Sequential Monte Carlo Methods for System Identification

Thomas Bo Schön, Fredrik Lindsten, Johan Dahlin, Johan Wågberg,
Christian Andersson Naesseth, Andreas Svensson and Liang Dai

Original Publication:

Thomas Bo Schön, Fredrik Lindsten, Johan Dahlin, Johan Wågberg, Christian Andersson Naesseth, Andreas Svensson and Liang Dai, Sequential Monte Carlo Methods for System Identification, 2015, Proceedings of the 17th IFAC Symposium on System Identification., 775-786.
http://dx.doi.org/10.1016/j.ifacol.2015.12.224

Sequential Monte Carlo Methods for System Identification

Thomas B. Schön, Fredrik Lindsten, Johan Dahlin,
Johan Wågberg, Christian A. Naesseth,
Andreas Svensson and Liang Dai*

October 16, 2015

Abstract

One of the key challenges in identifying nonlinear and possibly non-Gaussian state space models (SSMs) is the intractability of estimating the system state. Sequential Monte Carlo (SMC) methods, such as the particle filter (introduced more than two decades ago), provide numerical solutions to the nonlinear state estimation problems arising in SSMs. When combined with additional identification techniques, these algorithms provide solid solutions to the nonlinear system identification problem. We describe two general strategies for creating such combinations and discuss why SMC is a natural tool for implementing these strategies.

*This work was supported by the projects *Learning of complex dynamical systems* (Contract number: 637-2014-466) and *Probabilistic modeling of dynamical systems* (Contract number: 621-2013-5524), both funded by the Swedish Research Council. TS, JW, AS and LD are with Division of Systems and Control, Uppsala University, Uppsala, Sweden. E-mail: {thomas.schon, johan.wagberg, andreas.svensson, liang.dai}@it.uu.se. FL is with the Department of Engineering, University of Cambridge, Cambridge, United Kingdom. E-mail: fredrik.lindsten@eng.cam.ac.uk. JD and CAN are with the Division of Automatic Control, Linköping University, Linköping, Sweden. E-mail: {christian.a.naesseth,johan.dahlin}@liu.se.

1 Introduction

This paper is concerned with system identification of nonlinear state space models (SSMs) in discrete time. The general model that we consider is given by,

$$x_{t+1} \mid x_t \sim f_\theta(x_{t+1} \mid x_t, u_t), \tag{1a}$$
$$y_t \mid x_t \sim g_\theta(y_t \mid x_t, u_t), \tag{1b}$$
$$(\theta \sim \pi(\theta)), \tag{1c}$$

where the states, the known inputs and the observed measurements are denoted by $x_t \in \mathsf{X} \subseteq \mathbb{R}^{n_x}$, $u_t \in \mathsf{U} \subseteq \mathbb{R}^{n_u}$ and $y_t \in \mathsf{Y} \subseteq \mathbb{R}^{n_y}$, respectively. The dynamics and the measurements are modeled by the probability density functions (PDFs) $f_\theta(\cdot)$ and $g_\theta(\cdot)$, respectively, parameterised by the unknown vector $\theta \in \Theta \subseteq \mathbb{R}^{n_\theta}$. The initial state x_1 is distributed according to some distribution $\mu_\theta(x_1)$. For notational simplicity, we will from hereon (without loss of generality) drop the known input u_t from the notation. When considering Bayesian identification, meaning that θ is modelled as an unobserved random variable, we also place a prior $\pi(\theta)$ on θ. We are concerned with off-line identification, i.e. we wish to find the unknown parameters θ in (1) based on a *batch* of T measurements.

The *key challenge* that will drive the development throughout this paper is how to deal with the difficulty that the states $x_{1:T}$ in (1) are unknown. We will distinguish between two different strategies for handling this:

1. **Marginalisation** amounts to marginalising (integrating out) the states from the problem and viewing θ as the only unknown quantity of interest. In the frequentistic problem formulation, the prediction error method [Ljung, 1999] and direct maximisation of the likelihood belong to this strategy. In the Bayesian formulation, the Metropolis–Hastings algorithm [Metropolis et al., 1953, Hastings, 1970] can be used to approximate the posterior distribution of the parameters conditionally on the data.

2. **Data augmentation** treats the states as auxiliary variables that are estimated together with the parameters. The expectation maximisation (EM) algorithm of Dempster et al. [1977] solves the maximum likelihood formulation in this way and the Gibbs sampler of Geman and Geman [1984] solves the Bayesian problem by this strategy.

In the special case when the model (1) is linear and Gaussian, there are closed form expressions available from the Kalman filter and the associated smoother. The primary focus of this paper, however, is the more challenging nonlinear and/or non-Gaussian case. More than two decades ago [e.g., Gordon et al., 1993, Kitagawa, 1993] *sequential Monte Carlo* (SMC) methods started to emerge with the introduction of the *particle filter*. These methods have since then undergone a rapid development and today they constitute a standard solution to the problem of computing the latent (i.e. unobserved/unknown/hidden) states in nonlinear/non-Gaussian SSMs.

