mathematics

MDPI

Article

Sparse Simulation of Autoregressive Gaussian Processes

Tadej Krivec [1] **and Juš Kocijan** [1,2,*]

[1] Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
[2] Centre for Information Technologies and Applied Mathematics, University of Nova Gorica, 5000 Nova Gorica, Slovenia
* Correspondence: jus.kocijan@ijs.si

Abstract

This study proposes a novel and improved numerical approximation of the simulation of Gaussian process autoregressive models. As a Bayesian nonparametric regression method, Gaussian process models offer the unique advantage of providing closed-form uncertainty quantification. When Gaussian process models are used for autoregressive models, the validation procedure requires the model's simulation or multi-step-ahead prediction. However, simulating dynamical Gaussian process models is complex due to the intractable propagation of uncertain inputs through the nonlinear model. Numerical approximation, namely Monte Carlo simulation, is one of the most frequent options for simulating dynamical models based on Gaussian processes. The computational burden of Monte Carlo simulation algorithms increases cubically with data size, representing a challenge. This paper introduces a unified simulation framework invariant to sparse and variational approximations to obtain a static sample from the pseudo-point posterior. Furthermore, we propose an innovative method for simulating Gaussian process dynamical models. A single parameter is proposed to regulate the trade-off between computational complexity and algorithmic accuracy. This innovation demonstrates the potential to replace the conditionally independent Monte Carlo method with no additional computational burden, thereby enhancing estimates of latent responses. The proposed simulation method is demonstrated using two synthetic examples and a realistic case study.

Keywords: Gaussian process models; autoregressive models; Monte Carlo simulation; dynamical systems; uncertainty quantification

MSC: 37M05; 37M10; 60G15; 65C05; 65C20; 68U99

Academic Editors: Ning Cai and Yong Xie

Received: 4 May 2026
Revised: 1 June 2026
Accepted: 11 June 2026
Published: 13 June 2026

1. Introduction

The following investigation proposes improved numerical approximations of the simulation of Gaussian-process (GP) autoregressive models.

GP models deliver a well-calibrated predictive distribution at a model output, providing a systematic quantification of model fidelity [1]. This characteristic makes them particularly suitable for applications where a nuanced understanding of model uncertainty is crucial, such as in safety-critical scenarios [2], robust control systems [3], and fault detection. GPs, characterised by their utilisation of an infinite number of basis functions [4–7], belong to a class of probabilistic models. Leveraging a nonparametric structure, GPs effectively reduce the number of metaparameters and seamlessly scale model complexity with the size of the dataset. The model's ability to penalise overly complex structures is inherently embedded in the evidence [8].

While closed-form calculations are available for both the marginal and conditional distributions, the associated algorithm exhibits cubic computational complexity and quadratic memory complexity [9]. One systematic approach for reducing computational complexity involves employing sparse approximation methods, where a reduced set of pseudo-points, denoted by $m < n$, summarises the data [10–12].

Among the various sparse-based approximations of the prior distribution, the Fully Independent Training Conditional (FITC) method is one of the most popular. FITC involves a joint optimisation of pseudo-points and hyperparameters through gradient-based optimisation strategies [13].

A more recent extension of sparse approximations involves a direct approximation of the posterior using variational inference without imposing additional assumptions about the prior. Pseudo-inputs and hyperparameters are obtained by maximising the lower bound of the actual Marginal Log-Likelihood (MLL), a process equivalent to minimising the Kullback–Leibler (KL) divergence between the true and the approximated posterior. This approach rigorously defines the distance between the true and the approximated posterior. The parameters of the free variational distribution can be treated as model parameters, giving rise to a Scalable Variational Gaussian Process (SVGP) [14–16].

SVGP facilitates stochastic optimisation and provides an unbiased estimation of the lower bound of the true MLL from random subsets of training data. An alternative is deriving a collapsed bound, wherein the optimal parameters of the free variational distribution are analytically determined, resulting in a Variational Free Energy (VFE) approximation [17]. However, this reintroduces the full correlation between data points, making the bound unsuitable for stochastic optimisation.

While initially conceived, approximations were for static systems; many of these concepts seamlessly extend to **dynamical models**. A natural adaptation involves a direct functional relationship between time and output observations within a dynamical model, with the mapping captured by a GP. However, this type of static regression cannot inherently learn the system's dynamics. A viable approach to addressing this involves representing the system's dynamics through a Finite Impulse Response (FIR) model. In this model, the current output observation depends upon a finite set of lagged exogenous variables [18]. An inherent challenge with FIR models lies in the necessity to select a relatively large order, corresponding to the number of lagged exogenous inputs [19]. Alternatively, one may consider an Infinite Impulse Response (IIR) model, where a functional relationship between the current output observation and all past exogenous variables encapsulates the system's dynamics.

The introduction of lagged-output observations as additional inputs forms a nonlinear autoregressive model with exogenous inputs (NARX).

The orders of NARX models required to achieve comparable accuracy are notably smaller than those of FIR models. This makes NARX models using GPs, namely GP-NARX combinations, more suitable for real-world applications.

The elegance of the NARX model lies in its simplification of model training to that of the static case. This characteristic renders the NARX model particularly advantageous for scalability to large datasets, simplifying experimentation and offering practical advantages over more specialised models. Notably, the NARX approach can surpass the performance of alternative models, such as state-space models (GP-SSM) [20–23] or nonlinear output-error models (GP-NOE) [24]. The versatility and robustness of the NARX model make it a compelling choice for a wide range of applications, and this study focuses on these.

A particular case of the multiple-step-ahead prediction is the **simulation**, where the *prediction horizon is infinite* [6], or in practice, sufficiently large. There are two primary motivations for conducting simulations:

1. Forecasting future system states: Simulations enable the anticipation of the system's state over an extended prediction horizon, allowing insights into its behaviour over multiple future steps;
2. Model validation for dynamical systems: Simulations are crucial for validating the dynamical system model. This validation is particularly noteworthy in NARX models, as it assesses model performance in an operational regime for which the model was not explicitly trained.

While the first point underscores the intuitive need for predicting future states [25], the second point highlights the unique aspect of NARX models, where model performance is rigorously evaluated in real-world operating conditions that were not part of the model training data. A successful simulation accurately describes observations and validates the assumptions embedded in the model under practical circumstances.

GP-NARX models are favoured for straightforward training procedures, yet the simulation process necessitates an iterative approach. During simulation, predictions from previous timesteps are utilised as inputs for subsequent timesteps, extending indefinitely or until the conclusion of the prediction horizon. However, this introduces uncertainty into the inputs and the need for approximations [26–30]. A *significant challenge* in dynamical GP-NARX models arises from the propagation of uncertainty—as distributions of random variables at model input can be interpreted—through nonlinear models, which is intractable [31–33]. The propagation of uncertainty extends beyond training, presenting complexities in multi-step-ahead predictions. In GP-NARX models, the Gaussian distribution undergoes sequential propagation through the nonlinear model, resulting in an analytically intractable problem [6].

The approximated simulation procedures in GP-NARX models can be broadly categorised into three types:

- Naïve simulation: This method propagates the mean of the prediction, resulting in an underestimation of the forecasted uncertainty [6];
- Approximations of statistical moments: This category further subdivides into two approaches—those employing a Taylor expansion [29] and those employing exact matching of statistical moments [27,34]. Unfortunately, the former is considered a rough approximation. At the same time, the latter only applies to a limited set of covariance functions, such as the squared exponential function and a linear function [6]. An inherent drawback of methods based on the approximation of statistical moments is their neglect of higher statistical moments.
- Numerical approximations: This approach involves employing numerical techniques, i.e., the Monte Carlo (MC) method, to sample simulated trajectories [21,35–37];

Given the increasing computational capabilities, **numerical approximations** utilising MC sampling have gained prominence. MC-based simulation offers the advantage of not constraining the model to a specific choice of covariance function, enabling the identification of higher statistical moments.

Unfortunately, the existing work on improving the scalability of GP models does not translate well to MC sampling of trajectories in dynamical GP models. The MC approach exhibits cubic computational complexity with the number of predicted steps into the future [21]. When implementing an MC sampler for GP-NARX trajectories, the Gaussian distribution extends to all function evaluations, necessitating storing consecutive MC samples in memory. This substantially increases the computational complexity, especially for long prediction horizons in GP-NARX models [37].

A commonly adopted strategy to mitigate computational complexity during simulation involves assuming conditional independence between consecutive latent values,

given those associated with the training data. While this assumption introduces a rough simplification that is mathematically imprecise, it does offer a significant reduction of computational complexity [38]. Nevertheless, the assumption of conditional independence may not hold in all practical scenarios, potentially leading to inaccuracies in the simulation results. *Our challenge* in this investigation is getting less computational complexity and acceptable memory requirements at acceptable accuracy.

The main contribution presented is twofold.

1. This study, which is based on the thesis [39], proposes a unified and original framework for numerically simulating GP-NARX models, drawing inspiration from the static representation introduced in [40]. Our approach divides the simulation algorithm into a GP-specific static part and a dynamical part that remains consistent across various sparse approximations.
2. A novel approximation named Thresholded Correlated Monte Carlo (TCMC) simulation that advances the state-of-the-art in GP simulation of dynamical systems is introduced, contributing the following advancements:
   (a) An approximated MC-based simulation algorithm featuring a single parameter that balances the trade-off between the computational complexity and the accuracy of the simulation approximation, which can, in the limit, recover the ground truth.
   (b) Special case of the proposed simulation approximation, Pseudo Independent Monte Carlo (PIMC) simulation, enhances estimated simulations' accuracy while maintaining the same computational complexity as the existing Conditionally Independent MC simulation [38].
   (c) Development of compact and efficient simulation algorithms applicable to sparse and variational approximations. An open-source GP-NARX model training and simulation toolbox is also available, accessible at the following repository: https://github.com/tadejkrivec/GPflow-NARX (accessed on 10 June 2026).

This paper is organised as follows: In Section 2, we provide an in-depth introduction to GP-NARX models and their popular sparse and variational approximations. We establish a unified representation for predictions across these approximations, creating a crucial bridge toward developing a unified simulation algorithm. The section ends with the presentation of the unified simulation procedure. In Section 3, we list MC approximations to the unified GP-NARX model simulations. Section 4 introduces *the novel approximation method* for simulating GP-NARX models. This section explores a particular case of the approximation, demonstrating its capacity to significantly enhance independent MC simulations without incurring additional computational costs while ensuring a more accurate approximation to the ground truth. Section 5 systematically compares the proposed approximations through two synthetic illustrative examples and a realistic case study, namely a dynamical benchmark problem, highlighting the problematic training and test data size for conventional GP-NARX models. Section 6 briefly summarises the contributions and concludes with some final remarks.

2. Methodology

2.1. Vanilla GP-NARX Model

We assume the true dynamics satisfy

$$f_t = f\left(f_{t-n_a}, \ldots, f_{t-1}, x_{t-n_b}, \ldots, x_{t-1}\right), \tag{1}$$

where f is a nonlinear function, f_t is the function evaluation at timestep t, x are the exogenous variables, and $n_a, n_b > 0$ denote the number of lags for the delayed latent values and delayed exogenous variables, respectively. Equivalently, in matrix form,

$$\mathbf{f}_{1:t} = f(\mathbf{Z}_{1:t}), \tag{2}$$

where $\mathbf{f}_{1:t}$ is the vector of latent function values and $\mathbf{Z}_{1:t} \in \mathcal{R}^{t\times(n_a+n_b)}$ is the NARX input matrix (t observations). The i-th row of $\mathbf{Z}_{1:t}$ is

$$\mathbf{z}_i^T = [f_{i-n_a}, \ldots, f_{i-1}, x_{i-n_b}, \ldots, x_{i-1}]. \tag{3}$$

Because the latent variables are not attainable in practice, we instead observe a noisy output vector, $\mathbf{y}_{1:t} = \mathbf{f}_{1:t} + \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I}\sigma_n^2)$. While alternative observation models are possible, this additive assumption is the most common presumption [5,6].

The dynamical system is thus written as

$$\mathbf{y}_{1:t} = f(\mathbf{Z}_{1:t}) + \boldsymbol{\epsilon}, \tag{4}$$

where the i-th row of $\mathbf{Z}_{1:t}$ is now defined using the noisy outputs,

$$\mathbf{z}_i^T = [y_{i-n_a}, \ldots, y_{i-1}, x_{i-n_b}, \ldots, x_{i-1}]. \tag{5}$$

Our goal is to infer the latent function f in Equation (4). Here, we focus on the case where f is modelled with a GP [5].

A GP is a collection of random variables such that any finite subset has a joint Gaussian distribution. It is fully characterised by its mean $m(\cdot)$ and covariance function $k(\cdot,\cdot)$,

$$m(\mathbf{z}) = \mathbb{E}[f(\mathbf{z})], \tag{6a}$$
$$k(\mathbf{z}, \mathbf{z}') = \mathbb{E}[(f(\mathbf{z}) - m(\mathbf{z}))(f(\mathbf{z}') - m(\mathbf{z}')^T)]. \tag{6b}$$

We write the GP prior as

$$f(\cdot) \sim \mathcal{GP}(f(\cdot)|m(\cdot), k(\cdot,\cdot)), \tag{7}$$

where $m(\cdot)$ and $k(\cdot,\cdot)$ are parametrised by hyperparameters $\boldsymbol{\theta}$. Using Bayes' theorem, the prior over functions is transformed into the posterior,

$$\mathcal{GP}\left(f(\cdot)|\mathbf{y}_{1:t}, \boldsymbol{\theta}, \sigma_n^2\right) = \frac{p\left(\mathbf{y}_{1:t}|\mathbf{f}_{1:t}, \sigma_n^2\right)\mathcal{GP}(f(\cdot)|\boldsymbol{\theta})}{p(\mathbf{y}_{1:t}|\boldsymbol{\theta}, \sigma_n^2)}, \tag{8}$$

where the likelihood $p\left(\mathbf{y}_{1:t}|\mathbf{f}_{1:t}, \sigma_n^2\right)$ is assumed Gaussian. For notational simplicity, the conditioning on the choice of $m(\cdot)$ and $k(\cdot,\cdot)$ in Equation (8) is replaced by $\boldsymbol{\theta}$. We will typically omit conditioning on $\boldsymbol{\theta}$ and σ_n^2, since these are usually deterministic variables learned from data, and reintroduce them only when they are parameters of interest or random variables. Without loss of generality, we henceforth assume $m(\cdot) = \mathbf{0}$.

In practice, we work with finite sets of random variables and marginalise out latent quantities that are not of interest,

$$p(\mathbf{f}_{1:t+n}) = \int \mathcal{GP}(f(\cdot))d\mathbf{f}_{\setminus\{\mathbf{f}_{1:t+n}\}}, \tag{9a}$$
$$p(\mathbf{f}_{1:t+n}|\mathbf{y}_{1:t}) = \int \mathcal{GP}(f(\cdot)|\mathbf{y}_{1:t})d\mathbf{f}_{\setminus\{\mathbf{f}_{1:t+n}\}}, \tag{9b}$$

where $\mathbf{f}_{\setminus\{\mathbf{f}_{1:t+n}\}}$ is the set of all latent values outside $\{\mathbf{f}_{1:t+n}\}$. Note that $\{\mathbf{f}_{1:t+n}\} = \{\mathbf{f}_{1:t}, \mathbf{f}_{t+1:t+n}\}$, where $\mathbf{f}_{1:t}$ corresponds to the observations and $\mathbf{f}_{t+1:t+n}$ are the desired latent values up to

n steps into the future. The marginalisation property of Gaussian distributions allows all remaining latent variables to be ignored; see [5] for details.

Unfortunately, vanilla GP-NARX models scale cubically with the training data size. Accordingly, the dominant practical obstacles in training and simulating dynamical GPs are computational complexity and memory requirements. The sparse approximation methods reviewed next are introduced to reduce computational complexity.

2.2. Sparse GP-NARX Model

Sparse approximations aim to identify a representative set of m pseudo-inputs that summarise the data well [10]. This can be interpreted as a form of dimensionality reduction, but in the row space of the input matrix (data points) rather than its column space (features). We introduce m pseudo-inputs

$$\mathbf{z}'_{1:m} = [z'_1, \ldots, z'_m] \tag{10}$$

and the associated latent function values

$$\mathbf{u} = [u_1, \ldots, u_m], \tag{11}$$

which are drawn from the same joint distribution as $\mathbf{f}_{1:t+n}$. This construction mirrors the vanilla GP, where pseudo-points $\mathbf{u}$ are implicitly present but marginalised out.

