MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Recursive Bayesian Inference and Learning of Gaussian-Process State-Space Models

Berntorp, K.

TR2019-053 June 29, 2019

Abstract

Gaussian processes in combination with sequential Monte-Carlo methods have emerged as promising tools for offline nonlinear system identification. However, sometimes the dynamical system evolves in such a way that online learning is preferable. This paper addresses the online joint state estimation and learning problem for nonlinear dynamical systems. We leverage a recently developed reduced-rank formulation of Gaussian-process state-space models (GP-SSMs), and develop a recursive formulation for updating the sufficient statistics associated with the GP-SSM by exploiting marginalization and conjugate priors. The results indicate that our method efficiently learns the system jointly with estimating the state, and that the approach for certain scenarios gives similar performance as more computation-heavy offline approaches.

European Control Conference (ECC)

Recursive Bayesian Inference and Learning of Gaussian-Process State-Space Models

Karl Berntorp

***Abstract*— Gaussian processes in combination with sequential Monte-Carlo methods have emerged as promising tools for offline nonlinear system identification. However, sometimes the dynamical system evolves in such a way that online learning is preferable. This paper addresses the online joint state estimation and learning problem for nonlinear dynamical systems. We leverage a recently developed reduced-rank formulation of Gaussian-process state-space models (GP-SSMs), and develop a recursive formulation for updating the sufficient statistics associated with the GP-SSM by exploiting marginalization and conjugate priors. The results indicate that our method efficiently learns the system jointly with estimating the state, and that the approach for certain scenarios gives similar performance as more computation-heavy offline approaches.**

I. Introduction

The objective in system identification [1] is to learn models of dynamical systems from measurements. Recently, Gaussian processes (GPs, [2]) have emerged as a useful tool for learning of nonlinear systems on the form [3]–[8]

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, \tag{1a}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{e}_k, \tag{1b}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ is the state to be estimated, $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$ is the measurement, $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ is the known input, $\boldsymbol{w}_k$, $\boldsymbol{e}_k$, are the process and measurement noise, respectively, and $\boldsymbol{f}$, $\boldsymbol{h}$ are the functions to be learned from data. Modeling the state-transition $\boldsymbol{f}$ and observation function $\boldsymbol{h}$ as GPs, leading to GP state-space models (GP-SSMs) [9]–[11], has shown to be an efficient modeling approach to learn systems from uncertain data, in combination with particle filtering (PF, [12], [13]). Uncertain data can arise both due to limited amount of data or due to limited excitation of the system in regions of the state space. Using GP-SSMs, this uncertainty can be reflected in the learning process while avoiding the common issue of overfitting to data [7].

In this paper, we develop a PF [14] based approach for jointly estimating *online* the state trajectory and learning the state-transition function of the GP-SSM. We leverage a recently developed reduced-rank model formulation [6], [7], [15], [16] of the GP-SSM, in which connections between GPs and a basis-function expansion of $\boldsymbol{f}$ is made by introducing priors on the basis-function coefficients. We extend the work in [6], which treated offline batch learning in a particle Markov chain Monte-Carlo (PMCMC) setting, to the online setting, by tailoring a PF to GP-SSMs in combination with marginalization. Marginalization is crucial in our approach and is together with the reduced-rank formulation the key enabler of simultaneous state inference and system learning in a fully Bayesian setting. A bottleneck usually prohibiting the use of GP-SSM based methods is the poor scaling with the training data. In [6], it was shown that the reduced-rank approximation reduces the computational load significantly. In this paper several illustrative examples indicate that the reduced-rank formulation, with a proper implementation of the PF and suitable approximations, indeed can be used to do online joint Bayesian inference and learning. The reduced-rank formulation of the GP-SSM converges asymptotically [16], and after a burn-in period, the PMCMC implementation in [6] provides samples from the true posterior associated with the reduced-rank SSM for any number of particles $N \geq 2$. However, this strong property does not carry over to our online PF approach, similar to standard PFs.

GPs in system identification have by now a variety of use cases. For instance, in impulse response estimation [17], nonlinear ordinary differential equations [18], and force modeling [19]. There is an increasingly rich literature on filtering and smoothing in GP-SSMs (e.g., [20], [21]). When it comes to learning in GP-SSMs, the main difficulty is to infer the nonlinear function $\boldsymbol{f}$ from the latent state $\boldsymbol{x}_k$, which can only be observed from the measurements $\boldsymbol{y}_{0:k} = \{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_k\}$. Expectation maximization (EM) based procedures can be found in [3], [7], [22]. Various approaches based on PMCMC methods [12], [13] have been developed in [4], [6], [7]. When using PMCMC for learning, a PF conditioned on one of the state trajectories is iterated intertwined with updates of the sufficient statistics, using the whole available data set. Most of the learning methods for GP-SSMs are batch approaches, which means that they are not suitable for direct implementation in an online setting.

Our work extends [6] to the online case, providing a recursive implementation. In particular, we focus on discrete-time GP-SSMs on the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{w}_k, \tag{2a}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{e}_k, \tag{2b}$$

where $\boldsymbol{f}$ is assumed to be a realization from a GP prior over $\mathbb{R}^{n_x}$, $\boldsymbol{f}(\boldsymbol{x}_k) \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{K}_\theta(\boldsymbol{x}_k, \boldsymbol{x}_k'))$ for a given covariance function $\boldsymbol{K}_\theta(\boldsymbol{x}, \boldsymbol{x}')$ subject to (assumed known) hyperparameters $\boldsymbol{\theta}$. The process noise is Gaussian distributed with unknown covariance $\boldsymbol{Q}$ according to $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$. In the proposed method, each particle retains its own estimate of the unknowns $\boldsymbol{f}$ and $\boldsymbol{Q}$. We assume the observation function $\boldsymbol{h}$ to be known and the Gaussian measurement noise

Mitsubishi Electric Research Laboratories (MERL), 02139 Cambridge, MA, USA. Email: karl.o.berntorp@ieee.org