The *aim* of this paper is to show how SMC can be used in solving the nonlinear system identification problems that arise in finding the unknown parameters in (1). We do not aim to cover all different methods that are available, but instead we aim to clearly describe exactly where and how the need for SMC arises and focus on the key principles. Complementary overview paper are provided by Kantas et al. [2014] and by Andrieu et al. [2004].

We consider the Bayesian and the maximum likelihood formulations, as defined in Section 2. The rest of the paper is divided into three parts. In the first part (Sections 3 and 4) we describe the marginalisation and data augmentation strategies and show where the need for SMC arise. The second part (Section 5) provides a rather self-contained derivation of the particle filter and outlines some of its properties. Finally, in the third part (Section 6–8) we show how these *particle methods* can be used to implement the identification strategies described in the first part, resulting in several state-of-the-art algorithms for nonlinear system identification. Loosely speaking, the SMC-part of the various algorithms that we introduce is essentially a way of systematically exploring the state space X^T in a nonlinear SSM (1) in order to address the key challenge of dealing with the latent state sequence $x_{1:T}$.

2 Problem formulation

There are different ways in which the system identification problem can be formulated. Two common formalisms are grounded in frequentistic and Bayesian statistics, respectively. We will treat both of these formulations in this paper, without making any individual ranking among them. First, we consider the frequentistic, or *maximum likelihood* (ML), formulation. This amounts to finding a point estimate of the unknown parameter θ, for which the observed data is as likely as possible. This is done by maximising the data likelihood function according to

$$\widehat{\theta}_{\mathrm{ML}} = \arg\max_{\theta \in \Theta} p_\theta(y_{1:T}) = \arg\max_{\theta \in \Theta} \log p_\theta(y_{1:T}). \tag{2}$$

For a thorough treatment of the use of ML for system identification, see e.g., Ljung [1999], Söderström and Stoica [1989].

Secondly, in the *Bayesian* formulation, the unknown parameters θ are modeled as a random variable (or random vector) according to (1c). The system identification problem thus amounts to computing the *posterior* distribution of θ given the observed data. According to Bayes' theorem, the posterior distribution is given by,

$$p(\theta \,|\, y_{1:T}) = \frac{p_\theta(y_{1:T})\pi(\theta)}{p(y_{1:T})}. \tag{3}$$

Note that the likelihood function should now be interpreted as the conditional PDF of the observations given the parameters θ, i.e. $p_\theta(y_{1:T}) = p(y_{1:T} \,|\, \theta)$. However, to be able to discuss the different identification criteria in a common

setting, we will, with a slight abuse of notation, denote the likelihood by $p_\theta(y_{1:T})$ also in the Bayesian formulation. An early account of Bayesian system identification is provided by Peterka [1981] and a more recent description is available in Ninness and Henriksen [2010].

The *central object* in both formulations above is the observed data likelihood $p_\theta(y_{1:T})$, or its constituents $p_\theta(y_t \mid y_{1:t-1})$ as,

$$p_\theta(y_{1:T}) = \prod_{t=1}^{T} p_\theta(y_t \mid y_{1:t-1}), \tag{4}$$

where we have used the convention $y_{1\mid 0} \triangleq \emptyset$. For the nonlinear SSM (1), the likelihood (4) is not available in closed form. This is a result of the fact that the latent state sequence $x_{1:T}$, i.e. $p(x_{1:T} \mid y_{1:T})$, is unknown. Indeed, a relationship between the likelihood and the latent states can be obtained via marginalization of the joint density $p_\theta(x_{1:T}, y_{1:T})$ w.r.t. $x_{1:T}$ according to,

$$p_\theta(y_{1:T}) = \int p_\theta(x_{1:T}, y_{1:T}) \mathrm{d}x_{1:T}, \tag{5}$$

where the model provides a closed form expression for the integrand according to,

$$p_\theta(x_{1:T}, y_{1:T}) = \mu_\theta(x_1) \prod_{t=1}^{T} g_\theta(y_t \mid x_t) \prod_{t=1}^{T-1} f_\theta(x_{t+1} \mid x_t). \tag{6}$$

This expression does not involve any marginalisation, whereas the observed data likelihood $p_\theta(y_{1:T})$ is found by averaging the joint distribution $p_\theta(x_{1:T}, y_{1:T})$ over all possible state sequences according to (5). Equivalently, we can express $p_\theta(y_{1:T})$ as in (4) where the one-step predictive likelihood can be written (using marginalisation) as,

$$p_\theta(y_t \mid y_{1:t-1}) = \int g_\theta(y_t \mid x_t) p_\theta(x_t \mid y_{1:t-1}) \mathrm{d}x_t. \tag{7}$$

These expressions highlight the tight relationship between the system identification problem and the state inference problem. A key challenge that will drive the developments in this work is how to deal with the latent states. For nonlinear system identification, the need for computational methods, such as SMC, is tightly coupled to the intractability of the integrals in (5) and (7).