Computational savings arise once we assume $\mathbf{f}_{1:t}$ and $\mathbf{f}_{t+1:t+n}$ are conditionally independent given $\mathbf{u}$, yielding the approximate joint prior

$$p(\mathbf{f}_{1:t+n}) \approx \int p(\mathbf{f}_{1:t}|\mathbf{u})p(\mathbf{f}_{t+1:t+n}|\mathbf{u})p(\mathbf{u})d\mathbf{u}. \tag{12}$$

The joint Gaussian distribution is fully determined by the conditional distributions given the pseudo-points,

$$p(\mathbf{f}_{1:t}|\mathbf{u}) = \mathcal{N}(\mathbf{f}_{1:t}|\mathbf{K}_{f_{1:t},u}\mathbf{K}_{u,u}^{-1}\mathbf{u}, \tag{13a}$$

$$\mathbf{K}_{f_{1:t},f_{1:t}} - \mathbf{Q}_{f_{1:t},f_{1:t}}) \tag{13b}$$

$$p(\mathbf{f}_{t+1:t+n}|\mathbf{u}) = \mathcal{N}(\mathbf{f}_{t+1:t+n}|\mathbf{K}_{f_{t+1:t+n},u}\mathbf{K}_{u,u}^{-1}\mathbf{u}, \tag{13c}$$

$$\mathbf{K}_{f_{t+1:t+n},f_{t+1:t+n}} - \mathbf{Q}_{f_{t+1:t+n},f_{t+1:t+n}}), \tag{13d}$$

where $\mathbf{Q}_{ab} = \mathbf{K}_{a,u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,b}$. As in the vanilla GP-NARX case, the posterior is obtained via

$$p(\mathbf{f}_{1:t+n}|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_{1:t}|\mathbf{f}_{1:t})p(\mathbf{f}_{1:t+n})}{p(\mathbf{y}_{1:t})}, \tag{14}$$

with $p(\mathbf{f}_{1:t+n})$ defined by Equation (12).

The conditional-independence assumption in Equation (12) alone is still insufficient for practical computational improvements, so additional structure is typically imposed. In this work, we consider two widely used approaches:

1. Prior approximations;
2. Posterior approximations.

A comparison of the prior and posterior approximations can be found in [11]. The Fully Independent Training Conditional (FITC) approximation is a *prior approximation*. It keeps the test conditional $p(\mathbf{f}_{t+1:t+n}|\mathbf{u})$ exact, but replaces the training conditional with

$$p(\mathbf{f}_{1:t}|\mathbf{u}) \approx q(\mathbf{f}_{1:t}|\mathbf{u}) = \prod_{i=1}^{t} p(f_i|\mathbf{u}). \tag{15}$$

Variational approximations, in contrast, approximate the posterior directly and therefore belong to *posterior approximations*. We introduce a free variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{\Phi})$ and obtain the lower bound

$$p\left(\mathbf{y}_{1:t}|\boldsymbol{\theta}, \sigma_n^2\right) \geq L(\mathbf{m}, \mathbf{\Phi}, \mathbf{z}'_{1:m}, \boldsymbol{\theta}, \sigma_n^2) \tag{16a}$$

$$= \mathbb{E}_{q(\mathbf{f}_{1:t})}[\log p(\mathbf{y}_{1:t}|\mathbf{f}_{1:t})] - \mathrm{KL}[q(\mathbf{u})||p(\mathbf{u})], \tag{16b}$$

where $\mathbf{m}$ and $\mathbf{\Phi}$ parameterise the free variational distribution. By lower-bounding the MLL, the free variational distribution is encouraged towards the exact posterior over the pseudo-points, i.e., $q(\mathbf{u}) \approx p(\mathbf{u}|\mathbf{y}_{1:t})$ [17,41].

The most scalable sparse method considered here is the Scalable Variational Gaussian Process (SVGP) [14,15], where the variational distribution remains free and is learned by stochastically optimising the Evidence Lower Bound (ELBO) [14]. We refer to this approximation as the GP-NARX (SVGP) model. The less scalable alternative is the Variational Free Energy (VFE) method: its key benefit is that the variational distribution is determined optimally by maximising the ELBO in closed form. However, this reintroduces correlations between data points and prevents stochastic optimisation. We refer to this approximation as the GP-NARX (VFE) model.

2.3. Unified Prediction in GP-NARX Models

This paper does not focus on learning the model parameters (i.e., $\boldsymbol{\theta}, \sigma_n^2$). We therefore assume they are given and fixed; a description of model learning is provided in Appendix D.

Given learned parameters, the objective is to predict the model output at the next timestep $t+1$. The presentation below follows [10,11,16], to which we refer for full derivations and discussion. Starting from Equation (9b), prediction is obtained by marginalising the training latent function values $\mathbf{f}_{1:t}$. A unified description covering the vanilla GP and the sparse approximations is [10,11,16]

$$\begin{aligned} q(\mathbf{f}_{t+1}) &= \int p(\mathbf{f}_{1:t+n}|\mathbf{y}_{1:t}) d\mathbf{f}_{\setminus\{\mathbf{f}_{t+1}\}} \\ &\approx \int\int p(\mathbf{f}_{t+1:t+n}|\mathbf{u}) p(\mathbf{f}_{1:t}|\mathbf{y}_{1:t}, \mathbf{u}) q(\mathbf{u}) d\mathbf{u} d\mathbf{f}_{\setminus\{\mathbf{f}_{t+1}\}} \\ &\approx \int p(\mathbf{f}_{t+1}|\mathbf{u}) q(\mathbf{u}) d\mathbf{u}, \end{aligned} \tag{17}$$

where the distribution $q(\mathbf{u})$ depends on the chosen approximation,

$$\begin{cases} \underbrace{q(\mathbf{u}) = p(\mathbf{f}_{1:t}|\mathbf{y}_{1:t})}_{\text{Vanilla GP-NARX}}, \text{ defined by Equation (A18)}, \\ \underbrace{q(\mathbf{u}) = p(\mathbf{u}|\mathbf{y}_{1:t})}_{\text{GP-NARX (FITC)}}, \text{ defined by Equation (A19)}, \\ \underbrace{q(\mathbf{u}) \approx p(\mathbf{u}|\mathbf{y}_{1:t})}_{\text{GP-NARX (VFE)}} \text{ defined by Equation (A21)}, \\ \underbrace{q(\mathbf{u}) \approx p(\mathbf{u}|\mathbf{y}_{1:t})}_{\text{GP-NARX (SVGP)}}, \text{ defined by Equation (A22)}, \end{cases} \tag{18}$$

where the letter in the equation labels indicates that the corresponding expressions are collected in the Appendix G. In particular, vanilla GP-NARX prediction is recovered

when $q(\mathbf{u})$ is taken as the posterior over the training latent function values. In Figure 1, the prediction mechanism corresponds to the black interconnected part. The final integral in Equation (17) is analytically tractable; each approximation leads to a different closed-form expression. The resulting predictions are summarised in Appendix G.

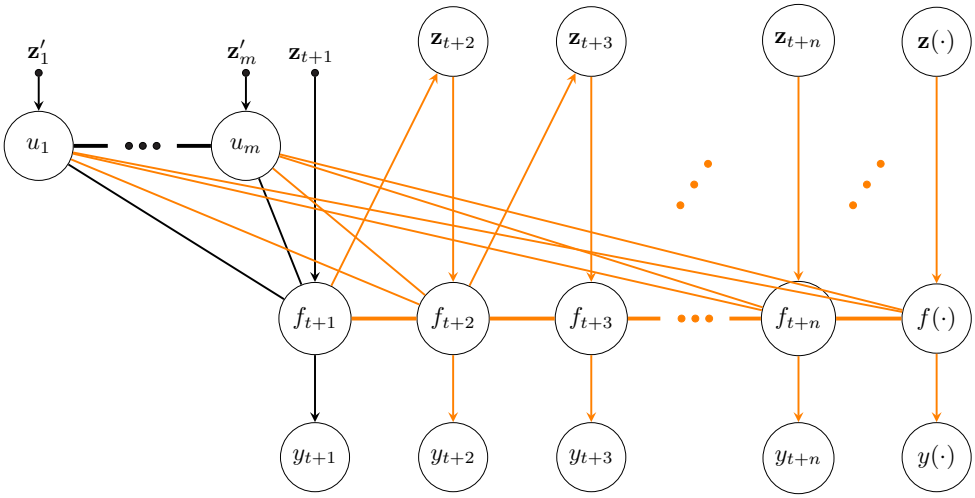


Figure 1. A probabilistic graphical model (PGM) representation of the posterior distribution of the latent function. The black interconnected components depict the prediction process, whereas the orange interconnected components correspond to the simulation of GP-NARX models. Because random inputs are propagated through a nonlinear function, the simulation does not admit a closed-form solution; in contrast, the prediction can be derived analytically.

2.4. Unified Simulation in GP-NARX Models

To enable a streamlined simulation algorithm, we retain the explicit form of $q(\mathbf{u})$ rather than analytically marginalising it. Although this choice offers limited benefit for prediction, it allows the simulation procedure to be separated into two components:

1. **Static part**: Approximation-specific, with readily available algorithms and implementations.
2. **Dynamical part**: Shared across the approximations considered here, enabling reuse of the same dynamical simulation code.

The decomposition into static and dynamical components separates the closed-form posterior prediction from the recursive simulation problem. The static component contains approximation-specific posterior predictive distributions, while the dynamical component describes the recursive evolution of the simulated trajectory. This separation enables a unified simulation framework across multiple sparse GP approximations.

If the final step in Equation (17) is marginalised, each approximation yields a different simulation algorithm, which obstructs a generalised implementation. Moreover, because latent function values up to timestamp $t + n$ depend on the propagation of random inputs through a nonlinear transformation, closed-form solutions are unavailable; consequently, the final integral in Equation (17) must be estimated numerically. The same numerical methodology then applies to later steps.

The unified simulation of latent function values is defined as

$$q(\mathbf{f}_{t+1:t+n}) = \int \prod_{i=2}^{n} p(\mathbf{f}_{t+i}|\mathbf{f}_{t+1:t+i-1}, \mathbf{u}) q(\mathbf{u}) d\mathbf{u}, \tag{19}$$

where $q(\mathbf{u})$ is the **static part** and $\prod_{i=2}^{n} p(\mathbf{f}_{t+i}|\mathbf{f}_{t+1:t+i-1}, \mathbf{u})$ is the **dynamical part**. For an arbitrary parameter value n_a, the dynamical part up to timestamp $t+n$ can be written as

$$\begin{aligned} p(\mathbf{f}_{t+i}|\mathbf{f}_{t+1:t+i-1}, \mathbf{u}) &= \int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u}) \\ &\cdot \Big(\prod_{i=2}^{n} \int p(\mathbf{f}_{t+i}|\mathbf{f}_{t+1:t+i-1}, \mathbf{u}, \mathbf{z}_{t+i}) \\ &\cdot p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1}) d\mathbf{z}_{t+i} \Big) d\mathbf{f}_{\setminus\{f_{t+1:t+i}\}} \end{aligned} \tag{20}$$

where

$$\begin{aligned} &p(\mathbf{z}_{t+i}^{T}|\mathbf{f}_{t+i-n_a:t+i-1}) \\ &\quad = \delta\Big(\mathbf{z}_{t+i}^{T} - [\mathbf{f}_{t+i-n_a:t+i-1}^{T}, \mathbf{x}_{t+i-n_b:t+i-1}^{T}]\Big), \end{aligned} \tag{21}$$

with δ denoting the Dirac delta, indicating that the autoregressive input is constructed deterministically from the known lagged latent values.

Because GP-NARX models are not trained sequentially in the output-error sense, a conservative alternative is to propagate noisy values through the inputs,

$$\begin{aligned} &p(\mathbf{z}_{t+i}^{T}|\mathbf{y}_{t+i-n_a:t+i-1}) \\ &\quad = \delta\Big(\mathbf{z}_{t+i}^{T} - [\mathbf{y}_{t+i-n_a:t+i-1}^{T}, \mathbf{x}_{t+i-n_b:t+i-1}^{T}]\Big), \end{aligned} \tag{22}$$

where $\mathbf{y}_{t+i-n_a:t+i-1} \sim p(\mathbf{y}_{t+i-n_a:t+i-1}|\mathbf{f}_{t+i-n_a:t+i-1})$. This choice can be motivated by the use of noisy observations in the definition of input regressors in GP-NARX models. However, because training emphasises step-ahead prediction, there is no definitive criterion that uniformly favours one propagation strategy. If the error-in-variables effect is negligible, latent propagation in the output-error sense may be preferable; if it is substantial, propagating noisy values may be more appropriate. In practice, the better option is determined empirically (e.g., via cross-validation). Unless stated otherwise, we adopt the latent—output-error—approach.

The simulation is obtained by sequential MC sampling. A k-th sequential sample unrolls through the chained Gaussian distributions,

$$\text{draw } \tilde{\mathbf{u}}^{(k)} \sim q(\mathbf{u}), \tag{23a}$$

$$\text{draw } \tilde{\mathbf{f}}_{t+2-n_a:t+1}^{(k)} \sim p(\mathbf{f}_{t+2-n_a:t+1}|\tilde{\mathbf{u}}^{(k)}), \tag{23b}$$

$$\text{draw } \tilde{\mathbf{z}}_{t+2}^{(k)} \sim p(\mathbf{z}_{t+2}|\tilde{\mathbf{f}}_{t+2-n_a:t+1}^{(k)}), \tag{23c}$$

$$\text{draw } \tilde{\mathbf{f}}_{t+2}^{(k)} \sim p(\mathbf{f}_{t+2}|\tilde{f}_{t+1}^{(k)}, \tilde{\mathbf{u}}^{(k)}, \tilde{\mathbf{z}}_{t+2}^{(k)}), \tag{23d}$$

$$\vdots \tag{23e}$$

$$\text{draw } \tilde{\mathbf{z}}_{t+n}^{(k)} \sim p(\mathbf{z}_{t+n}|\tilde{\mathbf{f}}_{t+n-n_a:t+n-1}^{(k)}), \tag{23f}$$

$$\text{draw } \tilde{\mathbf{f}}_{t+n}^{(k)} \sim p(\mathbf{f}_{t+n}|\tilde{\mathbf{f}}_{t+1:t+n-1}^{(k)}, \tilde{\mathbf{u}}^{(k)}, \tilde{\mathbf{z}}_{t+n}^{(k)}). \tag{23g}$$

Noisy values can then be drawn independently,

$$\text{draw } \tilde{\mathbf{y}}_{t+1:t+n}^{(k)} \sim p\Big(\mathbf{y}_{t+1:t+n}|\tilde{\mathbf{f}}_{t+1:t+n}^{(k)}\Big). \tag{24}$$

Posterior marginals, which can be interpreted as a Gaussian Mixture Model (GMM) [42] at an arbitrary timestep $t+i$, are approximated by

$$q(\mathbf{f}_{t+i}) \approx \frac{1}{r}\sum_{k=1}^{r} p(\mathbf{f}_{t+i}|\tilde{\mathbf{f}}_{t+1:t+i-1}^{(k)}, \tilde{\mathbf{u}}^{(k)}, \tilde{\mathbf{z}}_{t+i}^{(k)}), \tag{25}$$

where r denotes the number of MC samples. At each step, the latent distribution can be represented as a GMM. Overall, the unified simulation can be interpreted as a two-part procedure:

1. Batch sample latent function values with known inputs before the simulation begins (approximation-specific but static);
2. Iteratively sample the remaining quantities conditioned on the static latent function values (shared across approximations but dynamic).

Figure 1 provides a PGM of the unified simulation.

Finally, note that the distribution over $\mathbf{u}$ is approximation-specific. The black interconnected part of the graphical model is static and supports a straightforward batch sampling procedure. The orange interconnected part is the dynamical component; algorithmically, it is consistent across all approximations once the pseudo-input realisations are fixed.

3. Existing Approximations to the Unified GP-NARX Model Simulation

As presented in the previous section, the unified representation of the GP-NARX model simulation opens avenues for novel approximations within the simulation algorithm that are universally applicable across diverse GP-NARX models. Before introducing a novel approximation to the simulation's shared dynamical aspect, we explain in this section the existing approximations in the context of the presented simulation. A summary of existing numerical simulation methods is necessary for the subsequent exposition.