$\boldsymbol{e}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$ to have known covariance $\boldsymbol{R}$. However, the case of an unknown observation model follows analogously to the case of unknown $\boldsymbol{f}$, as does the case of nonzero inputs $\boldsymbol{u}_k$ similar to (1). There are also practical reasons for assuming a known observation model. First, introducing too many unknowns might have implications on observability of the system. Second, $\boldsymbol{h}$ usually corresponds to a sensor model that typically is known. We target a fully Bayesian solution to the joint state inference and learning problem, where the objective is to approximate the posterior distributions of $\boldsymbol{x}_k$, $\boldsymbol{f}$, and $\boldsymbol{Q}$ at each time step k,

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{0:k}), \tag{3a}$$

$$p(\boldsymbol{f}, \boldsymbol{Q}|\boldsymbol{y}_{0:k}). \tag{3b}$$

The outline of the paper is as follows. Sec. II provides background material on reduced-rank GP-SSMs and PFs necessary to understand our approach, which is explained in Sec. III. Sec. IV evaluates the proposed method using two numerical examples. Finally, Sec. V concludes the paper.

II. Reduced-Rank GP-SSMs and Particle Filtering

In this section, we briefly review background material on GP-SSMs and PFs that are necessary for understanding the proposed learning method described in Sec. III.

A. Reduced-Rank GP-SSMs

We rely on GPs for encoding the prior information of $\boldsymbol{f}$ for learning of the state-space model. The covariance function $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}')$ describes the prior assumptions on the function and is known a priori. Following the notation in [16], isotropic covariance functions (i.e., those only depend on the Euclidean norm $\|\boldsymbol{x} - \boldsymbol{x}'\|$) can be approximated in terms of Laplace operators,

$$\boldsymbol{K}_\theta(\boldsymbol{x}, \boldsymbol{x}') \approx \sum_{j_1,\ldots,j_{n_x}=1}^{m} \mathcal{S}_\theta(\lambda^{j_1,\ldots,j_{n_x}}) \phi^{j_1,\ldots,j_{n_x}}(\boldsymbol{x}) \phi^{j_1,\ldots,j_{n_x}}(\boldsymbol{x}'), \tag{4}$$

where, for simplicity, we assume m basis functions for each state dimension. In (4), $\mathcal{S}_\theta$ is the spectral density of $\boldsymbol{K}_\theta$ and

$$\lambda^{j_1,\ldots,j_{n_x}} = \sum_{n=1}^{n_x} \left(\frac{\pi j_n}{2L_n} \right)^2, \tag{5a}$$

$$\phi^{j_1,\ldots,j_{n_x}} = \prod_{n=1}^{n_x} \frac{1}{\sqrt{L_n}} \sin\left(\frac{\pi j_n (x_n + L_n)}{2L_n} \right), \tag{5b}$$

are the Laplace operator eigenvalues and eigenfunctions, respectively, on the interval $[-L_n, L_n] \in \mathbb{R}$ for each $n = 1, \ldots, n_x$. For brevity, we will in the rest of the paper denote $j_1, \ldots, j_{n_x}$ with $\boldsymbol{j}$. Note that according to (4), (5), only the spectral density depends on the hyperparameters $\boldsymbol{\theta}$. Furthermore, (4) can be interpreted as an optimal parametric expansion with respect to the covariance function in the GP prior [6]. As a special case when $n_x = 1$, (5) becomes $\lambda^j = (\pi j/(2L))^2$ and $\phi^j(x) = 1\sqrt{L} \sin(\pi j(x + L)/(2L))$.

From the approximation (4) using Laplace operators, [16] provides a relation between basis function expansions of a function f and GPs based on the Karhunen-Loeve expansion; with the basis functions chosen as (5b), then

$$f(\boldsymbol{x}) \sim \mathcal{GP}(0, \kappa(\boldsymbol{x}, \boldsymbol{x}')) \Leftrightarrow f(\boldsymbol{x}) \approx \sum_{\boldsymbol{j}} \gamma^{\boldsymbol{j}} \phi^{\boldsymbol{j}}(\boldsymbol{x}), \tag{6}$$

with

$$\gamma^{\boldsymbol{j}} \sim \mathcal{N}(0, \mathcal{S}_\theta(\lambda^{\boldsymbol{j}})). \tag{7}$$

For a state-space model $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{w}_k$, (6) implies the reduced-rank GP-SSM

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} \gamma_1^1 & \cdots & \gamma_1^m \\ \vdots & & \vdots \\ \gamma_{n_x}^1 & \cdots & \gamma_{n_x}^m \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} \phi^1(\boldsymbol{x}_k) \\ \vdots \\ \phi^m(\boldsymbol{x}_k) \end{bmatrix}}_{\boldsymbol{\varphi}(\boldsymbol{x}_k)} + \boldsymbol{w}_k, \tag{8}$$

where γ_n^j are the weights to be learned and m is the total number of basis functions (i.e., m^{n_x} in (4)). In Sec. III, (8) in combination with PF forms the basis for learning $\boldsymbol{A}$ and $\boldsymbol{Q}$ jointly with estimating the state $\boldsymbol{x}$. For later use, we express the prior on the coefficients $\gamma^{\boldsymbol{j}}$ in (7) at time step $k = 0$ as a Matrix-normal ($\mathcal{MN}$) distribution over $\boldsymbol{A}$ [23],

$$\boldsymbol{A} \sim \mathcal{MN}(\boldsymbol{M}, \boldsymbol{Q}, \boldsymbol{V}) \tag{9}$$

with mean $\boldsymbol{M} = \boldsymbol{0}$, right covariance $\boldsymbol{Q}$, and left precision $\boldsymbol{V}$ with diagonal elements $\mathcal{S}_\theta^{-1}(\lambda^{\boldsymbol{j}})$. We put an inverse-Wishart ($\mathcal{IW}$) prior on $\boldsymbol{Q}$ according to $\boldsymbol{Q} \sim \mathcal{IW}(\nu_0, \boldsymbol{\Lambda}_0)$, where $\nu_0 > n_x - 1$ is the degrees of freedom and $\boldsymbol{\Lambda}_0$ is a positive definite matrix. Assuming the covariance prior to be $\mathcal{IW}$ distributed is common in covariance estimation due to its properties [7], [24]–[26]. Since there exists different parametrizations of the $\mathcal{MN}$ and $\mathcal{IW}$ distributions, we provide the details in the Appendix.