To illustrate the strategies and algorithms introduced we will use them to solve two concrete problems, which are formulated below.

Example 1: Linear Gaussian model

Our first illustrative example is a simple linear Gaussian state space (LGSS) model, given by

$$x_{t+1} = 0.7x_t + v_t, \qquad v_t \sim \mathcal{N}(0, \theta^{-1}), \tag{8a}$$
$$y_t = 0.5x_t + e_t, \qquad e_t \sim \mathcal{N}(0, 0.1), \tag{8b}$$
$$(\theta \sim \mathcal{G}(0.01, 0.01)), \tag{8c}$$

where the unknown parameter θ corresponds to the *precision* (inverse variance) of the process noise v_t. Note that the prior for the Bayesian model is chosen as the Gamma ($\mathcal{G}$) distribution with known parameters, for reasons of simplicity, since this is the *conjugate prior* for this model. The initial distribution $\mu_\theta(x_1)$ is chosen as the stationary distribution of the state process. Identification of θ is based on a simulated data set consisting of $T = 100$ samples $y_{1:100}$ with true parameter $\theta_0 = 1$.

Example 2: Nonlinear non-Gaussian model

Our second example, involving real data, is related to a problem in paleoclimatology. Shumway and Stoffer [2011] considered the problem of modelling the thickness of ice varves (layers of sediments that are deposited from melting glaciers). The silt and sand that is deposited over one year makes up one varve and changes in the varve thicknesses indicates temperature changes. The data set that is used contains the thickness of 634 ice varves formed at a location in Massachusetts between years 9 883 and 9 250 BC. We make use of a nonlinear and non-Gaussian SSM proposed by Langrock [2011] to model this data:

$$x_{t+1} \,|\, x_t \sim \mathcal{N}(x_{t+1}; \phi x_t, \tau^{-1}), \tag{9a}$$
$$y_t \,|\, x_t \sim \mathcal{G}(y_t; 6.25, 0.256 \exp(-x_t)), \tag{9b}$$

with parameters $\theta = \{\phi, \tau\}$. The initial distribution $\mu_\theta(x_1)$ is chosen as the stationary distribution of the state process. The data set and a more complete description of the problem is provided by Shumway and Stoffer [2011].

3 Identification strategy – marginalisation

The marginalisation strategy amounts to solving the identification problem—either (2) or (3)—by computing the integral appearing in (7) (or, equivalently, (5)). That is, we *marginalize* (integrate out) the latent states $x_{1:T}$ and view θ as the only unknown quantity of interest.

In some special cases the marginalisation can be done exactly. In particular, for LGSS models the one-step predictive density in (7) is Gaussian and computable by running a Kalman filter. For the general case, however, some numerical approximation of the integral in (7) is needed. We will elaborate on

this in Section 6, where we investigate the possibility of using SMC to perform the marginalisation. For now, however, to illustrate the general marginalisation strategy, we will assume that the integral in (7) can be solved exactly.

3.1 ML identification via direct optimisation

Consider first the ML formulation (2). Direct optimisation (DO) amounts to working directly on the problem

$$\widehat{\theta}_{\mathrm{ML}} = \arg\max_{\theta\in\Theta} \sum_{t=1}^{T} \log \int g_\theta(y_t \mid x_t) p_\theta(x_t \mid y_{1:t-1}) \mathrm{d}x_t, \tag{10}$$

where we have rewritten the data log-likelihood using (4) and (7). Even though we momentarily neglect the difficulty in computing the integral above it is typically not possible to solve the optimisation problem (10) in closed form. Instead, we have to resort to numerical optimisation methods, see e.g. Nocedal and Wright [2006]. These methods typically find a maximiser of the log-likelihood function $\log p_\theta(y_{1:T})$ by iteratively refining an estimate θ_k of the maximiser $\widehat{\theta}_{\mathrm{ML}}$ according to

$$\theta_{k+1} = \theta_k + \alpha_k s_k. \tag{11}$$

Here, s_k denotes the search direction which is computed based on information about the cost function available from previous iterations. The step size α_k, tells us how far we should go in the search direction. The search direction is typically computed according to

$$s_k = H_k^{-1} g_k, \qquad g_k = \nabla_\theta \log p_\theta(y_{1:T})\big|_{\theta=\theta_k}, \tag{12}$$

where H_k denotes a positive definite matrix (e.g. the Hessian $\nabla^2_\theta \log p_\theta(y_{1:T})$ or approximations thereof) adjusting the gradient g_k.

