Hostettler, Roland; Särkkä, Simo; Godsill, Simon J.

**Rao-Blackwellized Particle MCMC for Parameter Estimation in Spatio-Temporal Gaussian Processes**

*Published in:*

*DOI:*
10.1109/MLSP.2017.8168171

Published: 01/01/2017

*Document Version*
Peer reviewed version

*Please cite the original version:*

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.
Rao–Blackwellized Particle MCMC for Parameter Estimation in Spatio-Temporal Gaussian Processes

Roland Hostettler, Simo Särkkä, and Simon J. Godsill

This is a post-print of a paper published in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. When citing this work, you must always cite the original article:


DOI:
10.1109/MLSP.2017.8168171

Copyright:
Copyright 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
RAO–BLACKWELLIZED PARTICLE MCMC FOR PARAMETER ESTIMATION IN SPATIO-TEMPORAL GAUSSIAN PROCESSES

Roland Hostettler*, Simo Särkkä†, and Simon J. Godsill†

*Department of Electrical Engineering and Automation, Aalto University, Espoo, Finland
E-Mail: {roland.hostettler, simo.sarkka}@aalto.fi
†Department of Engineering, University of Cambridge, Cambridge, UK
E-Mail: sjg@eng.cam.ac.uk

ABSTRACT

In this paper, we consider parameter estimation in latent, spatio-temporal Gaussian processes using particle Markov chain Monte Carlo methods. In particular, we use spectral decomposition of the covariance function to obtain a high-dimensional state-space representation of the Gaussian processes, which is assumed to be observed through a nonlinear non-Gaussian likelihood. We develop a Rao–Blackwellized particle Gibbs sampler to sample the state trajectory and show how to sample the hyperparameters and possible parameters in the likelihood. The proposed method is evaluated on a spatio-temporal population model and the predictive performance is evaluated using leave-one-out cross-validation.

Index Terms— Gaussian processes, statistical learning, Monte Carlo methods, parameter estimation

1. INTRODUCTION

Gaussian processes (GP) are a versatile Bayesian non-parametric modeling approach [1]. They have found widespread applications, for example in time series modeling in finance [2], meteorology [3], medical applications [4], and target tracking [5], to name a few. One well-known disadvantage is that the batch formulation in GP regression scales cubically with the number of training points. However, it has recently been shown that stationary, temporal and spatio-temporal GPs can be transformed into equivalent (infinite dimensional) linear state-space systems by decomposing the GP’s spectral density. This can subsequently be used together with Kalman filtering and Rauch–Tung–Striebel smoothing [6–8], which greatly alleviates the computational burden.

In this work, we consider learning (i.e., estimation of the model parameters) of models where a latent, spatio-temporal process, observed indirectly through some proxy variable, is modeled using a GP. In particular, we consider models of the form

\[
\begin{align}
 f(x, t) &\sim GP(m(x, t; \theta_f), k(x, t, x', t'; \theta_f)), \\
y(t) &\sim p(y(t) \mid f(x, t), \theta_y),
\end{align}
\]

where \(m(x, t)\) is the GP’s mean function (without loss of generality assumed to be zero for the remainder of this paper) and \(k(x, t, x', t') = k(\Delta x, \Delta t)\) with \(\Delta t = t - t'\) and \(\Delta x = x - x'\) its stationary covariance function, both parametrized by the hyperparameters \(\theta_f\). Furthermore, \(p(y(t) \mid f(x, t), \theta_y)\) is the measurement likelihood, which is parametrized by parameters \(\theta_y\), and \(t\) and \(x\) denote the temporal and spatial variables, respectively.