B. Particle Filtering

Sequential Monte-Carlo (SMC) methods, such as PFs, constitute a class of techniques that estimate the posterior distribution in SSMs, and SMCs have recently emerged as a useful tool in learning of SSMs (e.g., [27]). PFs approximate the posterior density $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k})$ by a set of N weighted state trajectories as

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k}) \approx \sum_{i=1}^{N} q_k^i \delta_{\boldsymbol{x}_{0:k}^i}(\boldsymbol{x}_{0:k}), \tag{10}$$

where q_k^i is the importance weight of the ith state trajectory $\boldsymbol{x}_{0:k}^i$ and $\delta(\cdot)$ is the Dirac delta mass. The PF recursively estimates (10) by repeated application of Bayes' rule as

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k}) \propto p(\boldsymbol{y}_k|\boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k-1}) p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k-1}) \cdot p(\boldsymbol{x}_{0:k-1}|\boldsymbol{y}_{0:k-1}). \tag{11}$$

Since it is hard to obtain samples from (10) directly, sampling is done from a tractable, user-designed *proposal distribution* π, as

$$\boldsymbol{x}_k \sim \pi(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k}). \tag{12}$$

Inserting (10) into (11) and accounting for the proposal, importance weight q_k^i is obtained as

$$q_k^i \propto q_{k-1}^i \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_{0:k}^i, \boldsymbol{y}_{0:k-1}) p(\boldsymbol{x}_k^i|\boldsymbol{x}_{0:k-1}^i, \boldsymbol{y}_{0:k-1})}{\pi(\boldsymbol{x}_k^i|\boldsymbol{x}_{0:k-1}^i, \boldsymbol{y}_{0:k})}. \quad (13)$$

In this work we choose the proposal as the predictive density, $\pi(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k-1})$, which leads to the simplified weight update

$$q_k^i \propto q_{k-1}^i p(\boldsymbol{y}_k|\boldsymbol{x}_k^i). \quad (14)$$

The PF algorithm iterates between prediction and weight update, combined with a resampling step that removes particles with low weights and replaces them with more likely particles.

III. Recursive Bayesian Inference and Learning

The joint state inference and learning problem using the reduced-rank formulation (8) amounts to estimating the posterior distributions of $\boldsymbol{x}_{0:k}$, $\boldsymbol{Q}$, and $\boldsymbol{A}$. To this end, we utilize the factorization

$$p(\boldsymbol{A}, \boldsymbol{Q}, \boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k}) = p(\boldsymbol{A}, \boldsymbol{Q}|\boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}) p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k}), \quad (15)$$

where

$$p(\boldsymbol{A}, \boldsymbol{Q}|\boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}) = p(\boldsymbol{A}|\boldsymbol{Q}, \boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}) p(\boldsymbol{Q}|\boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}). \quad (16)$$

From (16), we can approximate (3b) by marginalization. To estimate (15), we will first describe how to compute (16). This is followed by a procedure for computing the prediction density $p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k-1})$, needed to generate the particles $\{\boldsymbol{x}_k^i\}_{i=1}^N$. Computing the prediction density is complicated by that it is dependent on the unknown coefficients $\boldsymbol{A}$ and covariance $\boldsymbol{Q}$.

A. Recursive Bayesian Learning in GP-SSMs

For the learning of $\boldsymbol{A}$ and $\boldsymbol{Q}$, we use Bayes' rule on (16),

$$p(\boldsymbol{A}, \boldsymbol{Q}|\boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}) \propto p(\boldsymbol{x}_k, \boldsymbol{y}_k|\boldsymbol{A}, \boldsymbol{Q}, \boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k-1}) \, p(\boldsymbol{A}, \boldsymbol{Q}|\boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k-1}). \quad (17)$$

The observation function (2b) is known and independent of $\boldsymbol{A}$ and $\boldsymbol{Q}$. Hence, from the Markov property of (8),

$$p(\boldsymbol{x}_k, \boldsymbol{y}_k|\boldsymbol{A}, \boldsymbol{Q}, \boldsymbol{x}_{0:k-1}, \boldsymbol{y}_{0:k-1}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{A}, \boldsymbol{Q}) p(\boldsymbol{y}_k|\boldsymbol{x}_k), \quad (18)$$

which is the product of two Gaussian distributions. To get a computationally tractable solution of (17), we use conjugate priors. Given a likelihood, a conjugate prior is the prior distribution such that the prior and posterior are in the same family of distributions. From (9), the joint prior $p(\boldsymbol{A}, \boldsymbol{Q})$ at time step $k = 0$ is $\mathcal{MNIW}$ distributed with the hierarchical structure

$$\mathcal{MNIW}(\boldsymbol{A}, \boldsymbol{Q}|\boldsymbol{0}, \boldsymbol{V}, \boldsymbol{\Lambda}_0, \nu_0) = \mathcal{MN}(\boldsymbol{A}|\boldsymbol{0}, \boldsymbol{Q}, \boldsymbol{V}) \mathcal{IW}(\boldsymbol{Q}|\nu_0, \boldsymbol{\Lambda}_0). \quad (19)$$

For the Gaussian joint likelihood (18), the joint prior (19) is a conjugate prior [7], [23], [28]. Hence, for any $k = 1, 2, \ldots$, the posterior (17) is $\mathcal{MNIW}$ distributed when conditioning on $\boldsymbol{x}_{0:k}$ and $\boldsymbol{y}_{0:k}$.