Example 3: DO applied to the LGSS model

To apply the update in (10) for estimating θ in (8), we need to determine search direction s_k and the step lengths α_k. Here, we select the search direction as the gradient of the log-likelihood, i.e. H_k is the identity matrix. The log-likelihood for (8) can be expressed as

$$\log p_\theta(y_{1:T}) = \sum_{t=2}^{T} \log \mathcal{N}(y_t; 0.5\widehat{x}_{t|t-1}, P_{t|t-1} + 0.1), \tag{13}$$

where $\widehat{x}_{t|t-1}$ and $P_{t|t-1}$ denotes the predicted state estimate and its covariance obtained from a Kalman filter. The gradient g_k in (12) of the log-likelihood (13) can therefore be obtained by calculating $\nabla_\theta \widehat{x}_{t|t-1}$ and $\nabla_\theta P_{t|t-1}$, which can be obtained from the Kalman filter, using the so-called sensitivity derivatives introduced by Åström [1980]. In the upper part of Figure 1, we present the log-likelihood (blue) computed using the Kalman filter together with the estimate $\widehat{\theta}_{\mathrm{ML}}$ (orange) obtained by the DO algorithm.

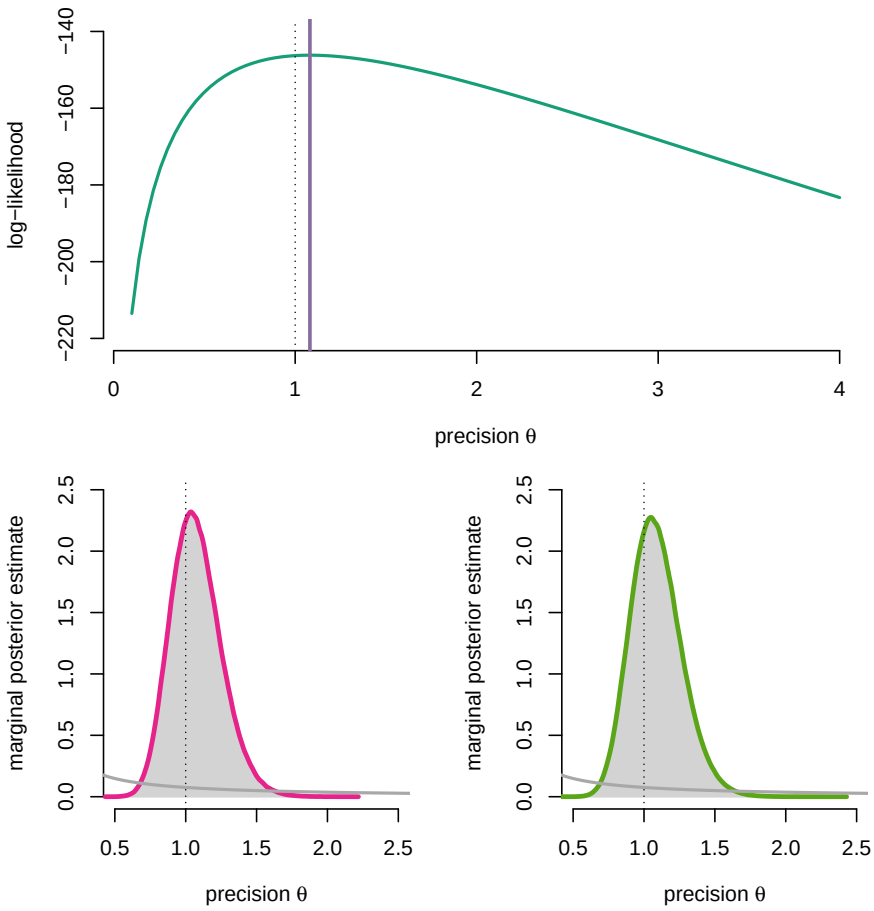

Figure 1: Upper: the log-likelihood estimates (green) together with the ML parameter estimates of the LGSS model obtain by the DO and the EM algorithms, respectively. The estimates sit on top of each other and are shown in blue. Lower: parameter posterior estimates obtained by the MH (left) and the Gibbs (right) algorithms, respectively. The vertical dashed lines indicate the true parameters of the model from which the data is generated and the dark grey lines indicate the prior density.

Within the DO method the use of SMC arise in evaluation of the cost function in (10) and its derivatives to compute the search directions s_k in (12).

