Sequential Monte Carlo Learning for Time Series Structure Discovery

Feras A. Saad [1 2] **Brian J. Patton** [2] **Matthew D. Hoffmann** [2] **Rif A. Saurous** [2] **Vikash K. Mansinghka** [3 2]

Abstract

This paper presents a new approach to automatically discovering accurate models of complex time series data. Working within a Bayesian nonparametric prior over a symbolic space of Gaussian process time series models, we present a novel structure learning algorithm that integrates sequential Monte Carlo (SMC) and involutive MCMC for highly effective posterior inference. Our method can be used both in "online" settings, where new data is incorporated sequentially in time, and in "offline" settings, by using nested subsets of historical data to anneal the posterior. Empirical measurements on real-world time series show that our method can deliver 10x–100x runtime speedups over previous MCMC and greedy-search structure learning algorithms targeting the same model family. We use our method to perform the first large-scale evaluation of Gaussian process time series structure learning on a prominent benchmark of 1,428 econometric datasets. The results show that our method discovers sensible models that deliver more accurate point forecasts and interval forecasts over multiple horizons as compared to widely used statistical and neural baselines that struggle on this challenging data.

1. Introduction

Many applications depend on models that can explain and predict time series data. A key challenge in time series modeling is the substantial expertise needed to design statistical models that capture complex temporal patterns such as changepoints, heteroskedastic noise, additive and multiplicative seasonal effects, and higher-order autocorrelations (Hyndman & Athanasopoulos, 2021, Chapters 12–13). Even experts who know how to construct sophisticated time series models may have trouble deciding what types of patterns to include when working with a given dataset. Motivated by these challenges, this work addresses the problem of automatically discovering models of time series data that exhibit a wide range of patterns which are a-priori unknown.

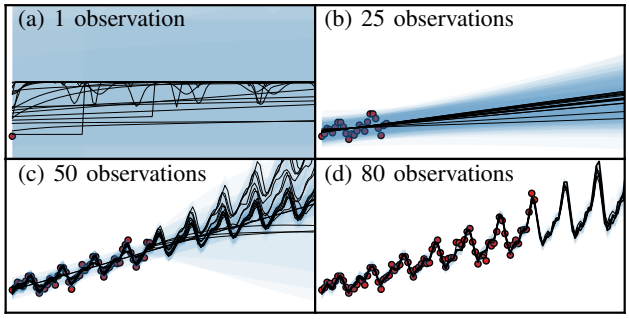

Figure 1. Real-time discovery of time series structure. Red dots show observed data so far, solid lines show mean forecasts from a weighted ensemble of learned symbolic model structures, where the thickness of a line is proportional to its corresponding model weight; blue regions show 95% prediction intervals. (a) Broad uncertainty. (b) Dominant linear trend. (c) Multimodal posterior, including linear trend with additive or with multiplicative seasonality. (d) Linear trend with multiplicative seasonality.

Given observations $\{(t_i, y_i)\}_{i=1}^n$ of time points t_i and time series values y_i, we focus on learning an unknown real function f and i.i.d. noise process ϵ such that $y(t) = f(t) + \epsilon(t), t \in \mathbb{R}$. These so-called "pure time series models", which are univariate in nature and do not include exogenous variables, are used in many disciplines. In econometrics, for example, the function f is intended to capture essential empirical features of the observed data that are believed to arise from an unknown macroeconomic structural model whose exogenous variables are too complex or impossible to accurately specify (Brooks, 2008, Chapter 5).

Figure 1 shows an example data stream and the evolution of various hypotheses about the unknown function f that our method explores as it encounters new observations. The plots illustrate some central features of our approach, which include: (i) adapting both the structure and parameters of f according to patterns in the data observed so far; and (ii) maintaining not one hypothesis about f but a weighted collection of hypotheses that together capture uncertainty over the unknown time series structure and parameters.

[1]Carnegie Mellon University [2]Google Research [3]Massachusetts Institute of Technology. Correspondence to: Feras Saad <fsaad@cmu.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

Overview To automatically discover time series models, we stochastically sample symbolic expressions in the Gaussian process family for nonparametric regression introduced in Duvenaud et al. (2013), which we specialize to univariate time series data (Roberts et al., 2013). Whereas Duvenaud et al. (2013) use a greedy-search algorithm for structure learning, our approach is based on fully Bayesian inference in a probabilistic generative model over symbolic descriptions of temporal patterns (i.e., Gaussian process covariance functions), numeric parameters, and observed data. This representation enables us to develop a novel sequential Monte Carlo (SMC) algorithm for posterior inference that naturally handles streaming data and coherently characterizes joint uncertainty about latent model structure and parameters.

Key Results The SMC algorithm introduced in this paper can deliver 10x–100x improvements in runtime vs. forecasting accuracy profiles for many real-world datasets (Fig. 5) over the greedy-search method in Duvenaud et al. (2013) (which finds a single "best-fit" model structure and parameters) and over the MCMC sampling method in Saad et al. (2019). We further evaluate our method on 1,428 monthly datasets from M3 (Makridakis & Hibon, 2000) against 9 statistical and neural baselines (Hyndman & Khandakar, 2008; Hyndman et al., 2008; Fiorucci et al., 2016; Golyandina et al., 2018; Taylor & Letham, 2018; Lindemann et al., 2021; Lim et al., 2021), many of which struggle to capture the breadth of patterns in the data (Makridakis et al., 2018). The results show that our structure-learned models produce more accurate point forecasts (improvement $\geq 14\%$) and interval forecasts (improvement $\geq 6\%$) across 1–18 step horizons (Figs. 6 and 7), with a runtime per dataset that is comparable to the most competitive baselines (Table 1).

Outline The rest of the paper is structured as follows: Section 2 reviews time series modeling using Gaussian processes; Section 3 presents our SMC structure learning method; Section 4 contains the evaluation; Section 5 discusses related work; and Section 6 offers closing remarks.

2. Gaussian Process Time Series Models

Preliminaries Let $D \coloneqq (\mathbf{t}, \mathbf{y})$ be a dataset with n time points $\mathbf{t} \coloneqq (t_1, \ldots, t_n)$ and observations $\mathbf{y} \coloneqq (y_1, \ldots, y_n)$. As the true function that generated the data is unknown, we use a Gaussian process function prior $f \sim \mathrm{GP}(0, k_\theta)$ and assume an additive observation noise model, so that $y_i = f(t_i) + \epsilon_i$, where $k_\theta \coloneqq (k, \theta)$ denotes a covariance kernel k with parameters $\theta = (\theta_1, \ldots, \theta_{d(k)}) \in \Theta_k \subset \mathbb{R}^{d(k)}$; $d(k)$ is the number of continuous parameters in k; and $\epsilon_i \overset{\text{iid}}{\sim} N(0, \eta)$. The probabilistic generative model $P_{\mathbf{t}}(y; k_\theta)$ is then

$$[f(t_1), \ldots, f(t_n)] \sim \text{MultivariateNormal}(0, k_\theta(\mathbf{t})) \quad (1)$$

$$y_i \sim \text{Normal}(f(t_i), \eta), \quad i \in [n]. \quad (2)$$

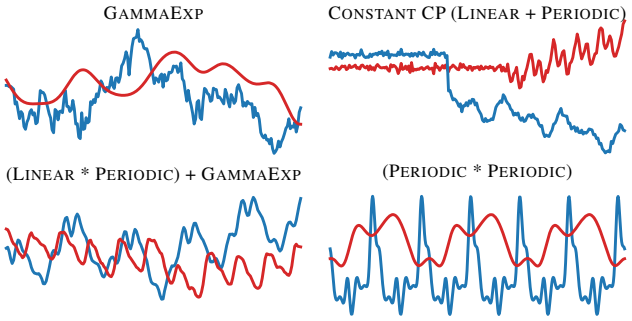

Figure 2. Example time series patterns that can be expressed in the Gaussian process model family (Eqs. (3)–(10)), including random-walk evolution, smooth variation, trends with additive and multiplicative seasonality, changepoints (CP), and multiple seasonality. For each kernel structure k, colored lines show draws of random functions $f \sim \mathrm{GP}(0, k_\theta)$ using different parameters $\theta \in \mathbb{R}^{d(k)}$.

A Flexible Modeling Language While the model (1)–(2) appears simple on the surface, Gaussian processes define a flexible class of distributions. The kernel structure k and parameters θ of the covariance function k_θ together dictate the properties of the random function f. Duvenaud et al. (2013) introduced a language $\mathcal{L}$ for expressing a wide range of Gaussian process patterns, which we restrict to time series data. Kernel structures $k \in \mathcal{L}$ are specified compositionally from simpler parts, using a context-free grammar (CFG):

$$B ::= \text{LINEAR} \mid \text{PERIODIC} \mid \text{GAMMAEXP} \mid \ldots \quad (3)$$

$$\oplus ::= + \mid * \mid \text{CP} \quad (4)$$

$$k ::= B \mid (k_1 \oplus k_2). \quad (5)$$

The symbol B denotes a finite set of "base" kernels that can be composed through binary operators $\oplus$, namely addition (+), pointwise multiplication (*), and changepoints (CP) (Rasmussen & Williams, 2006, Chapter 4). Base kernels $b \in B$ and the changepoint operator CP are associated with parameter spaces $\Theta_b \subset \mathbb{R}^{d(b)}$, $\Theta_{\text{CP}} \subset \mathbb{R}^2$, respectively. Figure 2 shows draws $f \sim \mathrm{GP}(0, k_\theta)$, for various k_θ.

A Prior Over Language Expressions To automate the process of discovering covariance functions k_θ for a given dataset, we treat all unknown quantities as latent variables in a probabilistic generative model $P_{\mathbf{t}}(k, \theta, \eta, \mathbf{y})$ defined by

$$k \sim \text{PCFG, c.f. (3)–(5)} \quad (6)$$

$$[\theta_1, \ldots, \theta_{d(k)}] \overset{\text{iid}}{\sim} \text{LogNormal}(0, 1) \quad (7)$$

$$\eta \sim \text{InverseGamma}(1, 1) \quad (8)$$

$$[f(t_1), \ldots, f(t_n)] \sim \text{MultivariateNormal}(0, k_\theta(\mathbf{t})) \quad (9)$$

$$y_i \sim \text{Normal}(f(t_i), \eta), \quad i \in [n]. \quad (10)$$

In Eq. (6), "PCFG" denotes a prior over $\mathcal{L}$ obtained by assigning a probability to each production rule in the CFG (3)–(5). To ease notation, we will use the symbol $\varphi \coloneqq (\theta, \eta)$ to denote the numeric kernel parameters and noise variance.

Problem Formulation In Eqs. (6)–(10), the posterior

$$P_{\mathbf{t}}(k, \varphi \mid \mathbf{y}) = \frac{P_{\mathbf{t}}(k, \varphi, \mathbf{y})}{\sum_{k \in \mathcal{L}} \int_{\Theta_k \times [0,\infty]} P_{\mathbf{t}}(k, \varphi, \mathbf{y}) \, \mathrm{d}\varphi}, \quad (11)$$

over (k, φ) given $\mathbf{y}$ is defined on a transdimensional space,

$$\text{support}\left(P_{\mathbf{t}}(k, \varphi \mid \mathbf{y})\right) \subset \bigcup_{k \in \mathcal{L}} \left[\{k\} \times \Theta_k \times \mathbb{R}\right]. \quad (12)$$

Our goal is to generate a set $\{(w^i, (k^i, \varphi^i))\}_{i=1}^M$ of $M \geq 1$ approximate posterior samples $(k^i, \varphi^i) \sim P_{\mathbf{t}}(k, \varphi \mid \mathbf{y})$ and weights $w^i > 0$ that can be used to solve queries, i.e., to compute posterior expectations of test functions ψ,

$$\mathbb{E}_{P_{\mathbf{t}}}\left[\psi(k, \varphi, \mathbf{t}, \mathbf{y}) \mid \mathbf{y}\right] \approx \sum_{i=1}^{M} \frac{w^i}{\sum_j w^j} \psi(k^i, \varphi^i, \mathbf{t}, \mathbf{y}). \quad (13)$$

