedback available so far, whereas two-parent crossover provides **structural innovation** by merging two configurations of inductive biases, $b'_{pq} = \mathrm{LLM}(b_p, b_q, C_t)$. Every offspring is then verified (§3.4), and the outcome updates the posterior of its parent. For mutation, with $y_p = \mathbb{I}[\,U(f_{b'_p}) > \max_{b\in\mathcal{B}^t} U(f_b)\,]$, the whole unit of evidence goes to the single parent, so $\alpha_p \leftarrow \alpha_p + y_p$ and $\beta_p \leftarrow \beta_p + (1 - y_p)$.

For crossover, attributing the outcome to one parent alone would discard the information contributed by the other, so credit is assigned in proportion to the current utility,

$$w_i = \frac{\exp\big(U(f_{b_i})/\rho\big)}{\sum_{j\in\{p,q\}} \exp\big(U(f_{b_j})/\rho\big)}, \qquad i \in \{p, q\}, \tag{8}$$

where $\rho > 0$ is a temperature and $w_p + w_q = 1$. A weaker parent keeps a non-zero weight, so that its combinatorial role is not discarded. The resulting update applies the same rule with a weighted credit, $\alpha_i \leftarrow \alpha_i + w_i\, y_{pq}$ and $\beta_i \leftarrow \beta_i + w_i\,(1 - y_{pq})$ for $i \in \{p, q\}$, with $y_{pq} = \mathbb{I}[\,U(f_{b'_{pq}}) > \max_{b\in\mathcal{B}^t} U(f_b)\,]$,

reads as a weighted pseudo-count update of the Bernoulli evidence. Because $w_p + w_q = 1$, one crossover adds exactly one unit of evidence, just as mutation does, and mutation is the special case $w_p = 1$, so both operators are served by a single sampler, which is what the name utility-weighted Thompson Sampling refers to. Appendix C.3 proves that splitting the evidence this way leaves the guarantees of the sampling rule intact, and the assignment keeps low-utility pieces that recombine well in play, so the search does not concentrate prematurely on a few already good pieces.

3.4 VERIFICATION OF INDUCTIVE BIASES

To verify whether a candidate meets the requirements that FOTSI places on an imputation strategy, we score it by the downstream forecasting loss together with two penalties. Effectiveness and efficiency enter explicitly, while reusability is guaranteed by the discovery procedure itself (Definition 2.3 and Proposition 2.1). Let b_{c_i} with $i \in \{p, pq\}$ be the candidate of a round, and $f_{b_{c_i}} = \mathrm{Inst}(b_{c_i})$. Its utility is

$$U(f_{b_{c_i}}) = -\,L_{\mathrm{pred}}(f_{b_{c_i}}) \;-\; \lambda_1 \cdot \mathbb{I}_{\mathrm{CONFLICT}} \cdot |\Delta_{\mathrm{imp}}| \cdot |\Delta_{\mathrm{pred}}| \;-\; \gamma \cdot \mathrm{Cost}(f_{b_{c_i}}), \tag{9}$$

in which the first two terms are the two faces of U_{eff} of Definition 2.2. The first is the downstream prediction loss, and the second penalizes a *conflict* between the two losses. The best piece of the pool, $b^t_{\mathrm{ref}} = \arg\max_{b\in\mathcal{B}^t} U(f_b)$, serves as a reference, $\Delta_{\mathrm{imp}} = L_{\mathrm{imp}}(f_{b_{c_i}}) - L_{\mathrm{imp}}(f_{b^t_{\mathrm{ref}}})$ and $\Delta_{\mathrm{pred}} = L_{\mathrm{pred}}(f_{b_{c_i}}) - L_{\mathrm{pred}}(f_{b^t_{\mathrm{ref}}})$, and a candidate is in *conflict* exactly when it reduces the imputation error while the forecasting error grows,

$$\mathbb{I}_{\mathrm{CONFLICT}} = \mathbb{I}[\,\Delta_{\mathrm{imp}} < 0 \;\text{ and }\; \Delta_{\mathrm{pred}} > 0\,], \tag{10}$$

which is the failure that FOTSI has to avoid. Its penalty grows with the magnitude of both deviations, because relying on the objective alone can drive a population towards solutions that deviate from the goal (Coello Coello, 2002). The third term realizes U_{effi}, since candidates are produced by an LLM and differ in code structure and cost, and ignoring this would let the search drift towards verbose or redundant functions (Cotroneo et al., 2025). The term penalizes both the number of code lines beyond a tolerated length and the measured runtime, and it is the only place where the efficiency requirement enters the search. A candidate is accepted only if its utility strictly exceeds the best utility of the pool, that is, if $y_i = 1$. Writing $\mathcal{B}'^t$ for the pool after this step, a candidate that satisfies it gives $\mathcal{B}'^t = \mathcal{B}^t \cup \{b_{c_i}\}$ and one that does not leaves $\mathcal{B}'^t = \mathcal{B}^t$. The pool is then truncated to its top k pieces,

$$\mathcal{B}^{t+1} = \operatorname*{arg\,max}_{S\subseteq\mathcal{B}'^t,\ |S|=k} \sum_{b\in S} U(f_b), \tag{11}$$

which keeps the cost of parent selection and the number of LLM calls bounded. The population size is a key control parameter, since a large population wastes computation while a small one limits

Algorithm 1 Utility-weighted Thompson Sampling for the evolution of inductive biases

Require: bias pool $\mathcal{B}^t$; posteriors $\{(\alpha_i, \beta_i)\}$; feedback C_t
Ensure: offspring configurations $\{b'_p, b'_{pq}\}$

$U_{\max} \leftarrow \max_{b \in \mathcal{B}^t} U(f_b)$
for each $b_i \in \mathcal{B}^t$ **do**
  $\tilde{\theta}_i \sim \mathrm{Beta}(\alpha_i, \beta_i)$
end for
$p \leftarrow \arg\max_i \tilde{\theta}_i; \quad q \leftarrow \arg\max_{i \neq p} \tilde{\theta}_i$
$b'_p \leftarrow \mathrm{LLM}(b_p, C_t)$
$b'_{pq} \leftarrow \mathrm{LLM}(b_p, b_q, C_t)$
$y_p \leftarrow \mathbb{I}[U(f_{b'_p}) > U_{\max}]; \quad y_{pq} \leftarrow \mathbb{I}[U(f_{b'_{pq}}) > U_{\max}]$
$(\alpha_p, \beta_p) \leftarrow (\alpha_p + y_p,\ \beta_p + 1 - y_p)$
for $i \in \{p, q\}$ **do**
  $w_i \leftarrow \exp(U(f_{b_i})/\rho) / \sum_{j \in \{p,q\}} \exp(U(f_{b_j})/\rho)$
  $(\alpha_i, \beta_i) \leftarrow (\alpha_i + w_i y_{pq},\ \beta_i + w_i(1 - y_{pq}))$
end for
return $\{b'_p, b'_{pq}\}$

coverage (Kitamura & Fukunaga, 2025). After T rounds, the best piece of the pool is returned as the discovered configuration, $b^* = \arg\max_{b \in \mathcal{B}^T} U(f_b)$, which is the configuration of Definition 2.4.

4 Experiments

4.1 Baseline Methods

We compare EvoFIF against seven state-of-the-art imputation methods whose realization mechanisms differ. (1) **BRITS** (Cao et al., 2018) (bidirectional recurrence); (2) **SAITS** (Du et al., 2023) (masked self-attention); (3) **PSW-I** (Wang et al., 2025a) (optimal transport, training-free); (4) **FIM** (Seifner et al., 2025) (zero-shot foundation model); (5) **Glocal-IB** (Yang et al., 2025) (information bottleneck); (6) **HELIX** (Zhang et al., 2026) (hybrid encoding); and (7) **HyFAD** (Gao et al., 2026) (time-frequency diffusion).

Existing FOTSI methods are not used as baselines, since they target variable subset forecasting under a fixed forecaster and would have to be re-fitted for every scenario of Section 2 (Appendix A.1).

4.2 Comparative Experiments

The main comparison uses TimesNet at a horizon of 96 steps over the four missingness patterns MCAR, MAR, MNAR and BLACKOUT, injected as described in Appendix D.2, and every number we report is a downstream prediction loss relative to the one of EvoFIF,

$$\Delta_{\mathrm{pred}}(m) = \frac{L_{\mathrm{pred}}(m) - L_{\mathrm{pred}}(\text{EvoFIF})}{L_{\mathrm{pred}}(\text{EvoFIF})}, \tag{12}$$

so that a value above zero means a method loses more than EvoFIF does. Appendix F lists the complete numbers, and Figure 3 shows EvoFIF with the two best baselines of each pattern, normalized by its own loss. EvoFIF attains the lowest loss on 21 of the 36 dataset–pattern combinations and ranks in the top two on 33 of them, and where the advantage lies is the informative part. Under MCAR and MAR the methods stay within a few percent of each other, while under MNAR and BLACKOUT, the two patterns with structured missingness, several baselines lose more than twice as much on the datasets with strong temporal structure, and the leading method changes between the two groups. Fitting an imputer to the distribution of one scenario therefore degrades exactly where the structure of the series matters, while EvoFIF holds its accuracy, which is the effectiveness that U_{eff} of Definition 2.2 measures.

The imputation cost, listed in Appendix F, shows the same asymmetry. EvoFIF imputes a scenario in a fraction of a second where every baseline needs tens of seconds to minutes, three orders of

magnitude more, which is the efficiency that U_{effi} of Definition 2.2 measures. The comparison is conservative, since the baselines are trained for the target dataset and pattern while EvoFIF never retrains, and it excludes the one-off search, which is amortized over every scenario that reuses the configuration.