The objective is then to infer the function values \(f_n \equiv f(x, t_n)\) at times \(t_n\) (for \(n = 1, \ldots, N\)) as well as the parameters \(\theta_f\) and \(\theta_y\) based on a set of measurements \(y_{1:N} = \{y_1, \ldots, y_N\}\). The most common approach to do this in batch settings is by maximizing the marginal likelihood of the data \(p(y_{1:N} \mid \theta_f, \theta_y)\) using, for example, gradient-based methods, or Markov chain Monte Carlo (MCMC) [11]. These approaches can also readily be employed when using the equivalent state-space formulations and Kalman filtering [7]. When facing non-Gaussian likelihoods \(p(y_n \mid f_n, \theta_y)\), one commonly has to resort to approximations such as expectation propagation or Laplace approximations [9–11]. These may overcome some of the difficulties involved with nonlinear non-Gaussian likelihoods, but introduce approximations to the posterior and may still be unfeasible for a large number of data points and require further approximations such as inducing points [12].

In this paper, we propose to use a fully Bayesian approach based on particle MCMC methods (particle Gibbs with ancestor sampling, PGAS [13, 14]), which can readily handle arbitrary likelihoods \(p(y_n \mid f_n, \theta_y)\), together with the state-space representation of spatio-temporal GPs. This has the advantage of not approximating the posterior by an (implicitly) assumed density but rather approximating the true posterior using samples from it, while retaining the beneficial properties of the state-space formulation. Since the conversion procedure can yield high-dimensional state-space systems, Rao–Blackwellization of the conditionally linear substructure is used to alleviate the computational burden and increase scalability. The proposed method is conceptually similar to the methods proposed in [15] and [16]; however, it solves a different problem. The latter methods aim at estimating the parameters in state-space systems where the dynamic model and observation model are modeled using GPs, while the method proposed in this work aims at inferring the parameters in GPs on state-space form, observed through a known likelihood (up to some parametrization \(\theta_y\)).

The remainder of this paper is organized as follows. Section 2 briefly reviews the procedure for converting spatio-temporal Gaussian processes to state-space models. The proposed method is introduced in Section 3, followed by numerical illustrations in Section 4. Some concluding remarks follow in Section 5.

2. CONVERSION OF GAUSSIAN PROCESSES

In this section, we briefly review the procedure for converting stationary GPs to state-space models. For a more detailed treatment, the reader is referred to [6–8].

The main idea of the conversion approach is to decompose the
spectral density of the stationary Gaussian process which is given by the Fourier transform of the covariance function

\[ k(\Delta x, \tau) \leftrightarrow S_f(\omega_x, \omega_t) \]

into a white noise process with spectral density \( S_w \) and a linear system with frequency response \( H(\omega_x, \omega_t) \) such that [17]

\[ S_f(\omega_x, \omega_t) = S_w H(\omega_x, \omega_t) H(\omega_x, \omega_t)^*, \tag{2} \]

where the superscript * denotes the complex conjugate. From the transfer function \( H(\omega_x, \omega_t) \) of the linear system, the corresponding infinite dimensional state-space representation can then readily be found using well known conversion techniques to obtain [7]

\[
\begin{align*}
\dot{x}(x, t) &= Ax(x, t) + Bu(x, t), \tag{3a} \\
f(x, t) &= Cx(x, t), \tag{3b} \\
x(x, 0) &\sim \mathcal{N}(0, P_0), \tag{3c}
\end{align*}
\]

where \( w(x, t) \) is the white random process with spectral density \( S_w \) as given by the decomposition in (2). Furthermore, \( A, B, \) and \( C \) are, in general, a matrix and vectors of linear operators, respectively, and are given through \( H(\omega_x, \omega_t) \). The covariance \( P_0 \) of the initial state \( x(x, 0) \) is given by the solution of the following continuous time Lyapunov equation

\[ AP_0 + P_0A^T + BS_wB^T = 0. \tag{4} \]

Finally, discretizing (3) with respect to time yields

\[
\begin{align*}
x_n &= F_n x_{n-1} + q_n, \\
f_n &= C x_n, \\
x_0 &\sim \mathcal{N}(0, P_0),
\end{align*}
\]

where \( F_n = \exp(A \Delta t) \) is the linear operator exponential with \( \Delta t = t_n - t_{n-1} \), and \( q_n \sim \mathcal{N}(0, Q_n) \) is white, Gaussian noise with covariance operator

\[
Q_n = \int_0^{\Delta t} \exp(A(\Delta t - \tau)) B S_w B^T \exp(A(\Delta t - \tau))^T d\tau. \tag{5c}
\]