The query may be about the posterior probability of the presence of a given temporal structure in the data, e.g.,

$$P_{\mathbf{t}}(\mathsf{Periodic} \in k \mid \mathbf{y}) = \mathbb{E}_{P_{\mathbf{t}}}\left[\mathbf{1}[\mathsf{Periodic} \in k] \mid \mathbf{y}\right], \quad (14)$$

or a prediction for new data $\mathbf{y}_*$ at time points $\mathbf{t}_*$, e.g.,

$$\mathbb{E}_{P_{\mathbf{t},\mathbf{t}_*}}\left[f(\mathbf{t}_*) \mid \mathbf{y}\right] = \mathbb{E}_{P_{\mathbf{t}}}\left[\psi^{\text{mean}}_{\mathbf{t}_*}(k, \varphi, \mathbf{t}, \mathbf{y}) \mid \mathbf{y}\right], \quad (15)$$

$$\psi^{\text{mean}}_{\mathbf{t}_*} \coloneqq \int_{-\infty}^{\infty} \mathbf{u}_* P_{\mathbf{t},\mathbf{t}_*}(f(\mathbf{t}_*) = \mathbf{u}_* \mid k, \varphi, \mathbf{y}) \, \mathrm{d}\mathbf{u}_*;$$

$$P_{\mathbf{t},\mathbf{t}_*}(\mathbf{y}_* \leq \mathbf{u} \mid \mathbf{y}) = \mathbb{E}_{P_{\mathbf{t}}}\left[\psi^{\text{CDF}}_{\mathbf{t}_*,\mathbf{u}}(k, \varphi, \mathbf{t}, \mathbf{y}) \mid \mathbf{y}\right], \quad (16)$$

$$\psi^{\text{CDF}}_{\mathbf{t}_*,\mathbf{u}} \coloneqq \int_{-\infty}^{\mathbf{u}} P_{\mathbf{t},\mathbf{t}_*}(\mathbf{y}_* = \mathbf{u}_* \mid k, \varphi, \mathbf{y}) \, \mathrm{d}\mathbf{u}_*.$$

Since $P_{\mathbf{t},\mathbf{t}_*}(f(\mathbf{t}_*) \mid k, \varphi, \mathbf{y})$ in Eqs. (15) and (16) is also a multivariate normal distribution, the test function $\psi^{\text{mean}}_{\mathbf{t}_*}$ can be analytically computed in closed form (as can other moments, see Rasmussen & Williams (2006, Chapter 2.2)), whereas $\psi^{\text{CDF}}_{\mathbf{t}_*,\mathbf{u}}$ can be computed numerically with high precision (Genz, 1992). Generally speaking, evaluating $\psi(k, \varphi, \mathbf{t}, \mathbf{y})$ pointwise is "easy"; whereas computing its posterior expectation as in Eqs. (15)–(16) requires the particle collection $\{(w^i, (k^i, \varphi^i))\}_{i=1}^M$ for the estimator (13).

3. A Sequential Monte Carlo Sampler for Time Series Structure Learning

In this section we briefly review sequential Monte Carlo (SMC) sampling and describe a novel SMC sampler for inferring a weighted collection of kernel structures and parameters that approximate the posterior distribution (11).

Background SMC methods (Del Moral et al., 2007) are designed to produce approximate samples from a sequence $\pi_0, \pi_1, \ldots, \pi_T$ of probability distributions, where each π_j is defined on a measurable space $(X_j, \mathcal{X}_j)$ can be evaluated pointwise up to a normalizing constant, $\pi_j(x) = \gamma_j(x)/Z_j$ $(j = 0, \ldots, T; Z_0 \equiv 1)$. The output of an SMC sampler at step j (for $j = 0, \ldots, T$) is a weighted "particle collection" $\{(w_j^i, x_j^i)\}_{i=1}^M$, such that each random pair (w_j^i, x_j^i) is "properly weighted" (Liu & Chen, 1998) for π_j, meaning

$$\mathbb{E}\left[w_j^i \, \psi(x_j^i)\right] = Z_j \mathbb{E}_{\pi_j}\left[\psi(x)\right], \quad (\varphi \in X_j \to \mathbb{R}). \quad (17)$$

As Eq. (17) implies that $\mathbb{E}[w_j^i] = Z_j$, the particles can be used to obtain biased but consistent estimators of expectations $\mathbb{E}_{\pi_j}[\psi(x)]$ via ratio estimation. Assuming that π_0 is tractable to sample from and $Z_0 = 1$, the first step in SMC is to generate $x_0^i \sim \pi_0$ and $w_0^i \leftarrow 1$ $(i = 1, \ldots, M)$, which gives properly weighted pairs for π_0. At step j, the particles $\{(w_{j-1}^i, x_{j-1}^i)\}_{i=1}^M$ are evolved through a two-step process

$$x_j^i \sim K_j(x_{j-1}^i, \cdot) \quad (18)$$

$$w_j^i \leftarrow w_{j-1}^i \left[\frac{\gamma_j(x_j^i) L_{j-1}(x_j^i, x_{j-1}^i)}{\gamma_{j-1}(x_{j-1}^i) K_j(x_{j-1}^i, x_j^i)}\right], \quad (19)$$

where $K_j : X_{j-1} \times \mathcal{X}_j \to [0, 1]$ and $L_{j-1} : X_j \times \mathcal{X}_{j-1} \to [0, 1]$ are Markov kernels (whose densities we also refer to as K_j and L_{j-1}). These kernels are typically designed on a per-problem basis. Subject to relatively mild conditions, the new particle collection $\{(w_j^i, x_j^i)\}_{i=1}^M$ obtained by Eqs. (18)–(19) is properly weighted for π_j. To help prevent particle collapse, if the weights $w_j^{1:M}$ at step j become skewed, each particle x_j^i is resampled to take value x_j^ι with (relative) probability w_j^ι $(\iota = 1, \ldots, M)$ and its weight is reset to the mean weight: $w_j^i \leftarrow (w_j^1 + \cdots + w_j^M)/M$.

3.1. Sequential Bayesian Structure Learning

To leverage SMC for inferring the posterior $P_{\mathbf{t}}(k, \varphi \,|\, \mathbf{y})$ in Eq. (11) we use a sequential Bayesian reasoning approach. The observations $D = (\mathbf{t}, \mathbf{y})$ are partitioned into T subsets $D_j \coloneqq (\mathbf{t}_j, \mathbf{y}_j)$ with $\mathbf{t}_j \coloneqq t_{j,1:N_j}$, and $\mathbf{y}_j \coloneqq y_{j,1:N_j}$ $(j = 1, \ldots, T; N_j > 0)$. We let $\mathbf{t}_{1:j} \coloneqq \cup_{s=1}^j \mathbf{t}_s$ and analogously for $\mathbf{y}_{1:j}$, so $(\mathbf{t}, \mathbf{y}) \equiv (\mathbf{t}_{1:T}, \mathbf{y}_{1:T})$. In online learning problems (e.g., Fig. 1) the D_j correspond to streaming new data and in offline problems to batches of previous data used to anneal the full posterior (11). The sequence of target distributions for SMC is then $P_{\mathbf{t}_{1:j}}(k, \varphi \,|\, \mathbf{y}_{1:j})$, $j = 0, \ldots, T$.

As all these target distributions are defined on the same transdimensional space (12), we use a *reweight-resample-rejuvenate* SMC method (Algorithm 1) that is similar to the iterated batched importance sampler (IBIS) of Chopin (2002). In the *reweight* step (lines 4–5), the target is updated from $\pi_{j-1} \coloneqq P_{\mathbf{t}_{1:j-1}}(k, \varphi \mid \mathbf{y}_{1:j-1})$ to $\pi_j \coloneqq P_{\mathbf{t}_{1:j}}(k, \varphi \mid \mathbf{y}_{1:j})$,. For Eq. (18), the particle $(k_{j-1}^i, \varphi_{j-1}^i)$ is evolved through a deterministic "copy" kernel K_j, i.e, $(k_j^i, \varphi_j^i) \sim \delta_{(k_{j-1}^i, \varphi_{j-1}^i)}$, so that the corresponding reverse kernel L_{j-1} is an atom at (k_j^i, φ_j^i). The incremental weight (bracketed term in Eq. (19)) is then $P_{\mathbf{t}_{1:j}}(\mathbf{y}_j \mid k_j^i, \varphi_j^i, \mathbf{y}_{1:j-1})$.

Algorithm 1 SMC Structure Learning via Data Annealing

Require: Dataset sequence $D_j := (\mathbf{t}_i, \mathbf{y}_j), j = 1, \ldots, T$; No. of particles $M > 0$, rejuv steps $N_{\text{rejuv}} \geq 0$.
Ensure: Properly weighted samples $\{(w^i, (k^i, \varphi^i))\}$ for posterior distribution $P_{\mathbf{t}_{1:T}}(k, \varphi \mid \mathbf{y}_{1:T})$.

1: ▷ *Repeat operations indexed by i over $i = 1, \ldots, M$* ◁
2: $(k_0^i, \varphi_0^i) \sim P(k, \varphi)$; $\quad w_0^i \leftarrow 1$
3: **for** $j = 1, \ldots, T$ **do**
4: $\quad (k_j^i, \varphi_j^i) \leftarrow (k_{j-1}^i, \varphi_j^i)$
5: $\quad w_j^i \leftarrow w_{j-1}^i \left[\dfrac{P_{\mathbf{t}_{1:j}}(k^i, \varphi^i, \mathbf{y}_{1:j})}{P_{\mathbf{t}_{1:j-1}}(k^i, \varphi^i, \mathbf{y}_{1:j-1})} \right]$
6: $\quad$ **if** $j < T$ **and** RESAMPLE?$(w_j^{1:M})$ **then**
7: $\quad\quad (\iota_1, \ldots, \iota_M) \leftarrow$ RESAMPLE$(w_j^{1:M})$
8: $\quad\quad (k_j^i, \varphi_j^i) \leftarrow (k_{j-1}^{\iota_i}, \varphi_{j-1}^{\iota_i})$
9: $\quad\quad w_j^i \leftarrow (w_j^1 + \cdots + w_j^M)/M$
10: $\quad$ **for** $u = 1, \ldots, N_{\text{rejuv}}$ **do**
11: $\quad\quad (k_j^i, \varphi_j^i) \sim$ INVOLUTIVEMCMC$($
$\quad\quad\quad (k, \varphi) \mapsto P_{\mathbf{t}_{1:j}}(k, \varphi, \mathbf{y}_{1:j}; (k_j^i, \varphi_j^i)))$
$\quad\quad\quad$ ▷ *refer to Section 3.2*
12: $\quad\quad \varphi_j^i \sim$ HAMILTONIANMONTECARLO$($
$\quad\quad\quad \varphi \mapsto P_{\mathbf{t}_{1:j}}(k_j^i, \varphi, \mathbf{y}_{1:j}); \varphi_j^i)$
$\quad\quad\quad$ ▷ *refer to Neal (2011)*

3.2. Involutive MCMC Rejuvenation Moves

The *rejuvenate* step (lines 10–12 of Algorithm 1) evolves the particle (k_j^i, w_j^i) through an MCMC kernel K_j that adapts the structure and parameters (k, φ) to the latest batch $(\mathbf{t}_j, \mathbf{y}_j)$ of observations. Rejuvenation is an SMC step (18)–(19) where the target distributions are equal, the backward kernel is the time reversal of K_j, and the weight is unchanged. Since K_j traverses a transdimensional space (12) over symbolic covariance expressions, we must perform posterior inference over tree-shaped data structures, à la Chipman et al. (1998); Neal (2003). We build a new class of structure learning moves based on involutive MCMC (Neklyudov et al., 2020; Cusumano-Towner et al., 2020) that generalize the greedy search operators from Duvenaud et al. (2013) and embed them in a Bayesian inference framework.