3.1. Fully Correlated Monte Carlo (FCMC) Simulation

We start with an FCMC simulation [37], as the conventional numerical simulation method for the simulation of dynamical GP models. The latent distribution at an arbitrary timestep $t+i$ for FCMC is defined by Equation (A23), where the letter in the equation label means that this equation can be found in the Appendix H.1, which is exact for $q(\mathbf{u}) = p(\mathbf{u}|\mathbf{y}_{1:t})$. This is a fully correlated simulation and enables an arbitrarily good approximation of the posterior if we disregard the numerical complexity with an increasing number of samples. Consequently, we will compare the results of other methods to the results of FCMC and will refer to it as the ground truth.

3.2. Conditionally Independent Monte Carlo (CIMC) Simulation

CIMC simulation [38] does not retain $q(\mathbf{u})$. Still, it analytically marginalises over the distribution (20) instead, pushing the integral in the product term, i.e., Equation (A25) defined in the Appendix H.3. Distribution $q(\mathbf{u})$ is again specific to each approximation and defined by Equation (18).

We are also adding two naive simulations to this list due to their frequent use for completeness and comparison.

3.3. Fully Correlated Naive (FCN) Simulation

FCN simulation [6], also known as naïve simulation, keeps the fully correlated latent distribution, defined by Equation (22), but considers only the mean of the prediction in defining the inputs. Therefore, the inputs are deterministic and numerical integra-

tion with respect to the latent values, which can be obtained sequentially in closed form. Equations (A26) and (A27) in the Appendix H.4 define the distribution.

3.4. Conditionally Independent Naive (CIN) Simulation

CIN simulation [43] further approximates the FCN simulation by considering the latent distribution to be conditionally independent. Equations (A28) and (A27) in the Appendix H define the distribution.

4. Thresholded Correlated Monte Carlo (TCMC) Simulation

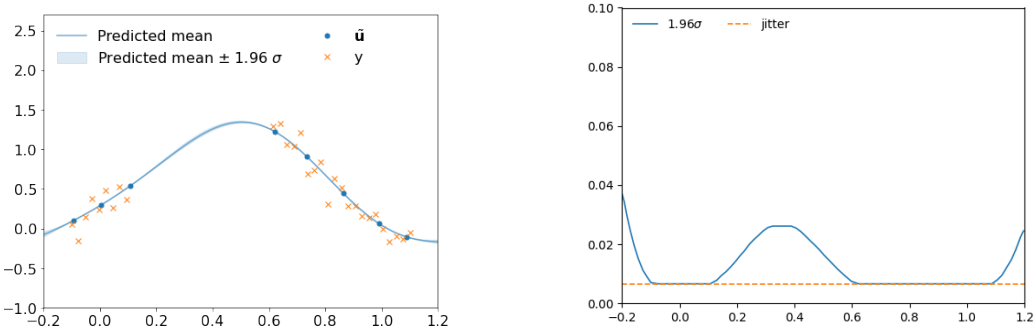
In this section, novel approximations for simulating autoregressive GP models will be introduced. We will explore a specific case that can seamlessly replace the current methodology to reduce computational complexity or memory requirements.

The conditionally independent assumptions made in the approximations may not hold in all practical scenarios, potentially leading to inaccuracies in the simulation results. Because of this, the TCMC simulation approximates the FCMC simulation and avoids the conditionally independent assumption, which introduces some additional uncertainty into the simulation.

4.1. Observing Latent Values

The ground-truth numerical simulation algorithm, as defined in Section 3.1, retains all past latent function values, a direct consequence of the GP formulation. It is crucial to emphasise that the successive realisations are latent. These latent realisations, being uncorrupted observations of the process, convey substantially more information than their noisy counterparts and reduce uncertainty about the underlying latent function. This is particularly important in nonlinear systems, where predictive uncertainty must be propagated through sampling. As more latent values are retained, the uncertainty of the latent function decreases, eventually becoming negligible, at which point additional retained latent values provide little benefit and the function behaves almost deterministically.

For instance, the frequently used Radial-Basis-Function (RBF) covariance function, also known as the squared-exponential covariance function, induces nonlinear functions with infinite differentiability. This results in exceptionally smooth functions, leading to highly correlated latent values. The residual variance in the latent function $p(\mathbf{f}_*|\tilde{\mathbf{u}})$ is depicted in Figure 2 following the acquisition of a sample from the pseudo-point posterior, where $\tilde{\mathbf{u}}$ is drawn from $q(\mathbf{u})$, and predictions are made conditioned on the realisations.



(**a**) (**b**)

Figure 2. Illustration of the residual variance following the observation of a sample from the pseudo-point posterior. The term jitter denotes the noise introduced to the matrix diagonals for numerical stability, typically set at 10^{-6}. (**a**) Prediction of the latent function conditioned on $\tilde{\mathbf{u}} \sim q(\mathbf{u})$ for the VFE approximation. (**b**) Residual variance of the latent function conditioned on $\tilde{\mathbf{u}} \sim q(\mathbf{u})$ for the VFE approximation.

The residual uncertainty in the latent function is minimal, suggesting that, given $\tilde{\mathbf{u}}$, only a limited range of distinct realisations for the latent functions persists.

4.2. Thresholded Simulation Algorithm

The variance of the predictive distribution given the realisation up to the timestep i completely specifies the level of uncertainty for the timestep considered.

We propose a straightforward algorithm incorporating a threshold parameter $\boldsymbol{T}_{\mathbb{V}}$. It is common to add jitter to all diagonal matrices in the algorithm implementation for numerical stability, and so it is added to the covariance matrix to ensure GP-model training stability. The value of jitter is supposed to be selected in a way that its influence on prediction accuracy is not significant. Consequently, delving into uncertainties smaller than the jitter is deemed impractical, introducing potential numerical complications without substantial practical utility.

The threshold parameter $\boldsymbol{T}_{\mathbb{V}}$ shall be selected as the value of jitter added to diagonal matrices in the implementations of training algorithms for numerical stability. We argue that setting $\boldsymbol{T}_{\mathbb{V}}$ below the jitter value is impractical because it can cause numerical problems. On the other hand, one should avoid selecting a larger value of the threshold parameter $\boldsymbol{T}_{\mathbb{V}}$ than the jitter value to retain prediction accuracy. In this way, the threshold for $\boldsymbol{T}_{\mathbb{V}}$ is selected unambiguously.

In this algorithm, the latent realisations are deemed informative and retained if the predictive variance at the specified timestep surpasses the designated threshold. Conversely, the corresponding latent realisations are discarded if the predictive variance falls below the threshold.

A draw from the thresholded simulation is generated using the following procedure: Initially, a sample is drawn from the posterior distribution over the pseudo-points, and this sample is subsequently added to the set of observed latent values $\mathcal{F}$, i.e.,

$$\begin{aligned} \text{draw } \tilde{\mathbf{u}} &\sim q(\mathbf{u}) \\ \mathcal{F} &= \{\tilde{\mathbf{u}}\}. \end{aligned} \tag{26}$$

Next, a posterior latent realisation is drawn at the timestep $t+1$, i.e.,

$$\begin{aligned} \text{draw } \tilde{\mathbf{f}}_{t+2-n_a:t+1} &\sim p(\mathbf{f}_{t+2-n_a:t+1}|\mathcal{F}), \\ \mathcal{F} &= \{\tilde{\mathbf{u}}, \tilde{\mathbf{f}}_{t+2-n_a:t+1}\}. \end{aligned} \tag{27}$$

At timestep $t+2$, the latent realisation is drawn conditioned on preceding realisations, i.e.,

$$\begin{aligned} &\text{draw } \tilde{\mathbf{z}}_{t+2} \sim p(\mathbf{z}_{t+2}|\tilde{\mathbf{f}}_{t+2-n_a:t+1}), \\ &\text{draw } \tilde{\mathbf{f}}_{t+2} \sim p(\mathbf{f}_{t+2}|\mathcal{F}, \tilde{\mathbf{z}}_{t+2}), \\ &\mathcal{F} = \begin{cases} \mathcal{F}, \text{ if } \mathbb{V}[p(\mathbf{f}_{t+2}|\mathcal{F}, \tilde{\mathbf{z}}_{t+2})] \leq T_{\mathbb{V}} \\ \{\tilde{\mathbf{f}}_{t+2}, \mathcal{F}\}, \text{ if } \mathbb{V}[p(\mathbf{f}_{t+2}|\mathcal{F}, \tilde{\mathbf{z}}_{t+2})] > T_{\mathbb{V}}, \end{cases} \end{aligned} \tag{28}$$

where the latent observations are added to the set $\mathcal{F}$ only if they are considered informative, i.e., the variance exceeds the user-defined threshold. Similarly, for the timestep $t+3$

$$\begin{aligned} &\text{draw } \tilde{\mathbf{z}}_{t+3} \sim p(\mathbf{z}_{t+3}|\tilde{\mathbf{f}}_{t+3-n_a:t+2}), \\ &\text{draw } \tilde{\mathbf{f}}_{t+3} \sim p(\mathbf{f}_{t+3}|\mathcal{F}, \tilde{\mathbf{z}}_{t+3}), \\ &\mathcal{F} = \begin{cases} \mathcal{F}, \text{ if } \mathbb{V}[p(\mathbf{f}_{t+3}|\mathcal{F}, \tilde{\mathbf{z}}_{t+3})] \leq T_{\mathbb{V}} \\ \{\tilde{\mathbf{f}}_{t+3}, \mathcal{F}\}, \text{ if } \mathbb{V}[p(\mathbf{f}_{t+3}|\mathcal{F}, \tilde{\mathbf{z}}_{t+3})] > T_{\mathbb{V}}. \end{cases} \end{aligned} \tag{29}$$

The procedure is sequentially repeated up to the timestep $t+n$. The latent sample at the timestep $t+n$ is obtained by

$$\begin{aligned} &\text{draw } \tilde{\mathbf{z}}_{t+n} \sim p(\mathbf{z}_{t+n}|\tilde{\mathbf{f}}_{t+i-n_a:t+n-1}), \\ &\text{draw } \tilde{\mathbf{f}}_{t+n} \sim p(\mathbf{f}_{t+n}|\mathcal{F}, \tilde{\mathbf{z}}_{t+n}) \end{aligned} \tag{30}$$

where $\mathcal{F}$ denotes the set of retained latent values. An MC approximation of the posterior latent distribution that can be seen as a GMM at an arbitrary timestep $t+i$ is defined by

$$p(\mathbf{f}_{t+i}|\mathbf{y}_{1:t}) \approx \frac{1}{r}\sum_{k=1}^{r} p(\mathbf{f}_{t+i}|\mathcal{F}^{(k)}, \tilde{\mathbf{z}}_{t+i}^{(k)}), \tag{31}$$

where k denotes a single sequential draw, r denotes the number of samples, and $\mathcal{F}^{(k)}$ represents the retained latent samples. We will refer to this approximation as the TCMC simulation. The algorithm is defined in the Appendix I as Algorithm A1.

A particular instance of TCMC simulation emerges when the threshold $T_{\mathbb{V}}$ is sufficiently large, leading to the exclusion of latent observations after drawing $\tilde{\mathbf{u}} \sim q(\mathbf{u})$. This condition paves the way for a specialised implementation we propose as an additional contribution. Henceforth, we shall denote this unique variant as a Pseudo Independent MC (PIMC) simulation. Appendix I details the algorithm defining PIMC simulation as Algorithm A2. Conditionally Independent MC simulation (PIMC or CIMC) assumes that the latent function values up to the horizon are conditionally independent given the inputs and the noisy observations, which is not always true. The approximation is reduced if the simulation is based on TCMC, which can, in the limit, recover the ground truth.

The FCMC simulation can also be regarded as a distinctive case within the TCMC simulation framework, characterised by a threshold $T_{\mathbb{V}} = 0$, where all latent observations are retained. Nevertheless, FCMC simulation is quite computationally complex (Table 1), while with the threshold $T_{\mathbb{V}} \neq 0$, the TCMC algorithm reduces the computational complexity. This can be seen in Appendices A and B's static and dynamic system simulation results.

This simulation representation allows us to balance computational complexity against the level of approximation in the simulation. The computational complexities and memory requirements for the unified simulation of GP-NARX models are detailed in Table 1. Notably, *the complexity and memory requirements of the TCMC simulation* are dependent on the parameter p, representing the count of retained latent observations.

Table 1. Computational complexity and memory requirements associated with the numerical approximations employed in the unified simulation of GP-NARX models. The notation used includes the number of samples denoted by r, the predictive horizon by n, the count of retained samples in the TCMC simulation by p, and the number of pseudo-points by m. In the vanilla GP-NARX, m is substituted with the number of training data denoted by t.

Simulation Algorithm	Computational Complexity	Memory Requirements
Fully Correlated Monte Carlo (FCMC) simulation	$\mathcal{O}(r \cdot n \cdot (n+m)^2)$	$\mathcal{O}(r \cdot (n+m)^2)$
Thresholded correlated Monte Carlo (TCMC) simulation	$\mathcal{O}(r \cdot n \cdot (p+m)^2)$	$\mathcal{O}(r \cdot (p+m)^2)$
Pseudo Independent Monte Carlo (PIMC) simulation	$\mathcal{O}(r \cdot n \cdot m^2)$	$\mathcal{O}(r \cdot m^2)$
Conditionally Independent Monte Carlo (CIMC) simulation	$\mathcal{O}(r \cdot n \cdot m^2)$	$\mathcal{O}(r \cdot m^2)$
Fully Correlated Naive (FCN) simulation	$\mathcal{O}(n \cdot (n+m)^2)$	$\mathcal{O}\left((n+m)^2\right)$
Conditionally Independent Naive (CIN) simulation	$\mathcal{O}(n \cdot m^2)$	$\mathcal{O}(m^2)$

Various factors influence the practical determination of the count of retained latent observations:

1. The threshold parameter $T_{\mathbb{V}}$.
2. The selection of the covariance function.
3. Problem-specific considerations, such as learned hyperparameters.

In the upcoming section, we will conduct empirical tests to assess the performance of the simulation approximations.

5. Experimental Evaluation

This section will provide an empirical evaluation of the approximated simulation algorithms on two synthetic examples and a realistic case study, focusing on two distinct cases:

1. Sequential sampling of a static function;
2. Simulation of a GP-NARX model.

Initially, we will analyse latent distributions, comparing the FCMC estimation of the static posterior against various approximations. Subsequently, we will explore running times, the impact of various covariance functions, different sparse approximations, and the number of pseudo-inducing points. The method was also tested on several other cases for approximation error, convergence, and numerical stability, but only two cases are presented in this paper.

The practical implementation of the TCMC algorithm introduces a key distinction from the previously described approach. In the practical TCMC algorithm, latent samples are retained if, for any of the r independent MC samples, the variance of the latent posterior at the considered timestep surpasses the threshold $T_{\mathbb{V}}$. Consequently, this pragmatic strategy retains more latent observations than an individual MC run. Including a few latent observations that might otherwise be discarded ensures the preservation of the dimensionality of retained latent observations and Cholesky factors across all independent MC samples. This approach enables efficient vectorisation of the algorithm, leading to a substantial reduction in running times.

The algorithms were implemented in contemporary software frameworks, namely, TensorFlow [44] and its GP-specific extension, GPflow [45]. The following system specifications were used for experiments conducted in this section:

- CPU: Intel(R) Core(TM) i7-8700K CPU 3.70 GHz;
- GPU: Quadro P4000 8 GB GDDR5 [GP104GL];
- RAM: 2 × 16 GB DIMM DDR4 3200 MHz;
- OS: Ubuntu 18.04.5 LTS;
- Software: GPflow 2.4.0, Tensorflow 2.4.0.