For a set of measurements $\boldsymbol{y}_{0:T}$, by introducing

$$\boldsymbol{\Phi}_T = \sum_{k=1}^{T-1} \boldsymbol{x}_{k+1} \boldsymbol{x}_{k+1}^{\mathrm{T}}, \quad (20a)$$

$$\boldsymbol{\Psi}_T = \sum_{k=1}^{T-1} \boldsymbol{x}_{k+1} \boldsymbol{\varphi}(\boldsymbol{x}_k)^{\mathrm{T}}, \quad (20b)$$

$$\boldsymbol{\Sigma}_T = \sum_{k=1}^{T-1} \boldsymbol{\varphi}(\boldsymbol{x}_k) \boldsymbol{\varphi}(\boldsymbol{x}_k)^{\mathrm{T}}, \quad (20c)$$

the posterior distribution of $\boldsymbol{A}$ and $\boldsymbol{Q}$ when using a batch of measurements in a PMCMC implementation for learning [6], [7], becomes

$$p(\boldsymbol{Q}|\boldsymbol{x}_{0:T}, \boldsymbol{y}_{0:T}) = \mathcal{IW}(\boldsymbol{Q}|\bar{\nu}, \bar{\boldsymbol{\Lambda}}), \quad (21a)$$

$$p(\boldsymbol{A}|\boldsymbol{Q}, \boldsymbol{x}_{0:T}, \boldsymbol{y}_{0:T}) = \mathcal{MN}(\boldsymbol{A}|\bar{\boldsymbol{M}}, \boldsymbol{Q}, \bar{\boldsymbol{\Sigma}}^{-1}), \quad (21b)$$

where

$$\begin{aligned} \bar{\boldsymbol{M}} &= \bar{\boldsymbol{\Psi}} \bar{\boldsymbol{\Sigma}}^{-1}, \quad \bar{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_T + \boldsymbol{V}, \quad \bar{\boldsymbol{\Psi}} = \boldsymbol{\Psi}_T + \boldsymbol{M}\boldsymbol{V}, \\ \bar{\boldsymbol{\Lambda}} &= \boldsymbol{\Lambda}_0 + \bar{\boldsymbol{\Phi}} - \bar{\boldsymbol{M}} \bar{\boldsymbol{\Psi}}^{\mathrm{T}}, \qquad \bar{\boldsymbol{\Phi}} = \boldsymbol{\Phi}_T + \boldsymbol{M}\boldsymbol{V}\boldsymbol{M}^{\mathrm{T}}, \end{aligned} \quad (22)$$

and $\bar{\nu} = T + \nu_0$. Since we assume a zero-mean prior on the coefficients in (9) at time step $k = 0$, in (22), $\boldsymbol{M} = \boldsymbol{0}$.

To get recursive update equations suitable for online learning, we note that from (20),

$$\boldsymbol{\Phi}_{k+1} = \boldsymbol{\Phi}_k + \boldsymbol{x}_{k+1} \boldsymbol{x}_{k+1}^{\mathrm{T}}, \quad (23a)$$

$$\boldsymbol{\Psi}_{k+1} = \boldsymbol{\Psi}_k + \boldsymbol{x}_{k+1} \boldsymbol{\varphi}(\boldsymbol{x}_k)^{\mathrm{T}}, \quad (23b)$$

$$\boldsymbol{\Sigma}_{k+1} = \boldsymbol{\Sigma}_k + \boldsymbol{\varphi}(\boldsymbol{x}_k) \boldsymbol{\varphi}(\boldsymbol{x}_k)^{\mathrm{T}}. \quad (23c)$$

By inserting (23) into (22), we can write the conditional densities in (16) as

$$p(\boldsymbol{A}|\boldsymbol{Q}, \boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}) = \mathcal{MN}(\boldsymbol{A}|\bar{\boldsymbol{M}}_{k|k}, \boldsymbol{Q}, \bar{\boldsymbol{\Sigma}}_{k|k}^{-1}), \quad (24a)$$

$$p(\boldsymbol{Q}|\boldsymbol{x}_{0:k}, \boldsymbol{y}_{0:k}) = \mathcal{IW}(\boldsymbol{Q}|\bar{\nu}_{k|k}, \bar{\boldsymbol{\Lambda}}_{k|k}). \quad (24b)$$

With the initialization

$$\bar{\boldsymbol{M}}_0 = \boldsymbol{0}, \ \bar{\boldsymbol{\Sigma}}_0 = \boldsymbol{V}, \ \bar{\boldsymbol{\Psi}}_0 = \boldsymbol{0}, \ \bar{\boldsymbol{\Phi}}_0 = \boldsymbol{0}, \quad (25)$$

it can be shown that the sufficient statistics in (24) can be recursively updated as

$$\bar{\boldsymbol{M}}_{k|k} = \bar{\boldsymbol{\Psi}}_{k|k} \bar{\boldsymbol{\Sigma}}_{k|k}^{-1}, \quad (26a)$$

$$\bar{\nu}_{k|k} = \bar{\nu}_{k|k-1} + 1, \quad (26b)$$

$$\bar{\boldsymbol{\Lambda}}_{k|k} = \boldsymbol{\Lambda}_0 + \bar{\boldsymbol{\Phi}}_{k|k} - \bar{\boldsymbol{M}}_{k|k} \bar{\boldsymbol{\Psi}}_{k|k}^{\mathrm{T}}, \quad (26c)$$

$$\bar{\boldsymbol{\Sigma}}_{k|k} = \bar{\boldsymbol{\Sigma}}_{k|k-1} + \boldsymbol{\varphi}(\boldsymbol{x}_{k-1}) \boldsymbol{\varphi}(\boldsymbol{x}_{k-1})^{\mathrm{T}}, \quad (26d)$$

$$\bar{\boldsymbol{\Phi}}_{k|k} = \bar{\boldsymbol{\Psi}}_{k|k-1} \bar{\boldsymbol{M}}_{k|k-1}^{\mathrm{T}} + \boldsymbol{x}_k \boldsymbol{x}_k^{\mathrm{T}}, \quad (26e)$$

$$\bar{\boldsymbol{\Psi}}_{k|k} = \bar{\boldsymbol{\Psi}}_{k|k-1} + \boldsymbol{x}_k \boldsymbol{\varphi}(\boldsymbol{x}_{k-1})^{\mathrm{T}}, \quad (26f)$$

with the statistics of the predictive distributions given by the time-update step