3.2.1. SUBTREE-REPLACE

Starting at (k, θ, η), the SUBTREE-REPLACE move (Fig. 3a) replaces a randomly selected subtree of (k, θ) with a fresh subtree and updates the noise η using a Gibbs-type move. More specifically, the proposal q_{R} simulates a path $b := (b_1, \ldots, b_m)$ ($m \geq 0$, $b_i \in \{0 := \text{left}, 1 := \text{right}\}$) to a node in the abstract syntax-tree representation of k. We then simulate a fresh subtree $(\tilde{k}, \tilde{\theta})$ and obtain a proposed state (k', θ') by replacing the current subtree $(\hat{k}, \hat{\theta})$ rooted at b with $(\tilde{k}, \tilde{\theta})$. We define notation for $(\hat{k}, \hat{\theta})$ and (k', θ') as

$$(\hat{k}, \hat{\theta}) \leftarrow (k_b, \theta_b), \; (k', \theta') \leftarrow \otimes_b \left(\ominus_b(k, \theta), (\tilde{k}, \tilde{\theta}) \right) \quad (20)$$

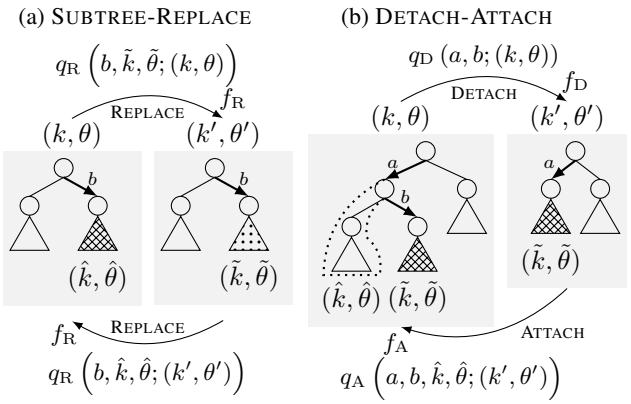

Figure 3. Involutive MCMC rejuvenation moves over the structure k and parameters θ of a Gaussian process covariance function k_θ.

where $(\cdot)_b$, $\ominus_b(\cdot)$ and $\otimes_b(\cdot, \cdot)$ are operators for extracting, deleting, and inserting subtrees rooted at b, respectively. To propose a new observation noise η' we perform a Gibbs-style move that leverages the conjugacy of the inverse-gamma prior (8) over η and normal likelihood (10) of the observations $\mathbf{y}$ for fixed $f(\mathbf{t})$ in Eq. (9). More specifically, given $(k', \theta', \eta, \mathbf{y})$, the means $\boldsymbol{\mu}' := [\mu_1', \ldots, \mu_n']$ of the random variables $[f(t_1), \ldots, f(t_n)]$ are known in closed form, which lets us sample η from its conditional distribution given $(k', \theta', f(\mathbf{t}) = \boldsymbol{\mu}', \mathbf{y})$ (proofs in Appendix A):

$$\eta' \sim \text{InverseGamma}(1 + n/2, 1 + \textstyle\sum_{i=1}^n (y_i - \mu_i')^2/2). \quad (21)$$

Let $u := (b, \tilde{k}, \tilde{\theta}, \eta')$ be the variables sampled by q_{R}. Since SUBTREE-REPLACE is its own reversal, the mapping and inverse mapping $[(k, \theta, \eta), u] \leftrightarrow [(k', \theta', \eta'), u']$ are given by a measure-preserving involution f_{R} defined by

$$f_{\text{R}}([(k, \theta, \eta), (b, \tilde{k}, \tilde{\theta}, \eta')]) := [(k', \theta', \eta'), (b, \hat{k}, \hat{\theta}, \eta)],$$

where Eq. (20) defines (k', θ') and $(\hat{k}, \hat{\theta})$ from $(k, \theta, b, \tilde{k}, \tilde{\theta})$.

Proposition 1. *If the proposal q_{R} samples the path b uniformly and $(\tilde{k}, \tilde{\theta})$ from the conditional prior given $(\ominus_b(k, \theta))$, then the involutive MCMC acceptance probability $\alpha_{\text{R}} = \min(1, r_{\text{R}})$, where*

$$r_{\text{R}} := \frac{P(\eta')}{P(\eta)} \frac{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')}{P_{\mathbf{t}}(\mathbf{y} \mid k, \theta, \eta)} \frac{1/|k'|}{1/|k|} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})}. \quad (22)$$

Remark. Even though the proposal (21) for η' resembles a Gibbs move, the ratio r_{R} includes terms for η' because it is sampled *before* deciding whether to accept (k', θ').

While the SUBTREE-REPLACE proposal is very flexible and, provided that b is root with positive probability, ensures irreducibility of the chain, it has a problematic limitation. Consider a proposed move $(k, \theta) \to (k', \theta')$ with

$$\underbrace{\text{LINEAR}_{\theta_1}}_{(k,\theta)} \to \underbrace{(\text{LINEAR}_{\theta_1'} + \text{PERIODIC}_{\theta_2', \theta_3'})}_{(k',\theta')} \quad (23)$$

which may arise in the transition from Fig. 1b to Fig. 1c. The only way for SUBTREE-REPLACE to perform such a move is to select the root and resimulate the entire expression, which (i) has a positive probability of not including LINEAR; and (ii) discards the inferred parameter θ_1 in the current state. The next move type is designed to address this limitation.

3.2.2. DETACH-ATTACH

Starting at (k, θ, η), the DETACH-ATTACH move (Fig. 3b) is composed of a pair of submoves that invert one another.

- DETACH: Replaces the subtree (k_a, θ_a) rooted at a with the subtree $(\tilde{k}, \tilde{\theta})$ rooted at (a, b); so the proposed state is $(k', \theta') \leftarrow \otimes_a(\ominus_a(k, \theta), (\tilde{k}, \tilde{\theta}))$. In Fig. 3b, the dotted region shows the discarded fragment $(\hat{k}, \hat{\theta})$ of (k_a, θ_a), which will be the "scaffold" that must be simulated in the corresponding ATTACH direction to reverse this move.
- ATTACH: Replaces the subtree $(\tilde{k}, \tilde{\theta})$ rooted at a with a new subtree (k_a, θ_a) that itself embeds $(\tilde{k}, \tilde{\theta})$ as a subtree rooted at b. The remaining components of (k_a, θ_a) other than $(\tilde{k}, \tilde{\theta})$, i.e., the scaffold, are precisely the discarded fragment $(\hat{k}, \hat{\theta})$ from the corresponding DETACH move. The proposed state is $(k', \theta') \leftarrow \otimes_a(\ominus_a(k, \theta), (k_a, \theta_a))$

Whereas SUBTREE-REPLACE is self-inverting, the DETACH-ATTACH move is associated with two proposals q_{D} and q_{A} and two maps f_{D} and f_{A} that invert one another. The DETACH direction is chosen with probability $\xi \in (0, 1)$. Proposal q_{D} simulates a path (a, b) whereas q_{A} simulates both a path (a, b) and a scaffold $(\hat{k}, \hat{\theta})$. Moreover

$$f_{\mathrm{D}}([(k, \theta, \eta), (a, b)]) := [(k', \theta', \eta'), (a, b, \hat{k}, \hat{\theta}, \eta)], \quad (24)$$

$$f_{\mathrm{A}}([(k, \theta, \eta), (a, b, \hat{k}, \hat{\theta})]) := [(k', \theta', \eta'), (a, b, \eta)]. \quad (25)$$

Proposition 2. *If q_{D} and q_{A} sample paths a uniformly and q_{A} samples scaffold $(\hat{k}, \hat{\theta})$ from the conditional prior given $[\ominus_a(k, \theta), b, (\tilde{k}, \tilde{\theta})]$, then the acceptance ratios $r_{\mathrm{D}} = r_{\mathrm{R}} \cdot (1 - \xi)/\xi \cdot q_{\mathrm{A}}(b \,|\, a; k', \theta')/q_{\mathrm{D}}(b \,|\, a; k, \theta)$ and $r_{\mathrm{A}} = r_{\mathrm{R}}^2 r_{\mathrm{D}}^{-1}$.*

3.2.3. IMPROVING ACCEPTANCE RATES VIA PSEUDO-MARGINAL PARAMETER PROPOSALS

It is well known that transdimensional MCMC algorithms (Green, 1995) often suffer from high rejection rates for moves $(k, \theta) \to (k', \theta')$ that change the structure (Brooks et al., 2003; Karagiannis & Andrieu, 2013). In particular, even though k' may model the data $\mathbf{y}$ well using some ideal parameter θ'_*, the transition is likely rejected if the proposed parameter θ' is such that $\mathbf{y}$ has low density given (k', θ'). The first two panels in the top row of Fig. 4 illustrate this idea for the move (23), where the proposed parameters θ'_2, θ'_3 in the PERIODIC subexpression have incorrect frequency and lengthscale, leading to a poor fit. We address this problem by developing an auxiliary-variable method to improve θ' during the transdimensional proposal.

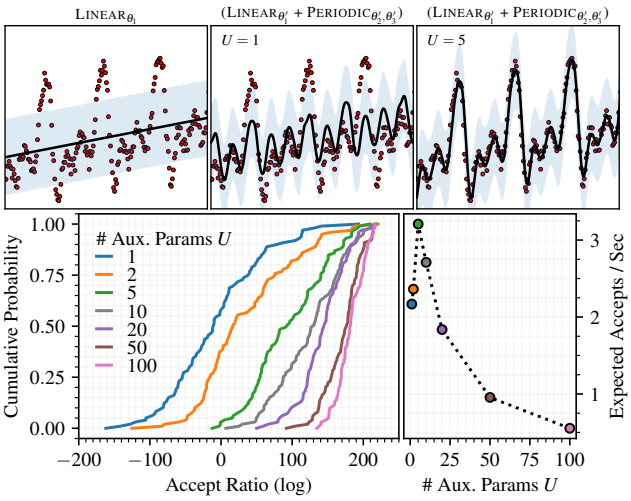

Figure 4. Top row: Structure proposal (23) using $U \in \{1, 5\}$ auxiliary parameters. Bottom row: Distribution of log acceptance ratios (58) and expected acceptances per second for various U.

For the move $(k, \theta) \to (k', \theta')$, we partition $\theta = (\theta_S, \theta_D)$ and $\theta' = (\theta'_S, \theta'_F)$, where S indexes the "shared" parameters that exist in both θ and θ' (which may have different values), D indexes the "discarded" parameters from θ that do not exist in θ', and F indexes the "fresh" parameters in θ' that do not exist in θ. The proposal simulates $U > 0$ i.i.d. auxiliary parameters $\{(\theta'^u_S, \theta'^u_F)\}_{u=1}^U$ using log-normal random walk proposals (for S) and prior proposals (for F) and selects one $\tilde{u} \in [U]$ to serve as the selected parameter θ':

$$\theta'^u_s \sim \text{LogNormal}(\theta_{s,\delta}, \delta) \quad (u = 1 \ldots U; s \in S) \quad (26)$$

$$\theta'^u_f \sim \text{LogNormal}(0, 1) \quad (u = 1 \ldots U; f \in F) \quad (27)$$

$$\tilde{u} \sim \text{Discrete}\left(\omega(\theta'^1_{S,F}), \ldots, \omega(\theta'^U_{S,F})\right), \quad (28)$$

where $\theta_{s,\delta} := \ln(\theta_s) - \delta/2$ ($\delta \geq 0$) and $\omega(\cdot)$ assigns a nonnegative weight to each $\theta'^u_{S,F}$. The optimal choice of ω that minimizes the variance of the estimate of the marginal likelihood $P_{\mathbf{t}}(k' \mid \mathbf{y})$ is precisely the importance weight

$$\omega(\theta'^u_{S,F}) = \frac{P_{\mathbf{t}}(k', \theta'^u_{S,F}, \eta, \mathbf{y})}{\prod_{s \in S} q(\theta'^u_s; \theta_s) \prod_{s \in S} q(\theta'^u_f)}, \quad (29)$$

although the general theory we develop allows any nonnegative ω, which may be faster to compute than Eq. (29). As part of the forward proposal $(k, \theta) \to (k', \theta')$, we also simulate $U - 1$ auxiliary "reverse" parameters,

$$\theta^u_s \sim \text{LogNormal}(\theta'^{\tilde{u}}_{s,\delta}, \delta) \quad (u = 1 \ldots U - 1; s \in S) \quad (30)$$

$$\theta^u_d \sim \text{LogNormal}(0, 1) \quad (u = 1 \ldots U - 1; d \in D) \quad (31)$$

$$\hat{u} \sim \text{Uniform}(1, \ldots, U). \quad (32)$$

The role of these parameters is described in the next proposition, which presents a proposal distribution, involution, and acceptance ratio for an involutive MCMC scheme using the auxiliary-variable proposals (26)–(28) and (30)–(32).