3.2 Bayesian identification via Metropolis–Hastings

Let us now turn to the Bayesian formulation (3). As above, to illustrate the idea we consider first the simple case in which the marginalisation in (7) can be done exactly. Still, in most nontrivial cases the posterior PDF in (3) cannot be computed in closed form. The difficulty comes from the factor $p(y_{1:T})$, known as the *marginal likelihood*, which ensures that the posterior PDF is properly normalised (i.e., that it integrates to one). The marginal likelihood can be expressed as,

$$p(y_{1:T}) = \int p_\theta(y_{1:T})\pi(\theta)\mathrm{d}\theta. \tag{14}$$

Even if the likelihood $p_\theta(y_{1:T})$ is analytically tractable, we see that we need to carry out an integration w.r.t. θ. Furthermore, computing a point estimate, say the posterior mean of θ, also amounts to solving an integral

$$\mathrm{E}\left[\theta \mid y_{1:T}\right] = \int \theta\, p(\theta \mid y_{1:T})\mathrm{d}\theta, \tag{15}$$

which may also be intractable.

A generic way of approximating such intractable integrals, in particular those related to Bayesian posterior distributions, is to use a Monte Carlo method. In Section 5 we will discuss, in detail, how SMC can be used to approximately integrate over the *latents states* of the system. The sequential nature of SMC makes it particularly well suited for integrating over latent stochastic processes, such as the states in an SSM. However, to tackle the present problem of integrating over the *latent parameters* we shall consider a different class of methods denoted as Markov chain Monte Carlo (MCMC).

The idea behind MCMC is to simulate a Markov chain $\{\theta[m]\}_{m\geq 1}$. The chain is constructed in such a way that its *stationary* distribution coincides with the so-called *target* distribution of interest, here $p(\theta \mid y_{1:T})$. If, in addition, the chain is *ergodic*—essentially meaning that spatial averages coincide with "time" averages—the sample path from the chain can be used to approximate expectations w.r.t. to the target distribution:

$$\frac{1}{M-k+1}\sum_{m=k}^{M} \varphi(\theta[m]) \xrightarrow{\text{a.s.}} \int \varphi(\theta)p(\theta \mid y_{1:T})\mathrm{d}\theta, \tag{16}$$

as $M \to \infty$, for some test function φ. Here $\xrightarrow{\text{a.s.}}$ denotes almost sure convergence. Note that the first k samples have been neglected in the estimator (16), to avoid the transient phase of the chain. This is commonly referred to as the *burn-in* of the Markov chain.

Clearly, for this strategy to be useful we need a systematic way of constructing the Markov chain with the desired properties. One such way is to use the

Metropolis–Hastings (MH) algorithm. The MH algorithm uses an accept/reject strategy. At iteration $m+1$, we propose a new sample θ' according to,

$$\theta' \sim q(\cdot \mid \theta[m]), \tag{17}$$

where $q(\cdot)$ denotes a *proposal* distribution designed by the user and $\theta[m]$ denotes the sample from the previous iteration m. The newly proposed sample θ' will then be added to the sequence (i.e. $\theta[m+1] = \theta'$) with probability

$$\alpha \triangleq 1 \wedge \frac{p(\theta' \mid y_{1:T})}{p(\theta[m] \mid y_{1:T})} \frac{q(\theta[m] \mid \theta')}{q(\theta' \mid \theta[m])} \tag{18a}$$

$$= 1 \wedge \frac{p_{\theta'}(y_{1:T})\pi(\theta')}{p_{\theta[m]}(y_{1:T})\pi(\theta[m])} \frac{q(\theta[m] \mid \theta')}{q(\theta' \mid \theta[m])}, \tag{18b}$$

where $a \wedge b$ is used to denote $\min(a, b)$ and α is referred to as the acceptance probability. Hence, with probability $1-\alpha$ the newly proposed sample is not added to the sequence (i.e. rejected) and in that case the previous sample is once more added to the sequence $\theta[m+1] = \theta[m]$.

There exists a well established theory for the MH algorithm, which for example establish that it is ergodic and converges to the correct stationary distribution. This has been developed since the algorithm was first introduced by Metropolis et al. [1953] and Hastings [1970]. Ninness and Henriksen [2010] provides a nice account on the use of the MH algorithm for Bayesian system identification.

Example 4: MH applied to the LGSS model

To apply the MH algorithm for parameter inference in the LGSS model, we require a proposal distribution $q(\cdot)$ in (17) and to calculate the acceptance probability α in (18b). A standard choice for $q(\cdot)$ is a Gaussian random walk

$$q(\theta' \mid \theta[m]) = \mathcal{N}\left(\theta' \mid \theta[m], \sigma_q^2\right), \tag{19}$$

where σ_q^2 denotes the variance of the random walk. For this choice of proposal, the acceptance probability is

$$\alpha = 1 \wedge \frac{p_{\theta'}(y_{1:T})\pi(\theta')}{p_{\theta[m]}(y_{1:T})\pi(\theta[m])},$$

where q cancels since it is symmetric in θ. Note that the likelihood can be computed using the Kalman filter in analogue with (13) for the LGSS model. In the lower left part of Figure 1, we present the resulting posterior estimate obtained from running the algorithm $M = 20\,000$ iterations (discarding the first $10\,000$ as burn-in).