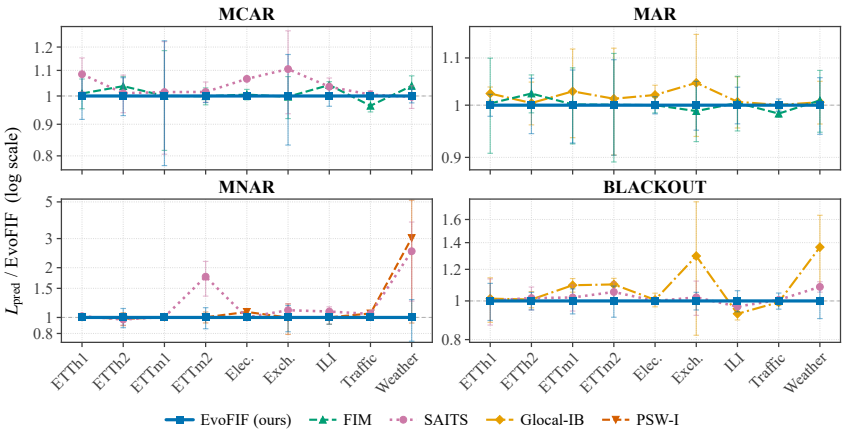


Figure 3: Downstream prediction loss of EvoFIF and the two best baselines per pattern, relative to EvoFIF.

4.3 Ablation Studies

To verify the role of each component we compare three variants, each removing one of the three designs. The variant *w/o Extraction* replaces the extraction of Section 3.2 by an initial pool drawn at random. The variant *w/o UWTS* removes the utility-weighted Thompson Sampling of Section 3.3 and evolves the single best configuration of the pool in every round, which turns the search into a greedy hill climb. The variant *w/o Utility* removes the downstream-utility and cost terms from the verification of Section 3.4.

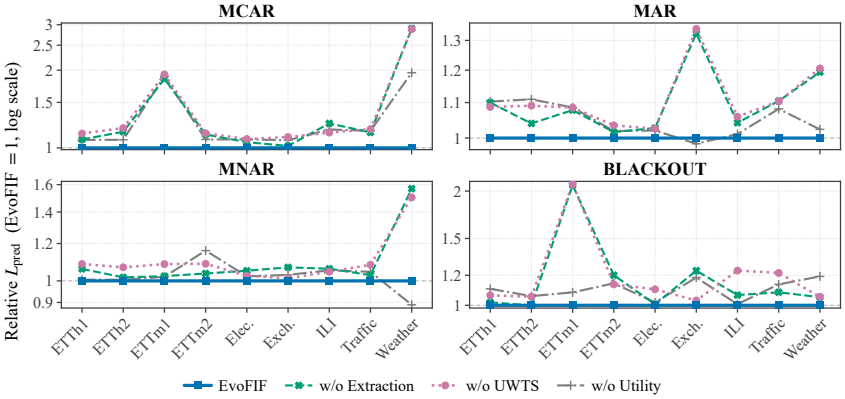


Figure 4: Ablation. Relative loss of each variant with respect to EvoFIF.

EvoFIF is better than all three variants under every missingness pattern, so none of the three designs can be dropped, and the variants differ in how much they lose. *w/o Extraction* and *w/o UWTS* degrade substantially more than *w/o Utility* under all four patterns, and on the hardest cells their loss exceeds twice that of EvoFIF while the loss of *w/o Utility* stays below that mark. The initial pool and the sampling rule therefore carry most of the gain, while the utility terms contribute a smaller but consistent improvement.

In Figure 4, 1.0 is EvoFIF itself.

4.4 Generalization Studies

This study validates the reusability term U_{reuse} of Definition 2.2 by changing the FOTSI scenario. Neither the downstream forecasting model nor the prediction horizon takes part in the search, so one and the same configuration of inductive biases faces scenarios that the search never saw, and every method is scored under the same protocol. Table 2 reports the average loss over the four missingness patterns, and Tables 7 and 8 list the complete per-dataset numbers of both axes.

EvoFIF attains the lowest loss in all six columns without any retraining, so the claim that a configuration of inductive biases stays reusable where model-centric methods have to be refitted holds beyond the forecaster and the horizon of the main comparison. The margin over the second-best method is of the same order as in Section 4.2, and the baselines are unstable across the scenario, since the second-best method changes with the forecaster.

Table 2: Generalization to forecasters and horizons unseen in the search. Best value per column in bold.

	Forecasting model			Prediction horizon		
Method	ARIMA	DLinear	XGBoost	H=192	H=336	H=720
BRITS	0.844 ± 0.032	0.444 ± 0.067	0.230 ± 0.092	0.561 ± 0.034	0.624 ± 0.027	0.671 ± 0.027
SAITS	0.823 ± 0.029	0.441 ± 0.051	0.290 ± 0.068	0.540 ± 0.025	0.594 ± 0.022	0.662 ± 0.024
PSW-I	0.838 ± 0.043	0.444 ± 0.067	0.208 ± 0.067	0.548 ± 0.025	0.624 ± 0.020	0.687 ± 0.024
FIM	0.841 ± 0.023	0.517 ± 0.093	0.312 ± 0.081	0.557 ± 0.023	0.615 ± 0.025	0.700 ± 0.032
Glocal-IB	0.821 ± 0.028	0.461 ± 0.056	0.265 ± 0.060	0.547 ± 0.024	0.598 ± 0.019	0.664 ± 0.024
HELIX	0.889 ± 0.051	0.476 ± 0.054	0.351 ± 0.087	0.594 ± 0.036	0.656 ± 0.028	0.727 ± 0.039
HyFAD	0.937 ± 0.033	5.645 ± 0.351	0.961 ± 0.224	0.606 ± 0.045	0.696 ± 0.039	0.740 ± 0.032
EvoFIF	0.773 ± 0.020	0.388 ± 0.064	0.178 ± 0.062	0.517 ± 0.020	0.568 ± 0.017	0.632 ± 0.021

4.5 Case Study

The configuration that the loop returns fills a gap according to the scale of the gap, and Appendix E reads its source line by line, traces it back to the initial pool and lists the complete code.

5 Discussion

EvoFIF meets the three requirements that FOTSI places on an imputation strategy, since it lowers the prediction loss of unseen forecasters, imputes a scenario in a fraction of a second where every baseline needs seconds to minutes, and reuses one configuration of inductive biases across datasets, patterns, forecasters and horizons without retraining.

- **Limitations.** The pool starts from classical imputation algorithms, whose assumptions are explicit, whereas the implicit inductive biases of a deep model are not part of it, so a scenario whose structure falls outside that library can only be reached by recombining what is available.
- **Insights for future work.** The extraction of inductive biases is most valuable for a task family as heterogeneous as FOTSI, so the next step is to recover the implicit biases of a deep model as explicit structure and to return them to the discovery loop, which would let the library grow with the data while keeping the reusable form that FOTSI requires.

AI use statement

AI tools assisted with translation, language polishing and the implementation of the algorithm and the experimental code.

REFERENCES

Marziyeh Afkanpour, Elham Hosseinzadeh, and Hamed Tabesh. Identify the most appropriate imputation method for handling missing values in clinical structured datasets: A systematic review. *BMC Medical Research Methodology*, 24(1):188, 2024.

David Berghaus. EVIL: Evolving interpretable algorithms for zero-shot inference on event sequences and time series with llms. *arXiv preprint arXiv:2604.15787*, 2026.

Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. BRITS: Bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pp. 6775–6785, 2018.

Dikshit Chauhan, Bapi Dutta, Indu Bala, Niki van Stein, Thomas Bäck, and Anupam Yadav. Large language models and evolutionary computation: A critical review of bidirectional interaction, automated algorithm design, and co-adaptive systems. *Computer Science Review*, 63:101050, 2026.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8(1):6085, 2018.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 785–794, 2016.

Hao Cheng, Qingsong Wen, Yang Liu, and Liang Sun. RobustTSF: Towards theory and design of robust time series forecasting with anomalies. In *International Conference on Learning Representations (ICLR)*, 2024.

MinGyu Choi and Changhee Lee. Conditional information bottleneck approach for time series imputation. In *International Conference on Learning Representations (ICLR)*, 2024.

Carlos A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11–12):1245–1287, 2002.

Domenico Cotroneo, Cristina Improta, and Pietro Liguori. Human-written vs. AI-generated code: A large-scale study of defects, vulnerabilities, and complexity. In *2025 IEEE 36th International Symposium on Software Reliability Engineering (ISSRE)*, pp. 252–263. IEEE, 2025.

Wenjie Du, David Cote, and Yan Liu. SAITS: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023.

Wenjie Du, Jun Wang, Linglong Qian, Yiyuan Yang, Zina Ibrahim, Fanxing Liu, Zepu Wang, Haoxin Liu, Zhiyuan Zhao, Yingjie Zhou, Wenjia Wang, Kaize Ding, Yuxuan Liang, B. Aditya Prakash, and Qingsong Wen. TSI-Bench: Benchmarking time series imputation. *arXiv preprint arXiv:2406.12747*, 2024.

Hongfan Gao, Wangmeng Shen, Bin Yang, and Jilin Hu. HyFAD: Hybrid time-frequency diffusion with frequency-aware embedding for time series imputation. *arXiv preprint arXiv:2606.05239*, 2026.

Qi Hao, Yue Gao, Runchang Liang, Yunhe Zhang, and Pengyang Wang. Generative imputation with multi-level causal consistency for variable subset forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2025a.

Qi Hao, Runchang Liang, Yue Gao, Hao Dong, Wei Fan, Lu Jiang, and Pengyang Wang. Is precise recovery necessary? a task-oriented imputation approach for time series forecasting on variable subset. *IEEE Transactions on Knowledge and Data Engineering*, 37(11):6464–6477, 2025b.