As an example, consider a covariance function composed of the product of two Matérn kernels \( k_M(\cdot) \)

\[
k(\Delta x, \tau) = k_M(\Delta x)k_M(\tau), \tag{6}
\]

with

\[
k_M(r) = \frac{2^{\nu-1} - 1}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu} |r|}{l} \right)^\nu K_\nu \left( \frac{\sqrt{2\nu} |r|}{l} \right), \tag{7}
\]

where \( \sigma^2, l, \) and \( \nu \) are the kernel’s hyperparameters, and \( K_\nu(\cdot) \) is the modified Bessel function of the second kind of order \( \nu \).

Using the Fourier transform with respect to \( \tau \) on (7)-(8) yields

\[
S(\Delta x, \omega_t) = k_M(\Delta x) \frac{2^\nu \sqrt{\pi} \lambda_t^{\nu+1/2} \Gamma(\nu + 1/2)}{\Gamma(\nu)} \times (\lambda_t^2 + \omega^2)^{-\nu+1/2} \tag{8}
\]

Finally, if the decomposition (2) can not be done exactly, it can be approximated by, for example, using rational approximations of \( H(\omega_x, \omega_t) \). Examples for when this is necessary include when \( \nu_t - 1/2 \) in the Matérn covariance is not a positive integer or the popular squared exponential covariance function [6, 18].

### 3. STATE AND PARAMETER ESTIMATION

In this section, we develop the proposed approach for estimating the posterior distribution of the parameters and the state, \( p(\theta | y_{1:N}, x_{0:N}) \). The approach is based on Gibbs sampling and particle MCMC methods (see [13, 19]) where the state trajectory \( x_{0:N} \), GP hyperparameters \( \theta_f \), and likelihood parameters \( \theta_y \) (if any) are sampled in turn, each conditioned on the other parameters and the data \( y_{1:N} \). This approach has the advantage of being able to accommodate a broad class of possibly nonlinear non-Gaussian models and thus can overcome some of the limitations of approaches maximizing the marginal likelihood \( p(y_{1:N} | \theta_f, \theta_y) \), at the expense of being computationally more demanding compared to Kalman filtering-based approaches. The basic sampler is shown in Algorithm 1 and each of its components is discussed below.

#### 3.1. State Sampling

Sampling the states is achieved by using the particle Gibbs with ancestor sampling (PGAS) kernel introduced in [13]. The basic idea in
PGAS is to construct a particle filter that approximates the smooth-
ingen density $p(x_{0:N} | y_{1:N}, \theta_f, \theta_y)$ (for brevity, we will drop the con-
ditioning on $\theta_f$ and $\theta_y$ throughout the remainder of this subsection) by generating a set of weighted trajectories $\{w^m_N, z^m_{0:N}\}_{m=1}^M$ from which a new trajectory is sampled with probability $w^m_N$. By introdu-
cing an a priori known, deterministic seed trajectory $\tilde{x}_{0:N}$, it can be 
shown that the resulting algorithm indeed is an invariant sampler over 
$p(x_{0:N} | y_{1:N}, \theta_f, \theta_y)$, see [13] or [19] for details.

As noted in Section 2 the conversion of the GP to state-space 
form generally produces an infinite dimensional system. In prac-
tice, this has to be approximated by using a discrete set of training 
and prediction points in the spatial domain $x$. This turns the system 
from an infinite dimensional into a high-dimensional system instead. 
Thus, the dimensionality becomes challenging when sampling the 
states using sequential Monte Carlo methods.