4 Identification strategy – Data augmentation

An alternative strategy to marginalisation is to make use of *data augmentation*. Intuitively, we can think of this strategy as a systematic way of separating *one* hard problem into *two* new and closely linked sub-problems, each of which is hopefully easier to solve than the original problem. For the SSM (1) these two problems amounts to

1. finding information about the state sequence $x_{1:T}$.
2. finding information about the parameters θ.

The state sequence $x_{1:T}$ is thus treated as an *auxiliary variable* that is estimated together with the parameters θ. Using the data augmentation terminology [Tanner and Wong, 1987, van Dyk and Meng, 2001], the state sequence $x_{1:T}$ is referred to as the *missing data*, as opposed to the *observed data* $y_{1:T}$. By augmenting the latter with the former, we obtain the *complete data* $\{x_{1:T}, y_{1:T}\}$.

Naturally, if the complete data were known, then identification of θ would have been much simpler. In particular, we can directly write down the complete data likelihood $p_\theta(x_{1:T}, y_{1:T})$ according to (6), which contrary to the observed data likelihood $p_\theta(y_{1:T})$, does not involve any marginalisation. The two likelihoods are related via (5), suggesting that the complete data likelihood can indeed be used for identifying the unknown parameters.

4.1 Expectation Maximisation (EM)

Analogously with the marginalisation strategy, we can make use of data augmentation to address both the frequentistic and the Bayesian identification problems. For the former identification criteria, (2), the result is the expectation maximisation (EM) algorithm [Dempster et al., 1977]. EM provides an iterative procedure to compute ML estimates of unknown parameters θ in probabilistic models involving latent variables, like the SSM in (1).

As a result of the conditional probability identity

$$p_\theta(x_{1:T}, y_{1:T}) = p_\theta(x_{1:T} \mid y_{1:T}) p_\theta(y_{1:T}),$$

we can relate the observed and complete data log-likelihoods as

$$\log p_\theta(y_{1:T}) = \log p_\theta(x_{1:T}, y_{1:T}) - \log p_\theta(x_{1:T} \mid y_{1:T}). \tag{20}$$

The EM algorithms operates by iteratively maximising the *intermediate quantity*

$$\mathcal{Q}(\theta, \theta') \triangleq \int \log p_\theta(x_{1:T}, y_{1:T}) p_{\theta'}(x_{1:T} \mid y_{1:T}) \mathrm{d}x_{1:T} \tag{21a}$$

$$= \mathrm{E}_{\theta'} \left[\log p_\theta(x_{1:T}, y_{1:T}) \mid y_{1:T} \right], \tag{21b}$$

according to:

(E) $\mathcal{Q}(\theta, \theta[k]) = \mathrm{E}_{\theta[k]}\left[\log p_\theta(x_{1:T}, y_{1:T}) \,|\, y_{1:T}\right]$,

(M) $\theta[k+1] = \arg\max_{\theta \in \Theta} \mathcal{Q}(\theta, \theta[k])$.

We can show that iterating the above Expectation (E) and Maximisation (M) steps implies

$$\mathcal{Q}(\theta, \theta') \geq \mathcal{Q}(\theta', \theta') \implies p_\theta(y_{1:T}) \geq p_{\theta'}(y_{1:T}). \tag{22}$$

Hence, $\{\theta[k]\}_{k\geq 1}$ will by construction result in a monotonic increase in likelihood values. Hence, the complete data log-likelihood $\log p_\theta(x_{1:T}, y_{1:T})$ can, via the intermediate quantity $\mathcal{Q}$ in (21), be used as a surrogate for the original observed data likelihood function $p_\theta(y_{1:T})$ in solving the ML problem (2). We are still required to compute the integral (21) and an important observation is now that we can approximate this integral (and its derivatives w.r.t. θ) using SMC.

Example 5: EM applied to the LGSS model

We need to compute the intermediate quantity to apply the EM algorithm for estimating the parameter in the LGSS model. For this model, we can write

$$\mathcal{Q}(\theta, \theta') \quad = \quad \text{const.} \; + \; \mathrm{E}_{\theta'}\left[\log \mu_\theta(x_1) + \sum_{t=2}^{T} \log \mathcal{N}(x_t; 0.7x_{t-1}, \theta^{-1}) \,\middle|\, y_{1:T}\right]$$

Note, that this expression results from that the parameter only is present in the latent state process.