Jongseon Kim, Hyungjoon Kim, HyunGi Kim, Dongjun Lee, and Sungroh Yoon. A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artificial Intelligence Review*, 58(216), 2025. doi: 10.1007/s10462-025-11223-9.

Tomofumi Kitamura and Alex Fukunaga. Is selection all you need in differential evolution? *arXiv preprint arXiv:2506.14425*, 2025.

Marine Le Morvan and Gaël Varoquaux. Imputation for prediction: Beware of diminishing returns. In *International Conference on Learning Representations (ICLR)*, 2025.

Runchang Liang, Qi Hao, Yue Gao, Kunpeng Liu, Lu Jiang, Pengyang Wang, and Minghao Yin. Imputation via domain adaptation: Rethinking variable subset forecasting from knowledge transfer. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2025.

Fei Liu, Xialiang Tong, Mingxuan Yuan, Xi Lin, Fu Luo, Zhenkun Wang, Zhichao Lu, and Qingfu Zhang. Evolution of heuristics: Towards efficient automatic algorithm design using large language model. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Alberto Mino Calero, Adil Rasheed, and Anastasios M. Lekkas. ChronoSHAP: Revealing temporal patterns from transformers and LTSF-linear models in time series forecasting. *Knowledge-Based Systems*, 331:114802, 2026.

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.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96, 2018.

Patrick Seifner, Kostadin Cvejoski, Antonia Körner, and Ramsés J. Sánchez. Zero-shot imputation with foundation inference models for dynamical systems. In *International Conference on Learning Representations (ICLR)*, pp. 24296–24342, 2025.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Weimin Song, Qianqian Ren, Xingfeng Lv, and Jinbao Li. Meta-contrastive task adaptation for time series forecasting: A structured dual-teacher framework. *Information Fusion*, 132:104238, 2026.

Giovanni Squillero and Alberto Tonda. Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization. *Information Sciences*, 329:782–799, 2016.

Daniel J. Stekhoven and Peter Bühlmann. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933. doi: 10.1093/biomet/25.3-4.285.

Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520–525, 2001.

Hao Wang, Zhengnan Li, Haoxuan Li, Xu Chen, Mingming Gong, Bin Chen, and Zhichao Chen. Optimal transport for time series imputation. In *International Conference on Learning Representations (ICLR)*, pp. 828–852, 2025a.

Jun Wang, Wenjie Du, Yiyuan Yang, Linglong Qian, Wei Cao, Keli Zhang, Wenjia Wang, Yuxuan Liang, and Qingsong Wen. Deep learning for multivariate time series imputation: A survey. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 10696–10704, 2025b.

Zhixian Wang, Linxiao Yang, Liang Sun, Qingsong Wen, and Yi Wang. Task-oriented time series imputation evaluation via generalized representers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations (ICLR)*, 2023.

Shengli Xue and Lijun Zhang. VarDim-Transformer: A unified framework for variable-dimension time series with unknown missing patterns. *Journal of Systems Science and Complexity*, 39(2): 872–894, 2026.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=Bb4VGOWELI.

Jie Yang, Kexin Zhang, Guibin Zhang, Philip S. Yu, and Kaize Ding. Glocal information bottleneck for time series imputation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park, and Guojie Song. Reevo: Large language models as hyper-heuristics with reflective evolution. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

Jiacheng You, Xinyang Chen, Yu Sun, Weili Guan, and Liqiang Nie. Meta guidance: Incorporating inductive biases into deep time series imputers. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/2c570b0f9938c7a58a612e5b00af9cc0-Abstract-Conference.html.

Hsiang-Fu Yu, Nikhil Rao, and Inderjit S. Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 847–855, 2016.

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

Aoqian Zhang, Shaoxu Song, Yu Sun, and Jianmin Wang. Learning individual models for imputation. In *Proceedings of the 35th IEEE International Conference on Data Engineering (ICDE)*, pp. 160–171, 2019.

Fengming Zhang, Wenjie Du, Huan Zhang, Ke Yu, and Shen Qu. HELIX: Hybrid encoding with learnable identity and cross-dimensional synthesis for time series imputation. In *Forty-third International Conference on Machine Learning (ICML)*, 2026.

Kexin Zhang, Baoyu Jing, K. Selçuk Candan, Dawei Zhou, Qingsong Wen, Han Liu, and Kaize Ding. Cross-domain conditional diffusion models for time series imputation. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, 2025.

Yingyi Zhong, Wen'an Zhou, and Zhixian Wang. A survey of data augmentation in domain generalization. *Neural Processing Letters*, 57(34), 2025. doi: 10.1007/s11063-025-11747-9.

A Related Work

A.1 Existing Methods for Forecast-Oriented Imputation

The methods that optimize imputation for a downstream forecaster fall into the three categories of Table 1, and they differ in where the forecasting signal enters the imputation.

Evaluation-guided methods keep the imputer and the forecaster separate and estimate the effect that an imputation has on the downstream task. Generalized Representers (Wang et al., 2024) treat the causal effect of an imputed value on the prediction loss as a sample influence and estimate it by a first-order Taylor expansion, so that a candidate imputation can be scored without retraining the forecaster. The estimate holds while the compared imputations stay close and the loss stays smooth, so its error grows for abrupt imputations and for predictors with sharp transitions, and the per-time-step integration of the influence ignores the temporal dependence between the steps that are filled.

Joint optimization methods train the two components together, so that the prediction loss reaches the imputation module. TOI-VSF back-propagates that loss into a variable-subset fitting procedure (Hao et al., 2025b), VIDA aligns the feature distributions of complete and partial observations by optimal transport so that the imputer sees a representation closer to the complete data (Liang et al., 2025), and GIMCC anchors imputation on time-invariant causal relations that are shared between the training and the test period (Hao et al., 2025a). All three address variable subset forecasting (VSF), in which the observed variable set and the forecasting task are fixed in advance and the imputation is learned together with the forecaster that consumes it. The forecasting signal is exact in all three, since it is the training objective itself, and the price is the cost of the joint training plus the coupling between the learned imputer and the forecaster whose gradient shaped it.

Representation learning methods constrain the imputation representation instead of the imputation output. TimeCIB (Choi & Lee, 2024) optimizes an information-theoretic objective that preserves the task-relevant information of the series while discarding the noise that the missing values introduce, which makes the representation transferable across tasks, at the cost of a fitting procedure for every dataset and of a proxy objective in place of the forecasting loss itself.

What the three categories share is the level at which they operate. Each of them delivers its strategy as part of a model fitted to a particular set of scenarios, so the strategy is bound to the data distribution, the missingness pattern, the forecaster and the horizon that the fitting saw (Zhang et al., 2025; Xue & Zhang, 2026), and a change on any of those axes invalidates it.

A.2 The Classical Algorithms Behind the Initial Pool

The initial pool of Section 3.2 is extracted from seven classical imputation algorithms, chosen so that each contributes a distinct and explicit inductive bias and, together, they span the classical imputation practice that time series benchmarks routinely evaluate (Du et al., 2024).

We use mean, forward fill (LOCF), KNN, MissForest, IIM, TRMF and XGBoost, because each of them carries a distinct and explicit inductive bias and, together, they span the classical imputation practice that time series benchmarks routinely evaluate (Du et al., 2024). Mean embodies level preservation and forward fill embodies state persistence at the temporal boundary (Du et al., 2024). IIM learns an individual regression model from the nearest observed samples of each missing entry (Zhang et al., 2019), while KNN exploits the similarity of neighbouring samples (Troyanskaya et al., 2001). TRMF models temporal dependence through a latent temporal structure (Yu et al., 2016), and MissForest and XGBoost contribute priors from non-linear variable dependence and from gradient-boosted ensembles (Stekhoven & Bühlmann, 2012; Chen & Guestrin, 2016). Each of them encodes an interpretable and verifiable imputation assumption, which the search can later recombine. We start from classical algorithms because their assumptions are explicit and can be evolved further, whereas the biases inside a deep network are distributed over its parameters and are hard to convert into configurations that are explicit and verifiable (Mino Calero et al., 2026), and structurally simple imputation methods remain competitive at lower cost (Afkanpour et al., 2024).

A.3 Explicit Structure in Time Series Modeling

Deep imputation models are the predominant approach and reach their accuracy by learning temporal patterns end to end, with recurrent networks (Cao et al., 2018), attention-based models (Du et al., 2023) and conditional diffusion models (Tashiro et al., 2021) as representative designs. Their structure, however, remains implicit. Whatever regularity such a model relies on is distributed over its parameters, and the model has to be retrained for a new dataset or missingness pattern (Wang et al., 2025b). A parallel line of work makes part of the structure explicit. Trend and seasonality are separated by hand-designed decomposition inside forecasting architectures (Zeng et al., 2023). Distribution shift is handled by instance normalization that is detached from the architecture. Periodicity and local shape are encoded by two-dimensional variation modeling and by patching (Wu et al., 2023). Cross-variable dependence is expressed through a graph that is either given or learned , and a learned spatio-temporal prior can be inserted into a diffusion imputer.

The work closest to ours in spirit is Meta Guidance (You et al., 2025), which designs two domain-specific inductive biases for imputation, one for non-stationarity through a proximity principle and one for periodicity through weights over historical periods, and injects them into deep imputers with a small learnable module. Its experiments report an average reduction of imputation error by 27.39 percent across nine datasets and several architectures, which supports the premise of this paper that explicit structure deserves to become part of the model. The biases there are designed by hand and are bound to the module that injects them, so the choice of biases and their relative weight stay a manual design problem, and the resulting object is a network component that has to be fitted to the data at hand. EvoFIF takes a different starting point. The biases themselves are the object of the search, they are decoupled from any model, and their value is measured by the downstream forecasting utility instead of imputation error.