To compare the sequential estimation of the static posterior or the simulation of a GP-NARX model, we will consider a 2-Wasserstein distance W_2 [46] defined by

$$W_2(\mu,\nu)^2 = ||\mathbf{m}_1 - \mathbf{m}_2||^2 + \mathrm{Tr}\left[\mathbf{K}_1 + \mathbf{K}_2 - 2\left(\mathbf{K}_2^{\frac{1}{2}}\mathbf{K}_1\mathbf{K}_2^{\frac{1}{2}}\right)^{\frac{1}{2}}\right], \tag{32}$$

where $\mu \sim \mathcal{N}(\mu|\mathbf{m}_1, \mathbf{K}_1)$, $\nu \sim \mathcal{N}(\nu|\mathbf{m}_2, \mathbf{K}_2)$, and $\mathbf{K} = \mathbf{K}^{\frac{1}{2}}\mathbf{K}^{\frac{1}{2}}$.

In a static problem, the ground truth can be computed in closed form. However, in both static estimation with approximation and the simulation of GP-NARX models, all distributions are estimated from r independent MC samples.

5.1. Sequential Sampling of a Static Function

Firstly, we will consider sequential sampling on a static problem defined by

$$\mathbf{y} = \sin(3\mathbf{x}) + 0.2\cos(10\mathbf{x}) + \mathcal{N}\left(\boldsymbol{\epsilon}|\mathbf{0}, 0.15^2\mathbf{I}\right). \tag{33}$$

The algorithmic structure of sequential sampling in a static model closely resembles the simulation process for GP-NARX models. However, in the static model, the latent distribution is Gaussian and can be analytically evaluated, providing an analytical solution that serves as the ground truth for comparison.

To model the problem defined by Equation (33), we employed both a VFE approximation and a FITC approximation with 30 pseudo-inputs. The latent distribution was estimated using $r = 500$ MC samples. Model training involved maximising the MLL through the LBFGS optimiser. In thresholded MC sampling, a threshold value of $T_{\mathbb{V}} = 10^{-6}$ was chosen.

Discussion

In Figure 3, the 2-Wasserstein distance is illustrated between the closed-form solution of the latent distribution and the estimated distribution from samples for the Matérn($\frac{5}{2}$) covariance function [6]. Results for the other covariance functions in the Matérn family can be found in Appendix A and the definitions in Appendix C. The figures additionally present the running times of various numerical approximations for sequential sampling, demonstrating their performance with respect to an increasing number of input locations.

The computational demands of FCMC static posterior estimation exhibit cubic computational complexity, rendering them impractical for large numbers of test inputs. Conversely, TCMC estimation proves computationally feasible and is accomplished in a significantly shorter timeframe than FCMC. For instance, when considering the Matérn($\frac{5}{2}$) covariance function, the running time of TCMC is only 10% of that required for FCMC estimation when processing 600 test inputs. Nevertheless, the effectiveness of TCMC depends on the smoothness of the covariance function. The selection of a suitable covariance function is an important part of efficient model design, and it also affects the efficiency of model simulation.

Upon closer examination of the 2-Wasserstein distance, computed up to the 600 test inputs for the FCMC, it becomes apparent that the TCMC estimation error closely mirrors that of the FCMC estimation. This similarity arises from the adoption of a very low threshold value, i.e., $T_{\mathbb{V}} = 10^{-6}$. By setting the threshold as low as the numerical jitter, the TCMC estimation recovers the FCMC results, achieving this outcome at only a fraction of the running time for covariance functions that generate relatively smooth functions.

While even less computationally demanding, PIMC and CIMC estimations introduce a discernible error, as indicated by the 2-Wasserstein distance in the earlier figures. This discrepancy is particularly noticeable in CIMC estimation, where the 2-Wasserstein distance is relatively high compared to FCMC estimation.

If the functions induced by the covariance functions exhibit greater roughness, as seen with the Matérn($\frac{3}{2}$) and Matérn($\frac{1}{2}$) covariance functions in Appendix A, the computational advantages of the thresholded sampler diminish. For the Matérn($\frac{1}{2}$) covariance function, these benefits are eliminated. Therefore, the effectiveness of TCMC depends on the smoothness of the covariance function. The selection of a suitable covariance function is an important part of efficient model design, and it also affects the efficiency of model simulation. In situations involving the Matérn($\frac{1}{2}$) covariance function, defaulting to CIMC and PIMC estimation is often necessary. However, a considerable error is expected, as demonstrated by the 2-Wasserstein distance.

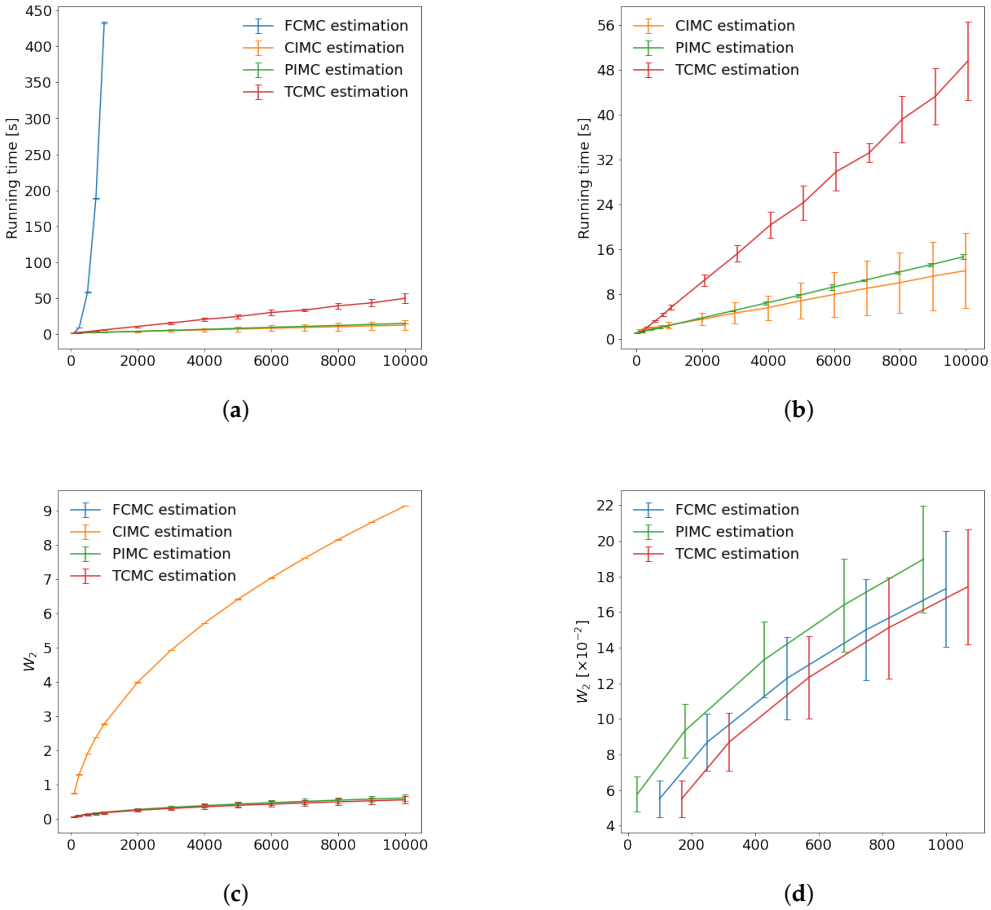
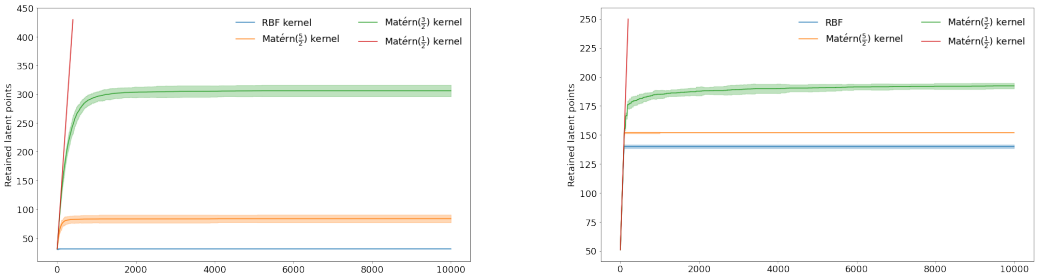


Figure 3. The 2-Wasserstein distance and the running times for the VFE approximation in the sequential estimation of the latent distribution using the Matérn($\frac{5}{2}$) covariance function. (**a**) Running times for the VFE approximation and the Matérn($\frac{5}{2}$) covariance function. (**b**) Closer view of running times for the VFE approximation and the Matérn($\frac{5}{2}$) covariance function. (**c**) 2-Wasserstein distance for the VFE approximation and the Matérn($\frac{5}{2}$) covariance function. (**d**) Closer view of the 2-Wasserstein distance for the VFE approximation and the Matérn($\frac{5}{2}$) covariance function.

In Figure 4, the plot illustrates the retained latent points concerning the increasing number of test locations for various covariance functions and GP approximations. It is evident that, at a certain point, the number of retained points reaches saturation, and keeping more latent values does not provide benefits. Compared to rougher kernel functions, the saturation point arrives earlier for smoother covariance functions.

For instance, in the case of the Matérn($\frac{1}{2}$) covariance function, characterised by highly uncorrelated data points between neighbouring inputs, nearly all latent points are retained. The curve for Matérn($\frac{1}{2}$) can be regarded as a representative proxy for the retained latent points in scenarios involving FCMC estimation. The retained latent points derived from the FCMC estimation of the Matérn($\frac{3}{2}$) covariance function would, for example, closely align with the curve from the TCMC estimation of the Matérn($\frac{1}{2}$) covariance function.

While the sequential estimation in static problems shares similarities with the simulation of GP-NARX models, it is important to note that the issues are distinct. Nevertheless, having a closed-form solution for comparison purposes is advantageous. In static problems, consecutive latent samples have no input/output dependency between consecutive latent samples. As a result, a single Cholesky factor can be computed for multiple independent MC samples, leading to a substantial reduction in running times and memory requirements for the algorithms. In the subsequent section, we will explore these approximations within the context of an illustrative dynamical system.



(**a**) (**b**)

Figure 4. The plot depicts retained latent points for various covariance functions with respect to the number of static data points or predicted steps into the future. The solid regions represent the two standard deviations computed from 10 random restarts. (**a**) Sequential sampling of a static function. (**b**) Simulation of a dynamical system.

5.2. Simulation of a Dynamical System

We consider an illustrative problem [37], where the difference equation governs the true process

$$\begin{aligned} f_{t+1}^1 &= f_t^1 \exp\left(1 - 0.4 f_t^1 - \frac{(2 + 1.2 x_t^1) f_t^2}{1 + (f_t^1)^2}\right), \\ f_{t+1}^2 &= f_t^2 \exp\left(1 - 0.5 x_t^1 - \frac{(1.5 - x_t^2) f_t^2}{f_t^1}\right), \\ x_t^1 &= \cos(2\pi t), \\ x_t^2 &= \sin(2\pi t). \end{aligned} \tag{34}$$

We observe noisy values

$$\begin{aligned} y_t^1 &\sim \mathcal{N}\left(y_t^1 | f_t^1, \sigma_n^2\right), \\ y_t^2 &\sim \mathcal{N}\left(y_t^2 | f_t^2, \sigma_n^2\right). \end{aligned} \tag{35}$$

For the training data, the system was initialised with $[f_0^1, f_0^2] = [0.300, 0.800]$ and simulated for 900 samples. The test data utilised an initialisation of $[f_0^1, f_0^2] = [0.380, 0.326]$, and the simulation was conducted for the specified number of samples. Gaussian noise with $\sigma_n^2 = 10^{-2}$ was introduced to corrupt the data. NARX model parameters were set as $n_a = n_b = 4$. The system was modelled by a GP-NARX (VFE) model with 50 pseudo-inputs. The latent distribution was estimated using $r = 500$ MC samples. Hyperparameters were determined through MLL maximisation with the LBFGS optimiser. In the TCMC simulation, a threshold of $T_{\mathbb{V}} = 10^{-6}$ was selected.

Discussion

In Figure 5, the running times of the simulation are illustrated up to 10,000 steps into the future for the Matérn($\frac{5}{2}$) covariance function. The results for the other covariance functions in the Matérn family are shown in Appendix A. The running times for the FCMC simulation escalate rapidly, mirroring the trend observed in the static case. As the prediction horizon extends, the FCMC simulation cannot be practically realised.

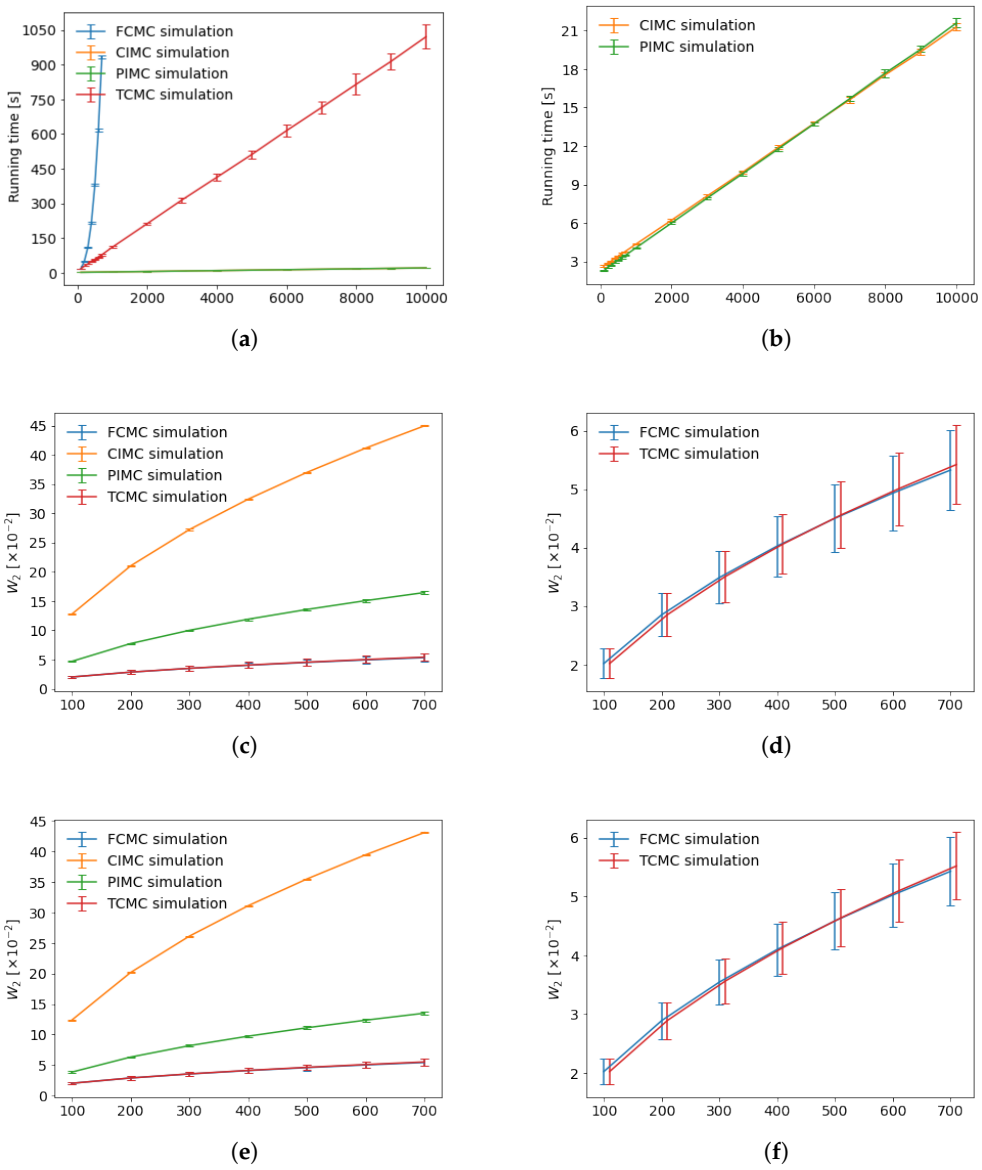


Figure 5. The 2-Wasserstein distance and simulation running times are presented for the GP-NARX (VFE) model using the Matérn($\frac{5}{2}$) covariance function. (**a**) Running time comparison of the simulation. (**b**) Closer view of the running time comparison of the simulation. (**c**) 2-Wasserstein distance for f^1. (**d**) Closer view of the 2-Wasserstein distance for f^1. (**e**) 2-Wasserstein distance for f^2. (**f**) Closer view of the 2-Wasserstein distance for f^2.