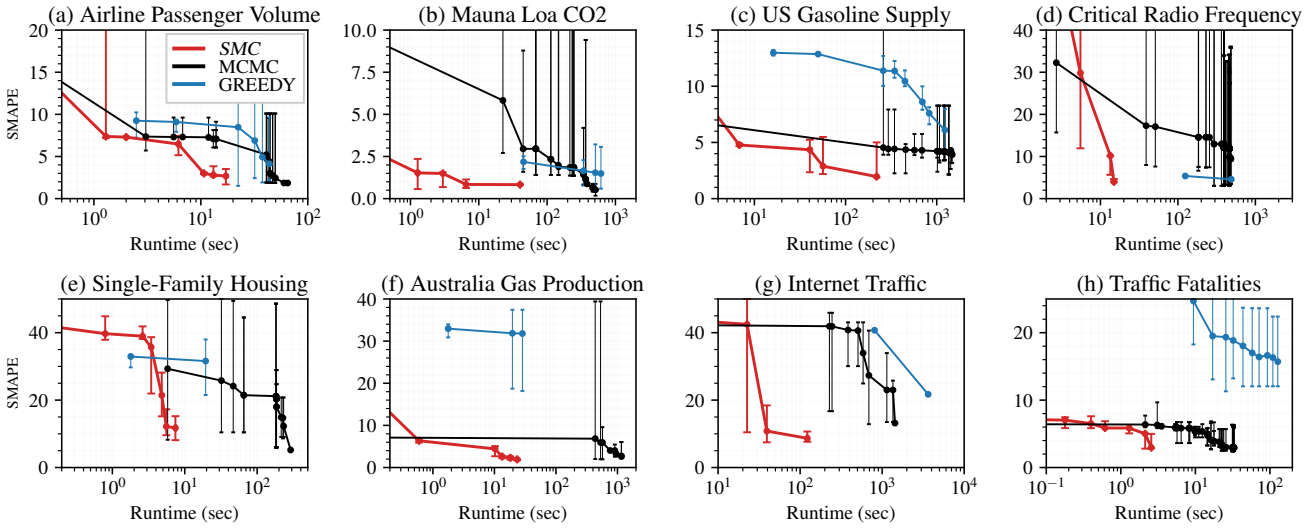

Figure 5. Runtime (x-axes, log scale) vs. forecast error (y-axes, SMAPE over 18-step forecast horizon) for eight datasets in the Time Series Data Library (Hyndman & Yangzhuoran, 2018) using three different structure discovery algorithms for temporal Gaussian processes: SMC sampling (Algorithm 1), MCMC sampling (Saad et al., 2019) and GREEDY search (Duvenaud et al., 2013).

Proposition 3. *Consider the* SUBTREE-REPLACE *move using an extended proposal* q^U_{R} *that samples auxiliary variables* (26)–(28) *and* (30)–(32)*, equipped with the involution*

$$f^U_{\mathrm{R}}([(k,\theta,\eta),(b,\tilde{k},\theta'^{1:U}_{S,F},\tilde{u},\theta^{1:U-1}_{S,D},\hat{u},\eta')]) = [(k',\theta',\eta'),(b,\hat{k},(\theta^{1:\hat{u}-1}_{S,D},\theta_{S,D},\theta^{\hat{u}:U-1}_{S,D}),\hat{u},(\theta'^{1:\tilde{u}-1}_{S,F},\theta'^{\tilde{u}+1:U}_{S,F}),\tilde{u},\eta)] \quad (33)$$

$$\text{where } (k',\theta') := \otimes_b\left(\ominus_b(k,(\theta'^{\tilde{u}}_S,\theta_D)),(\tilde{k},\theta'^{\tilde{u}}_F)\right). \quad (34)$$

If q^U_{R} *samples the path* b *uniformly and samples* $\tilde{k}$ *from the conditional prior given* $(\ominus_b k)$*, then involutive MCMC with acceptance probability* $\alpha^U_{\mathrm{R}} := \min(1,\tilde{r}^U_{\mathrm{R}})$ *defines an irreducible, aperiodic Markov chain with stationary distribution* $P_{\mathbf{t}}(k,\varphi\,|\,\mathbf{y})$*, where* $\tilde{r}^U_{\mathrm{R}}$ *is defined in Eq.* (58) *of Appendix A.*

Analogous results hold for the ATTACH-DETACH move.

Example Consider the move in Eq. (23). The top row of Fig. 4 shows how the current LINEAR model (k,θ) and two proposed models (k',θ') fit the data, with $U = 1, 5$ auxiliary parameters for sampling θ'. The second row shows the empirical distribution of acceptance ratios $\tilde{r}^U_{\mathrm{R}}$ for various U. Since k' is marginally orders of magnitude more likely for the observed data than k, increasing U leads to $\min(1,\tilde{r}^U_{\mathrm{R}}) = \alpha^U_{\mathrm{R}} \approx 1$ with overwhelming probability. Letting t_U denote the number of seconds needed to compute $\tilde{r}^U_{\mathrm{R}}$ and $N^{\text{accept}}_U = \alpha^U_{\mathrm{R}}/t_U$ the number of accepts per second, Fig. 4 shows that increasing U eventually leads to diminishing returns of $\mathbb{E}\left[N^{\text{accept}}_U\right]$, because α_U is essentially 1 for $U \geq 5$. While the optimal value of U is difficult to quantify theoretically for arbitrary moves $(k,\theta) \to (k',\theta')$, one possible choice is to set $U \propto |\theta'| \equiv d(k')$.

4. Evaluation

We implemented[1] the time series structure discovery method described in Section 3 using the Gen probabilistic programming system (Cusumano-Towner et al., 2019) and evaluated its performance against multiple automated forecasting methods. Section 4.1 compares runtime versus accuracy profiles to two previous algorithms for Gaussian processes structure learning and Section 4.2 presents runtime and accuracy results on challenging econometric forecasting benchmarks. All the experiments were conducted on a Google Cloud n2d-standard-48 instance (server specs: AMD EPYC™ 7B12 48vCPU @2.25GHz, 192 GB RAM).

4.1. Runtime Comparisons to MCMC & Greedy Search

Figure 5 shows runtime vs. forecasting accuracy profiles for eight TSDL datasets (Hyndman & Yangzhuoran, 2018) using our SMC sampler (Algorithm 1), MCMC sampling (Saad et al., 2019), and GREEDY search (Duvenaud et al., 2013). As in Duvenaud et al. (2013), for GREEDY the maximum search depth is 10, random restarts are used during parameter optimization, and structures are scored using the BIC. As our server has $M = 48$ parallel threads, we used M particles for SMC, ran M parallel chains for MCMC, and scored M structures in parallel for greedy search.

Greedy search delivers forecasts of comparable accuracy to SMC and MCMC in three datasets (Figs. 4a, 4b and 4d) and lower accuracy in five datasets (Figs. 4c and 4e–4h). In Figs. 4f and 4h, the errors from greedy search after 10 and 100 seconds have high variance (due to random restarts)

[1] Available online at https://github.com/fsaad/AutoGP.jl

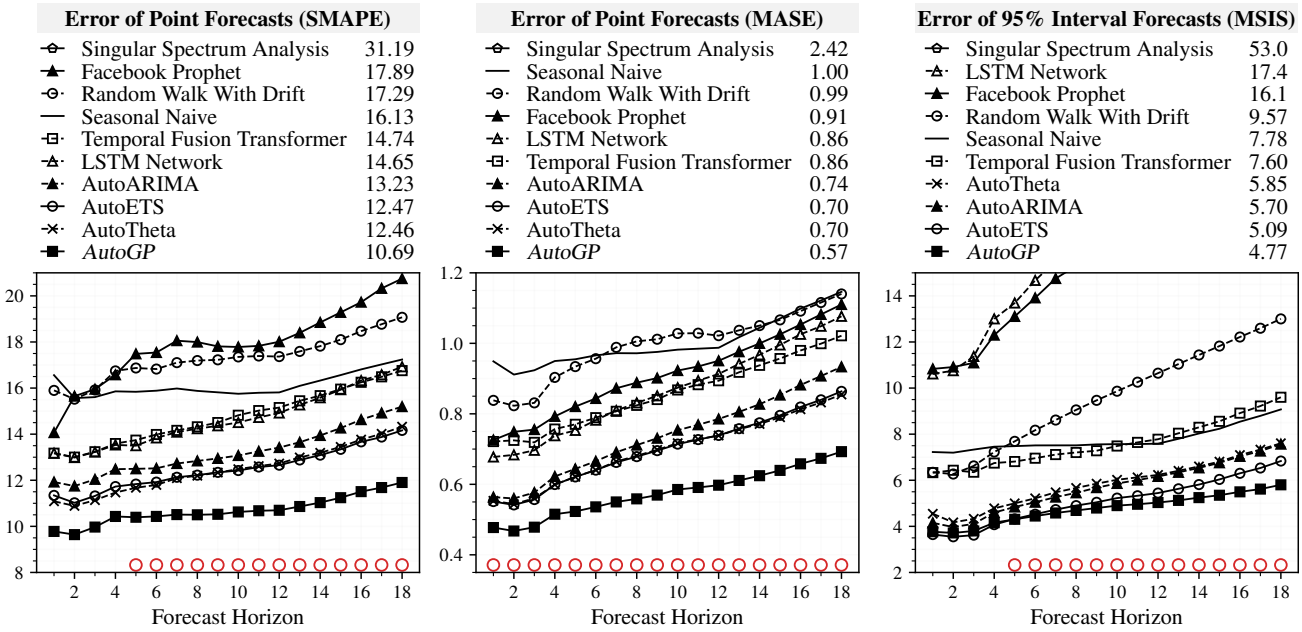

Figure 6. Error of point forecasts (SMAPE and MASE) and 95% interval forecasts (MSIS) using our method (labeled *AutoGP*) and several baselines over 18 forecast horizons. Each dot at a given horizon shows the mean error (SMAPE, MASE, MSIS) across 1,428 forecasting problems in the M3 monthly econometric time series (Makridakis & Hibon, 2000). Red circles indicate horizons where errors of *AutoGP* are statistically significantly less than errors of the next-best method (Mann-Whitney U test, $p = 0.05$ with Bonferroni correction for 54 tests). Legends show in sorted order the overall error of each method, which is the mean error across all 18 horizons.

and are over 2x higher than those obtained from SMC in 1 second. A main distinction between greedy search operators and MCMC rejuvenation operators is that the former can never decrease the size of the kernel structure at given step, whereas the latter makes stochastic accept/reject choices using reversible moves that leave the posterior invariant.

The error distributions using SMC and MCMC typically converge to similar values, although the MCMC chains exhibit higher variance and can require between 10x (Figs. 4g and 4h) to 100x (Figs. 4d and 4f) more wall-clock time. It is helpful to compare SMC and MCMC using the same overall rejuvenation budget. Consider n observations and an SMC scheme with a logarithmic annealing schedule $\mathbf{t}_{1:2}, \mathbf{t}_{1:4}, \ldots, \mathbf{t}_{1:n}$ (i.e., $\log n$ steps) and r rejuvenation moves per step. As the rejuvenation cost with n observations is $O(n^3)$ for a dense Gaussian process, the total cost of rejuvenation within an SMC particle is $\sum_{i=0}^{\log n} r(n/2^i)^3 = O(rn^3)$. An MCMC chain with the same budget performs $O(r \log(n) n^3)$ work. We thus expect SMC to deliver speedups when structures can be inferred using $n_* \ll n$ data points relatively early in the annealing schedule.

4.2. Forecasting Accuracy on Econometric Data

Motivation The next evaluation builds on the study of Makridakis et al. (2018) (which itself builds on the earlier study of Ahmed et al. (2010)) comparing the accuracy of popular machine learning forecasting methods to statistical baselines in a benchmark of over 1,428 monthly time series from the M3 dataset (Makridakis & Hibon, 2000). This previous study found that statistical methods consistently outperform more complex machine learning methods across all 18 forecasting horizons. Follow-on work by Cerqueira et al. (2022) showed that increasing the number of observations improves the performance of machine learning methods to some extent, but they still do not outperform statistical baselines on the 18-step forecast horizon.[2]

We benchmarked our time series method on the M3 monthly data and compared the results to top-performing baselines from Makridakis et al. (2018) and other popular methods that also deliver probabilistic forecasts, including: AutoARIMA (Hyndman & Khandakar, 2008), AutoETS (Hyndman et al., 2008), AutoTheta (Fiorucci et al., 2016), Facebook Prophet (Taylor & Letham, 2018), Singular Spectrum Analysis (Golyandina et al., 2018), LSTM (Lindemann et al., 2021), and Temporal Fusion Transformer (Lim et al., 2021). Refer to Appendix C for additional experimental details.

[2]The more recent study of Makridakis et al. (2023) found that deep learning ensembles can perform competitively to statistical ensembles on the M3 benchmark, but they require several months of training and hyperparameter tuning. In particular, the authors of the study write that: *"the time spent for tuning the four deep learning methods was 3,193, 512, 507 and 3,077 hours for DeepAR, Feed-Forward, Transformer, and WaveNet, respectively"*, which renders these methods highly impractical for the typical user.