Table 3: Structure that prior work makes explicit in time series modeling and how it is obtained.

Work	Explicit structure	How it is obtained	Coupled to one model
Autoformer, DLinear (Zeng et al., 2023)	trend and seasonality	hand-designed decomposition	yes, architectural
RevIN	distribution shift	hand-designed normalization	no, detached
TimesNet, PatchTST (Wu et al., 2023)	periodicity, local shape	hand-designed transforms	yes, architectural
GRIN, PriSTI	cross-variable dependence, spatio-temporal prior	graph that is given or learned, learned prior	yes, coupled to the imputer
Meta Guidance (You et al., 2025)	non-stationarity, periodicity	hand-designed biases, fused by a learnable module	yes, injected into the imputer
EvoFIF (this work)	the biases and the rules that route them	discovered by search over classical seeds	no, instantiated per scenario on demand

A.4 LLM-guided Evolutionary Search

Searching over programs rather than parameters has a long history in evolutionary computation, and the recent change is that a language model can act as the variation operator. OPRO shows that a language model can improve a written object such as a prompt from its own evaluations (Yang et al., 2024). EoH and ReEvo move the search object to executable heuristics and add reflection on the search history (Liu et al., 2024; Ye et al., 2024). FunSearch and AlphaEvolve search programs that are scored by an automatic evaluator and reach new results in mathematics and in computing systems (Romera-Paredes et al., 2024; Novikov et al., 2025). EVIL applies the same scheme to executable algorithms for event sequences and time series (Berghaus, 2026). Surveys of the interaction between language models and evolutionary computation summarise these mechanisms and their remaining limits (Chauhan et al., 2026), and a language model tends to converge to a restricted region of program space, which is why EvoFIF combines local mutation with crossover and allocates its budget by posterior sampling (Russo et al., 2018; Squillero & Tonda, 2016).

The feedback that the search receives differs from these applications. In FunSearch and AlphaEvolve the score is a property of the program itself, whereas an imputation rule is only as good as the

forecasting performance it enables on the series it fills, so the evaluator of EvoFIF has to run the downstream forecaster and to charge a candidate for the computation it needs. The role of the discovered object also differs. The programs those works return are used as solvers, whereas the configuration that EvoFIF returns is executed as a preprocessing step that is reused across datasets, patterns, forecasters and horizons without retraining, which is the property formalised in Section 2.

Table 4: LLM-guided evolutionary search. The last row is the setting of this paper.

Work	Search object	Feedback	Domain
OPRO (Yang et al., 2024)	prompts for a solver	task score	optimisation
EoH (Liu et al., 2024)	heuristics in code	benchmark score	combinatorial optimisation
ReEvo (Ye et al., 2024)	heuristics with reflection	benchmark score	combinatorial optimisation
FunSearch (Romera-Paredes et al., 2024)	programs	exact or automatic evaluator	mathematics
AlphaEvolve (Novikov et al., 2025)	programs	evaluators	mathematics, computing systems
EVIL (Berghaus, 2026)	executable algorithms	zero-shot inference score	event sequences, time series
EvoFIF (this work)	configurations of inductive biases	downstream forecasting utility and cost	forecast-oriented imputation

B PROOFS

This appendix develops the reformulation of Section 2. We first analyse how a scenario-bound parameterization behaves when a fitted imputation model is transferred to another scenario (§B.1), then establish the identification between imputation inductive biases and reusable imputation functions (§B.2), and finally derive Proposition 2.1 from the two (§B.3).

B.1 TRANSFER OF A SCENARIO-BOUND IMPUTATION MODEL

Let $\ell_\tau(\theta) = L_{\text{pred}}(f_\theta; \tau)$ be the objective that a parametric imputation model f_θ induces on scenario τ, and let

$$\Theta^*_\tau = \arg\min_\theta \ell_\tau(\theta) \tag{13}$$

be the set of its minimizers, i.e. the parameters that a fitting procedure returns on τ.

Lemma B.1. For any two scenarios τ_s, τ_t and any $\theta \in \Theta^*_{\tau_s}$, $\ell_{\tau_t}(\theta) \geq \min_{\theta'} \ell_{\tau_t}(\theta')$, with equality if and only if $\Theta^*_{\tau_s} \cap \Theta^*_{\tau_t} \neq \emptyset$.

Proof. The inequality is the definition of the minimum. Since $\Theta^*_{\tau_t}$ collects all minimizers of ℓ_{τ_t}, every parameter, θ included, satisfies $\ell_{\tau_t}(\theta) \geq \min_{\theta'} \ell_{\tau_t}(\theta')$. Equality holds exactly when θ is itself a minimizer of ℓ_{τ_t}, that is, exactly when $\Theta^*_{\tau_s} \cap \Theta^*_{\tau_t} \neq \emptyset$. If the two optimal sets are disjoint, every $\theta \in \Theta^*_{\tau_s}$ is non-optimal for τ_t and the inequality is strict.

The lemma says nothing yet about how likely the two sets are to intersect, and that depends on how the scenario enters the objective. With $au = (D, R, g, H)$, the pattern R decides which entries are observed, the dataset D changes the distribution over which the expectation is taken, and the forecaster g together with the task H changes the composition $g(\cdot\,; H) \circ f_\theta$ and hence the gradient of the loss with respect to θ. Each of the four coordinates therefore changes the objective as a function of θ, and the minimizer of a changed objective is not in general a minimizer of the original one, so the two optimal sets are disjoint generically and the transfer regret

$$\text{Reg}\big(f_{\theta^*_{\tau_s}}; \tau_t\big) = \ell_{\tau_t}\big(\theta^*_{\tau_s}\big) - \min_\theta \ell_{\tau_t}(\theta) \geq 0 \tag{14}$$

is strictly positive. A model-centric strategy can thus guarantee reuse only when the target scenario happens to share the optimal parameters of the source scenario, precisely the condition that the diversity of missing patterns, datasets, forecasters and horizons violates. A guarantee that holds for *all* admissible target scenarios cannot be built from parameters.

B.2 Imputation Inductive Biases and Reusable Imputation Functions

Let $\mathcal{F}_{\text{reuse}}$ be the set of imputation functions that can be executed on any target scenario without retraining.

Lemma B.2. $\text{Inst}(\mathcal{B}) = \mathcal{F}_{\text{reuse}}$.

Proof. ($\subseteq$) Take $b \in \mathcal{B}$. By Definition 2.3, b is decoupled from the scenario and able to guide the inference of missing values on any target scenario, so $\text{Inst}(b)$ requires no retraining and $\text{Inst}(\mathcal{B}) \subseteq \mathcal{F}_{\text{reuse}}$. ($\supseteq$) Take $f \in \mathcal{F}_{\text{reuse}}$. Since f can be applied to scenarios it was never fitted for, it is a structural assumption for inferring missing values that does not depend on a particular scenario, and it is executable by assumption. By Definition 2.3 it is therefore an imputation inductive bias, so some $b \in \mathcal{B}$ satisfies $\text{Inst}(b) = f$ and $\mathcal{F}_{\text{reuse}} \subseteq \text{Inst}(\mathcal{B})$.

The abstract requirement that the object of optimization must be usable on unseen scenarios and the concrete statement that this object is a configuration of inductive biases therefore pick out the same set of functions. The plural carries meaning. A single inductive bias covers one structural aspect, such as local continuity, periodicity, non-stationarity or cross-variable dependence, whereas a FOTSI scenario typically exhibits several at once, so the elements of $\mathcal{B}$ are configurations of complementary biases and the identification of Lemma B.2 holds at the level of configurations. A configuration of inductive biases is thus the name of the class whose reusability the term of Definition 2.2 already enforces, and it adds no modeling choice of its own.

B.3 Proof of Proposition 2.1

Proof. The reusability term equation 4 vanishes for a function f exactly when its transfer regret vanishes on every target scenario, that is, exactly when $f \in \mathcal{F}_{\text{reuse}}$. A strictly positive weight γ on that term therefore confines the maximization of equation 2 to $\mathcal{F}_{\text{reuse}}$. Any function outside the class performs no better on the optimization scenarios than some function inside it, and it performs worse on the target scenarios, so it cannot be optimal. By Lemma B.2 the feasible set can be reparameterized by the biases space, and for every b the objective takes the value $U(f_b)$ of its instantiation, so

$$\max_{f \in \mathcal{F}_{\text{reuse}}} U(f) = \max_{b \in \mathcal{B}} U\big(\text{Inst}(b)\big) = \max_{b \in \mathcal{B}} U(f_b), \tag{15}$$

which is equation 6.

Remark. If $\gamma = 0$, the utility ignores reuse and the reformulation collapses to the model-centric problem, in which the optimizer is free to return the parameters θ^*_τ fitted to the optimization scenarios. The equivalence of Proposition 2.1 is thus driven by the requirement of retraining-free reuse rather than by a preference for an explicit bias representation. Effectiveness and efficiency alone do not force it, and the requirement of retraining-free reuse is exactly what model-centric FOTSI methods fail to satisfy.

C Analysis of the Sampling Strategy

C.1 The Bandit Model of Parent Selection

We build on the randomized probability matching rule of Thompson sampling (Thompson, 1933).