Fortunately, the resulting system (5) exhibits a large simple linear-
structure, which can be exploited. In particular, the states can be 
divided into nonlinear states $s_n$ (the GP $f_n$, at the training points) and 
linear states $z_n$ (the GP’s derivatives $\partial f_n/\partial t$ at the training 
points as well as the prediction points). Thus, the dynamic model 
can be written as

$$p(s_n, z_n | s_{n-1}, z_{n-1}) = \mathcal{N}\left( \begin{array}{c} s_n \\ \tilde{z}_{n-1} \\ \hat{z}_{n-1}^{\top} \end{array} ; \begin{array}{c} g_{n-1} \\ Q_n^{s} \\ Q_n^{z} \end{array} , \begin{array}{c} Q_n^{s} \\ Q_n^{z} \end{array} \right) \right),$$

where $g_{n-1} \triangleq g(s_{n-1})$ and $h_{n-1} \triangleq h(s_{n-1})$. Then, Rao– 
Blackwellization can be used to reduce the dimensionality of the 
state that has to be targeted using Monte Carlo sampling as fol-

Starting from the joint filtering density, we obtain

$$p(z_n, s_n | y_{1:n}) = p(z_n | s_n, y_{1:n})p(s_n | y_{1:n}),$$

(12)

where $p(z_n | s_n, y_{1:n})$ is analytically tractable using a Kalman fil-
ter and $p(s_n | y_{1:n})$ will be the target density for sequential Monte 
Carlo sampling. This yields

$$p(z_n, s_{0:n} | y_{1:n}) \approx \sum_{m=1}^M w^m_N \mathcal{N}(z_n; \tilde{z}_n^m, P_n^e) \delta(s_{0:n} - s_{0:n}^m),$$

(13)

where $\delta(\cdot)$ denotes Dirac’s delta function, $w^m_N$ is the $m$th marginal 
trajectory, $s_{0:n}^m$ the corresponding weight, $\tilde{z}_n^m$ the conditional mean 
of the linear states, and $P_n^e$ their covariance. Note that since neither 
$Q_n$ nor $G$ or $H$ depend on $s_{n-1}$, the covariance $P_n^e$ does not depend 
on the state trajectory $s_{0:n}$ either.

3.1. Linear States

Assume that at time $t_{n-1}$, the density $p(z_{n-1} | s_{0:n-1}, y_{1:n-1})$ is 
given by

$$p(z_{n-1} | s_{0:n-1}, y_{1:n-1}) = \mathcal{N}(\hat{z}_{n-1}; \hat{z}_{n-1}, P_{n-1}^e)$$

(14)

and that $s_n$ has been sampled. Then, the update for $z_n$ reduces to a 
Kalman filter prediction

$$p(z_n | s_n, y_{1:n}) \propto p(y_n | s_n, z_n)p(z_n | s_n, y_{1:n-1})$$

$$\propto p(z_n | s_n, y_{1:n-1})$$

(15)

since the likelihood is independent of $z_n$. Considering the joint den-
sity of $z_n$ and $s_n$ yields

$$p(z_n, s_n | s_{0:n-1}, y_{1:n-1})$$

(16)

$$= \int p(z_n, s_n | z_{n-1}, s_{n-1})p(z_{n-1} | s_{0:n-1}, y_{1:n-1}) dz_{n-1}$$

$$= \mathcal{N}(\hat{z}_{n-1}; \hat{z}_{n-1}, Q_n + G^T H^T P_{n-1}^e H G + G^T P_{n-1}^e G^T, S)$$

Thus, by conditioning on $s_n$, it follows that

$$p(z_n | s_n, y_{1:n}) = p(z_n | s_n, y_{1:n-1})$$

(17)

with

$$\hat{z}_n = h_{n-1} + H \hat{z}_{n-1} + K(s_n - g_{n-1} - G \hat{z}_{n-1}),$$

(18a)

$$P_n^e = Q_n^s + H P_{n-1}^e H^T - K S K^T,$$

(18b)

$$S = Q_n^s + G P_{n-1}^e G^T,$$

(18c)

$$K = (Q_n^s + H P_{n-1}^e G^T)S^{-1}.$$
the marginalization. However, they can be calculated recursively as follows. First, note that

\[
p(s_j | s_{n-1}, y_{1:n-1})
\]

\[
= \int p(s_j | z_j, \bar{s}_j) \times p(z_j | s_{n-1}, \bar{s}_j, y_{1:n-1}) dz_j - 1.
\]

The second term of the integrand in (22) can be found from the joint density

\[
p(z_j, \bar{s}_j | s_{n-1}, y_{1:n-1}) = \int p(z_j, \bar{s}_j | z_{j-2}, \bar{s}_{j-2}) \times p(z_{j-2} | s_{n-2}, \bar{s}_{n-2}, y_{1:n-1}) dz_{j-2},
\]

where \(p(z_{j-2} | s_{n-2}, \bar{s}_{n-2}, y_{1:n-1})\) is the same density at \(j - 2\).