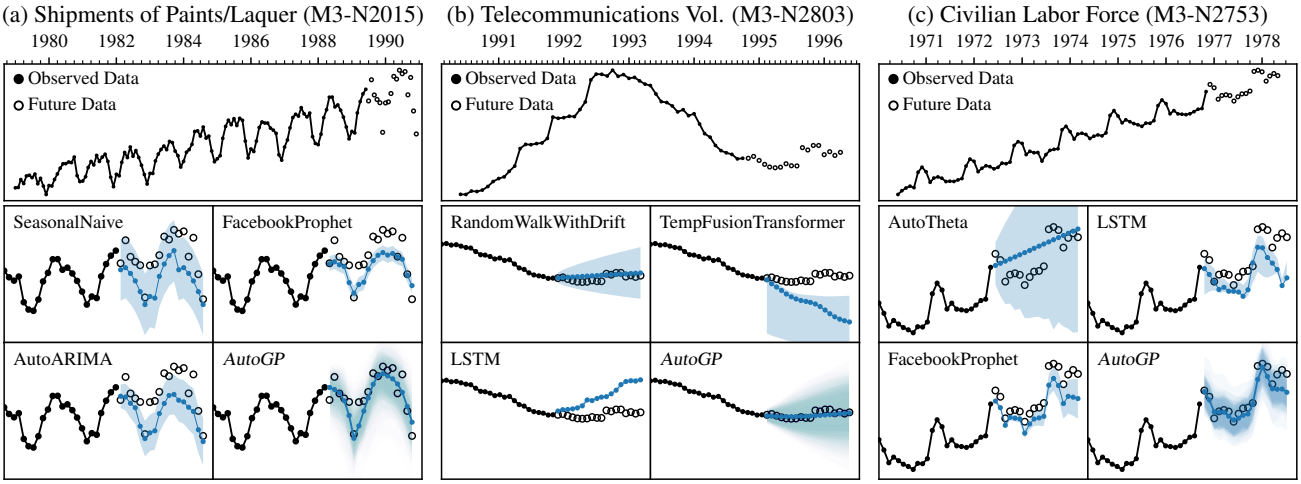

Figure 7. Forecasts on three example time series in the M3 data using our method (labeled *AutoGP*) and baselines from the evaluation in Fig. 6. Observed data is shown in black; future data is shown in white; and point forecasts and 95% interval forecasts are shown in blue.

Accuracy Figure 6 shows the forecasting accuracy results. As in Makridakis et al. (2018), point forecasts are evaluated using the SMAPE and MASE metrics and 95% interval forecasts are evaluated using the MSIS metric (refer to Eqs. (66)–(68) in Appendix C). Since fitting procedures for most forecasting methods are themselves stochastic, we report the lowest error across 10 different random seeds, which lets us better quantify the predictive capacity of each baseline independently of stochasticity in the fitting process. Our method (named *AutoGP*, analogously to e.g., AutoARIMA) delivers the lowest forecast errors across all three metrics, which are statistically significant in horizons 5–18 for SMAPE, 1–18 for MASE, and 5–18 for MSIS.

Qualitative Plots To gain more insight into the qualitative differences between our method and the baselines, we investigate forecasts on three example M3 time series in more detail. In Fig. 7a, the baselines capture the seasonal variation, however Seasonal Naive and AutoARIMA produce forecasts that are too low and, in the latter's case, with insufficient uncertainty. The forecast intervals from Facebook Prophet are also too narrow, a property that appears in several datasets (e.g., Fig. 7c) and is also observed in Hyndman & Athanasopoulos (2021, Section 12.2). In Fig. 7b, Random Walk With Drift accurately explains the data and this structure is emulated by our method, whereas LSTM and Temporal Fusion Transformer both capture incorrect dynamics. Figure 7c shows an example of how AutoTheta, a top-performing and robust statistical baseline on M3, often learns conservative models that underfit. LSTM and Facebook Prophet capture the seasonal structure but their point forecasts are too low and interval forecasts poorly calibrated. Our method also under-predicts the last three data points in Fig. 7c but they fall within the 95% prediction interval.

Runtime Table 1 shows runtime statistics per M3 dataset of each method. The average runtime of our method (*AutoGP*, 34.68 sec) lies between that of AutoETS (21.09 sec) and that of LSTM (36.71 sec). Its higher variance (8.59 sec) as compared to the baselines is due to the fact that the complexity of the learned Gaussian process structures (which take values in an unbounded space, c.f. Eqs. (3)–(5)) depends on the complexity of the underlying temporal patterns in the observed data, whereas the baselines typically optimize over a fixed model and parameter space, making their runtime more stable across different datasets.

The AutoARIMA, AutoTheta, and AutoETS baselines all leverage stepwise, enumeration-based model search algorithms that do not provide a straightforward way for users to control accuracy as a function of runtime. In contrast, users can control the runtime and accuracy trade-offs for *AutoGP* (as in Fig. 5, for example) by modifying inference hyperparameters of Algorithm 1 such as the number M of SMC particles, number N_{rejuv} of involutive MCMC rejuvenation steps, or number T of annealing steps.

Table 1. Mean and standard deviation of wall-clock runtime per dataset for various forecasting baselines in the M3 evaluation.

	Runtime / Dataset (sec)	
	Mean	Std
Facebook Prophet	0.87	0.41
Historic Average	4.02	1.07
RandomWalk With Drift	0.01	1.25
Naive	0.01	1.40
Seasonal Naive	5.20	1.64
AutoARIMA	12.79	2.31
AutoTheta	20.39	2.61
AutoETS	21.09	3.26
AutoGP	34.68	8.59
LSTM	36.71	0.34
Temporal Fusion Transformer	414.80	0.34

5. Related Work

Gaussian Process Modeling Gaussian processes are widely used models for time series data (Roberts et al., 2013). Our work builds on the automated covariance kernel discovery approach from Duvenaud et al. (2013) by introducing a new structure learning algorithm based on sequential Monte Carlo sampling in a Bayesian model over covariance expressions, numeric parameters, and data. Using this new algorithm, this work presents a quantitative evaluation that goes beyond previous work in the automated Gaussian process model discovery literature in terms of the number of datasets (D=1428) and baselines (B=9), e.g., Duvenaud et al. (2013, D=5, B=5); Lloyd et al. (2014, D=13, B=7); Janz et al. (2016, D=0, B=0); Kim & Teh (2018, D=6, B=0); Schaechtle et al. (2017, D=4, B=5); and Saad et al. (2019, D=7, B=5). Berns et al. (2022) provide a survey of automated covariance kernel discovery for Gaussian process regression models with multiple input dimensions. While our work focuses specifically on a language (3)–(5) of kernel structures that specify covariance functions for a single temporal input, it would be fruitful to apply our SMC structure learning algorithm to more expressive modeling languages that support multiple inputs.

SMC Structure Learning Sequential Monte Carlo learning algorithms have been used for Bayesian inference over structured latent spaces in a variety of settings, including probabilistic graphical models (Hamze & de Freitas, 2005; Yu et al., 2021; Naesseth et al., 2014), phylogenetic tree models (Wang et al., 2015; 2019; Moretti et al., 2021), and Dirichlet process-based models (Obermeyer et al., 2014; Saad & Mansinghka, 2018). Our approach in Algorithm 1 uses resample-move SMC (Chopin, 2002), where the sequence of target distributions is obtained via data annealing and involutive MCMC rejuvenation moves (Fig. 3) are used to adapt the structure given new observations; which is a specific instantiation of the general algorithm template in Saad (2022, Section 3.3.2) for Bayesian structure learning via SMC. Figure 5 suggests that applying this SMC algorithm template in an analogous manner to Algorithm 1 may improve upon MCMC algorithms for structure learning in other settings, such as the model families in Adams et al. (2010); Mansinghka et al. (2016); Jin et al. (2019); Cranefield & Dhiman (2021); Saad & Mansinghka (2021).

Transdimensional Auxiliary Variables Section 3.2.3 introduces a sampling-importance-resampling proposal for parameters in a transdimensional move, which we formally justify as an involutive MCMC step (Neklyudov et al., 2020; Cusumano-Towner et al., 2019) over an extended state-space with auxiliary variables (Lew et al., 2022). Previous works have also used auxiliary-variable strategies for transdimensional MCMC (Brooks et al., 2003; Al-Awadhi et al., 2004; Andrieu & Roberts, 2009)—the most closely related methods are those in Yeh et al. (2012); Karagiannis & Andrieu (2013). These two works use a single-particle, multi-step annealed importance sampler (AIS) for proposing parameters in a transdimensional move; whereas our method in Section 3.2.3 can be seen as a multi-particle, single-step AIS proposal. A natural idea is to combine these approaches by using a multi-particle, multi-step AIS transdimensional parameter proposal, which can be formally understood as an involutive MCMC move with a proposal pair as in Saad et al. (2022, Algorithms 3 and 4) and an involution function that generalizes the one in Eqs. (33)–(34).

6. Conclusion

We have presented a new approach for automatically discovering models of time series data. Our method leverages sequential Monte Carlo inference (Algorithm 1) to effectively infer the posterior distribution over symbolic Gaussian process model structures and numeric parameters that can accurately model a range of real-world datasets. We introduced involutive MCMC rejuvenation moves (Sections 3.2.1–3.2.2) that adapt the structure given new observations and a new pseudo-marginal parameter proposal (Section 3.2.3) to improve acceptance rates of transdimensional moves. These algorithmic contributions may also be of independent interest for Bayesian structure learning in many other symbolic probabilistic model families.

Empirical measurements in Section 4.1 show that our SMC sampler enables 10x–100x improvements in runtime versus accuracy profiles over previous greedy search and MCMC structure learning methods that target the same model class. Our algorithms enabled us to perform in Section 4.2 a large evaluation containing $\sim$1,400 forecasting problems drawn from the challenging M3 dataset, which show that our method can produce commonsense forecasts whose accuracy outperforms prominent statistical and neural baselines across multiple forecast horizons. These results present the first illustration of the practicality of Gaussian process structure learning on a large benchmark set that is challenging for both statistical and machine learning methods (Ahmed et al., 2010; Makridakis et al., 2018; 2023).

As our structure learning algorithm operates over dense Gaussian processes, it may be possible to work around the fundamental $O(n^3)$ scaling bottleneck by using sparse approximations (Quiñonero-Candela & Rasmussen, 2005; Rossi et al., 2021), state-space model representations (Hartikainen & Särkkä, 2010; Grigorievskiy & Karhunen, 2016), or combinations thereof (Tebbutt et al., 2021; Hamelijnck et al., 2021). These modeling variants would enable scalability beyond 10,000 observations at the expense of approximation accuracy, however, introducing new trade-offs in the modeling and inference algorithm design space.

References

Adams, R. P., Wallach, H., and Ghahramani, Z. Learning the structure of deep sparse graphical models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 1–8. PMLR, 2010.

Ahmed, N. K., Ahmed, A. F., El Gayar, N., and Hisham, E.-S. An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29(5-6): 594–621, 2010. doi:10.1080/07474938.2010.481556.

Al-Awadhi, F., Merrilee, H., and Jennison, C. Improving the acceptance rate of reversible jump MCMC proposals. *Statistics & Probability Letters*, 69(2):189–198, 2004. doi:10.1016/j.spl.2004.06.025.

Andrieu, C. and Roberts, G. O. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009. doi:10.1214/07-AOS574.

Berns, F., Hüwel, J., and Beecks, C. Automated model inference for Gaussian processes: An overview of state-of-the-art methods and algorithms. *SN Computer Science*, 3(4):300, 2022. doi:10.1007/s42979-022-01186-x.

Brooks, C. *Introductory Econometrics for Finance*. Cambridge University Press, Cambridge, 3rd edition, 2008. doi:10.1007/978-3-540-71918-2.

Brooks, S. P., Giudici, P., and Roberts, G. O. Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1):3–39, 2003. doi:10.1111/1467-9868.03711.

Cerqueira, V., Torgo, L., and Soares, C. A case study comparing machine learning with statistical methods for time series forecasting: Size matters. *Journal of Intelligence Information Systems*, 59(4):415–433, 2022. doi:10.1007/s42979-022-01186-x.

Chipman, H. A., George, E. I., and McCulloch, R. E. Bayesian CART model search. *Journal of the American Statistical Association*, 93(443):935–948, 1998. doi:10.1080/01621459.1998.10473750.

Chopin, N. A sequential particle filter method for static models. *Biometrika*, 89(3):539–551, 2002. doi:10.1093/biomet/89.3.539.

Corani, G., Benavoli, A., and Zaffalon, M. Time series forecasting with Gaussian processes needs priors. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track*, volume 12978 of *Lecture Notes in Computer Science*, pp. 103–117. Springer, 2021. doi:10.1007/978-3-030-86514-6_7.