The selection rule of Section 3.3.1 is a stochastic bandit. Every element b_i of the pool $\mathcal{B}^t$ is an arm, one round pulls the arm whose offspring will be generated next, and the reward of a pull is the success of that offspring, that is, whether it beats the best utility of the pool. Two properties of the setting are what make the problem tractable. The arms are independent, because a candidate is scored on the same evaluation cells whatever the rest of the pool contains, so the merit of one arm does not depend on the outcomes of the others. The merit of an arm is also stationary while the search runs, because the pool changes only through the accept and reject decisions of the search itself, so the latent probability that an arm yields an improvement does not drift. Both properties hold for any fixed evaluation protocol, and we will refer to them as the bandit model,

$$y_i^{(t)} \mid \theta_i \;\sim\; \text{Bernoulli}(\theta_i), \qquad \theta_i \;\sim\; \text{Beta}(1,1), \tag{16}$$

where the uniform prior is the honest choice before the first evaluation, since no base algorithm is universally better than another (Wolpert & Macready, 1997) and none stays best across datasets and missingness settings (Du et al., 2024). The Beta prior is conjugate to the Bernoulli likelihood, so after s_i successes and f_i failures the posterior of an arm is again a Beta, $\theta_i \mid H_t \sim \mathrm{Beta}(1+s_i, 1+f_i)$, which is the same Beta form the sampler maintains, and the posterior of the pool factorises over the arms because the arms are independent. The rule of Algorithm efalg:exploration draws one sample per arm from this posterior and takes the two largest samples.

C.2 Why the Rule Balances Exploration and Exploitation

Two properties of the rule explain the balance it reaches. The first is that the probability with which an arm is pulled equals the posterior probability that this arm is optimal.

Proposition C.1 (Probability matching). Let H_t denote the history of the first t rounds and let i_t be the arm that the rule of Algorithm efalg:exploration pulls, and write $\theta^\star = \max_j \theta_j$ for the best latent success probability in the pool. Under the model of equation 16,

$$\Pr(i_t = i \mid H_t) \;=\; \Pr(\theta_i = \theta^\star \mid H_t)\,.$$

Proof. Conditional on H_t the latent parameters are independent with $\theta_j \mid H_t \sim \mathrm{Beta}(1+s_j, 1+f_j)$, so the sampled vector $(\tilde\theta_1, \ldots, \tilde\theta_n)$ is a draw from the posterior law of $(\theta_1, \ldots, \theta_n)$. The rule pulls i if and only if $\tilde\theta_i$ exceeds every other sample, and the probability of that event under the posterior is the posterior probability that θ_i is the largest coordinate, which is $\Pr(\theta_i = \theta^\star \mid H_t)$. □

Proposition C.1 is the formal content of the balance. An arm is played in proportion to how likely it is to be the best one, so an arm whose posterior is concentrated above the others is exploited almost every round, while an arm whose posterior is still wide keeps being explored, because a wide posterior places mass on large values even when its mean is low.

The second property is that the probability of pulling an arm that is not optimal decays with the evidence behind the pool, which is the mechanism behind the logarithmic regret of the rule. Write $\Delta_i = \theta^\star - \theta_i$ for the gap of such an arm.

Lemma C.1 (Optimism and pessimism). On the event that the rule pulls arm i, at least one of two statements holds. Either the sample of arm i overestimates its own mean by half the gap, $\tilde\theta_i \geq \theta_i + \Delta_i/2$, or the sample of every arm j with $\theta_j = \theta^\star$ underestimates its mean by half the gap, $\tilde\theta_j \leq \theta^\star - \Delta_i/2$.

Proof. Assume that neither statement holds. Then the sample of arm i satisfies $\tilde\theta_i < \theta_i + \Delta_i/2 = \theta^\star - \Delta_i/2$, while every optimal arm satisfies $\tilde\theta_j > \theta^\star - \Delta_i/2$. The sample of arm i is therefore strictly smaller than the sample of an optimal arm, so a rule that pulls a largest sample cannot pull arm i, which contradicts the event. □

Concentration makes both events rare. The posterior of an arm is the law of a Bernoulli mean after $s_i + f_i$ observations, so Hoeffding's inequality applies to it up to the contribution of the uniform prior, and the tail of $\mathrm{Beta}(1+s_i, 1+f_i)$ decays as

$$\Pr\Big(\tilde\theta_i \geq \theta_i + \varepsilon \mid H_t\Big) \leq e^{-2n_i\varepsilon^2}, \qquad \Pr\Big(\tilde\theta_i \leq \theta_i - \varepsilon \mid H_t\Big) \leq e^{-2n_i\varepsilon^2}, \tag{17}$$

where $n_i = s_i + f_i$ counts the times the offspring of arm i has been evaluated. Combining equation 17 with Lemma C.1 bounds the probability that a suboptimal arm i is pulled at round t by $2e^{-n_i\Delta_i^2/2} + 2e^{-n_{j^\star}\Delta_i^2/2}$, the sum of one term that vanishes as arm i is measured and one term that vanishes as the best arm accumulates evidence. Summing over the rounds shows that a suboptimal arm is pulled only $O(\Delta_i^{-2}\log T)$ times in expectation, so the regret $\sum_i \Delta_i\,\mathbb{E}[n_i(T)]$ grows logarithmically in the number of rounds and matches the lower bound of the problem up to constants. The derivation also shows what the rule does not need. The optimism in a sample is exactly the width of the posterior, so no confidence bonus has to be calibrated as in an upper-confidence-bound rule, and the rate at which an arm is revisited falls out of the posteriors instead of being scheduled as in an epsilon-greedy rule.

Two elements of the derivation carry over to the search as a whole. The uniform prior of equation 16 requires that the first round treats the seeds as equals, and the contraction of equation 17 requires that every pull contributes one observation of the arm that was pulled. The first is satisfied by construction, since the pool is initialized from the seeds before any of them is scored. The second is what the credit assignment of the next subsection has to preserve for the two-parent operator.

C.3 The Weighted Update for Two Parents Preserves the Guarantees

In its classical form Thompson Sampling attributes the reward of a pull to the arm that was pulled, so one experiment yields one unit of evidence for one posterior. A crossover offspring has two parents, which raises the question of where its reward belongs.

Attributing the observation to both parents in full would add two units of evidence for a single experiment, and because the width of a Beta posterior is governed by the total pseudo-count, both posteriors would concentrate faster than the data warrant, exploration would shrink early, and the sampling rule would lose the balance it was chosen for. Attributing the observation to one parent alone would discard the contribution of the other parent and leave its record inconsistent with the role it played, so a parent that is weak on its own yet combines well would be punished for evidence it helped to produce.

Our update gives each parent a strictly positive weight and normalises the two weights to sum to one, as in Eq. equation 8, so that the posteriors move by

$$\alpha_i \leftarrow \alpha_i + w_i\, y_{pq}, \qquad \beta_i \leftarrow \beta_i + w_i\,(1 - y_{pq}), \qquad i \in \{p, q\}, \tag{18}$$

with $w_p + w_q = 1$. Three consequences follow. First, the posterior remains a Beta posterior that is a valid posterior for a Bernoulli mean, because a fractional pseudo-count is the conjugate update for a weighted observation, and the total amount of evidence grows by exactly one per round, which is the same budget that the single-parent case consumes. Second, the rate at which the posteriors concentrate is unchanged, so the exploration that the rule derives from posterior width is preserved. The weighting decides how the evidence is distributed, while the number of experiments fixes how much evidence exists. Third, both weights are strictly positive, so a parent whose recombinations succeed keeps receiving evidence even when its own utility is low and retains a chance to be revisited, while the larger weight follows the higher utility and keeps the ordering of the pool informative. The two operators are consistent with a single rule, since mutation is the special case $w_p = 1$, and the temperature ρ of Eq. equation 8 only determines how the unit of evidence is split. The guarantees of the sampling rule therefore hold for any $\rho > 0$, which is what allows one posterior mechanism to serve both operators.

C.4 Cost of the Sampling Machinery

Let n be the size of the pool, k the number of candidates that are kept after truncation and T the number of rounds. One round samples a Beta variate for every candidate and takes the two largest samples, which costs $O(n)$, and it truncates the pool, which costs $O(n \log n)$ when the pool is sorted and $O(n)$ when the k best elements are selected by partitioning. The posterior update costs $O(1)$. Against this stand the two LLM calls and the two evaluations that a round requires, and an evaluation runs an instantiated rule together with the downstream forecaster on a scenario, which exceeds the selection cost by orders of magnitude. Over T rounds the machinery therefore costs $O(T\, n \log n)$ while the budget is spent on the $O(T)$ evaluations, so the sampling policy adds no measurable overhead to the search.

D Experimental Setup

D.1 Datasets and Splits

Table 5 lists the nine multivariate benchmarks. They cover the domains and the sampling frequencies that the literature on time series imputation and forecasting routinely evaluates (Du et al., 2024), and they differ by two orders of magnitude in the number of variables, which is what makes a reusable configuration a meaningful object on this suite. Every series is split chronologically into 60 percent for training, 20 percent for validation and 20 percent for testing, without shuffling, and all reported

errors come from the test segment. Each dataset is normalized per variable with the statistics of its training segment, and the missingness is injected after normalization, so that a mask and a rate mean the same thing on every series.

Table 5: The nine multivariate benchmarks, with the number of variables, the number of steps, the sampling frequency and the domain.

Dataset	Variables	Steps	Frequency	Domain
ETTh1	7	17,420	hourly	transformer temperature
ETTh2	7	17,420	hourly	transformer temperature
ETTm1	7	69,680	15 minutes	transformer temperature
ETTm2	7	69,680	15 minutes	transformer temperature
Electricity	321	26,304	hourly	client consumption
Exchange	8	7,588	daily	exchange rates
ILI	7	966	weekly	influenza-like illness
Traffic	862	17,544	hourly	road occupancy
Weather	21	52,696	10 minutes	weather station

D.2 Missingness Injection

We inject four patterns into the complete training and test segments and keep the masks fixed afterwards, so every method faces exactly the same missing entries in a given cell. Let $R_{t,d}$ be the indicator that the entry at time t and variable d is observed.