$$\bar{\boldsymbol{\Phi}}_{k|k-1} = \lambda \bar{\boldsymbol{\Phi}}_{k-1|k-1}, \quad (27a)$$

$$\bar{\boldsymbol{\Psi}}_{k|k-1} = \lambda \bar{\boldsymbol{\Psi}}_{k-1|k-1}, \quad (27b)$$

$$\bar{\nu}_{k|k-1} = \lambda \bar{\nu}_{k-1|k-1}. \quad (27c)$$

The scalar real-valued number $\lambda \in [0,1]$ provides exponential forgetting in the data that allows the algorithm to adapt to (slowly time-varying) changes in $\boldsymbol{A}$ and $\boldsymbol{Q}$ over time, and also mitigates path degeneracy [24]. To find the posterior distribution of $\boldsymbol{A}$ and $\boldsymbol{Q}$, we marginalize out the state trajectory as

$$p(\boldsymbol{A},\boldsymbol{Q}|\boldsymbol{y}_{0:k}) = \int p(\boldsymbol{A},\boldsymbol{Q}|\boldsymbol{x}_{0:k},\boldsymbol{y}_{0:k})p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k})\,\mathrm{d}\boldsymbol{x}_{0:k} \approx \sum_{i=1}^{N} q_k^i p(\boldsymbol{A}^i,\boldsymbol{Q}^i|\boldsymbol{x}_{0:k}^i,\boldsymbol{y}_{0:k}). \quad (28)$$

The unknown function and covariance can be extracted from (28). For instance, an estimate $\hat{\boldsymbol{f}}_k$ of $\boldsymbol{f}$ is

$$\hat{\boldsymbol{f}}_k = \sum_{i=1}^{N} q_k^i \boldsymbol{A}^i \boldsymbol{\varphi}(\boldsymbol{x}_k^i). \quad (29)$$

B. Recursive Bayesian State Inference in GP-SSMs

To estimate the state trajectory, we need an expression for the predictive (proposal) density $p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^i,\boldsymbol{y}_{0:k-1})$ in (12). Using the lemma on transformation of variables in probability density functions,

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^i,\boldsymbol{y}_{0:k-1}) \propto p(\boldsymbol{A},\boldsymbol{Q}|S_{k-1}^i), \quad (30)$$

where $S_k^i = \{\bar{\boldsymbol{M}}_{k|k}^i, \bar{\nu}_{k|k}^i, \bar{\boldsymbol{\Lambda}}_{k|k}^i, \bar{\boldsymbol{\Sigma}}_{k|k}^i, \bar{\boldsymbol{\Phi}}_{k|k}^i, \bar{\boldsymbol{\Psi}}_{k|k}^i\}$. By integrating out the unknown parameters,

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^i,\boldsymbol{y}_{0:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{A},\boldsymbol{Q},\boldsymbol{x}_{k-1}^i,\boldsymbol{y}_{0:k-1}) p(\boldsymbol{A},\boldsymbol{Q}|\boldsymbol{x}_{0:k-1}^i,\boldsymbol{y}_{0:k-1})\mathrm{d}\boldsymbol{A}\mathrm{d}\boldsymbol{Q}. \quad (31)$$

Hence, the predictive distribution (31) is a Student-t density, which from (30) can be written as

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}^i,\boldsymbol{y}_{0:k-1}) = \mathrm{St}(\bar{\boldsymbol{M}}_{k|k-1}^i\boldsymbol{\varphi}(\boldsymbol{x}_{k-1}^i), \bar{\boldsymbol{\Lambda}}_{k|k-1}^i, \bar{\nu}_{k|k-1}^i - n_x + 1). \quad (32)$$

The samples $\{\boldsymbol{x}_k^i\}_{i=1}^N$ generated from (32) are used to compute the weights (14) and to update the statistics (26).

Note that in the proposed method, each particle i retains its own estimate of the unknown parameters $\boldsymbol{A}^i$ and $\boldsymbol{Q}^i$, associated statistics, as well as the state $\boldsymbol{x}_k^i$ and weight q_k^i. Algorithm 1 summarizes the proposed method.

IV. Numerical Evaluation

We evaluate against two different benchmark examples and compare with two offline PMCMC methods. The first one is the fully Bayesian batch method in [6] which our method extends to the online case. The second one is the regularized maximum likelihood method proposed in [7]. Both methods use a particle Gibbs Markov kernel for finding the state smoothing distribution within an MCMC procedure based on the whole sequence of measurements. In both examples, we use a squared exponential covariance function [2] $\kappa(r) = s_f \exp\left(-r^2/(2l^2)\right)$ with spectral density

$$S(s) = s_f\sqrt{2\pi l^2}\exp\left(-\frac{\pi^2 l^2 s^2}{2}\right). \quad (33)$$

Algorithm 1 Pseudo-code of proposed algorithm

Initialize: Set $\{\boldsymbol{x}_0^i\}_{i=1}^N \sim p_0(\boldsymbol{x}_0)$, $\{q_{-1}^i\}_{i=1}^N = 1/N$, $\{S_0^i\}_{i=1}^N = \{\boldsymbol{0}, \nu_0, \boldsymbol{\Lambda}_0, \boldsymbol{V}, \boldsymbol{0}, \boldsymbol{0}\}$

for $k = 0, 1, \ldots$ **do**
  for $i \in \{1, \ldots, N\}$ **do**
    Update weight $\bar{q}_k^i$ using (14):
    $$\bar{q}_k^i = q_{k-1}^i p(\boldsymbol{y}_k|\boldsymbol{x}_k^i)$$
    Update relevant statistics using (26).
  end for
  Normalize weights as $q_k^i = \bar{q}_k^i/(\sum_{i=1}^N \bar{q}_k^i)$.
  Compute $N_{\text{eff}} = 1/(\sum_{i=1}^N (q_k^i)^2)$
  if $N_{\text{eff}} \leq N_{\text{thr}}$ **then**
    Resample particles and copy the corresponding statistics. Set $\{q_k^i\}_{i=1}^N = 1/N$.
  end if
  Compute state estimate $\hat{\boldsymbol{x}}_k = \sum_{i=1}^N q_k^i \boldsymbol{x}_k^i$.
  Compute function estimate using (29).
  for $i \in \{1, \ldots, N\}$ **do**
    Predict relevant statistics using (27).
    Sample $\boldsymbol{x}_{k+1}^i$ from (32).
  end for
end for