The approximations are computed within a reasonable time frame, as demonstrated for the Matérn($\frac{5}{2}$) and Matérn($\frac{3}{2}$) covariance functions, i.e., covariance functions that generate relatively smooth functions. Nevertheless, it is noteworthy that running times in simulation are considerably higher compared to static estimation, as a distinct Cholesky factor needs to be computed for each independent MC sample.

For the Matérn($\frac{1}{2}$) covariance function, the running times of the TCMC simulation do not exhibit a reduction when compared to the FCMC simulation. In this scenario, the TCMC simulation is not practically feasible for large prediction horizons since nearly all latent samples are retained, resulting in a simulation closely resembling the FCMC counterpart. One alternative is to employ an approximation with additional assumptions, such as the PIMC and CIMC simulations. However, as the 2-Wasserstein distance demonstrates, it is essential to anticipate some error in estimating the latent response.

However, the commonly employed CIMC approximation in the literature shows no discernible advantages compared to the proposed PIMC simulation. The response estimated through a PIMC simulation offers a superior approximation to the FCMC and can be obtained within the same computational time as the CIMC simulation. Consequently, in situations where the TCMC simulation cannot be obtained, opting for the PIMC simulation is recommended over the CIMC simulation.

For smoother covariance functions, such as Matérn($\frac{5}{2}$) and Matérn($\frac{3}{2}$) covariance functions, the TCMC simulation significantly reduces the running time of the simulation while maintaining a 2-Wasserstein distance practically identical to that in the FCMC simulation. The chosen threshold $T_{\mathbb{V}} = 10^{-6}$ in this section is equivalent to the jitter added to covariance matrices for numerical stability, representing the level of variance we are willing to sacrifice for improved numerical conditioning. When the threshold matches the numerical jitter, the TCMC simulation can be considered essentially equivalent to the FCMC simulation. This ensures a reasonable computation time and scalability to larger predictive horizons.

Figure 4 illustrates the retained latent points concerning the increasing number of predicted steps into the future for different covariance functions. Once more, it is apparent that, at a certain point, the number of retained points reaches saturation. Again, we can observe that the Matérn($\frac{1}{2}$) covariance function serves as a proxy for the FCMC simulation.

The synthetic examples show that the TCMC simulation efficiently recovers the FCMC results within significantly less computational time when dealing with relatively smooth functions. However, the computational advantages diminish for covariance functions that induce rougher functions. This, however, is generally not problematic in practice for the following reasons:

1. Smooth functions are often preferred for small data sets as they tend to generalise better;
2. The choice of the covariance function becomes less critical for larger data sets;
3. In the case of large data sets, the uncertainty of the latent function posterior is typically relatively small.

The first point emphasises the preference for well-regularised models, often achieved through some smoothness assumption. In the context of GP-specific models, this assumption is defined on a meta-level, reflecting the properties of functions induced by respective covariance functions. An example of such a smoothness assumption is the level of differentiability, indicating how many times the function is differentiable.

The second point pertains to selecting the covariance function, particularly in the context of large data sets. In such cases, the abundance of data is a penalty for overly flexible models, rendering the choice of the covariance function less critical. Models employing various covariance functions, yet appropriately flexible to mitigate bias, are anticipated to exhibit relatively similar performance. This consideration becomes relevant when the objective is to simulate data over extended prediction horizons in the simulation of GP-NARX models. A preference for a smoother covariance function may be justified, especially if the model's performance does not change substantially.

The third point emphasises that the uncertainty associated with the expected latent function posterior tends to be relatively small in large data sets. This latent uncertainty captures the epistemic uncertainty arising from the lack of observed data. As more data become available, the latent uncertainty diminishes, requiring fewer latent samples to adequately characterise the latent function.

The upcoming section will showcase the simulation approximations using a real-world example.

5.3. Case Study

In the previous examples, our focus was not on model selection, such as determining the optimal covariance function, but rather on benchmarking the performance of simulation algorithms. In this section, we will delve into modelling a more complex dynamical benchmark. This benchmark entails a relatively large training data set, making it impractical to employ the vanilla GP-NARX model in practice [38]. Despite sparse models approximating the training process, the test data sets are too extensive for the FCMC simulation to be feasible. Our aim here is to showcase the application of the TCMC simulation on this benchmark.

Silverbox benchmark represents the second-order linear time-invariant system with the third-degree polynomial static nonlinearity around it in feedback. It is a real-world system that collects data from an electric circuit [47]. The system theoretically obeys the equation

$$m\frac{d^2y(t)}{dt^2} - d\frac{dy(t)}{dt} + ay(t) + by(t)^3 = u(t). \tag{36}$$

We selected the training data from samples 40,586 to 127,410 and the test data as the first 40,495 samples [48]. The system was excited with a white Gaussian noise sequence filtered by a ninth-order discrete-time Butterworth filter. Training data were obtained by exciting ten successive realisations of a random odd multi-sine signal [47]. We additionally standardised the data and added Gaussian noise to the observations with the standard deviation of $\sigma_n = 5 \times 10^{-2}$.

The meta-parameters of the NARX model were selected as $n_b = n_a = 5$. We used 300 pseudo-inputs initialised with the k-means clustering algorithm as suggested in [11]. The hyperparameters and the pseudo-input locations were determined by MLL maximisation using the Limited-Memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) optimiser [49].

Discussion

Table 2 presents the outcomes of the 10-fold cross-validation on the training data. Notably, for the Silverbox dataset, the most effective combination involves a linear and a Matérn($\frac{5}{2}$) covariance function, closely followed by a combination of a linear and a Matérn($\frac{3}{2}$) covariance function. The performance metrics indicate that the model describes the data well in terms of both mean values, as evident in the RMSE and standardised mean squared error (SMSE) metrics [5], and probability distributions, as indicated by the mean standardised log loss (MSLL) metric [5].

Table 3 presents the outcomes of prediction and simulation on the test data using the top two covariance functions selected through cross-validation. Analogous to the prediction results, the performance metrics affirm that the model adeptly describes the data, extending its capability to simulation. The simulation results exhibit negligible variations among the employed approximations, as seen with the performance metrics.

Notably, the TCMC simulation, which practically recovers the FCMC results, in our case considered as ground truth, is attainable on a personal computer. However, its running time remains impractical for scenarios where the simulation needs to be repeated multiple times (as in cross-validation or rolling forecasts) or when more real-time forecasts are required.

In contrast, the PIMC simulation, previously observed to outperform the CIMC approximation, can be obtained in just 5% of the time required for the TCMC simulation. Hence, the PIMC simulation emerges as a convenient alternative, effectively balancing computational complexity with the accuracy of the estimated latent response.

Table 2. Means and the respective 95% confidence intervals of the RMSE, SMSE, and MSLL for the prediction on the Silverbox data. The presented performance metrics are for 10-fold cross-validation over the training data. The best values are emphasised in bold. A linear covariance function (L.) was added to the respective covariance functions.

	L. + Matérn($\frac{5}{2}$)	**L. + Matérn($\frac{3}{2}$)**	**L. + Matérn($\frac{1}{2}$)**
RMSE [$\times 10^{-3}$]	**3.149 ± 0.049**	3.151 ± 0.049	3.164 ± 0.051
SMSE [$\times 10^{-3}$]	**3.333 ± 0.137**	3.337 ± 0.135	3.363 ± 0.142
MSLL	**−2.851 ± 0.020**	−2.851 ± 0.020	−2.847 ± 0.020

Table 3. Results of prediction and simulation on the Silverbox test data are presented for the top two covariance functions selected through cross-validation. The wall time is provided for both prediction and various simulation approximations. For context, the training, conducted with the LBFGS optimiser and spanning 100 steps, took 200 s as a reference. These experiments were executed on a computer (Apple MacBook Pro, M2, 16 GB). The best values are emphasised in bold.

		Prediction	**Simulation (TCMC)**	**Simulation (PIMC)**	**Simulation (CIMC)**
L. + Matérn($\frac{5}{2}$)	RMSE [$\times 10^{-3}$]	**3.156**	**2.999**	**3.042**	**3.002**
	SMSE [$\times 10^{-3}$]	**3.480**	**3.143**	**3.232**	**3.148**
	MSLL	**−2.832**	**−2.643**	**−2.627**	**−2.627**
	Wall time [s]	2	4195	219	198
L. + Matérn($\frac{3}{2}$)	RMSE [$\times 10^{-3}$]	3.202	3.526	3.486	3.445
	SMSE [$\times 10^{-3}$]	3.583	4.344	4.245	4.146
	MSLL	−2.820	2.596	−2.583	−2.585
	Wall time [s]	2	4217	228	179

6. Conclusions

GP models offer a probabilistic regression approach characterised by the elegance of the nonparametric function estimation. They reduce the need for strict assumptions or at least elevate them to a higher level. This mechanism presents a more systematic regression approach compared to other contemporary forecasting methods, where the selection of meta-parameters often involves an element of uncertainty.

Various approximations have partially alleviated the computational burden of GP models, marked by cubic complexity regarding the training data. A common strategy involves learning a set of pseudo-points to capture a sparse representation of the underlying data. Nevertheless, this approach primarily benefits static systems, where MLL and predictions can be obtained analytically.

Given the inherent propagation of uncertain inputs during training and multi-step-ahead predictions, the challenge intensifies when dealing with dynamical systems. In the training phase, autoregression can simplify the dynamical problem to that of the static case. However, simulating such autoregressive models remains a significant challenge due to the absence of analytical solutions, often necessitating MC estimation. This, in turn, imposes substantial computational demands and mandates specific algorithms tailored to each sparse approximation.

In this investigation, we proposed a unified simulation framework invariant to sparse and variations approximations up to obtaining a static sample from the pseudo-point posterior. This framework incorporates all known numerical simulations of GP-NARX models.

Moreover, we introduced a novel MC estimation algorithm for simulating autoregressive GP models. This algorithm addresses the computational complexity and specificity challenges associated with conventional MC algorithms tailored for individual sparse approximations. The presented algorithm is a versatile plug-and-play solution applicable across various sparse approximations.

When acquiring an utterly accurate simulation is infeasible, our proposed algorithm, Thresholded Conditional MC (TCMC), provides a close approximation to what can be regarded as the ground truth, i.e., FCMC, up to numerical precision. A specialised TCMC variant, PIMC simulation, can be employed for scenarios with stricter time constraints. PIMC attains simulation speeds comparable to static counterparts while outperforming existing CIMC approximations in accuracy.

The PIMC simulation does not keep the consecutive latent observations in memory. However, some latent uncertainty is still reduced, especially in kernels that induce smooth functions. PIMC should be preferred over the CIMC simulation since it better approximates the ground truth and has the same computational complexity.

TCMC simulation can reduce computational requirements and improve estimation of the latent response compared with CIMC and PIMC simulations. It introduces a tradeoff parameter that can maximise the computational resources and, consequently, the quality of the estimated latent response. If the threshold equals the numerical jitter, the TCMC simulation retrieves the FCMC simulation for only a fraction of the computational cost of the ground truth when kernels that induce smooth functions are used.

To assess the efficacy of these approximations, we conducted comparisons using a static system with analytically obtainable ground truth and an illustrative dynamical example. Furthermore, we validated the algorithmic performance by applying it to model a real-world case study with a substantial dataset. Notably, the simulations were successfully executed on a personal computer, a task previously deemed unattainable.

Following the proof of concept presented in this publication, more practical applications of the method are envisaged. Potential examples of uncertainty propagation in GP-ARX models include continuous-time Simultaneous Localisation and Mapping (SLAM) [50] and Gaussian Process Occupancy Mapping (GP-OM) for 3D point clouds [51] in the field of robotics. In SLAM, the propagation of robot odometry uncertainty over time is inherently an autoregressive process, and modelling the continuous probability distribution of dense point clouds with GPs is computationally expensive due to the large number of data points. The core philosophy of the TCMC algorithm—dynamically discarding latent variables when the predictive variance falls below the threshold $T_{\mathbb{V}}$—is mathematically analogous to dynamically downsampling redundant Lidar point clouds in map regions where the surface probability is already highly confident.

Moreover, for future endeavours in advancing the simulation of the GP-NARX model, a focus on refining the trade-off between computational complexity and the precision of the estimated response is warranted. In addition, special attention will be given to assessing uncertainty calibration, long-horizon stability, or preservation of temporal dependence structures when using FCMC. An intriguing avenue for future exploration is a hybrid approach that combines the efficiency of the TCMC simulation with samplers designed to approximate the latent posterior using basis functions at test time. This integrated strategy is expected to enhance computational efficiency and response accuracy, offering a promising direction for further research.

Author Contributions: Conceptualisation, T.K. and J.K.; methodology, T.K.; software, T.K.; validation, T.K. and J.K.; investigation, T.K.; resources, T.K.; data curation, T.K.; writing—original draft preparation, T.K.; writing—review and editing, T.K. and J.K.; project administration, J.K.; funding acquisition, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge that the Slovenian Research and Innovation Agency financially supported research core funding No. P2-0001 and a PhD grant for Tadej Krivec.

Data Availability Statement: The data presented in this study are openly available in GitHub at https://github.com/tadejkrivec/GPflow-NARX (accessed on 10 June 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Sequential Sampling of a Static Function

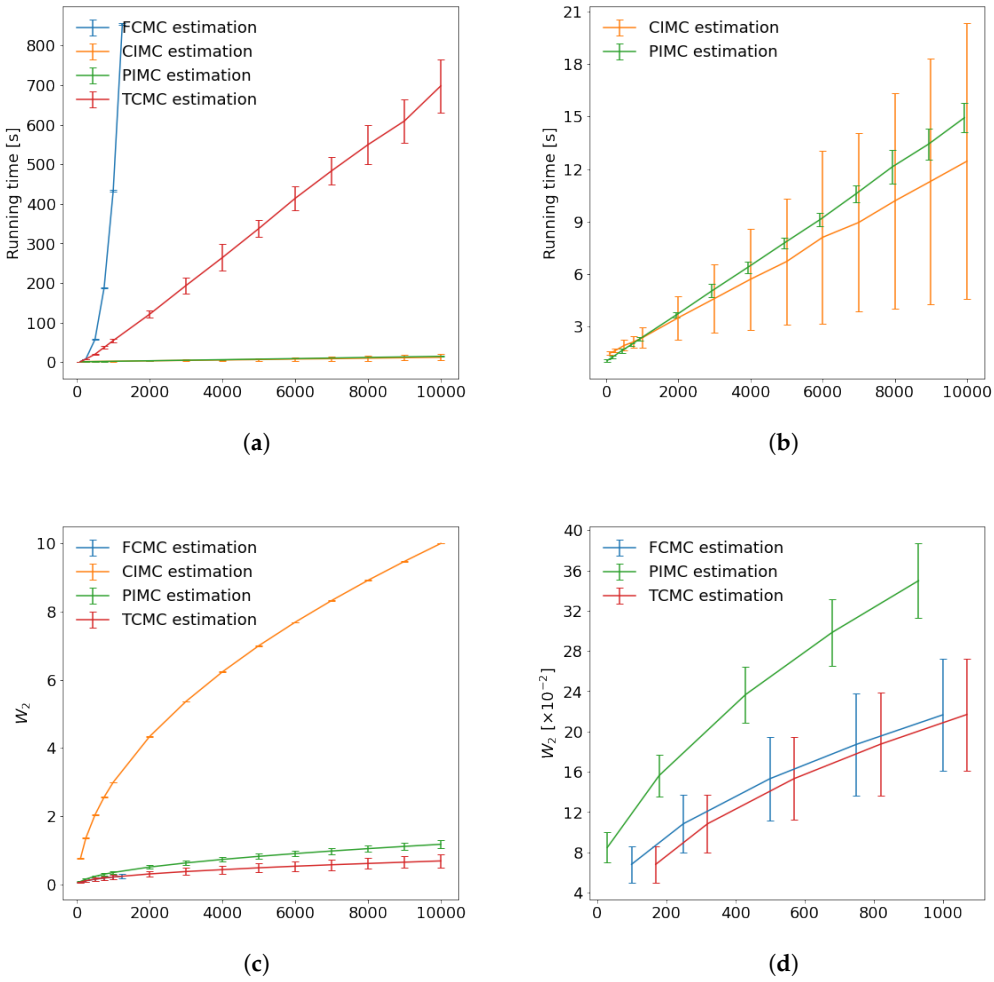


Figure A1. The 2-Wasserstein distance and the running times for the VFE approximation in the sequential estimation of the latent distribution using the Matérn($\frac{3}{2}$) covariance function. (**a**) Running times for the VFE approximation and the Matérn($\frac{3}{2}$) covariance function. (**b**) Closer view of running times for the VFE approximation and the Matérn($\frac{3}{2}$) covariance function. (**c**) 2-Wasserstein distance for the VFE approximation and the Matérn($\frac{3}{2}$) covariance function. (**d**) Closer view of the 2-Wasserstein distance for the VFE approximation and the Matérn($\frac{3}{2}$) covariance function.