Under *MCAR* each entry is removed independently with probability r, so $\Pr(R_{t,d} = 0) = r$ for every pair (t, d).

Under *MAR* the missingness of a variable depends on the observed values of another variable. Half of the variables are drawn as drivers, and every remaining variable d is paired with one driver g, whose standardized value $z_{g,t}$ sets the removal probability

$$\Pr(R_{t,d} = 0) = \sigma(z_{g,t} + c)\,, \tag{19}$$

where σ is the logistic function and the offset c is calibrated so that the average of the probabilities equals r. The removal then depends on the data through a variable that is itself observed, which is the defining property of MAR.

Under *MNAR* the missingness depends on the value that is removed. For every variable the entries whose absolute standardized value exceeds the $(1 - r)$ quantile of that variable are set to missing, so the removed entries are the extreme ones and no observed variable reveals them.

Under *BLACKOUT* whole stretches of the series disappear. Contiguous segments covering all variables are placed at random positions, with lengths drawn between one and fifty steps and at least ten steps between adjacent segments, until the fraction r of the time steps is covered. This pattern simulates the failure of a sensor over an interval rather than the loss of scattered entries, and it is the pattern in which the temporal structure of the data is the only remaining source of information.

The rate protocol follows the search. Three missingness rates are sampled at the beginning of a search and each of the three runs of a cell uses one of them, so a reported number averages over both the randomness of the mask and the strength of the missingness, and no method can be tuned to one particular rate. The three rates of the main comparison are 0.250, 0.393 and 0.480.

D.3 Downstream Evaluation and Metrics

Every method is evaluated by what its imputation enables rather than by how closely it reproduces the removed entries. The filled series is passed to a forecaster that predicts the next H steps of every variable, and the main comparison uses TimesNet (Wu et al., 2023) with a backcast window of 96 steps and a label window of 48 steps, which is the standard configuration of that model. The generalization experiments replace the forecaster with ARIMA, DLinear (Zeng et al., 2023) and XGBoost, and the horizon experiments use $H \in \{96, 192, 336, 720\}$. The main comparison reports $H = 96$.

The reported prediction error is the mean squared error of the forecast,

$$L_{\text{pred}} = \frac{1}{|\Omega|} \sum_{(t,d)\in\Omega} \left(y_{t,d} - \hat{y}_{t,d}\right)^2, \tag{20}$$

where Ω is the forecast window and y and $\hat{y}$ are the ground truth and the prediction. The reported imputation error is the root mean squared error over the entries that were removed,

$$L_{\text{imp}} = \sqrt{\frac{1}{|\Omega_M|} \sum_{(t,d)\in\Omega_M} \left(x_{t,d} - \hat{x}_{t,d}\right)^2}, \tag{21}$$

where $\Omega_M = \{(t,d) : R_{t,d} = 0\}$ is the set of missing entries of the test segment. The two measures answer different questions, since a low reconstruction error does not imply a low forecasting error, which is why the main comparison reports L_{pred} and Appendix F reports both. We also report the wall-clock time of the imputation step, measured on the same machine for every method.

Each cell is run three times, once per missingness rate with a fixed mask seed, and the tables report the mean over the three runs together with the mean of the per-run standard deviations. A discovered configuration is applied to a new dataset, pattern, forecaster or horizon without any retraining, so the same object is scored in all of the cells of Section 4.4.

D.4 Implementation Details

The search runs for a fixed number of rounds with a pool that keeps the most useful candidates after every round, and every candidate is scored on a bounded set of evaluation cells so that one slow candidate cannot stall the search. Offspring are generated by a commercial large language model accessed through a public API, with a fixed sampling temperature of 0.7, and the same prompts are used for the extraction, the mutation and the crossover of inductive biases. For every baseline we use the official implementation with the configuration recommended in the corresponding paper, cross-checked against a unified reference implementation where available. PSW-I and FIM require no training on the target scenario, while the remaining baselines are trained on the target dataset and pattern, and EvoFIF performs a one-off search whose result is reused without retraining. All comparisons share the machine, the normalization, the masks and the evaluators, so the differences between methods come from the imputation step alone.

E An Example of EvoFIF

This appendix states what the discovery loop of Section 3 returns on the datasets of the evaluation and traces the returned configuration back to the pool it was built from.

E.1 The Discovered Configuration

The loop returns one configuration of inductive biases, **a gap-scale-adaptive rule that interpolates a short gap along time, copies the observation of the same phase one period earlier across a longer one, and falls back to the global level of the series**, whose instantiation is a single executable function. The header recorded at generation time reports the iteration in which the configuration was produced and the utility its instantiation obtained. Its structure does not depend on any dataset and can be stated on its own as in Algorithm 2. Gaps no longer than half of the dominant period are closed by interpolating along time, which preserves the local trend and level, while longer gaps are closed with the observation of the same phase one period earlier, or one period later where that one is missing, which preserves the periodic structure. Columns that carry no observation at all fall back to the global level of the series, and every originally observed entry is restored, so its instantiation never modifies what is already known.

E.2 Where It Comes From

The function can be read back onto the pool of Section 3.2. Its interpolation backbone refines the continuity and level priors of the mean-type seeds and reuses the local-regression idea of Zhang et al.

Algorithm 2 Gap-scale-adaptive imputation. The discovered configuration of inductive biases, abstracted from its realization.

Require: observed series X and mask M
Ensure: imputed series $\tilde{X}$
hide the unobserved entries of X, keeping the observed ones aside
$\tilde{X} \leftarrow$ linear interpolation of X along time, edges filled by the nearest observation
for each column d **do**
    $P_d \leftarrow$ lag of maximal autocorrelation of column d over lags $2, \ldots, P_{\max}$
    if the autocorrelation at P_d exceeds the seasonality threshold **then**
        for each unobserved entry (t, d) whose surrounding gap is longer than $P_d/2$ **do**
            if $(t - P_d, d)$ is observed **then**
                $\tilde{X}_{t,d} \leftarrow X_{t-P_d,d}$
            else if $(t + P_d, d)$ is observed **then**
                $\tilde{X}_{t,d} \leftarrow X_{t+P_d,d}$
            end if
        end for
    end if
end for
fill the remaining undefined entries with the global level of X
restore all originally observed entries into $\tilde{X}$
return $\tilde{X}$

(2019), namely to infer an entry from the observations closest to it. Its periodic copy generalizes state persistence, since taking the value one period earlier is forward fill (LOCF) when the period is one, so the seed that knows only about the immediately preceding value has been extended into a rule that works at any period the data supports. The global-mean fallback is inherited directly from the mean seed. What the configuration adds beyond this is a mechanism that none of the seeds contains, namely an explicit, data-adaptive period detector together with the gap-scale switch that decides which of the two filling rules applies. TRMF models temporal dependence through latent factors, but it does not expose a period as an executable rule, and no other seed represents seasonality at all. The search thus recombined the priors it was given and introduced one mechanism that the priors lack, which is the qualitative counterpart of the ablation in Section 4.3, where removing the extraction of inductive biases degrades every pattern. The cost term of Section 3.4 also left a visible trace. The realization is vectorized rather than written entry by entry, so a rule that could be expensive on long series is expressed in a form cheap enough to be reused, unchanged, across all scenarios evaluated in Section 4.

F Additional Experimental Results

This appendix lists the complete per-dataset results of Section 4.2 for all seven baselines, the imputation cost behind the efficiency claim, and the per-dataset detail of the ablation.

Table 6 lists the loss of every method on every dataset under each of the four missingness patterns, which is the complete basis of Figure 3.

Table 6: Complete per-dataset results. Downstream prediction loss under each missingness pattern, that is the MSE of the TimesNet forecaster, reported as mean ± std over three independent runs. The best value of each dataset is in bold and the row of EvoFIF is highlighted. Elec., Exch. and ILI denote electricity, exchange_rate and national_illness.