Cranefield, S. and Dhiman, A. Identifying norms from observation using MCMC sampling. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pp. 118–124. International Joint Conferences on Artificial Intelligence Organization, 2021. doi:10.24963/ijcai.2021/17.

Cusumano-Towner, M., Lew, A. K., and Mansinghka, V. K. Automating involutive MCMC using probabilistic and differentiable programming. *arXiv*, 2007.09871, 2020. doi:arXiv.2007.09871.

Cusumano-Towner, M. F., Saad, F. A., Lew, A. K., and Mansinghka, V. K. Gen: A general-purpose probabilistic programming system with programmable inference. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Design and Implementation*, pp. 221–236. ACM, 2019. doi:10.1145/3314221.3314642.

Del Moral, P., Doucet, A., and Ajay, J. Sequential Monte Carlo for Bayesian computation. In *Proceedings of the 8th Valencia International Meeting on Bayesian Statistics*, volume 8 of *Bayesian Statistics*, pp. 1–34. Oxford University Press, 2007.

Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J. B., and Ghahramani, Z. Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1166–1174. PMLR, 2013.

Fiorucci, J. A., Pellegrini, T. R., Louzada, F., Petropoulos, F., and Koehler, A. B. Models for optimising the theta method and their relationship to state space models. *International Journal of Forecasting*, 32(4):1151–1161, 2016. doi:10.1016/j.ijforecast.2016.02.005.

Garza, F., Canseco, M. M., Challú, C., and Olivares, K. G. StatsForecast: Lightning fast forecasting with statistical and econometric models. PyCon US, Salt Lake City, Utah, 2022. URL https://github.com/Nixtla/statsforecast.

Genz, A. Numerical computation of multivariate normal probabilities. *Journal of Computational and Graphical Statistics*, 1(2):141–149, 1992. doi:10.1080/10618600.1992.10477010.

Golyandina, N., Korobeynikov, A., and Zhigljavsky, A. *Singular Spectrum Analysis with R*. Use R! Springer-Verlag, Berlin, 2018. doi:10.1007/978-3-662-57380-8.

Green, P. J. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995. doi:10.1093/biomet/82.4.711.

Grigorievskiy, A. and Karhunen, J. Gaussian process kernels for popular state-space time series models. In *Proceedings of the 2016 International Joint Conference on Neural Networks*, pp. 3354–3363. IEEE Press, 2016. doi:10.1109/IJCNN.2016.7727628.

Hamelijnck, O., Wilkinson, W., Loppi, N., Solin, A., and Damoulas, T. Spatio-temporal variational gaussian processes. In *Proceedings of the 35th Conference on Neural Information Processing Systems*, volume 34 of *Advances in Neural Information Processing Systems*, pp. 23621–23633. Curran Associates, Inc., 2021.

Hamze, F. and de Freitas, N. Hot coupling: A particle approach to inference and normalization on pairwise undirected graphs. In *Proceedings of the 19th Conference on Neural Information Processing Systems*, volume 18 of *Advances in Neural Information Processing Systems*, pp. 491–498. MIT Press, 2005.

Hartikainen, J. and Särkkä, S. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *Proceedings of the 2010 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 379–384. IEEE Press, 2010. doi:10.1109/MLSP.2010.5589113.

Hyndman, R. and Yangzhuoran, Y. tsdl: Time series data library, 2018. URL https://pkg.yangzhuoranyang.com/tsdl.

Hyndman, R. J. and Athanasopoulos, G. *Forecasting: Principles and Practice*. OTexts, Melbourne, 3rd edition, 2021. URL https://otexts.com/fpp3/.

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

Hyndman, R. J., Koehler, A. B., Ord, J. K., , and Snyder, R. *Forecasting with Exponential Smoothing: The State Space Approach*. Springer-Verlag, Berlin, 2008. doi:10.1007/978-3-540-71918-2.

Janz, D., Paige, B., Tom, R., van de Meent, J.-W., and Wood, F. Probabilistic structure discovery in time series data. *arXiv*, 1611.06863, 2016. doi:10.48550/arXiv.1611.06863.

Jin, Y., Fu, W., Kang, J., Guo, J., and Guo, J. Bayesian symbolic regression. *arXiv*, 1910.08892, 2019. doi:10.48550/arXiv.1910.08892.

Karagiannis, G. and Andrieu, C. Annealed importance sampling reversible jump MCMC algorithms. *Journal of Computational and Graphical Statistics*, 22(3):623–648, 2013. doi:10.1080/10618600.2013.805651.

Kim, H. and Teh, Y. W. Scaling up the automatic statistician: Scalable structure discovery using Gaussian processes. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pp. 575–584. PMLR, 2018.

Lew, A., Cusumano-Towner, M., and Mansinghka, V. Recursive Monte Carlo and variational inference with auxiliary variables. In *Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pp. 1096–1106. PMLR, 2022.

Lim, B., Arık, S. O., Loeff, N., and Pfister, T. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021. doi:10.1016/j.ijforecast.2021.03.012.

Lindemann, B., Müller, T., Vietz, H., Jazdi, N., and Jazdi, M. A survey on long short-term memory networks for time series prediction. *Procedia CIRP*, 99:650–655, 2021. doi:10.1016/j.procir.2021.03.088.

Liu, J. S. and Chen, R. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998. doi:10.1080/01621459.1998.10473765.

Lloyd, J., Duvenaud, D., Grosse, R., Tenenbaum, J., and Ghahramani, Z. Automatic construction and natural-language description of nonparametric regression models. *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, 28(1), 2014. doi:10.1609/aaai.v28i1.8904.

Loossens, T., Tuerlinckx, F., and Verdonck, S. A comparison of continuous and discrete time modeling of affective processes in terms of predictive accuracy. *Scientific Reports*, 11(6218), 2021. doi:10.1038/s41598-021-85320-4.

Makridakis, S. and Hibon, M. The M3-Competition: Results, conclusions and implications. *International Journal of Forecasting*, 16(4):451–476, 2000. doi:10.1016/S0169-2070(00)00057-1.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLos ONE*, 13(3):e0194889, 2018. doi:10.1371/journal.pone.0194889.

Makridakis, S., Spiliotis, E., Assimakopoulos, V., Semenoglou, A.-A., Mulder, G., and Nikolopoulos, K. Statistical, machine learning and deep learning forecasting methods: Comparisons and ways forward. *Journal of the Operational Research Society*, 74(3):840–859, 2023. doi:10.1080/01605682.2022.2118629.

Mansinghka, V., Shafto, P., Jonas, E., Petschulat, C., Gasner, M., and Tenenbaum, J. B. CrossCat: A fully Bayesian nonparametric method for analyzing heterogeneous, high dimensional data. *Journal of Machine Learning Research*, 17(138):1–49, 2016.

Moretti, A. K., Zhang, L., Naesseth, C. A., Venner, H., Blei, D., and Pe'er, I. Variational combinatorial sequential Monte Carlo methods for Bayesian phylogenetic inference. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 971–981. PMLR, 2021.

Naesseth, C. A., Fredrik Lindsten, S., and B., T. Sequential Monte Carlo for graphical models. In *Proceedings of the 28th Conference on Neural Information Processing Systems*, volume 27 of *Advances in Neural Information Processing Systems*, pp. 1862–1870. MIT Press, 2014.

Neal, R. M. Density modeling and clustering using Dirichlet diffusion trees. In *Proceedings of the 7th Valencia International Meeting*, volume 7 of *Bayesian Statistics*, pp. 619–629. Oxford University Press, 2003.

Neal, R. M. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*, chapter 5. CRC Press, New York, NY, 2011.

Neklyudov, K., Welling, M., Egorov, E., and Vetrov, D. Involutive MCMC: A unifying framework. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7273–7282. PMLR, 2020.

Obermeyer, F., Glidden, J., and Jonas, E. Scaling nonparametric Bayesian inference via subsample-annealing. In *Proceedings of the 17th Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pp. 696–705. PMLR, 2014.

Quiñonero-Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(65):1939–1959, 2005.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006.

Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., and Aigrain, S. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371 (1984):20110550, 2013. doi:10.1098/rsta.2011.0550.

Rossi, S., Heinonen, M., Bonilla, E., Shen, Z., and Filippone, M. Sparse Gaussian processes revisited: Bayesian approaches to inducing-variable approximations. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1837–1845. PMLR, 2021.

Saad, F. A. and Mansinghka, V. K. Temporally-reweighted chinese restaurant process mixtures for clustering, imputing, and forecasting multivariate time series. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pp. 755–764. PMLR, 2018.

Saad, F. A. and Mansinghka, V. K. Hierarchical infinite relational model. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1067–1077. PMLR, 2021.

Saad, F. A., Cusumano-Towner, M. F., Schaechtle, U., Rinard, M. C., and Mansinghka, V. K. Bayesian synthesis of probabilistic programs for automatic data modeling. *Proceedings of the ACM on Programming Languages*, 3 (POPL):37.1–37.32, 2019. doi:10.1145/3290350.

Saad, F. A., Cusumano-Towner, M., and Mansinghka, V. K. Estimators of entropy and information via inference in probabilistic models. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 5604–5621. PMLR, 2022.

Saad, F. A. K. *Scalable Structure Learning, Inference, and Analysis with Probabilistic Programs*. PhD thesis, Massachusetts Institute of Technology, 2022.

Schaechtle, U., Saad, F., Radul, A., and Mansinghka, V. Time series structure discovery via probabilistic program synthesis. *arXiv*, 1611.07051, 2017. doi:10.48550/arXiv.1611.07051.

Taylor, S. J. and Letham, B. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018. doi:10.1080/00031305.2017.1380080.

Tebbutt, W., Solin, A., and Turner, R. E. Combining pseudo-point and state space approximations for sum-separable Gaussian processes. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1607–1617. PMLR, 2021.

Wang, L., Bouchard-Côté, A., and Doucet, A. Bayesian phylogenetic inference using a combinatorial sequential Monte Carlo method. *Journal of the Ameri-*

can Statistical Association, 110(512):1362–1374, 2015. doi:10.1080/01621459.2015.1054487.

Wang, L., Wang, S., and Bouchard-Côté, A. An annealed sequential Monte Carlo method for Bayesian phylogenetics. *Systematic Biology*, 69(1):155–183, 2019. doi:10.1093/sysbio/syz028.

Yeh, Y.-T., Yang, L., Watson, M., Goodman, N. D., and Hanrahan, P. Synthesizing open worlds with constraints using locally annealed reversible jump MCMC. *ACM Transactions on Graphics*, 31(4):56.1–56.32, 2012. doi:10.1145/2185520.2185552.

Yu, K., Cui, Z., Sui, X., Qiu, X., and Zhang, J. Biological network inference with GRASP: A Bayesian network structure learning method using adaptive sequential Monte Carlo. *Frontiers in Genetics*, 12, 2021. doi:10.3389/fgene.2021.764020.

A. Proofs

Proof of Equation (21). Let $(k, \theta, \eta, \mathbf{y})$ be the current state. The conditional distribution of $[f(t_1), \ldots, f(t_n)]$ is given by:

$$[f(t_1), \ldots, f(t_n)] \mid (k, \theta, \eta, \mathbf{y}) \sim \text{MultivariateNormal}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{35}$$

$$\boldsymbol{\mu} \coloneqq k_\theta(\mathbf{t}) \left[k_\theta(\mathbf{t}) + \eta I\right]^{-1} \mathbf{y} \tag{36}$$

$$\boldsymbol{\Sigma} \coloneqq k_\theta(\mathbf{t}) - k_\theta(\mathbf{t}) \left[k_\theta(\mathbf{t}) + \eta I\right]^{-1} k_\theta(\mathbf{t}). \tag{37}$$

Conditioning on $f(\mathbf{t}) = \boldsymbol{\mu}$ and applying conjugacy of the inverse-gamma prior (8) to the normal likelihood (10) gives

$$\eta \mid (k, \theta, f(\mathbf{t}) = \boldsymbol{\mu}, \mathbf{y}) \sim \text{InverseGamma}\left(1 + n/2, 1 + \sum_{i=1}^{n} (y_i - \mu_i)^2/2\right). \tag{38}$$

□

Proof of Proposition 1. We will first show that f_{R} is an involution. Let (k, θ, η) be the current state and $(b, \tilde{k}, \tilde{\theta}, \eta') \sim q_{\text{R}}(\cdot; k, \theta, \eta)$. We then have

$$f_{\text{R}}([(k, \theta, \eta), (b, \tilde{k}, \tilde{\theta}, \eta')]) = [(k', \theta', \eta'), (b, \hat{k}, \hat{\theta}, \eta)], \qquad \text{where } (k', \theta') \coloneqq \otimes_b \left(\ominus_b(k, \theta), (\tilde{k}, \tilde{\theta})\right) \tag{39}$$

The self-inversion property holds for the variables (b, η, η'), because they each undergo a swap-position operation under f_{R}. It remains to show that the kernel structure k and parameters θ are self-inverted, that is,

$$\otimes_b \left(\ominus_b \left(\otimes_b \left(\ominus_b(k, \theta), (\tilde{k}, \tilde{\theta})\right)\right), (\hat{k}, \hat{\theta})\right) = (k, \theta), \tag{40}$$

which follows because $(\hat{k}, \hat{\theta}) \coloneqq (k_b, \theta_b)$ is the subtree of (k, θ) rooted at b detached by the innermost $\ominus_b$ operation.