A. Example 1

We first consider the system [6]

$$x_{k+1} = \tanh(2x_k) + w_k, \quad w_k \sim \mathcal{N}(0, 0.1), \quad (34a)$$
$$y_k = x_k + e_k, \quad e_k \sim \mathcal{N}(0, 0.1), \quad (34b)$$

where the objective is to learn f and Q. We initialize our algorithm with $\nu_0 = 10$, $\Lambda_0 = 1$, (i.e., with prior $Q \sim \mathcal{IW}(10, 1)$) and use $s_f = 50$, $l = 1$ in (33). Furthermore, we set $L = 4$ in (5), use $N = 20$ particles, and $m = 16$ basis functions. The posterior estimate of the method in [6] and the maximum likelihood estimate of the method in [7] for $T = 500$ data points and $K = 500$ MCMC iterations are shown in Fig. 1. Fig. 2 displays the results from our method for $k = 5$, $k = 50$, and $k = 500$ time steps, respectively. Our method converges as the number of data points increases, and when comparing Fig. 1 with Fig. 2 for $k = 500$, the performance is rather similar in terms of the posterior mean where most of the data are gathered (see lowest plot in Fig. 1).

B. Example 2

In this example we consider the following model,

$$x_{k+1} = 10\mathrm{sinc}\left(\frac{x_k}{7}\right) + w_k, \quad w_k \sim \mathcal{N}(0, 1), \quad (35a)$$
$$y_k = x_k + e_k, \quad e_k \sim \mathcal{N}(0, 1), \quad (35b)$$

where the objective is to learn f when the noise variance is known. We use $L = 30$ and $m = 40$ basis functions, with $s_f = 50$ and $l = 3$, and the number of particles is $N = 50$. Fig. 3 displays the results for $k = 5, 20, 40$. For $k = 5$, the estimator has not gathered enough information for it to make a sensible estimate of the dynamical model. However, already

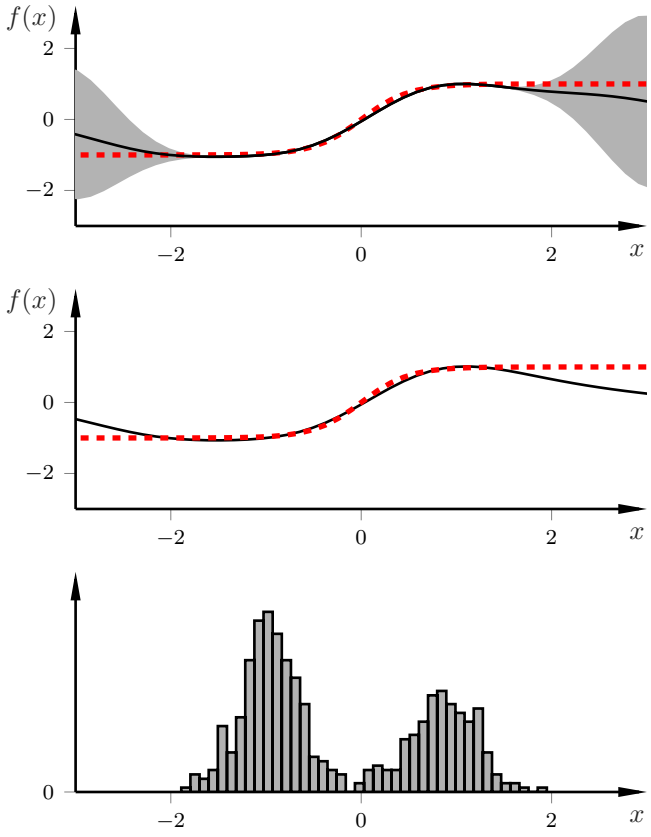

Fig. 1. The posterior estimates of the method in [6] (upper) and the regularized maximum likelihood method in [7] (middle) for $T = 500$ data points and 500 MCMC iterations for Example 1. The true function is in dashed red, the estimate in black, and for the Bayesian learning, the distribution is shown in gray. The bars in the lowest plot show the distribution of data in the state space.

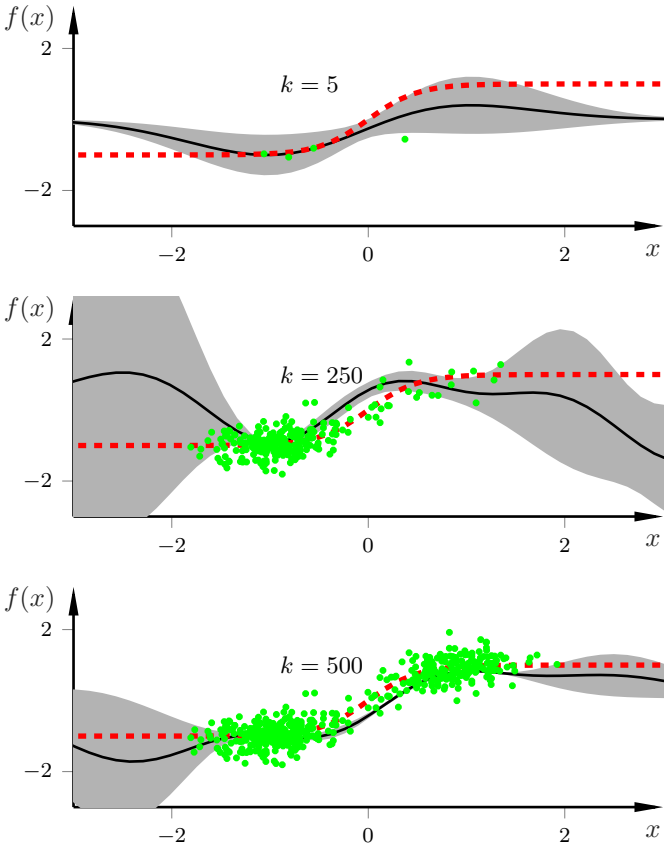

Fig. 2. The posterior estimates of our proposed method (Algorithm 1) for $k = 5$ (upper), $k = 250$ (middle), and $k = 500$ (lower) data points, for Example 1. True function in dashed red, estimated function in black, and estimated distribution in gray. The green dots indicate the measurements gathered for the different time steps.

for $k = 20$ the estimate looks reasonable and for $k = 40$, the main characteristics are correctly identified in the regions where data has been gathered. When comparing to the offline PMCMC methods in Fig. 4, the respective function estimates are very similar.