Thus, assuming that

\[
p(z_j, \bar{s}_j | s_{n-1}, y_{1:n-1}) = N(z_j; \bar{z}_j, P_{j-2}),
\]

it follows that \(p(z_{j-1}, \bar{s}_{j-1} | s_{n-2}, \bar{s}_{n-2}, y_{1:n-1})\) is as in (16) but with \(\bar{z}_j - 1\) and \(P_{j-2}\). Conditioning on \(s_{j-1}\) then yields the same update as in (17)-(18). Finally, this yields

\[
p(s_j | s_{n-1}, y_{1:n-1}) = N(s_j; g_{j-1} + G\hat{z}_{j-1}, Q_j + GP_{j-1}G^T),
\]

and the recursion is initialized with \(\hat{z}_{n-1} = \hat{a}_{n-1}, P_{n-1} = P_{a_{n-1}}\).

In practice, evaluating (21) for the whole future time horizon from \(n\) to \(N\) and for each time step and particle is impractical, due to the computational complexity. Hence, we propose to make use of the approximation suggested in [13] based on the fact that the state \(s_{n+N}\) for \(n+N > n\) only weakly correlates with \(s_{n-1}\). Hence, we can truncate (22) at some point \(n+N \leq N\) without introducing significant bias.

Finally, once a new trajectory \(s_{0:N}\) has been sampled, smoothing the linear states is achieved by using a Rauch–Tung–Striebel smoothing pass [22]. This results in the method summarized in Algorithm 2, where \(\pi(a_{n} | s_{n-1}, y_{1:n})\) and \(\pi(s_{n} | s_{0:n-1}, y_{1:n})\) denote the proposal densities for the ancestor weights and nonlinear states, respectively.

**Algorithm 2** Rao–Blackwellized PGAS

**Input:** Trajectory \(s_{0:M}\)

**Output:** Trajectory \(s_{0:N}\)

1: Sample \(s_{0:M}^m \sim N(0, P_0^m)\) for \(m = 1, \ldots, M - 1\) and set \(s_0^M = \bar{s}_0\)

2: Set \(\bar{s}_0^M = P_0^m(P_0^m)^{-1}s_0^m, P_0 = P_0^m - P_0^m(P_0^m)^{-1}P_0^m,\) and \(w_0^M = 1/M\) for \(m = 1, \ldots, M\)

3: for \(n = 1, \ldots, N\) do

4: Sample \(a_n^m \sim \pi(a_n | s_{0:n-1}, y_{1:n})\) for \(m = 1, \ldots, M - 1\)

5: Sample \(s_n^m \sim \pi(s_n | s_{0:n-1}, y_{1:n})\) for \(m = 1, \ldots, M - 1\)

6: Set \(s_n^M = \bar{s}_n^M\)

7: Calculate the ancestor weights \(\bar{w}_n^M\) using (22)-(25)

8: Sample \(a_n^M \sim C(\{w_i^M\}_{i=1}^M)\)

9: Calculate \(s_n^M\) and \(P_n^m\) according to (18)

10: Set \(s_{0:n} = \{s_{0:n-1}, s_n^M\}\)

11: Calculate the particle weights \(w_n^m \| w_n^m = p(y_n | s_n^m)p(s_n^m | s_{0:n-1}, y_{1:n})/\pi(s_n^m | s_{0:n-1}, y_{1:n})\)

12: end for

13: Sample \(s_{0:N}^M \sim C(\{w_i^M\}_{i=1}^M)\)

14: Calculate \(p(z_{0:N}^m | s_{0:N}, y_{1:N})\) using a Rauch–Tung–Striebel smoother

Depending on how the hyperparameters \(\theta_f\) enter (27), it may be possible to sample all or a subset of \(\theta_f\) from \(p(\theta_f | x_{0:N})\) directly (with an appropriate prior). If this is not possible, we note that (27) can readily be evaluated numerically (up to proportionality). Thus, a Metropolis-within-Gibbs approach can be used in this case to sample \(\theta_f\) (or a subset thereof).