To derive the acceptance ratio (22), the assumption that q_{R} samples the path b uniformly and $(\tilde{k}, \tilde{\theta})$ from the conditional prior given $(\ominus_b(k, \theta))$ means that

$$r_{\text{R}} = \frac{P_{\mathbf{t}}(k', \theta', \eta', \mathbf{y})}{P_{\mathbf{t}}(k, \theta, \eta, \mathbf{y})} \frac{q_{\text{R}}(b, \hat{k}, \hat{\theta}, \eta; k', \theta', \eta', \mathbf{y})}{q_{\text{R}}(b, \tilde{k}, \tilde{\theta}, \eta'; k, \theta, \eta, \mathbf{y})} \cancelto{1}{|\text{Det } J_{f_{\text{R}}}(k, \theta, \eta, b, \tilde{k}, \tilde{\theta}, \eta')|} \tag{41}$$

$$= \frac{P(\ominus_b(k', \theta')) P(\tilde{k}, \tilde{\theta} \mid \ominus_b(k', \theta'))}{P(\ominus_b(k, \theta)) P(\hat{k}, \hat{\theta} \mid \ominus_b(k, \theta))} \frac{P(\eta')}{P(\eta)} \frac{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')}{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')} \frac{1/|k'|}{1/|k|} \frac{P(\hat{k}, \hat{\theta} \mid \ominus_b(k', \theta'))}{P(\tilde{k}, \tilde{\theta} \mid \ominus_b(k, \theta))} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})} \tag{42}$$

$$= \frac{P(\eta')}{P(\eta)} \frac{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')}{P_{\mathbf{t}}(\mathbf{y} \mid k, \theta, \eta)} \frac{1/|k'|}{1/|k|} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})}, \tag{43}$$

where Eq. (42) uses the equality $\ominus_b(k, \theta) \equiv \ominus_b(k', \theta')$. □

Proof of Proposition 2. We first describe the implicit involution f_{DA} and proposal q_{DA} for ATTACH-DETACH. The proposal q_{DA} first samples an auxiliary "direction" variable $d \in \{0, 1\}$ (Neklyudov et al., 2020, Trick 3) with probability $(\xi, 1 - \xi)$, where $d = 0$ corresponds to the DETACH direction and $d = 1$ to ATTACH direction. Letting $\mathbf{z}$ be the current state and $\mathbf{v}$ be the variables sampled by the proposals (q_{D} or q_{A}), let $q_{\text{DA}}(d = 0, \mathbf{v}) \coloneqq \xi q_{\text{D}}(\mathbf{v})$, $q_{\text{DA}}(d = 1, \mathbf{v}) \coloneqq (1 - \xi) q_{\text{A}}(\mathbf{v})$. Then

$$f_{\text{DA}}(d = 0, \mathbf{z}, \mathbf{v}) \coloneqq (1, f_{\text{D}}(\mathbf{z}, \mathbf{v})), \tag{44}$$

$$f_{\text{DA}}(d = 1, \mathbf{z}, \mathbf{v}) \coloneqq (0, f_{\text{A}}(\mathbf{z}, \mathbf{v})). \tag{45}$$

To establish that f_{DA} is an involution, it suffices to show that $f_{\text{D}} = f_{\text{A}}^{-1}$. As in the previous proof, the variables (a, b, η, η') each undergo a swap-position operation under application of $f_{\text{A}}, f_{\text{D}}$ that invert one another. It remains to be shown that

$$\otimes_a(\ominus_a(\otimes_a(\ominus_a(k, \theta), (\tilde{k}, \tilde{\theta}))), (k_a, \theta_a)), \tag{46}$$

which follows from the properties (i) $\ominus_a(\otimes_a(\ominus_a T, T')) = \ominus_a T$; and (ii) $\otimes_a(\ominus_a T, T_a) = T$; which hold for any tree T.

We now prove the expression for the acceptance ratio r_{D} corresponding to the DETACH direction ($d = 0$). Under the hypotheses in Proposition 2 and the fact that f_{DA} is volume preserving, the acceptance ratio r_{A} is given by

$$r_{\mathrm{A}} = \frac{P_{\mathbf{t}}(k', \theta', \eta', \mathbf{y})}{P_{\mathbf{t}}(k, \theta, \eta, \mathbf{y})} \frac{q_{\mathrm{DA}}(d = 1, a, b, \hat{k}, \hat{\theta}; k', \theta', \eta', \mathbf{y})}{q_{\mathrm{DA}}(d = 0, a, b; k, \theta; k, \theta, \eta, \mathbf{y})} \tag{47}$$

$$= \frac{P_{\mathbf{t}}(k', \theta', \eta', \mathbf{y})}{P_{\mathbf{t}}(k, \theta, \eta, \mathbf{y})} \frac{(1 - \xi)}{\xi} \frac{q_{\mathrm{A}}(a, b, \hat{k}, \hat{\theta}; k', \theta', \eta', \mathbf{y})}{q_{\mathrm{A}}(a, b; k, \theta, \eta, \mathbf{y})} \tag{48}$$

$$= \frac{P(\ominus_a(k', \theta'))P(\tilde{k}, \tilde{\theta} \mid \ominus_a(k', \theta'))}{P(\ominus_a(k, \theta))P(k_a, \theta_a \mid \ominus_a(k, \theta))} \frac{(1 - \xi)}{\xi} \frac{P(\eta')}{P(\eta')} \frac{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')}{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')} \frac{1/|k'|}{1/|k|} \frac{q_{\mathrm{A}}(b \mid a; k', \theta')}{q_{\mathrm{D}}(b \mid a; k', \theta')} \tag{49}$$

$$\frac{P(\hat{k}, \hat{\theta} \mid \ominus_a(k', \theta'), b, (\tilde{k}, \tilde{\theta}))}{1} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})}$$

$$= \frac{P(\eta')}{P(\eta)} \frac{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')}{P_{\mathbf{t}}(\mathbf{y} \mid k, \theta, \eta)} \frac{1/|k'|}{1/|k|} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})} \frac{(1 - \xi)}{\xi} \frac{q_{\mathrm{A}}(b \mid a; k', \theta')}{q_{\mathrm{D}}(b \mid a; k, \theta)}, \tag{50}$$

where Eq. (50) uses the following properties:

$$\ominus_a(k, \theta) = \ominus_a(k', \theta'), \qquad \text{(by construction)} \tag{51}$$

$$P(\tilde{k}, \tilde{\theta} \mid \ominus_a(k', \theta')) = P(\tilde{k}, \tilde{\theta}), \qquad \text{(context-free prior)} \tag{52}$$

$$P(k_a, \theta_a \mid \ominus_a(k, \theta)) = P(k_a, \theta_a), \qquad \text{(context-free prior)} \tag{53}$$

$$P(k_a, \theta_a) = P(\tilde{k}, \tilde{\theta})P(k_a, \theta_a \mid (k_{ab}, \theta_{ab}) = (\tilde{k}, \tilde{\theta})) \qquad \text{(chain rule)} \tag{54}$$

$$= P(\tilde{k}, \tilde{\theta})P(\hat{k}, \hat{\theta} \mid b, (\tilde{k}, \tilde{\theta})) \qquad \text{(notation).} \tag{55}$$

An identical argument establishes the expression for the acceptance ratio r_{D} of the ATTACH direction ($d = 1$). □

Proof of Proposition 3. Recalling the definition of the involution f^U_{R} (33)–(34) for the SUBTREE-REPLACE proposal on the extended state-space, the acceptance ratio is

$$\tilde{r}^U_{\mathrm{R}} = \frac{P_{\mathbf{t}}(k', \theta', \eta', \mathbf{y})}{P_{\mathbf{t}}(k, \theta, \eta, \mathbf{y})} \frac{q_{\mathrm{R}}(b, \hat{k}, (\theta^{1:\hat{u}-1}_{S,D}, \theta_{S,D}, \theta^{\hat{u}+1:U}_{S,D}), \hat{u}, (\theta'^{1:\tilde{u}-1}_{S,F}, \theta^{\tilde{u}+1:U}_{S,F}), \tilde{u}, \eta)}{q_{\mathrm{R}}(b, \tilde{k}, \theta'^{1:U}_{S,F}, \tilde{u}, \theta^{1:U-1}_{S,D}, \hat{u}, \eta')} \tag{56}$$

$$= \frac{P_{\mathbf{t}}(k', \theta', \eta', \mathbf{y})}{P_{\mathbf{t}}(k, \theta, \eta, \mathbf{y})} \frac{1/|k|}{1/|k'|} \frac{P(\hat{k} \mid \ominus_b k')}{P(\tilde{k} \mid \ominus_b k)} \frac{q(\theta_{S,D}) \prod_{u=1}^{U-1} q(\theta^u_{S,D})}{\prod_{u=1}^{U} q(\theta'^u_{S,F})} \frac{\frac{\omega(\theta_{S,D})}{\sum_{u=1}^{U} \omega(\theta^u_{S,D})}}{\frac{\omega(\theta'^{\tilde{u}}_{S,F})}{\sum_{u=1}^{U} \omega(\theta'^u_{S,F})}} \frac{\prod_{\substack{u=1 \\ u \neq \tilde{u}}}^{U} q(\theta'^u_{S,F})}{\prod_{u=1}^{U-1} q(\theta^u_{S,D})} \frac{1/U}{1/U} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})} \tag{57}$$

$$= \frac{P(\theta'^{\tilde{u}}_{S,F})}{P(\theta_{S,D})} \frac{P(\eta')}{P(\eta)} \frac{P_{\mathbf{t}}(\mathbf{y} \mid k', \theta', \eta')}{P_{\mathbf{t}}(\mathbf{y} \mid k, \theta, \eta)} \frac{1/|k|}{1/|k'|} \frac{q(\theta_{S,D})}{q(\theta'^{\tilde{u}}_{S,F})} \frac{\omega(\theta_{S,D})}{\omega(\theta'^{\tilde{u}}_{S,F})} \frac{\sum_{u=1}^{U} \omega(\theta'^u_{S,F})}{\sum_{u=1}^{U} \omega(\theta^u_{S,D})} \frac{P_{\mathbf{t}}(\eta \mid \boldsymbol{\mu}, \mathbf{y})}{P_{\mathbf{t}}(\eta' \mid \boldsymbol{\mu}', \mathbf{y})}. \tag{58}$$

It can be readily verified that if $U = 1$ and $\delta = 0$ in the log-normal proposals (26) and (30), then Eq. (58) recovers Eq. (43), as these conditions give $\theta_S \equiv \theta'^{\tilde{u}}_S$ which in turn implies that $P(\theta'^{\tilde{u}}_{S,F})q(\theta_{S,D}) = P(\theta_{S,D})q(\theta'^{\tilde{u}}_{S,F})$. To show that f^U_{R} is an involution, we apply it twice to the current state to obtain

$$f^U_{\mathrm{R}}\left(f^U_{\mathrm{R}}([(k, \theta, \eta), (b, \tilde{k}, \theta'^{1:U}_{S,F}, \tilde{u}, \theta^{1:U-1}_{S,D}, \hat{u}, \eta')])\right) \tag{59}$$

$$= f^U_{\mathrm{R}}\left(\left[\left(\otimes_b\left(\ominus_b(k, (\theta'^{\tilde{u}}_S, \theta_D)), (\tilde{k}, \theta'^{\tilde{u}}_F)\right), \eta'\right), (b, \hat{k}, (\theta^{1:\hat{u}-1}_{S,D}, \theta_{S,D}, \theta^{\hat{u}:U-1}_{S,D}), \hat{u}, (\theta'^{1:\tilde{u}-1}_{S,F}, \theta'^{\tilde{u}+1:U}_{S,F}), \tilde{u}, \eta)\right]\right) \tag{60}$$

$$= \left[\underbrace{(k, (\theta_S, \theta_D))}_{\text{Eq. (40)}}, \eta), (b, \tilde{k}, (\theta^{1:\tilde{u}-1}_{S,F}, \theta'^{\tilde{u}}_{S,F}, \theta'^{\tilde{u}:U-1}_{S,F}), \tilde{u}, (\theta^{1:\hat{u}-1}_{S,D}, \theta^{\hat{u}:U-1}_{S,D}), \hat{u}, \eta')\right]. \tag{61}$$