V. Conclusion

We developed a novel approach for real-time joint state estimation and system identification. The method is fully Bayesian and is based on a combination of particle filtering and Gaussian-process state-space models. The method relies on conjugate priors and the marginalization concept. In combination with a recently proposed truncated basis-expansion formulation, this gives an online algorithm where each particle retains its own estimate of the unknown state-transition function and process-noise covariance. The method is applicable to general nonlinear systems, and a comparison with state-of-the-art offline learning approaches indicates that the method can, at least for certain problems, give similar performance but in real-time. It is future work to include estimation of the involved hyperparameters, as it can be nontrivial to tune these.

References

[1] L. Ljung, *System identification: theory for the user*. Prentice-hall, 1987.

[2] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[3] R. Turner, M. Deisenroth, and C. E. Rasmussen, "State-space inference and learning with Gaussian processes," in *Int. Conf. Artificial Intelligence and Statistics*, Sardinia, Italy, May 2010.

[4] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen, "Bayesian inference and learning in Gaussian process state-space models with particle MCMC," in *Adv. Neural Information Processing Systems*, Lake Tahoe, NV, Dec. 2013.

[5] R. Frigola, Y. Chen, and C. E. Rasmussen, "Variational Gaussian process state-space models," in *Advances in Neural Information Processing Systems*, Montreal, Canada, Dec. 2014.

[6] A. Svensson, A. Solin, S. Särkkä, and T. B. Schön, "Computationally efficient Bayesian learning of Gaussian process state space models," in *Int. Conf. Artificial Intelligence and Statistics*, Cadiz, Spain, May 2016.

[7] A. Svensson and T. B. Schön, "A flexible state-space model for learning nonlinear dynamical systems," *Automatica*, vol. 80, pp. 189–199, 2017.

[8] C. L. C. Mattos, Z. Dai, A. Damianou, J. Forth, G. A. Barreto, and N. D. Lawrence, "Recurrent Gaussian processes," *arXiv preprint arXiv:1511.06644*, 2015.

[9] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE Trans Pattern Anal Mach Intell*, vol. 30, no. 2, pp. 283–298, 2008.

[10] A. McHutchon, "Nonlinear modelling and control using Gaussian processes," phdthesis, Univ. Cambridge, 2014.

[11] R. Frigola-Alcade, "Bayesian time series learning with Gaussian processes," phdthesis, Univ. Cambridge, 2015.

[12] F. Lindsten, M. I. Jordan, and T. B. Schön, "Particle Gibbs with ancestor sampling." *J. Machine Learning Res.*, vol. 15, no. 1, pp. 2145–2184, 2014.

[13] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain

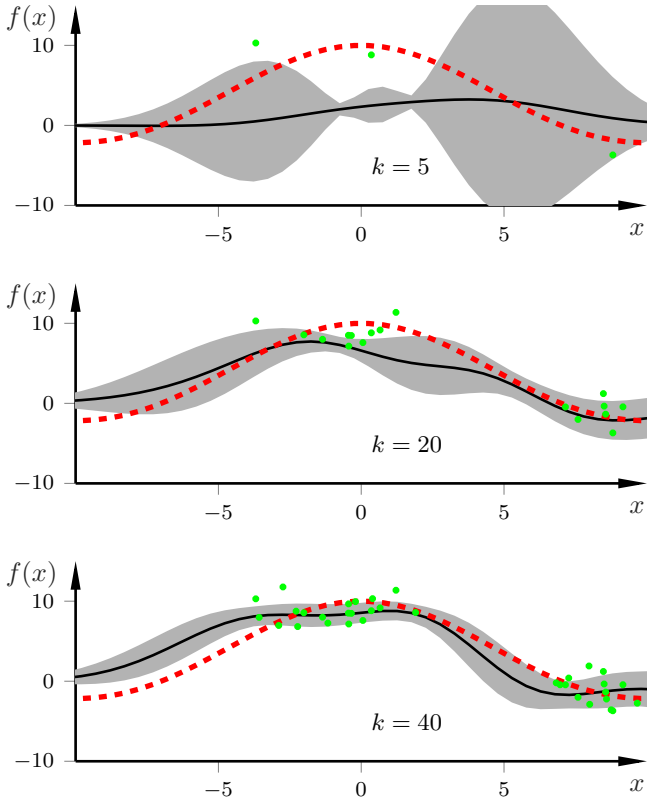

Fig. 3. The posterior estimates of our proposed method in Example 2 for $T = 5$ (upper), $T = 20$ (middle), and $T = 40$ (lower) data points. Same notation as in Fig. 2.

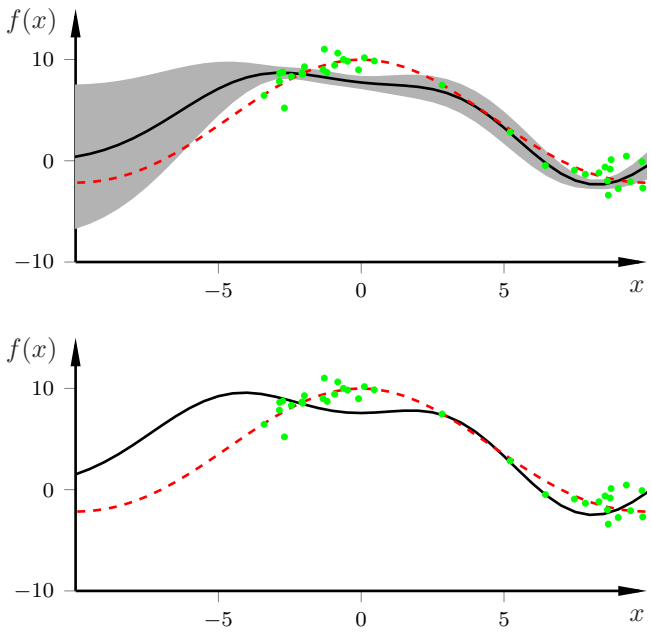

Fig. 4. The posterior estimates of the fully Bayesian method (upper plot) in [6] and the regularized maximum likelihood method (lower plot) in [7] for $T = 40$ data points and 400 MCMC iterations for Example 2. The green dots indicate the measurements gathered for the $T = 40$ time steps.