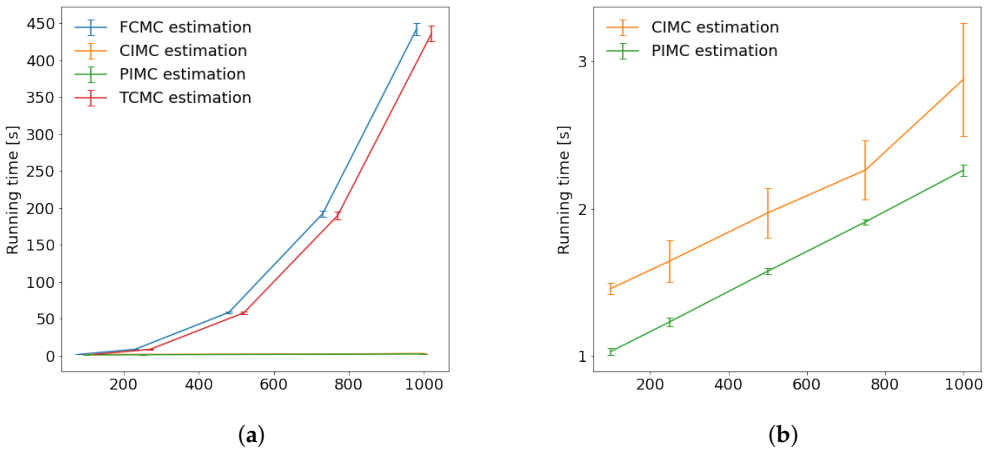
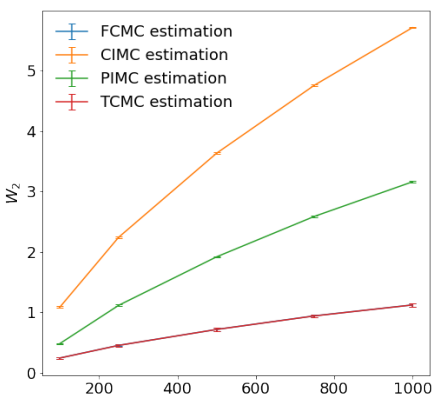
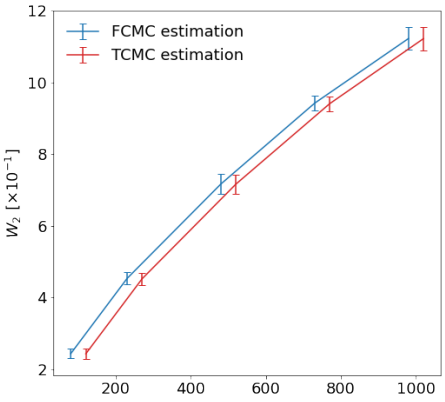


Figure A2. *Cont.*



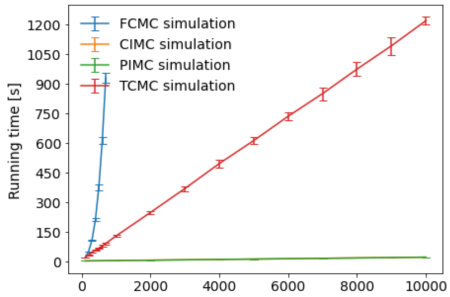
(**c**)



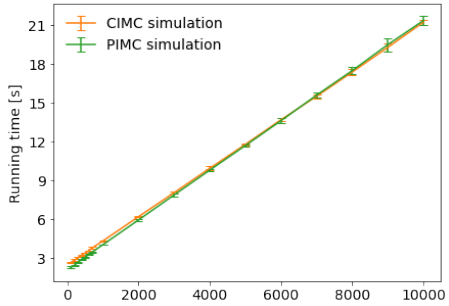
(**d**)

Figure A2. The 2-Wasserstein distance and the running times for the VFE approximation in the sequential estimation of the latent distribution using the Matérn($\frac{1}{2}$) covariance function. (**a**) Running times for the VFE approximation and the Matérn($\frac{1}{2}$) covariance function. (**b**) Closer view of running times for the VFE approximation and the Matérn($\frac{1}{2}$) covariance function. (**c**) 2-Wasserstein distance for the VFE approximation and the Matérn($\frac{1}{2}$) covariance function. (**d**) Closer view of the 2-Wasserstein distance for the VFE approximation and the Matérn($\frac{1}{2}$) covariance function.

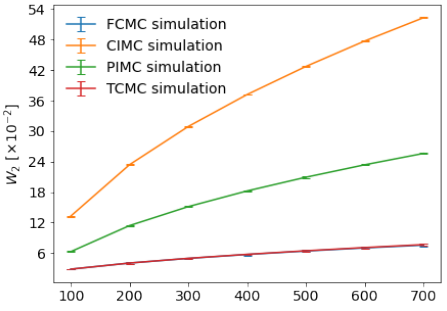
Appendix B. Simulation of a Dynamical System



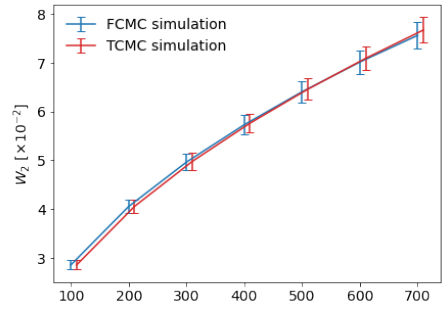
(**a**)



(**b**)



(**c**)



(**d**)

Figure A3. *Cont.*

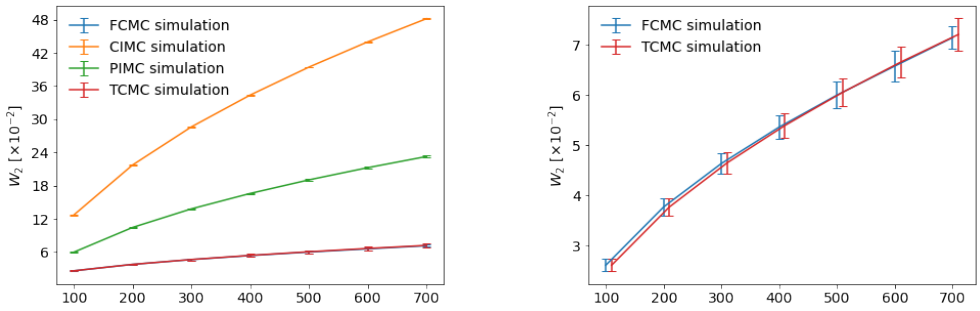


Figure A3. The 2-Wasserstein distance and the running times for the simulation of the GP-NARX (VFE) model using the Matérn($\frac{3}{2}$) covariance function. (**a**) Running time comparison of the simulation. (**b**) Closer view of the running time comparison of the simulation. (**c**) 2-Wasserstein distance for f^1. (**d**) Closer view of the 2-Wasserstein distance for f^1. (**e**) 2-Wasserstein distance for f^2. (**f**) Closer view of the 2-Wasserstein distance for f^2.

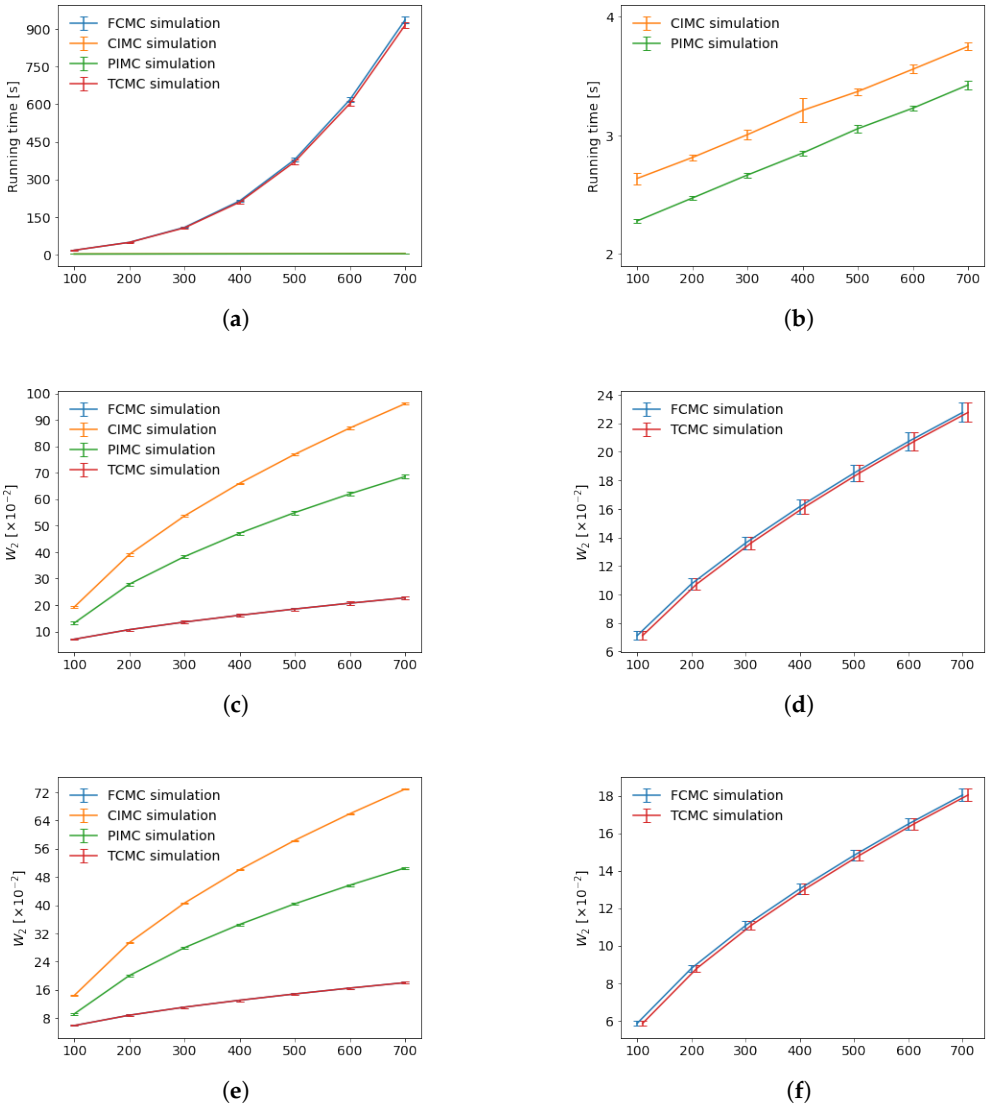


Figure A4. The 2-Wasserstein distance and the running times for the simulation of the GP-NARX (VFE) model using the Matérn($\frac{1}{2}$) covariance function. (**a**) Running time comparison of the simulation. (**b**) Closer view of the running time comparison of the simulation. (**c**) 2-Wasserstein distance for f^1. (**d**) Closer view of the 2-Wasserstein distance for f^1. (**e**) 2-Wasserstein distance for f^2. (**f**) Closer view of the 2-Wasserstein distance for f^2.

Appendix C. Covariance Functions

Matérn Covariance Functions

The definitions of the Matérn covariance functions are the following:

$$k_{\text{Matérn}(\nu=\frac{1}{2})}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-r}, \tag{A1a}$$

$$k_{\text{Matérn}(\nu=\frac{3}{2})}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 (1 + \sqrt{3}r) e^{-\sqrt{3}r}, \tag{A1b}$$

$$k_{\text{Matérn}(\nu=\frac{5}{2})}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 (1 + \sqrt{5}r + \frac{5}{3}r^2) e^{-\sqrt{5}r}, \tag{A1c}$$

$$k_{\text{Matérn}(\nu=\infty)}(\mathbf{x}_i, \mathbf{x}_j) = k_{\text{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-\frac{r^2}{2}}, \tag{A1d}$$

where $r = \frac{1}{l}||\mathbf{x}_i - \mathbf{x}_j||$. Kernels with the automatic relevance determination (ARD) property define

$$r = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{\Lambda}^{-1} (\mathbf{x}_i - \mathbf{x}_j)}, \tag{A2}$$

where $\mathbf{\Lambda}^{-1} = \text{diag}([l_1^{-2}, \ldots, l_d^{-2}])$ and d is the number of columns in $\mathbf{x}$.

Appendix D. Model Learning

This appendix provides a quick overview of training sparse Gaussian process models.

When the datasets are relatively large, a popular approach is to maximise the logarithm of the marginal likelihood or the ELBO with respect to the hyperparameters, i.e.,

$$\hat{\mathbf{\Theta}} = \arg\max_{\mathbf{\Theta}} [\log F(\mathbf{\Theta})], \tag{A3}$$

where $\mathbf{\Theta}$ represents the free parameters of F. Explicit definitions of $F(\mathbf{\Theta})$ for various approximations can be found in Appendix D. The evaluation of $F(\mathbf{\Theta})$ of a vanilla GP-NARX model results in a computational complexity of $\mathcal{O}(t^3)$, where t denotes the number of observed time steps. The training of the GP-NARX (FITC) and GP-NARX (VFE) results in a computational complexity of $\mathcal{O}(tm^2)$, where m denotes the number of pseudo-inputs. The training of the GP-NARX (SVGP) models results in a computational complexity $\mathcal{O}(m^3)$. The memory requirements for the vanilla GP-NARX model are $\mathcal{O}(t^2)$ and $\mathcal{O}(m^2)$ for sparse approximations. The following sections provide unified descriptions of the GP-NARX model's prediction and simulation.

Appendix E. Marginal Likelihoods

Appendix E.1. GP-NARX (Vanilla)

$$\begin{aligned} F(\mathbf{\Theta}) = F\left(\boldsymbol{\theta}, \sigma_n^2\right) = \\ -\frac{1}{2}\mathbf{y}_{1:t}^T \left(\mathbf{K}_{f_{1:t}, f_{1:t}} + \mathbf{I}\sigma_n^2\right)^{-1} \mathbf{y}_{1:t} - \frac{1}{2}\log\left|\mathbf{K}_{f_{1:t}, f_{1:t}} + \mathbf{I}\sigma_n^2\right| - \frac{n}{2}\log 2\pi \end{aligned} \tag{A4}$$

Appendix E.2. GP-NARX (FITC)

$$\begin{aligned} F(\mathbf{\Theta}) &= F\left(\boldsymbol{\theta}, \sigma_n^2, \mathbf{z}'_{1:m}\right) \\ &= -\frac{1}{2}\mathbf{y}_{1:t}^T \left(\mathbf{Q}_{f_{1:t}, f_{1:t}} - \text{diag}\left[\mathbf{Q}_{f_{1:t}, f_{1:t}} - \mathbf{K}_{f_{1:t}, f_{1:t}}\right] + \mathbf{I}\sigma_n^2\right)^{-1} \mathbf{y}_{1:t} \\ &\quad - \frac{1}{2}\log\left|\mathbf{Q}_{f_{1:t}, f_{1:t}} - \text{diag}\left[\mathbf{Q}_{f_{1:t}, f_{1:t}} - \mathbf{K}_{f_{1:t}, f_{1:t}}\right] + \mathbf{I}\sigma_n^2\right| - \frac{n}{2}\log 2\pi \end{aligned} \tag{A5}$$

Appendix E.3. GP-NARX (VFE)

$$\begin{aligned} F(\boldsymbol{\Theta}) &= F\left(\boldsymbol{\theta}, \sigma_n^2, \mathbf{z}'_{1:m}\right) \\ &= \log\left[\mathcal{N}(\mathbf{y}_{1:t}|\mathbf{0}, \mathbf{Q}_{f_{1:t},f_{1:t}} + \mathbf{I}\sigma_n^2)\right] - \frac{1}{2\sigma_n^2}\mathrm{tr}(\mathbf{K}_{f_{1:t},f_{1:t}} - \mathbf{Q}_{f_{1:t},f_{1:t}}) \end{aligned} \tag{A6}$$

Appendix E.4. GP-NARX (SVGP)

$$\begin{aligned} F(\boldsymbol{\Theta}) &= F\left(\boldsymbol{\theta}, \sigma_n^2, \mathbf{z}'_{1:m}, \mathbf{m}, \boldsymbol{\Phi}\right) \\ &= \sum_{n=1}^{t} \mathbb{E}_{q(f_n)}[\log p(y_n|f_n)] - \mathrm{KL}[q(\mathbf{u})||p(\mathbf{u})], \end{aligned} \tag{A7}$$

where

$$\begin{aligned} p(y_n|f_n) &= \mathcal{N}\left(y_n|f_n, \sigma_n^2\right), \\ q(\mathbf{u}) &= \mathcal{N}(\mathbf{u}|\mathbf{m}, \boldsymbol{\Phi}), \\ p(\mathbf{u}) &= \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{u,u}), \end{aligned} \tag{A8}$$

and KL defines the Kullback–Leibler divergence between $q(\mathbf{u})$ and $p(\mathbf{u})$.