Pattern	Method	ETTh1	ETTh2	ETTm1	ETTm2	Elec.	Exch.	ILI	Traffic	Weather
MCAR	BRITS	0.905 ±0.065	0.181 ±0.013	1.603 ±0.212	0.182 ±0.004	1.179 ±0.007	0.049 ±0.015	2.599 ±0.092	0.713 ±0.005	0.040 ±0.005
	SAITS	0.904 ±0.057	0.181 ±0.013	1.051 ±0.217	0.142 ±0.005	1.180 ±0.003	0.039 ±0.006	2.484 ±0.082	0.706 ±0.011	**0.033 ±0.001**
	PSW-I	0.911 ±0.064	0.186 ±0.012	1.045 ±0.205	0.148 ±0.004	1.169 ±0.024	0.061 ±0.014	2.664 ±0.090	0.761 ±0.007	0.034 ±0.001
	FIM	0.842 ±0.047	0.186 ±0.007	1.036 ±0.190	**0.140 ±0.004**	1.113 ±0.022	**0.035 ±0.003**	2.501 ±0.031	**0.677 ±0.015**	0.035 ±0.001
	Glocal-IB	0.901 ±0.063	0.182 ±0.014	1.037 ±0.198	0.142 ±0.005	1.180 ±0.003	0.039 ±0.006	2.570 ±0.088	0.710 ±0.004	0.034 ±0.001
	HELIX	0.898 ±0.058	**0.176 ±0.011**	1.064 ±0.217	0.145 ±0.003	1.432 ±0.012	0.040 ±0.005	2.504 ±0.044	0.864 ±0.003	0.034 ±0.001
	HyFAD	0.884 ±0.049	0.186 ±0.013	1.605 ±0.119	0.182 ±0.004	1.427 ±0.023	0.109 ±0.097	5.281 ±0.538	0.811 ±0.013	0.043 ±0.001
	EvoFIF	**0.834 ±0.070**	0.179 ±0.013	**1.036 ±0.237**	0.140 ±0.003	**1.107 ±0.007**	0.035 ±0.006	**2.400 ±0.090**	0.702 ±0.007	0.033 ±0.001
MAR	BRITS	0.941 ±0.025	0.165 ±0.007	1.343 ±0.007	0.192 ±0.022	1.186 ±0.023	0.039 ±0.002	2.666 ±0.236	0.725 ±0.004	0.044 ±0.002
	SAITS	0.923 ±0.007	0.163 ±0.007	1.002 ±0.110	0.150 ±0.016	1.184 ±0.023	0.037 ±0.000	2.574 ±0.215	0.721 ±0.004	0.034 ±0.002
	PSW-I	0.935 ±0.020	0.167 ±0.008	1.001 ±0.078	0.156 ±0.014	1.177 ±0.021	0.041 ±0.004	2.634 ±0.115	0.773 ±0.004	0.033 ±0.002
	FIM	0.908 ±0.087	0.167 ±0.006	0.987 ±0.076	0.147 ±0.016	1.160 ±0.018	**0.035 ±0.002**	2.482 ±0.137	**0.708 ±0.005**	0.033 ±0.002
	Glocal-IB	0.927 ±0.012	0.163 ±0.007	1.013 ±0.090	0.149 ±0.016	1.184 ±0.023	0.037 ±0.004	2.488 ±0.128	0.721 ±0.003	0.033 ±0.001
	HELIX	0.930 ±0.011	**0.162 ±0.005**	1.005 ±0.092	0.150 ±0.014	1.448 ±0.020	0.037 ±0.002	**2.441 ±0.155**	0.873 ±0.003	0.033 ±0.003
	HyFAD	0.916 ±0.041	0.166 ±0.007	1.343 ±0.103	0.205 ±0.015	1.160 ±0.020	0.037 ±0.005	16.142 ±1.645	0.771 ±0.003	0.044 ±0.005
	EvoFIF	**0.905 ±0.020**	0.163 ±0.009	**0.985 ±0.072**	**0.147 ±0.014**	**1.160 ±0.020**	0.035 ±0.002	2.471 ±0.091	0.720 ±0.009	**0.033 ±0.002**
MNAR	BRITS	0.916 ±0.035	0.189 ±0.029	1.211 ±0.183	0.446 ±0.380	1.114 ±0.003	0.387 ±0.175	3.238 ±0.211	0.716 ±0.015	0.235 ±0.109
	SAITS	0.932 ±0.027	**0.172 ±0.013**	1.097 ±0.042	0.304 ±0.072	1.113 ±0.002	0.411 ±0.039	3.082 ±0.209	0.717 ±0.013	0.157 ±0.079
	PSW-I	0.923 ±0.015	0.173 ±0.016	1.093 ±0.041	0.173 ±0.013	1.199 ±0.058	**0.371 ±0.078**	2.841 ±0.258	0.722 ±0.012	0.188 ±0.131
	FIM	1.051 ±0.180	0.261 ±0.165	1.410 ±0.032	0.884 ±0.019	1.369 ±0.023	0.589 ±0.297	2.886 ±0.205	0.878 ±0.013	0.298 ±0.030
	Glocal-IB	0.942 ±0.026	0.174 ±0.015	1.106 ±0.046	0.291 ±0.078	1.117 ±0.003	0.463 ±0.117	2.939 ±0.227	0.715 ±0.014	0.222 ±0.015
	HELIX	0.993 ±0.060	0.180 ±0.017	1.129 ±0.028	0.494 ±0.285	1.312 ±0.029	0.647 ±0.137	2.924 ±0.132	0.777 ±0.018	0.202 ±0.125
	HyFAD	1.016 ±0.036	0.279 ±0.091	1.217 ±0.012	0.290 ±0.108	1.269 ±0.041	0.764 ±0.006	3.270 ±0.097	0.717 ±0.022	0.091 ±0.030
	EvoFIF	**0.916 ±0.023**	0.177 ±0.024	**1.090 ±0.025**	**0.172 ±0.025**	**1.111 ±0.011**	0.372 ±0.067	**2.837 ±0.256**	**0.683 ±0.012**	**0.062 ±0.018**
BLACKOUT	BRITS	0.944 ±0.115	0.188 ±0.028	1.306 ±0.020	0.185 ±0.031	1.188 ±0.013	0.095 ±0.087	2.355 ±0.040	0.834 ±0.092	0.047 ±0.013
	SAITS	0.916 ±0.121	0.171 ±0.011	0.970 ±0.073	0.142 ±0.008	**1.156 ±0.024**	0.034 ±0.003	2.339 ±0.032	0.722 ±0.009	0.036 ±0.001
	PSW-I	0.960 ±0.102	**0.166 ±0.006**	0.952 ±0.067	0.139 ±0.019	1.181 ±0.031	0.055 ±0.021	2.464 ±0.158	0.815 ±0.019	0.034 ±0.003
	FIM	0.947 ±0.118	0.171 ±0.009	0.966 ±0.082	0.141 ±0.012	1.321 ±0.206	0.036 ±0.006	2.446 ±0.232	0.718 ±0.024	0.035 ±0.004
	Glocal-IB	0.926 ±0.118	0.169 ±0.008	1.041 ±0.041	0.149 ±0.005	1.163 ±0.046	0.044 ±0.016	**2.244 ±0.079**	**0.713 ±0.013**	0.046 ±0.009
	HELIX	0.922 ±0.134	0.168 ±0.007	0.977 ±0.107	0.151 ±0.002	1.867 ±0.032	0.037 ±0.005	2.351 ±0.072	0.887 ±0.027	0.037 ±0.002
	HyFAD	0.977 ±0.073	0.239 ±0.061	1.106 ±0.040	0.218 ±0.020	1.263 ±0.040	0.311 ±0.367	2.578 ±0.098	0.781 ±0.039	0.063 ±0.005
	EvoFIF	**0.914 ±0.097**	0.168 ±0.009	**0.951 ±0.069**	**0.135 ±0.012**	1.156 ±0.004	**0.034 ±0.002**	2.416 ±0.147	0.718 ±0.033	**0.033 ±0.003**

Tables 7 and 8 give the per-dataset detail behind Table 2. Both axes are scored with the configuration that the search returned, without any retraining.

Table 7: Complete results across forecasting models. Downstream prediction loss, that is the MSE of ARIMA, DLinear and XGBoost, averaged over the four missingness patterns and reported as mean ± std over three independent runs. The best value of each dataset is in bold and the rows of EvoFIF are highlighted.