**3.3. Likelihood Parameter Sampling**

Finally, the third step is to sample the likelihood’s parameters \(\theta_y\). Similar to the previous section, if \(p(\theta_y | y_{1:N}, x_{0:N}, \theta_f)\) can be written as

\[
p(\theta_y | y_{1:N}, x_{0:N}, \theta_f) = p(\theta_y | y_{1:N}, x_{0:N}) \times p(y_n | x_{0:N}, \theta_y)p(\theta_y)
\]

where the first equality is due to the independence of \(\theta_y\) on \(\theta_f\) given \(x_{0:N}\).

Depending on the likelihood \(p(y_n | x_n, \theta_y)\), it is often possible to sample from (29) directly. For example, when the likelihood consists of additive Gaussian noise and the noise variance is unknown, the conjugate prior can be used for \(\theta_y\) to obtain a closed form solution for the posterior, from which we can draw samples. If this is not possible, we can still resort to a Metropolis-within-Gibbs approach as discussed in the previous section.

**4. NUMERICAL ILLUSTRATIONS**

To illustrate the proposed method, we consider a spatio-temporal population regression problem.
4.1. Setup

We model the logarithm of a time-varying population \( r_n \) in an area as a Gaussian process, that is, \( f_n = \log(r_n) \). As for the covariance function, we use the covariance function \( k(\Delta x, \tau) = k_M(\Delta x)k_M(\tau) \) discussed in Section 2 and assume known orders \( \nu_0 = 3.5 \) and \( \nu_0 = 4.5 \). The unknown GP parameters are thus the length scales \( l_s \) and \( l_t \), and the variance \( \sigma^2 \).

The population size measurements for the \( j \)th location \( x_j \) (\( j = 1, \ldots, J \)) at time \( t_n \) are assumed to be Poisson distributed according to the likelihood

\[
y_{j,n} \sim \mathcal{P}(\exp(f_{j,n})),
\]

where \( \mathcal{P}(\cdot) \) denotes the Poisson distribution and \( f_{j,n} \triangleq f_n(x_j) \) is the population’s logarithm at the \( j \)th location \( x_j \).

The simulation data at each location is generated from a Ricker model [23] given by

\[
r_{j,n} = r_{j,n-1} \exp \left( \rho \left( 1 - \frac{r_{j,n-1}}{\kappa} + e_{j,n} \right) \right),
\]

where \( e_{j,n} \) is the process noise with \( \text{Cov}(e_{i,n}, e_{j,n}) = \sigma_{ij}^2 \), \( \rho = 0.1 \) is the intrinsic growth rate, and \( \kappa = 100 \) is the environmental carrying capacity.

In total, \( N = 1000 \) time samples in an area of 40 km by 20 km at 8 locations (separated by 10 km) are simulated. The performance of the proposed algorithm is evaluated by leave-one-out cross-validation and calculating the time-averaged root mean squared error for the prediction at the test location. This yields a total of 7000 training points and 1000 test points.

In total, 200 MCMC samples are drawn, of which 100 are discarded as burn-in. For the particle filter, \( M = 2000 \) particles are used. The ancestor indices are sampled according to the particle weight at time \( n = 1 \) (i.e. \( a_n \sim \mathcal{C}(w_{n-1}^{-1})_{M=1} \)) while the bootstrap proposal \( p(s_n | s_{n-1}, y_{1:n-1}) \) (20) is used for \( s_n \). Finally, \( N = 1 \) was chosen (without noticeable performance degradation).