Appendix F. Predictive Posteriors

This section defines the predictions in GP-NARX models, i.e., $p(f_{t+1}|\mathbf{y}_{1:t}) = \mathcal{N}(\mathbb{E}[f_{t+1}|\mathbf{y}_{1:t}], \mathbb{V}[f_{t+1}|\mathbf{y}_{1:t}])$.

Appendix F.1. GP-NARX

The mean of the predictive posterior is defined by

$$\mathbb{E}[f_{t+1}|\mathbf{y}_{1:t}] = \mathbf{K}_{f_{t+1},f_{1:t}}\left(\mathbf{K}_{f_{1:t},f_{1:t}} + \mathbf{I}\sigma_n^2\right)^{-1}\mathbf{y}_{1:t}. \tag{A9}$$

The variance of the predictive posterior is defined by

$$\mathbb{V}[f_{t+1}|\mathbf{y}_{1:t}] = \mathbf{K}_{f_{t+1},f_{t+1}} - \mathbf{K}_{f_{t+1},f_{1:t}}\left(\mathbf{K}_{f_{1:t},f_{1:t}} + \mathbf{I}\sigma_n^2\right)^{-1}\mathbf{K}_{f_{1:t},f_{t+1}}. \tag{A10}$$

Appendix F.2. GP-NARX (FITC)

The mean of the predictive posterior is defined by

$$\mathbb{E}[f_{t+1}|\mathbf{y}_{1:t}] = \mathbf{K}_{f_{t+1},u}\left(\mathbf{K}_{u,u} + \mathbf{K}_{u,f_{1:t}}\boldsymbol{\Lambda}^{-1}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,f_{1:t}}\boldsymbol{\Lambda}^{-1}\mathbf{y}_{1:t}, \tag{A11}$$

where

$$\boldsymbol{\Lambda} = \mathrm{diag}\left[\mathbf{K}_{f_{1:t},f_{1:t}} - \mathbf{K}_{f_{1:t},u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f_{1:t}} + \mathbf{I}\sigma_n^2\right]. \tag{A12}$$

https://doi.org/10.3390/math14122111

The variance of the predictive posterior is defined by

$$\mathbb{V}[f_{t+1}|\mathbf{y}_{1:t}] = \mathbf{K}_{f_{t+1},f_{t+1}} - \mathbf{K}_{f_{t+1},u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f_{t+1}}. \tag{A13}$$

Appendix F.3. GP-NARX (VFE)

The mean of the predictive posterior is defined by

$$\mathbb{E}[f_{t+1}] = \sigma_n^{-2}\mathbf{K}_{f_{t+1},u}\left(\mathbf{K}_{u,u} + \sigma_n^{-2}\mathbf{K}_{u,f_{1:t}}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,f_{1:t}}\mathbf{y}_{1:t}. \tag{A14}$$

The variance of the predictive posterior is defined by

$$\mathbb{V}[f_{t+1}] = \mathbf{K}_{f_{t+1},f_{t+1}} - \mathbf{K}_{f_{t+1},u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f_{t+1}} + \mathbf{K}_{f_{t+1},u}\left(\mathbf{K}_{u,u} + \sigma_n^{-2}\mathbf{K}_{u,f_{1:t}}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,f_{t+1}}. \tag{A15}$$

Appendix F.4. GP-NARX (SVGP)

The mean of the predictive posterior is defined by

$$\mathbb{E}[f_{t+1}] = \mathbf{K}_{f_{t+1},u}\mathbf{K}_{u,u}^{-1}\mathbf{m}. \tag{A16}$$

The variance of the predictive posterior is then defined by

$$\mathbb{V}[f_{t+1}] = \mathbf{K}_{f_{t+1},f_{t+1}} - \mathbf{K}_{f_{t+1},u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f_{t+1}} + \mathbf{K}_{f_{t+1},u}\mathbf{K}_{u,u}^{-1}\mathbf{\Phi}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f_{t+1}}, \tag{A17}$$

where $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{\Phi})$.

Appendix G. Pseudo-Point Posteriors

Appendix G.1. GP-NARX (Vanilla)

In GP-NARX (Vanilla), the pseudo-point posterior is defined by $q(\mathbf{u}) = p(\mathbf{f}_{1:t}|\mathbf{y}_{1:t})$, where the mean and the variance of the respective Gaussian distribution are defined by

$$\mathbb{E}[\mathbf{f}_{1:t}|\mathbf{y}_{1:t}] = \mathbf{K}_{f_{1:t},f_{1:t}}\left(\mathbf{K}_{f_{1:t},f_{1:t}} + \mathbf{I}\sigma_n^2\right)^{-1}\mathbf{y}_{1:t}, \tag{A18a}$$

$$\mathbb{V}[\mathbf{f}_{1:t}|\mathbf{y}_{1:t}] = \mathbf{K}_{f_{1:t},f_{1:t}} - \mathbf{K}_{f_{1:t},f_{1:t}}\left(\mathbf{K}_{f_{1:t},f_{1:t}} + \mathbf{I}\sigma_n^2\right)^{-1}\mathbf{K}_{f_{1:t},f_{1:t}}. \tag{A18b}$$

Appendix G.2. GP-NARX (FITC)

In GP-NARX (FITC), the pseudo-point posterior is defined by $q(\mathbf{u}) = p(\mathbf{u}|\mathbf{y}_{1:t})$, where the mean and the variance of the respective Gaussian distribution are defined by

$$\mathbb{E}[\mathbf{u}|\mathbf{y}_{1:t}] = \mathbf{K}_{u,u}\left(\mathbf{K}_{u,u} + \mathbf{K}_{u,f_{1:t}}\mathbf{\Lambda}^{-1}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,f_{1:t}}\mathbf{\Lambda}^{-1}\mathbf{y}_{1:t}, \tag{A19a}$$

$$\mathbb{V}[\mathbf{u}|\mathbf{y}_{1:t}] = \mathbf{K}_{u,u}\left(\mathbf{K}_{u,u} + \mathbf{K}_{u,f_{1:t}}\mathbf{\Lambda}^{-1}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,u}, \tag{A19b}$$

where

$$\mathbf{\Lambda} = \operatorname{diag}\left[\mathbf{K}_{f_{1:t},f_{1:t}} - \mathbf{K}_{f_{1:t},u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f_{1:t}} + \mathbf{I}\sigma_n^2\right]. \tag{A20}$$

Appendix G.3. GP-NARX (VFE)

In GP-NARX (VFE), the pseudo-point posterior is defined by $q(\mathbf{u}) \approx p(\mathbf{u}|\mathbf{y}_{1:t})$, where the mean and the variance of the respective Gaussian distribution are defined by

$$\mathbb{E}[\mathbf{u}] = \sigma_n^{-2}\mathbf{K}_{u,u}\left(\mathbf{K}_{u,u} + \sigma_n^{-2}\mathbf{K}_{u,f_{1:t}}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,f_{1:t}}\mathbf{y}_{1:t}, \tag{A21a}$$

$$\mathbb{V}[\mathbf{u}] = \mathbf{K}_{u,u}\left(\mathbf{K}_{u,u} + \sigma_n^{-2}\mathbf{K}_{u,f_{1:t}}\mathbf{K}_{f_{1:t},u}\right)^{-1}\mathbf{K}_{u,u}. \tag{A21b}$$

Appendix G.4. GP-NARX (SVGP)

In GP-NARX (SVGP), the pseudo-point posterior is defined by $q(\mathbf{u}) \approx p(\mathbf{u}|\mathbf{y}_{1:t})$, where the mean and the variance of the respective Gaussian distribution are directly parametrised by the free variational parameters, i.e.,

$$\mathbb{E}[\mathbf{u}] = \mathbf{m}, \tag{A22a}$$

$$\mathbb{V}[\mathbf{u}] = \mathbf{\Phi}. \tag{A22b}$$

Appendix H. Simulation Approximations

This section presents the simulation approaches used throughout this study and summarises their underlying posterior assumptions. In summary, Fully Correlated Monte Carlo simulation conditions on all latent values. Pseudo Independent Monte Carlo simulation conditions on posterior pseudo-point realisations, rendering the simulated values conditionally independent given these realisations. Conditionally Independent Monte Carlo simulation further marginalises over the posterior pseudo-point realisations and assumes conditional independence among the simulated latent values. Naive simulations provide a non-Monte Carlo alternative by propagating only the posterior means. Finally, thresholded correlated simulation conditions are applied to a subset of latent values, where retention is determined by the uncertainty associated with each posterior prediction.

Appendix H.1. Fully Correlated Monte Carlo Simulation

$$\begin{aligned} &q(\mathbf{f}_{t+1:t+n}) \\ &\quad = \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\ &\quad \cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\mathbf{f}_{t+1:t+i-1}, \mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right) d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u}. \end{aligned} \tag{A23}$$

Appendix H.2. Pseudo Independent Monte Carlo Simulation

$$\begin{aligned} &q(\mathbf{f}_{t+1:t+n}) \\ &\quad = \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\ &\quad \cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\cancel{\mathbf{f}_{t+1:t+i-1}}, \mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right) d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u} \\ &\quad = \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\ &\quad \cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right) d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u}. \end{aligned} \tag{A24}$$

Appendix H.3. Conditionally Independent Monte Carlo Simulation

$$\begin{aligned}
&q(\mathbf{f}_{t+1:t+n}) \\
&= \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\
&\cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\cancel{\mathbf{f}_{t+1:t+i-1}}, \mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u} \\
&= \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\
&\cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u} \\
&= \int q(\mathbf{f}_{t+2-n_a:t+1}) \\
&\cdot \left(\prod_{i=2}^{n}\int\int p(\mathbf{f}_{t+i}|\mathbf{u}, \mathbf{z}_{t+i})q(\mathbf{u})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}d\mathbf{u}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}} \\
&= \int q(\mathbf{f}_{t+2-n_a:t+1})\left(\prod_{i=2}^{n}\int q(\mathbf{f}_{t+i}|\mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}.
\end{aligned} \tag{A25}$$

Appendix H.4. Fully Correlated Naive Simulation

$$\begin{aligned}
&q(\mathbf{f}_{t+1:t+n}) = \\
&= \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\
&\cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\mathbf{f}_{t+1:t+i-1}, \mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\boldsymbol{\mu}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u}.
\end{aligned} \tag{A26}$$

where

$$\begin{aligned}
p(\mathbf{z}^T_{t+i}|\boldsymbol{\mu}_{t+i-n_a:t+i-1}) &= \delta\Big(\mathbf{z}^T_{t+i} - [\boldsymbol{\mu}^T_{t+i-n_a:t+i-1}, \mathbf{x}^T_{t+i-n_b:t+i-1}]\Big), \\
\boldsymbol{\mu}_{t+i-n_a:t+i-1} &= \mathbb{E}[p(\mathbf{f}_{t+i-n_a:t+i-1}|\mathbf{y}_{1:t})].
\end{aligned} \tag{A27}$$

Appendix H.5. Conditionally Independent Naive Simulation

$$\begin{aligned}
&q(\mathbf{f}_{t+1:t+n}) = \\
&= \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\
&\cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\cancel{\mathbf{f}_{t+1:t+i-1}}, \mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\boldsymbol{\mu}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u} \\
&= \int q(\mathbf{f}_{t+2-n_a:t+1})\left(\prod_{i=2}^{n}\int q(\mathbf{f}_{t+i}|\mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\boldsymbol{\mu}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}},
\end{aligned} \tag{A28}$$

where $p(\mathbf{z}_{t+i}|\boldsymbol{\mu}_{t+i-n_a:t+i-1})$ is defined by Equation (A27).

Appendix H.6. Thresholded Correlated Simulation

$$\begin{aligned}
&q(\mathbf{f}_{t+1:t+n}) \\
&= \int\int p(\mathbf{f}_{t+2-n_a:t+1}|\mathbf{u})q(\mathbf{u}) \\
&\cdot \left(\prod_{i=2}^{n}\int p(\mathbf{f}_{t+i}|\mathbf{f}_{\mathcal{F}}, \mathbf{u}, \mathbf{z}_{t+i})p(\mathbf{z}_{t+i}|\mathbf{f}_{t+i-n_a:t+i-1})d\mathbf{z}_{t+i}\right)d\mathbf{f}_{\setminus\{f_{t+1:t+n}\}}d\mathbf{u},
\end{aligned} \tag{A29}$$

where $\mathcal{F}$ denotes the set of retained latent function values based on the Algorithm A1.

Appendix I. Algorithms

This section presents the algorithms for the Monte Carlo simulation methods introduced in this publication. We assume that the pseudo-input locations $\mathbf{Z}'_{1:m}$ have been obtained during model training and treat them as fixed, as their estimation is not the focus of this work. The posterior distribution of the lagged latent values is available in closed form, and the initial realisations are obtained by sampling from this distribution.

Appendix I.1. Thresholded Correlated Monte Carlo Simulation

Algorithm A1: Algorithm for a single sample in the TCMC simulation of the GP-NARX models. The pseudo-point posteriors for the respective GP-NARX model are defined in Appendix G. PIMC simulation corresponds to the variance threshold $T_{\mathbb{V}} = \infty$. FCMC simulation corresponds to the variance threshold $T_{\mathbb{V}} = 0$. NP denotes noise propagation.

Data: $T_{\mathbb{V}}$: variance threshold
Data: $\mathbf{Z}'_{1:m}$: pseudo-input locations
Data: $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{\Phi})$: posterior at the pseudo-input locations $\mathbf{Z}'_{1:m}$ for the respective GP-NARX model
Result: $\tilde{\mathbf{f}}_{t+1:t+n}$: sample of the latent trajectory
Result: $\tilde{\mathbf{y}}_{t+1:t+n}$: sample of the noisy trajectory
$\tilde{\mathbf{f}}_{t+1-n_a:t} \sim q(\mathbf{f}_{t+1-n_a:t})$;
$\tilde{\mathbf{f}}_{t+1:t+n} \leftarrow [\ ]$;
$\tilde{\mathbf{y}}_{t+1:t+n} \leftarrow [\ ]$;
$i \leftarrow t+1$;
$\mathbf{K}_{f,f} \leftarrow \mathbf{K}_{u,u}$;
$\mathbf{L}\mathbf{L}^T \leftarrow \text{cholesky}(\mathbf{K}_{f,f})$;
$\tilde{\mathbf{u}} \sim q(\mathbf{u})$;
$\mathbf{Z} \leftarrow \mathbf{Z}'_{1:m}, \boldsymbol{\eta} \leftarrow \tilde{\mathbf{u}}, \mathcal{D} \leftarrow \{\mathbf{Z}, \boldsymbol{\eta}\}$;
while $i \leq t+n$ **do**
  if *NP* **then**
    $\tilde{\mathbf{z}}_i^T \leftarrow [\tilde{\mathbf{y}}_{i-n_a:i-1}^T, \mathbf{x}_{i-n_b:i-1}^T]$;
  else
    $\tilde{\mathbf{z}}_i^T \leftarrow [\tilde{\mathbf{f}}_{i-n_a:i-1}^T, \mathbf{x}_{i-n_b:i-1}^T]$;
  $\boldsymbol{\alpha} \leftarrow \mathbf{L} \backslash \mathbf{K}_{f_i,\eta}$;
  $\boldsymbol{\beta} \leftarrow \mathbf{L} \backslash \boldsymbol{\eta}$;
  $\mu \leftarrow \boldsymbol{\alpha}^T \boldsymbol{\beta}$;
  $\sigma^2 \leftarrow \mathbf{K}_{f_i,f_i} - \boldsymbol{\alpha}^T \boldsymbol{\alpha}$;
  $\tilde{f}_i \sim p(f_i|\mathcal{D}, \tilde{\mathbf{z}}_i) = \mathcal{N}(f_i|\mu, \sigma^2)$;
  $\tilde{y}_i \sim p(y_i|f_i)$;
  if $\sigma^2 \geq T_{\mathbb{V}}$ **then**
    $\mathbf{Z}^T \leftarrow [\mathbf{Z}^T, \tilde{\mathbf{z}}_i^T], \boldsymbol{\eta}^T \leftarrow [\boldsymbol{\eta}^T, \tilde{f}_i], \mathcal{D} \leftarrow \{\mathbf{Z}, \boldsymbol{\eta}\}$;
    $\mathbf{a} \leftarrow \mathbf{K}_{f,f_i}, b \leftarrow \mathbf{K}_{f_i,f_i}$;
    $\mathbf{K}_{f,f} \leftarrow \begin{bmatrix} \mathbf{K}_{f,f} & \mathbf{a} \\ \mathbf{a}^T & b \end{bmatrix}$;
    $\mathbf{L} \leftarrow$ update cholesky$(\mathbf{L}, \{\mathbf{a}, b\})$ (Algorithm A3);
  $\tilde{\mathbf{f}}_{t+1:t+n}^T \leftarrow [\tilde{\mathbf{f}}_{t+1:t+n}^T, \tilde{f}_i]$;
  $\tilde{\mathbf{y}}_{t+1:t+n}^T \leftarrow [\tilde{\mathbf{y}}_{t+1:t+n}^T, \tilde{y}_i]$;
  $i \leftarrow i+1$;

Appendix I.2. Pseudo Independent Monte Carlo Simulation

Algorithm A2: Algorithm for a single sample in the PIMC simulation of the GP-NARX models. NP denotes noise propagation.