We can directly maximise the intermediate quantity for θ since the system is linear in the parameters. By taking the gradient of $\mathcal{Q}(\theta, \theta')$, we obtain terms proportional to $E_{\theta'}\left[x_{t-1}(x_t - 0.7x_{t-1}) \,|\, y_{1:T}\right]$, where $\widehat{x}_{t|T}\widehat{x}_{t|T}$ and $\widehat{x}_{t|T}\widehat{x}_{t-1|T}$ denotes smoothed state estimates and $P_{t,t|T}$ and $P_{t-1,t|T}$ their covariances, respectively. These can be computed using a Kalman smoother and we refer the reader to Gibson and Ninness [2005] for the explicit expressions for implementing this. In the upper part of Figure 1, we present the parameter estimate $\widehat{\theta}_{\mathrm{ML}}$ (blue) obtained by the EM algorithm. We note that the parameter estimates obtained by the DO and EM algorithms are identical and overlapping. However, they differ from the true parameter due to the finite value of T.

4.2 Gibbs sampler

The Gibbs sampler is an MCMC method that produce samples from the joint distribution by alternatively sampling from its conditionals. Let us consider the Bayesian formulation, with the aim of computing (3). Inspired by the data augmentation strategy, start by assuming that the complete data $\{x_{1:T}, y_{1:T}\}$ is available. Bayes' theorem then results in

$$p(\theta \,|\, x_{1:T}, y_{1:T}) = \frac{p_\theta(x_{1:T}, y_{1:T})\pi(\theta)}{p(x_{1:T}, y_{1:T})}. \tag{23}$$

Intuitively, if the states $x_{1:T}$ were known, then the identification problem would be much simpler and, indeed, computing the posterior in (23) is typically easier than in (3). Firstly, the complete data likelihood is provided in (6), whereas the likelihood $p_\theta(y_{1:T})$ is intractable in the general case. Secondly, in many cases of interest it is possible to identify a prior for θ that is conjugate to the *complete data likelihood*, in which case the posterior (23) is available in closed form.

The problem with (23) is of course that it hinges upon the state sequence being known. However, assume that we can simulate a sample of the state trajectory $x_{1:T}$ from its conditional distribution given the observed data $y_{1:T}$ and the system parameters θ, i.e. from the joint smoothing distribution. Furthermore, assume that it is possible to sample from the distribution in (23). We can then implement the following algorithm: Initialise $\theta[0] \in \Theta$ arbitrarily and, for $m \geq 0$,

$$\text{Sample } x_{1:T}[m] \sim p_{\theta[m]}(x_{1:T} \mid y_{1:T}). \tag{24a}$$

$$\text{Sample } \theta[m+1] \sim p(\theta \mid x_{1:T}[m], y_{1:T}). \tag{24b}$$

This results in the generation of the following sequence of random variables

$$\theta[0], x_{1:T}[0], \theta[1], x_{1:T}[1], \theta[2], x_{1:T}[2], \ldots, \tag{25}$$

which forms a mutual Markov chain in the parameters θ and the states $x_{1:T}$, $\{\theta[m], x_{1:T}[m]\}_{m\geq 1}$. The procedure in (24) represents a valid MCMC method. More specifically, it is a particular instance of the so-called *Gibbs sampler*. The simulated Markov chain admits the *joint distribution* $p(\theta, x_{1:T} \mid y_{1:T})$ as a stationary distribution. Furthermore, under weak conditions it can be can be shown to be ergodic. That is, the Markov chain generated by the procedure (24) can be used to estimate posterior expectations, and the estimators are consistent in the sense of (16). Note that, if we are only interested in the marginal distribution $p(\theta \mid y_{1:T})$, then it is sufficient to store the sub-sequence constituted by $\{\theta[m]\}_{m\geq 1}$, obtained by simply discarding the samples $\{x_{1:T}[m]\}_{m\geq 1}$ from (25).

It is worth pointing out that it is possible to combine Gibbs sampling with other MCMC methods. For instance, if it is not possible to sample from posterior distribution (23) exactly, then a valid approach is to replace step (24b) of the Gibbs sampler with, e.g., an MH step with target distribution $p(\theta \mid x_{1:T}, y_{1:T})$. Similarly, for nonlinear state space models, the joint smoothing distribution in (24a) is not available in closed form, but it is still possible to implement the strategy above by sampling the state trajectory from an MCMC kernel targeting the joint smoothing distribution; see Section 8.2.

Example 6: Gibbs applied to the LGSS model

To implement the Gibbs sampler for parameter inference in the LGSS model, we need to sample from the conditional distributions in (24). To generate samples of state trajectories given θ and $y_{1:T}$, we can make use of the factorisation

$$p_\theta(x_{1:T} \mid y_{1:T}) = \left(\prod_{t=1}^{T-1} p_\theta(x_t \mid x_{t+1},\ y_{1:t}) \right) p_\theta(x_T \mid y_{1:T}) \tag{26}$$

of the joint smoothing distribution. Consequently, we can sample $x_{1:T}[m]$ using the following *backward simulation* strategy: Sample $\widetilde{x}_T \sim p_\theta(x_T \,|\, y_{1:T})$ and, for $t = T-1, \ldots, 1$, sample

$$\widetilde{x}_t \sim p_\theta(x_t \,|\, \widetilde{x}_{t+1},\ y_{1:t}) \propto p_\theta(\widetilde{x}_{t+1} \,|\, x_t) p_\theta(x_t \,|\, y_{1:t}). \tag{27}$$

We see that the backward simulation relies on the filtering distributions denoted by $\{p_\theta(x_t \,|\, y_{1:t})\}_{t=1}^T$ and, indeed, for the LGSS model we can obtain closed form expressions for all the involved densities by running a Kalman filter.