Monte Carlo methods," *J. Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[14] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *Handbook of Nonlinear Filtering*, D. Crisan and B. Rozovsky, Eds. Oxford University Press, 2009.

[15] A. Wills, T. B. Schön, F. Lindsten, and B. Ninness, "Estimation of linear systems using a Gibbs sampler," in *IFAC Symp. System Identification*, Brussels, Belgium, 2012.

[16] A. Solin and S. Särkkä, "Hilbert space methods for reduced-rank Gaussian process regression," *arXiv preprint arXiv:1401.5508*, 2014.

[17] G. Pillonetto and G. De Nicolao, "A new kernel-based approach for linear system identification," *Automatica*, vol. 46, no. 1, pp. 81–93, 2010.

[18] B. Macdonald, C. Higham, and D. Husmeier, "Controversy in mechanistic modelling with Gaussian processes," in *Int. Conf. Machine Learning*, Lille, France, July 2015.

[19] M. A. Alvarez, D. Luengo, and N. D. Lawrence, "Linear latent force models using Gaussian processes," *IEEE Trans Pattern Anal Mach Intell*, vol. 35, no. 11, pp. 2693–2705, 2013.

[20] M. P. Deisenroth, R. D. Turner, M. F. Huber, U. D. Hanebeck, and C. E. Rasmussen, "Robust filtering and smoothing with Gaussian processes," *IEEE Trans. Autom. Control*, vol. 57, no. 7, pp. 1865–1871, 2012.

[21] M. Deisenroth and S. Mohamed, "Expectation propagation in Gaussian process dynamical systems," in *Adv. Neural Information Processing Systems*, Lake Tahoe, NV, Dec. 2012.

[22] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen, "Identification of Gaussian process state-space models with particle stochastic approximation EM," in *IFAC World Congress*, Cape Town, Sout Africa, Aug. 2014.

[23] A. P. Dawid, "Some matrix-variate distribution theory: notational considerations and a Bayesian application," *Biometrika*, vol. 68, no. 1, pp. 265–274, 1981.

[24] E. Özkan, V. Šmídl, S. Saha, C. Lundquist, and F. Gustafsson, "Marginalized adaptive particle filtering for nonlinear models with unknown time-varying noise parameters," *Automatica*, vol. 49, no. 6, pp. 1566–1575, 2013.

[25] K. Berntorp and S. Di Cairano, "Tire-stiffness and vehicle-state estimation based on noise-adaptive particle filtering," *IEEE Trans. Control Syst. Technol.*, 2018, in press.

[26] K. Berntorp and S. Di Cairano, "Noise-statistics learning of automotive-grade sensors using adaptive marginalized particle filtering," *J. Dynamic Systems, Measurement, and Control*, 2019, in press.

[27] N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, N. Chopin, *et al.*, "On particle methods for parameter estimation in state-space models," *Statistical science*, vol. 30, no. 3, pp. 328–351, 2015.

[28] K. P. Murphy, "Conjugate Bayesian analysis of the Gaussian distribution," UBC, Tech. Rep., 2007.

APPENDIX

The $\mathcal{MNIW}$ distribution is the conjugate prior for SSMs linear in its parameters $\boldsymbol{A} \in \mathbb{R}^{n_x \times m}$ and $\boldsymbol{Q} \in \mathbb{R}^{n_x \times n_x}$. The $\mathcal{MNIW}$ distribution can be described by $\mathcal{MNIW}(\boldsymbol{A}, \boldsymbol{Q} | \boldsymbol{M}, \boldsymbol{V}, \nu, \boldsymbol{\Lambda}) = \mathcal{MN}(\boldsymbol{A} | \boldsymbol{M}, \boldsymbol{Q}, \boldsymbol{V}) \mathcal{IW}(\boldsymbol{Q} | \nu, \boldsymbol{\Lambda})$, where each part is

$$\mathcal{MN}(\boldsymbol{A} | \boldsymbol{M}, \boldsymbol{Q}, \boldsymbol{V}) = \frac{|\boldsymbol{V}|^{n_x/2}}{(2\pi)^{n_x m} |\boldsymbol{Q}|^{m/2}} \times \exp\left(-\frac{1}{2} \mathrm{tr}\left((\boldsymbol{A} - \boldsymbol{M})^{\mathrm{T}} \boldsymbol{U}^{-1} (\boldsymbol{A} - \boldsymbol{M}) \boldsymbol{V} \right) \right), \quad (36)$$

where $|\cdot|$ is the determinant, $\mathrm{tr}(\cdot)$ is the trace operator, and

$$\mathcal{IW}(\boldsymbol{U} | \nu, \boldsymbol{\Lambda}) = \frac{|\boldsymbol{\Lambda}|^{\nu/2} |\boldsymbol{U}|^{-(n_x + \nu + 1)/2}}{2^{\nu n_x / 2} \Gamma_{n_x}(\nu/2)} \times \exp\left(-\frac{1}{2} \mathrm{tr}(\boldsymbol{U}^{-1} \boldsymbol{\Lambda}) \right), \quad (37)$$

where ν is the degrees of freedom, $\boldsymbol{\Lambda} \in \mathbb{R}^{n_x \times n_x}$ is a positive definite scale matrix, and $\Gamma_{n_x}(\cdot)$ is the multivariate gamma function.