Data: $\mathbf{Z}'_{1:m}$: pseudo-input locations
Data: $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{\Phi})$: posterior at the pseudo-input locations $\mathbf{Z}'_{1:m}$ for the respective GP-NARX model
Result: $\tilde{\mathbf{f}}_{t+1:t+n}$: sample of the latent trajectory
Result: $\tilde{\mathbf{y}}_{t+1:t+n}$: sample of the noisy trajectory
$\tilde{\mathbf{f}}_{t+2-n_a:t+1} \sim q(\mathbf{f}_{t+2-n_a:t+1})$;
$\tilde{y}_{t+1} \sim p(y_{t+1}|f_{t+1})$;
$\tilde{\mathbf{f}}_{t+1:t+n} \leftarrow [\tilde{f}_{t+1}]^T$;
$\tilde{\mathbf{y}}_{t+1:t+n} \leftarrow [\tilde{y}_{t+1}]^T$;
$i \leftarrow t+2$;
$\tilde{\mathbf{u}} \sim q(\mathbf{u})$;
$\mathcal{D} \leftarrow \{\mathbf{Z}'_{1:m}, \tilde{\mathbf{u}}\}$;
$\mathbf{L}_u \mathbf{L}_u^T \leftarrow \text{cholesky}(\mathbf{K}_{u,u})$;
$\mathbf{L}_u^{-1} \leftarrow \mathbf{L}_u \backslash \mathbf{I}$;
$\boldsymbol{\gamma} \leftarrow \mathbf{L}_u^{-T} \mathbf{L}_u^{-1}$;
$\boldsymbol{\delta} \leftarrow \boldsymbol{\gamma} \tilde{\mathbf{u}}$;
while $i \leq t+n$ **do**
  if *NP* **then**
    $\tilde{\mathbf{z}}_i^T \leftarrow [\tilde{\mathbf{y}}_{i-n_a:i-1}^T, \mathbf{x}_{i-n_b:i-1}^T]$;
  else
    $\tilde{\mathbf{z}}_i^T \leftarrow [\tilde{\mathbf{f}}_{i-n_a:i-1}^T, \mathbf{x}_{i-n_b:i-1}^T]$;
  $\mu \leftarrow \mathbf{K}_{f_i,u} \boldsymbol{\delta}$;
  $\sigma^2 \leftarrow \mathbf{K}_{f_i,f_i} - \mathbf{K}_{f_i,u} \boldsymbol{\gamma} \mathbf{K}_{u,f_i}$;
  $\tilde{f}_i \sim p(f_i|\mathcal{D}, \tilde{\mathbf{z}}_i) = \mathcal{N}(f_i|\mu, \sigma^2)$;
  $\tilde{y}_i \sim p(y_i|f_i)$;
  $\tilde{\mathbf{f}}_{t+1:t+n}^T \leftarrow [\tilde{\mathbf{f}}_{t+1:t+n}^T, \tilde{f}_i]$;
  $\tilde{\mathbf{y}}_{t+1:t+n}^T \leftarrow [\tilde{\mathbf{y}}_{t+1:t+n}^T, \tilde{y}_i]$;
  $i \leftarrow i+1$;

Appendix I.3. Cholesky Update

Algorithm A3: Algorithm for the iterative update of the Cholesky factor. Updates the Cholesky factor based on a row/column addition to the covariance matrix.

Data: $\mathbf{L} \in \mathbb{R}^{m \times m}$: cached Cholesky factor
Data: $\{\mathbf{a}, b\}$: column addition to the covariance matrix, **a** represents the cross-covariance between the new point and the existing points, and b the covariance of the added point.
Result: $\hat{L} \in \mathbb{R}^{(m+1)\times(m+1)}$: updated Cholesky factor
$\boldsymbol{\alpha} \leftarrow \mathbf{L} \backslash \mathbf{a}$;
$\beta \leftarrow \sqrt{b - \boldsymbol{\alpha}^T \boldsymbol{\alpha}}$;
$\hat{\mathbf{L}} \leftarrow \begin{bmatrix} \mathbf{L} & \mathbf{0} \\ \boldsymbol{\alpha}^T & \beta \end{bmatrix}$;

References

1. Gregorčič, G.; Lightbody, G. Gaussian process approach for modelling of nonlinear systems. *Eng. Appl. Artif. Intell.* **2009**, *22*, 522–533. [CrossRef]
2. Capone, A.; Lederer, A.; Hirche, S. Gaussian process uniform error bounds with unknown hyperparameters for safety-critical applications. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 2609–2624.
3. Yi, Y.; Verbič, G. Operating Envelopes under Probabilistic Electricity Demand and Solar Generation Forecasts. *arXiv* **2022**, arXiv:2207.09818. [CrossRef]
4. Krige, D.G. A statistical approach to some basic mine valuation problems on the Witwatersrand. *J. South. Afr. Inst. Min. Metall.* **1951**, *52*, 119–139.
5. Rasmussen, C.E.; Williams, C.K. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006. [CrossRef]
6. Kocijan, J. *Modelling and Control of Dynamic Systems Using Gaussian Process Models*; Springer: Berlin/Heidelberg, Germany, 2016. [CrossRef]
7. Gibbs, M.N. Bayesian Gaussian Processes for Regression and Classification. Ph.D. Thesis, University of Cambridge, Cambridge, UK, 1998.
8. MacKay, D.J. *Information Theory, Inference and Learning Algorithms*; Cambridge University Press: Cambridge, UK, 2003.
9. Liu, H.; Ong, Y.S.; Shen, X.; Cai, J. When Gaussian process meets big data: A review of scalable GPs. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 4405–4423. [CrossRef]
10. Quinonero-Candela, J.; Rasmussen, C.E. A unifying view of sparse approximate Gaussian process regression. *J. Mach. Learn. Res.* **2005**, *6*, 1939–1959.
11. Bauer, M.; van der Wilk, M.; Rasmussen, C.E. Understanding probabilistic sparse Gaussian process approximations. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 1533–1541. [CrossRef]
12. Seeger, M.W.; Williams, C.K.; Lawrence, N.D. Fast forward selection to speed up sparse Gaussian process regression. In Proceedings of the International Workshop on Artificial Intelligence and Statistics, PMLR, Key West, FL, USA, 3–6 January 2003; pp. 254–261.
13. Snelson, E.; Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. *Adv. Neural Inf. Process. Syst.* **2005**, *18*, 1257–1264.
14. Hensman, J.; Fusi, N.; Lawrence, N.D. Gaussian processes for big data. *arXiv* **2013**, arXiv:1309.6835. [CrossRef]
15. Hensman, J.; Matthews, A.; Ghahramani, Z. Scalable variational Gaussian process classification. In Proceedings of the Artificial Intelligence and Statistics, PMLR, San Diego, CA, USA, 9–12 May 2015; pp. 351–360.
16. Sun, S.; Shi, J.; Wilson, A.G.; Grosse, R. Scalable Variational Gaussian Processes via Harmonic Kernel Decomposition. *arXiv* **2021**, arXiv:2106.05992. [CrossRef]
17. Titsias, M. Variational learning of inducing variables in sparse Gaussian processes. In Proceedings of the Artificial Intelligence and Statistics, PMLR, Clearwater Beach, FL, USA, 16–18 April 2009; pp. 567–574.
18. Hamming, R.W. *Digital Filters*; Courier Corporation: Chelmsford, MA, USA, 1998.
19. Nelles, O. Nonlinear dynamic system identification. In *Nonlinear System Identification*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 547–577. [CrossRef]
20. Deisenroth, M.P.; Turner, R.D.; Huber, M.F.; Hanebeck, U.D.; Rasmussen, C.E. Robust filtering and smoothing with Gaussian processes. *IEEE Trans. Autom. Control* **2011**, *57*, 1865–1871. [CrossRef]
21. Frigola, R.; Lindsten, F.; Schön, T.B.; Rasmussen, C.E. Bayesian inference and learning in Gaussian process state-space models with particle MCMC. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 3156–3164.
22. Frigola, R.; Chen, Y.; Rasmussen, C.E. Variational Gaussian process state-space models. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 3680–3688.
23. Frigola, R. Bayesian Time Series Learning with Gaussian Processes. Ph.D. Thesis, University of Cambridge, Cambridge, UK, 2015.
24. Kocijan, J.; Petelin, D. Output-Error Model Training for Gaussian Process Models. In *Proceedings of the Adaptive and Natural Computing Algorithms*; Dobnikar, A., Lotrič, U., Šter, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 312–321. [CrossRef]
25. Hachino, T.; Kadirkamanathan, V. Multiple Gaussian process models for direct time series forecasting. *IEEJ Trans. Electr. Electron. Eng.* **2011**, *6*, 245–252. [CrossRef]
26. Quinonero-Candela, J.; Girard, A.; Rasmussen, C.E. *Prediction at an Uncertain Input for Gaussian Processes and Relevance Vector Machines-Application to Multiple-Step Ahead Time-Series Forecasting*; Technical report; Informatics and Mathematical Modelling, Technical University of Denmark, DTU: Copenhagen, Denmark, 2003.
27. Candela, J.Q.; Girard, A.; Larsen, J.; Rasmussen, C.E. Propagation of uncertainty in Bayesian kernel models-application to multiple-step ahead forecasting. In *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing*; Proceedings (ICASSP'03); IEEE: Piscataway, NJ, USA, 2003; Volume 2, p. II -701. [CrossRef]
28. Deisenroth, M.P.; Fox, D.; Rasmussen, C.E. Gaussian processes for data-efficient learning in robotics and control. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *37*, 408–423. [CrossRef]

29. Girard, A.; Rasmussen, C.; Candela, J.Q.; Murray-Smith, R. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. *Adv. Neural Inf. Process. Syst.* **2002**, *15*, 545–552.
30. Groot, P.; Lucas, P.; Bosch, P. Multiple-step time series forecasting with sparse Gaussian processes. In Proceedings of the 23rd Benelux Conference on Artificial Intelligence, BNAIC, Ghent, Belgium, 3–4 November 2011.
31. Girard, A.; Murray-Smith, R. Gaussian processes: Prediction at a noisy input and application to iterative multiple-step ahead forecasting of time-series. In *Switching and Learning in Feedback Systems*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 158–184. [CrossRef]
32. McHutchon, A.; Rasmussen, C. Gaussian process training with input noise. *Adv. Neural Inf. Process. Syst.* **2011**, *24*, 1341–1349 .
33. Bijl, H.; Schön, T.B.; van Wingerden, J.W.; Verhaegen, M. System identification through online sparse Gaussian process regression with input noise. *IFAC J. Syst. Control* **2017**, *2*, 1–11. [CrossRef]
34. Deisenroth, M.P. *Efficient Reinforcement Learning Using Gaussian Processes*; KIT Scientific Publishing: Karlsruhe, Germany, 2010; Volume 9. [CrossRef]
35. Wang, Y.; Chaib-Draa, B. A marginalized particle Gaussian process regression. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1187–1195.
36. Worden, K.; Becker, W.E.; Rogers, T.; Cross, E. On the confidence bounds of Gaussian process NARX models and their higher-order frequency response functions. *Mech. Syst. Signal Process.* **2018**, *104*, 188–223. [CrossRef]
37. Beckers, T.; Hirche, S. Prediction with Approximated Gaussian Process Dynamical Models. *IEEE Trans. Autom. Control* **2021**, *67*, 6460–6473. [CrossRef]
38. Krivec, T.; Papa, G.; Kocijan, J. Simulation of variational Gaussian process NARX models with GPGPU. *ISA Trans.* **2021**, *109*, 141–151. [CrossRef] [PubMed]
39. Krivec, T. Simulation of Approximated Gaussian Process Autoregressive Models: Doctoral Dissertation. Ph.D. Thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, 2023.
40. Wilson, J.; Borovitskiy, V.; Terenin, A.; Mostowsky, P.; Deisenroth, M. Efficiently sampling functions from Gaussian process posteriors. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020; pp. 10292–10302.
41. Titsias, M.K. *Variational Model Selection for Sparse Gaussian Process Regression*; Report; University of Manchester: Manchester, UK, 2009.
42. Girard, A. Approximate Methods for Propagation of Uncertainty with Gaussian Process Models. Ph.D. Thesis, University of Glasgow, Glasgow, UK, 2004.
43. Kocijan, J.; Perne, M.; Grašic, B.; Božnar, M.Z.; Mlakar, P. Sparse and hybrid modelling of relative humidity: The Krško basin case study. *CAAI Trans. Intell. Technol.* **2020**, *5*, 42–48. [CrossRef]
44. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. *arXiv* **2015**, arXiv:1603.04467. [CrossRef]
45. Matthews, A.G.d.G.; van der Wilk, M.; Nickson, T.; Fujii, K.; Boukouvalas, A.; León-Villagrá, P.; Ghahramani, Z.; Hensman, J. GPflow: A Gaussian process library using TensorFlow. *J. Mach. Learn. Res.* **2017**, *18*, 1–6. [CrossRef]
46. Wikipedia. Wasserstein Metric—Wikipedia, The Free Encyclopedia. 2022. Available online: http://en.wikipedia.org/w/index.php?title=Wasserstein%20metric&oldid=1089055041 (accessed on 3 June 2022).
47. Wigren, T.; Schoukens, J. Three free data sets for development and benchmarking in nonlinear system identification. In *Proceedings of the 2013 European Control Conference (ECC)*; IEEE: Piscataway, NJ, USA, 2013; pp. 2933–2938.
48. Ljung, L.; Zhang, Q.; Lindskog, P.; Juditski, A. Estimation of grey box and black box models for non-linear circuit data. *IFAC Proc. Vol.* **2004**, *37*, 399–404. [CrossRef]
49. Byrd, R.H.; Lu, P.; Nocedal, J.; Zhu, C. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.* **1995**, *16*, 1190–1208. [CrossRef]
50. Ali, M.; Jardali, H.; Roy, N.; Liu, L. Autonomous navigation, mapping and exploration with Gaussian processes. In Proceedings of the Robotics: Science and Systems (RSS), Virtual, 10–14 July 2023.
51. Li, E.; Razani, R.; Xu, Y.; Liu, B. CPSeg: Cluster-free Panoptic Segmentation of 3D LiDAR Point Clouds. In *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: Piscataway, NJ, USA, 2023; pp. 8239–8245. [CrossRef]