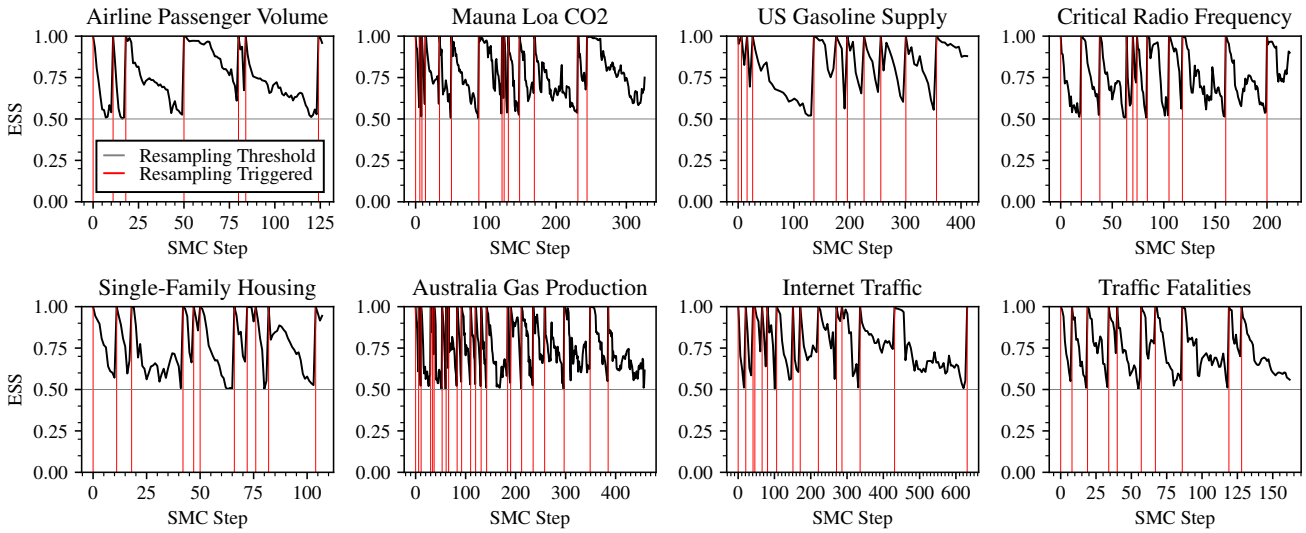

Figure 8. SMC step (x-axes) vs. normalized effective sample size (y-axes, ESS) for the eight datasets used in Fig. 5.

The transition kernel with acceptance probability $\alpha_{\mathrm{R}}^{U} = \min(1, \tilde{r}_{\mathrm{R}}^{U})$ thus leaves $P_{\mathbf{t}}(k, \varphi \mid \mathbf{y})$ invariant. Irreducibility and aperiodicity follow from (i) the hypothesis that q_{R}^{U} selects a zero-length path b (i.e., the root node) with positive probability; and (ii) the full support of the proposal $q(\theta'; \theta)$ over $\Theta_{d(k')}$ and of the proposal $P(\eta' \mid \boldsymbol{\mu}', \mathbf{y})$ over $\mathbb{R}_{>0}$. These conditions guarantee that any positive probability set in the transdimensional space (12) can be reached from (k, θ, η) in one step. □

B. Profiles of Effective Sample Size vs. SMC Step

Figure 8 shows the normalized effective sample size (ESS) versus SMC step j, defined from the weights according to $\mathrm{ESS}_j := (\sum_{i=1}^{M}(w_j^i))^2/(M\sum_{i=1}^{M}(w_j^i)^2)$, for eight datasets analyzed in Fig. 5 (Section 4.1 of the main text). Horizontal lines at ESS = 0.5 show the resampling threshold and vertical red lines show steps at which resampling occurred. There are typically several rejuvenation steps between resampling steps, suggesting that particle degeneracy is not a significant issue.

C. M3 Evaluation Details

Table 2 enumerates the baselines used in the evaluation from Section 4.2.[3] For each method, the default inference settings were used where applicable. Baselines from the statsforecast and neuralforecast packages (Garza et al., 2022) require the seasonal period m as input, which was set as $m = 12$ for monthly data. The LSTM and Temporal Fusion Transformer were trained with the multi-quantile loss at level 95 to produce the required 95% forecast intervals. For Singular Spectrum Analysis (SSA), multi-horizon forecasts were obtained using the recurrent SSA forecasting algorithm with bootstrap estimation for the forecast intervals. For *AutoGP*, Algorithm 1 was run using a linear annealing schedule with 5% of the data introduced at each step; $M = 48$ particles; adaptive resampling with $\mathrm{ESS} = M/2 = 24$; and $N_{\mathrm{rejuv}} = 100$ MCMC rejuvenation steps. The training time points $\mathbf{t}$ and (demeaned) values $\mathbf{y}$ are linearly transformed to $[0, 1]$. We use a kernel language $\mathcal{L}$ with three base kernels B (Eq. (3)) and a changepoint operator (CP, Eq. (4)) with the following parameterization:

$$\textsc{Linear}_{\theta_1,\theta_2,\theta_3} := \lambda t, t'.\theta_1 + \theta_2(t-\theta_3)(t'-\theta_3) \tag{62}$$

$$\textsc{GammaExponential}_{\theta_1,\theta_2,\theta_3} := \lambda t, t'.\theta_1 \exp\left(-(|t-t'|/\theta_2)^{\theta_3}\right) \qquad \theta_3 \in (0, 2] \tag{63}$$

$$\textsc{Periodic}_{\theta_1,\theta_2,\theta_3} := \lambda t, t'.\theta_1 \exp\left((-2/\theta_2^2)(\sin(\pi|t-t'|/\theta_3))^2\right) \tag{64}$$

$$k_1 \ \mathrm{CP}_{\theta_1,\theta_2}\ k_2 := \lambda t, t'.[\sigma_1 \cdot k_1(t,t') \cdot \sigma_2] + [(1-\sigma_1) \cdot k_2(t,t') \cdot (1-\sigma_2)] \tag{65}$$

$$\text{where } \sigma_1 := (1 + \tanh((t-\theta_1)/\theta_2))/2$$

$$\sigma_2 := (1 + \tanh((t'-\theta_1)/\theta_2))/2$$

[3] We also considered the hand-designed Gaussian process model from Corani et al. (2021), which uses a fixed structure with five kernels (PERIODIC + LINEAR + RBF + SM_1 + SM_2) tailored to the M3 data, where SM is a spectral mixture kernel. However, we encountered two challenges: (i) the model hyperpriors are pre-tuned on 350/1428 monthly time series, so the forecasts apply only to 1078/1428 datasets; and (ii) the prediction intervals spanned many orders of magnitude, leading to MSIS errors of $\approx 1.8 \times 10^4$.

Table 2. Baseline methods used in the evaluation from Section 4.2 on the M3 monthly time series.

Method	Reference	Implementation
AutoARIMA	Hyndman et al. (2008)	https://nixtla.github.io/statsforecast/
AutoETS	Hyndman et al. (2008)	https://nixtla.github.io/statsforecast/
AutoTheta	Fiorucci et al. (2016)	https://nixtla.github.io/statsforecast/
Custom M3 Gaussian Process	Corani et al. (2021)	https://github.com/IDSIA/gpforecasting/
Facebook Prophet	Taylor & Letham (2018)	https://facebook.github.io/prophet/
LSTM	Lindemann et al. (2021)	https://nixtla.github.io/neuralforecast/
Random Walk with Drift	Hyndman & Athanasopoulos (2021)	https://nixtla.github.io/statsforecast/
Seasonal Naive	Hyndman & Athanasopoulos (2021)	https://nixtla.github.io/statsforecast/
Singular Spectrum Analysis	Golyandina et al. (2018)	https://cran.r-project.org/web/packages/Rssa/
Temporal Fusion Transformer	Lim et al. (2021)	https://nixtla.github.io/neuralforecast/

C.1. Performance Metrics for Evaluating Forecasts

The definitions of Symmetric Mean Absolute Percentage Error (SMAPE), Mean Absolute Scaled Error (MASE), and Mean Scaled Interval Score (MSIS) appearing in Figs. 5 and 6 are taken from Makridakis et al. (2018):

$$\text{SMAPE}_h(x, \hat{x}) \coloneqq 100 \times \frac{2}{h} \sum_{t=n+1}^{n+h} \frac{|x_t - \hat{x}_t|}{|x_t| + |\hat{x}_t|} \tag{66}$$

$$\text{MASE}_{h,m}(x, \hat{x}) \coloneqq \frac{1}{h} \frac{\sum_{t=n+1}^{n+h} |x_t - \hat{x}_t|}{\frac{1}{n-m} \sum_{t=m+1}^{n} |x_t - x_{t-m}|} \tag{67}$$

$$\text{MSIS}_{h,m}(x, \hat{u}, \hat{\ell}) \coloneqq \frac{1}{h} \frac{\sum_{t=n+1}^{n+h} (\hat{u}_t - \hat{\ell}_t) + \frac{2}{a}(\hat{\ell}_t - x_t)\mathbf{1}[x_t < \hat{\ell}_t] + \frac{2}{a}(x_t < \hat{u}_t)\mathbf{1}[\hat{u}_t < x_t]}{\frac{1}{n-m} \sum_{t=m+1}^{n} |x_t - x_{t-m}|}, \tag{68}$$

where n is the number of observed data points; $h > 0$ is the forecasting horizon; $m > 0$ is the seasonal period ($m = 12$ for monthly data); $x = (x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+h})$ denotes the n observed data points and h test data points; $\hat{x} = (\hat{x}_{n+1}, \ldots, \hat{x}_{n+h})$ denotes the h point forecasts; and $\hat{u} = (\hat{u}_{n+1}, \ldots, \hat{u}_{n+h})$ and $\hat{\ell} = (\hat{\ell}_{n+1}, \ldots, \hat{\ell}_{n+h})$ denote the h upper and lower bounds of the $(1-a)\%$ prediction interval ($a = 0.05$ for 95% intervals).

D. Modeling Discrete-Time Data with Gaussian Processes

Recall from Eqs. (9) and (10) that the Gaussian process time series model specifies a distribution over data $y_i \sim \text{Normal}(f(t_i), \eta)$ where $t_i \in \mathbb{R}$ is a real time index. For discrete-time datasets such as M3, the observed time index t_i is instead a time stamp of the form YYYY-MM-DD. We convert time stamps to numbers by computing their age in seconds since the UNIX epoch 1970-01-01 00:00:00. This strategy is preferable to converting time stamps to integer indexes in applications where even slight differences in the duration between two time stamps influences the magnitude of the observations (Loossens et al., 2021). In a monthly time series, for example, the month 2020-01-01—2020-02-01 contains 31 days while the month 2020-02-01—2020-03-01 contains only 29 days. The UNIX epoch encoding of these time stamps (1577836800, 1580515200, 1583020800) captures the unequal spacing whereas the integer encoding (1, 2, 3) does not. Another benefit of epoch encoding with continuous-time models is that it provides a coherent way of learning seasonal periods with non-integer periodicity (such as weekly data with an annual pattern which has a seasonal period of 365.25/7 ≈ 52.179 steps on average). Traditional ARIMA or ETS models do not handle non-integer periodicity (Hyndman & Athanasopoulos, 2021, Section 12.1) and require users to manually include Fourier features with the correct periodicity.

Caution must be taken when computing predictions with a Gaussian process trained on epoch encodings, however, because the time series may not have meaningful semantics at arbitrary time points. Consider, for example, a monthly time series that records the total number of home sales as measured on the final day (2020-01-31, 2020-02-28, ...). The output $f(t)$ of a Gaussian process f at a real number t that encodes an arbitrary time stamp (e.g., 2020-01-18, 8pm) may not correspond to a semantically well-defined measurement of the underlying time series. Time series that evolve in continuous time can generally be modeled without these subtleties.