4.2. Parameter Sampling

In this scenario, the unknown parameters are \( \theta_f = [l_s, l_t, \sigma^2]^\top \) with no unknowns in the likelihood. First, note that the Lyapunov equation (4) can be solved analytically in this case and \( P_0 \) can be written as

\[ P_0 = \sigma^2 \tilde{P}_0 \]

where \( \tilde{P}_0 \) is independent of \( \sigma^2 \). Similarly, from (6) it can be seen that \( Q_n \) can be written as

\[ Q_n = \sigma^2 \tilde{Q}_n. \]

Thus, it follows from (27)–(28) that

\[
p(\sigma^2 | l_t, l_s, x_{0:N}) \propto p(\sigma^2)\mathcal{N}(x_0; 0, \sigma^2 \tilde{P}_0) \prod_{n=1}^N \mathcal{N}(x_n; F_n x_{n-1}, \sigma^2 \tilde{Q}_n). \]

Using the inverse Gamma prior \( p(\sigma^2) = IG(\sigma^2; \alpha, \beta) \) with prior parameters \( \alpha \) and \( \beta \) yields the posterior

\[ p(\sigma^2 | l_t, l_s, x_{0:N}) = IG(\sigma^2; \tilde{\alpha}, \tilde{\beta}) \]

with posterior parameters

\[ \tilde{\alpha} = \alpha + \frac{N N_y}{2}, \]
\[ \tilde{\beta} = \beta + \frac{1}{2}||x_0||_2^2 + \frac{1}{2} \sum_{n=1}^N ||x_n - F_n x_{n-1}||_2^2, \]

with \( \tilde{\beta} \) as defined above.

Fig. 1. Prediction example with true trajectory (---) simulated from a Ricker model, together with the MMSE estimate (----), and the sampled trajectories (----) for the time period \( n = 100, \ldots, 200 \).

where \( ||x||_2^2 = x^T C x \) and \( N_x \) is the state dimension. Thus, sampling \( \sigma^2 \) in this example can be done efficiently by sampling from the posterior.

Sampling the length scales \( l_s \) and \( l_t \) cannot be done directly as they enter the model in a more complicated way (see Section 2). Hence, we use the Metropolis-within-Gibbs approach as discussed in Section 3.

4.3. Results and Discussion

The mean of the time-averaged RMSEs of the leave-one-out cross-validation for the presented scenario is 8.86 (±2.47). An example for the predicted population for the time between \( n = 100 \) and \( n = 200 \) is shown in Fig. 1. The illustration depicts the true population over time (blue, solid), the sampled trajectories (grey, solid), as well as the minimum mean squared error (MMSE) estimate (red, dashed). As it can be seen, the sampled trajectories are distributed around the true state and the MMSE estimate matches the true trajectory well. The RMSE is 6.21 in this case, which is somewhat lower than the mean RMSE. Furthermore, Fig. 2 shows the histograms of the corresponding parameter samples.

The example shows that the learned model exhibits good predictive performance based on the estimated parameters and trajectories. A major challenge that remains is the scalability of the proposed method with respect to the number of spatial training points: The dimension of the nonlinear state \( s_n \) grows linearly with the number of spatial training points, which in turn increases the number of particles required for sampling the state trajectories \( s^p_n \). Experiments showed that for a low number of points (e.g. 2–3), as few as 100 particles can be enough to accurately learn the system and generate predictions, while, at around 10 to 15 spatial training points, several thousand particles are required, slowing down the learning. Note, however, that increasing the number of temporal training points does not affect the state dimension, and only increases the number of time samples to be processed.

5. CONCLUSIONS

In this paper, we proposed a particle MCMC method for estimating the posterior distribution of the state and parameters in latent, spatio-temporal Gaussian process regression problems. As shown in the illustrations, the proposed method can handle nonlinear non-Gaussian likelihoods, which can be a limiting factor for other methods. For problems with a moderate to large number of spatial training points,
scaling can become an issue as the number of required particles in such scenarios increases quickly.

6. REFERENCES