Forecaster	Method	ETTh1	ETTh2	ETTm1	ETTm2	Elec.	Exch.	ILI	Traffic	Weather
ARIMA	BRITS	0.945 ±0.015	0.371 ±0.009	1.222 ±0.029	0.409 ±0.017	1.214 ±0.022	0.274 ±0.052	**2.442 ±0.116**	0.644 ±0.004	0.074 ±0.025
	SAITS	0.956 ±0.020	0.376 ±0.013	1.159 ±0.023	0.403 ±0.013	1.195 ±0.010	0.219 ±0.064	**2.388 ±0.092**	0.640 ±0.011	0.067 ±0.016
	PSW-I	0.945 ±0.017	0.355 ±0.008	1.142 ±0.033	0.346 ±0.015	1.287 ±0.009	0.214 ±0.033	**2.367 ±0.061**	0.798 ±0.024	0.086 ±0.185
	FIM	0.962 ±0.016	0.378 ±0.008	1.224 ±0.025	0.475 ±0.011	1.263 ±0.052	0.178 ±0.011	**2.309 ±0.071**	0.691 ±0.006	0.092 ±0.005
	Glocal-IB	0.966 ±0.024	0.364 ±0.013	1.180 ±0.030	0.385 ±0.017	1.201 ±0.025	0.234 ±0.057	**2.332 ±0.075**	0.646 ±0.011	0.082 ±0.005
	HELIX	0.963 ±0.028	0.387 ±0.025	1.157 ±0.038	0.443 ±0.069	1.483 ±0.016	0.332 ±0.142	**2.398 ±0.109**	0.783 ±0.008	0.054 ±0.021
	HyFAD	0.982 ±0.024	0.394 ±0.028	1.358 ±0.020	0.555 ±0.011	1.380 ±0.011	0.441 ±0.114	**2.516 ±0.070**	0.732 ±0.006	0.077 ±0.009
	EvoFIF	0.924 ±0.018	0.335 ±0.008	1.119 ±0.025	0.334 ±0.009	1.171 ±0.012	0.158 ±0.025	**2.236 ±0.076**	0.638 ±0.005	0.040 ±0.006
DLinear	BRITS	0.323 ±0.055	0.175 ±0.021	0.360 ±0.080	0.117 ±0.025	0.400 ±0.069	0.212 ±0.037	**1.987 ±0.274**	0.312 ±0.022	0.115 ±0.025
	SAITS	0.358 ±0.062	0.167 ±0.012	0.359 ±0.069	0.093 ±0.020	0.397 ±0.061	0.151 ±0.033	**2.060 ±0.166**	0.298 ±0.024	0.082 ±0.016
	PSW-I	0.342 ±0.064	0.169 ±0.013	0.303 ±0.046	0.064 ±0.012	0.496 ±0.048	0.210 ±0.033	**1.930 ±0.349**	0.380 ±0.014	0.103 ±0.022
	FIM	0.446 ±0.096	0.196 ±0.020	0.431 ±0.058	0.142 ±0.017	0.527 ±0.084	0.186 ±0.012	**2.246 ±0.488**	0.368 ±0.046	0.111 ±0.020
	Glocal-IB	0.365 ±0.060	0.167 ±0.011	0.396 ±0.074	0.099 ±0.024	0.408 ±0.069	0.193 ±0.038	**2.110 ±0.199**	0.307 ±0.022	0.099 ±0.006
	HELIX	0.392 ±0.069	0.171 ±0.017	0.363 ±0.061	0.139 ±0.060	0.554 ±0.062	0.224 ±0.015	**1.961 ±0.164**	0.407 ±0.027	0.071 ±0.006
	HyFAD	0.451 ±0.097	0.211 ±0.037	0.484 ±0.035	0.160 ±0.095	0.526 ±0.187	0.342 ±0.081	**48.138 ±2.578**	0.371 ±0.033	0.122 ±0.014
	EvoFIF	0.311 ±0.072	0.157 ±0.013	0.294 ±0.047	0.067 ±0.013	0.382 ±0.059	0.149 ±0.020	**1.781 ±0.320**	0.290 ±0.022	0.062 ±0.006
XGBoost	BRITS	0.210 ±0.077	0.034 ±0.020	0.258 ±0.078	0.063 ±0.022	0.201 ±0.110	0.107 ±0.068	**1.059 ±0.168**	0.123 ±0.056	0.015 ±0.233
	SAITS	0.261 ±0.077	0.030 ±0.016	0.320 ±0.102	0.079 ±0.033	0.219 ±0.105	0.099 ±0.034	**1.439 ±0.183**	0.130 ±0.048	0.028 ±0.013
	PSW-I	0.232 ±0.096	0.023 ±0.014	0.220 ±0.050	0.034 ±0.021	0.297 ±0.126	0.107 ±0.034	**0.763 ±0.184**	0.169 ±0.069	0.028 ±0.008
	FIM	0.401 ±0.116	0.034 ±0.017	0.360 ±0.052	0.109 ±0.026	0.469 ±0.260	0.100 ±0.012	**1.073 ±0.152**	0.207 ±0.092	0.052 ±0.006
	Glocal-IB	0.262 ±0.084	0.027 ±0.013	0.335 ±0.101	0.066 ±0.028	0.227 ±0.105	0.119 ±0.039	**1.160 ±0.102**	0.133 ±0.059	0.057 ±0.008
	HELIX	0.332 ±0.136	0.049 ±0.041	0.289 ±0.061	0.098 ±0.043	0.492 ±0.155	0.154 ±0.006	**1.488 ±0.222**	0.240 ±0.096	0.017 ±0.019
	HyFAD	0.357 ±0.128	0.109 ±0.060	0.438 ±0.263	0.096 ±0.012	0.421 ±0.299	0.300 ±0.118	**6.680 ±1.039**	0.204 ±0.071	0.045 ±0.024
	EvoFIF	0.204 ±0.108	0.022 ±0.014	0.208 ±0.051	0.025 ±0.014	0.183 ±0.106	0.074 ±0.018	**0.764 ±0.182**	0.113 ±0.056	0.008 ±0.007

Table 8: Complete results across prediction horizons. Downstream prediction loss of the TimesNet forecaster at $H = 192$, 336 and 720 steps, averaged over the four missingness patterns and reported as mean ± std over three independent runs. The best value of each dataset is in bold and the rows of EvoFIF are highlighted. ILI has no cell at the two longer horizons, because its test segment is shorter than the horizon.

Horizon	Method	ETTh1	ETTh2	ETTm1	ETTm2	Elec.	Exch.	ILI	Traffic	Weather
H=192	BRITS	0.804 ±0.033	0.171 ±0.009	1.155 ±0.039	0.241 ±0.111	1.249 ±0.023	0.121 ±0.040	**2.307 ±0.139**	0.673 ±0.012	0.076 ±0.005
	SAITS	0.811 ±0.033	0.168 ±0.007	1.056 ±0.051	0.195 ±0.036	1.233 ±0.022	0.111 ±0.026	**2.290 ±0.171**	0.668 ±0.014	0.077 ±0.012
	PSW-I	0.814 ±0.039	0.170 ±0.009	1.054 ±0.039	0.177 ±0.017	1.240 ±0.018	0.135 ±0.027	**2.385 ±0.144**	0.716 ±0.028	0.080 ±0.026
	FIM	0.815 ±0.041	0.170 ±0.006	1.120 ±0.041	0.176 ±0.014	1.249 ±0.034	0.133 ±0.014	**2.309 ±0.164**	0.689 ±0.023	0.103 ±0.009
	Glocal-IB	0.809 ±0.032	0.168 ±0.008	1.059 ±0.046	0.194 ±0.027	1.239 ±0.026	0.135 ±0.031	**2.294 ±0.132**	0.674 ±0.012	0.097 ±0.008
	HELIX	0.828 ±0.049	0.172 ±0.013	1.072 ±0.034	0.254 ±0.074	1.407 ±0.033	0.190 ±0.037	**2.298 ±0.168**	0.740 ±0.017	0.087 ±0.033
	HyFAD	0.826 ±0.036	0.203 ±0.027	1.136 ±0.031	0.238 ±0.079	1.293 ±0.035	0.285 ±0.086	**4.665 ±0.632**	0.770 ±0.056	0.095 ±0.008
	EvoFIF	0.784 ±0.043	0.165 ±0.011	1.032 ±0.037	0.161 ±0.015	1.206 ±0.016	0.092 ±0.017	**2.167 ±0.142**	0.647 ±0.012	0.052 ±0.007
H=336	BRITS	0.982 ±0.036	0.176 ±0.011	1.108 ±0.034	0.276 ±0.083	**1.224 ±0.009**	0.153 ±0.021	–	0.922 ±0.019	0.148 ±0.003
	SAITS	0.983 ±0.028	0.166 ±0.009	0.978 ±0.035	0.233 ±0.021	**1.212 ±0.018**	0.149 ±0.029	–	0.922 ±0.018	0.106 ±0.016
	PSW-I	0.983 ±0.030	0.168 ±0.012	0.976 ±0.027	0.230 ±0.014	**1.237 ±0.015**	0.170 ±0.024	–	1.082 ±0.012	0.143 ±0.026
	FIM	0.979 ±0.038	0.170 ±0.013	1.040 ±0.032	0.230 ±0.014	**1.233 ±0.061**	0.163 ±0.012	–	0.961 ±0.020	0.143 ±0.009
	Glocal-IB	0.985 ±0.028	0.168 ±0.011	0.975 ±0.032	0.229 ±0.019	**1.217 ±0.016**	0.158 ±0.028	–	0.923 ±0.014	0.132 ±0.006
	HELIX	0.997 ±0.042	0.169 ±0.010	0.983 ±0.041	0.259 ±0.045	**1.439 ±0.023**	0.191 ±0.020	–	1.086 ±0.015	0.122 ±0.029
	HyFAD	1.016 ±0.032	0.215 ±0.035	1.136 ±0.048	0.311 ±0.058	**1.396 ±0.039**	0.250 ±0.060	–	1.082 ±0.011	0.159 ±0.029
	EvoFIF	0.943 ±0.030	0.162 ±0.014	0.952 ±0.032	0.207 ±0.012	**1.163 ±0.010**	0.132 ±0.014	–	0.909 ±0.019	0.079 ±0.007
H=720	BRITS	0.877 ±0.026	0.194 ±0.012	0.983 ±0.059	0.301 ±0.009	1.184 ±0.015	0.274 ±0.045	–	**1.211 ±0.030**	0.343 ±0.018
	SAITS	0.877 ±0.029	0.194 ±0.009	0.908 ±0.046	0.269 ±0.013	1.185 ±0.016	0.289 ±0.027	–	**1.224 ±0.023**	0.352 ±0.029
	PSW-I	0.880 ±0.023	0.197 ±0.012	0.916 ±0.056	0.282 ±0.017	1.225 ±0.017	0.331 ±0.023	–	**1.273 ±0.038**	0.392 ±0.009
	FIM	0.873 ±0.040	0.199 ±0.011	0.977 ±0.064	0.284 ±0.014	**1.292 ±0.072**	0.314 ±0.009	–	1.247 ±0.025	0.416 ±0.017
	Glocal-IB	0.878 ±0.027	0.194 ±0.010	0.897 ±0.047	0.262 ±0.018	1.180 ±0.021	0.301 ±0.027	–	**1.214 ±0.026**	0.389 ±0.014
	HELIX	0.898 ±0.041	0.195 ±0.013	0.921 ±0.072	0.298 ±0.039	**1.440 ±0.025**	0.307 ±0.041	–	1.375 ±0.035	0.381 ±0.048
	HyFAD	0.896 ±0.035	0.236 ±0.040	1.011 ±0.068	0.320 ±0.019	**1.371 ±0.016**	0.324 ±0.033	–	1.309 ±0.015	0.452 ±0.032
	EvoFIF	0.866 ±0.029	0.185 ±0.015	0.868 ±0.053	0.243 ±0.012	1.136 ±0.010	0.258 ±0.008	–	**1.190 ±0.026**	0.310 ±0.014

Table 9: Imputation cost, averaged over the 36 settings. The final column reports the mean cost relative to EvoFIF.

Method	Mean (s)	Median (s)	×EvoFIF
BRITS	220.901	207.169	2587×
SAITS	23.033	23.879	270×
PSW-I	35.922	39.156	421×
FIM	29.095	10.606	341×
Glocal-IB	34.566	28.679	405×
HELIX	70.095	31.076	821×
HyFAD	208.138	123.930	2437×
EvoFIF	0.085	0.014	1×