In the second step, (24b), we sample the parameters θ by

$$\theta \sim p(\theta \,|\, \widetilde{x}_{1:T}, y_{1:T}) = \mathcal{G}(\theta \,|\, \alpha, \beta), \tag{28a}$$

$$\alpha = 0.01 + \frac{T}{2}, \tag{28b}$$

$$\beta = 0.01 + \frac{1}{2}\left(0.51\widetilde{x}_1^2 + \sum_{t=1}^{T-1}(\widetilde{x}_{t+1} - 0.7\widetilde{x}_t)^2\right), \tag{28c}$$

which is the result of a standard prior-posterior update with a Gamma prior and Gaussian likelihood. The closed-form expression for $p(\theta \,|\, x_{1:T}, y_{1:T})$ is an effect of using a prior which is conjugate to the complete data likelihood.

In the lower right part of Figure 1, we present the resulting posterior estimate obtained from the Gibbs sampler with the same settings as for the MH algorithm. We note that the posterior estimates are almost identical for the two methods, which corresponds well to the established theory.

The data augmentation strategy (here implemented via the Gibbs sampler) enabled us to approximate the posterior distribution $p(\theta \,|\, y_{1:T})$ by separating the problem into two connected sub-problems (24).

5 Sequential Monte Carlo

Sequential Monte Carlo (SMC) methods offer numerical approximations to the state estimation problems associated with the nonlinear/non-Gaussian SSM (1). The particle filter (PF) approximates the filtering PDF $p_\theta(x_t \,|\, y_{1:t})$ and the particle smoother (PS) approximates the joint smoothing PDF $p_\theta(x_{1:t} \,|\, y_{1:t})$, or some of its marginals. The PF can intuitively be thought of as the equivalent of the Kalman filter for nonlinear/non-Gaussian SSMs.

The SMC approximation is an empirical distribution of the form

$$\widehat{p}_\theta(x_t \,|\, y_{1:t}) = \sum_{i=1}^N w_t^i \delta_{x_t^i}(x_t). \tag{29}$$

The samples $\{x_t^i\}_{i=1}^N$ are often referred to as *particles*—they are point-masses "spread out" in the state space, each particle representing one hypothesis about

the state of the system. We can think of each particle x_t^i as one possible system state and the corresponding weight w_t^i contains information about how probable that particular state is.

To make the connection to the marginalisation and data augmentation strategies introduced in the two previous sections clear, we remind ourselves where the need for SMC arise in implementing these strategies to identify θ in (1). The PF is used to compute the cost function (10) and its derivatives in order to find the search directions (12). To set up an MH samler, we can make use of a likelihood estimate provided by the PF in order to compute the acceptance probabilities in (18). When it comes to data augmentation strategies, the PS is used to approximate the intermediate quantity in (21) and in order to set up the Gibbs sampler, particle methods are used to draw a realisation from the joint smoothing distribution in (24a).

5.1 Particle filter

A principled solution to the nonlinear filtering problem is provided by the following two recursive equations:

$$p_\theta(x_t \mid y_{1:t}) = \frac{g_\theta(y_t \mid x_t) p_\theta(x_t \mid y_{1:t-1})}{p_\theta(y_t \mid y_{1:t-1})}, \tag{30a}$$

$$p_\theta(x_t \mid y_{1:t-1}) = \int f_\theta(x_t \mid x_{t-1}) p_\theta(x_{t-1} \mid y_{1:t-1}) \mathrm{d}x_{t-1}. \tag{30b}$$

These equations can only be solved in closed form for very specific special cases, e.g., the LGSS model which results in the Kalman filter. We will derive the particle filter as a general approximation of (30) for general nonlinear/non-Gaussian SSMs.

The particle filter—at least in its most basic form—can be interpreted as a sequential application of *importance sampling*. At each time step t we use importance sampling to approximate the filtering PDF $p_\theta(x_t \mid y_{1:t})$. This is made possible by using the expressions in (30) and by exploiting the already generated importance sampling approximation of $p_\theta(x_{t-1} \mid y_{1:t-1})$. At time $t = 1$ we can find an empirical distribution (29) by approximating $p_\theta(x_1 \mid y_1) \propto g_\theta(y_1 \mid x_1)\mu_\theta(x_1)$ using importance sampling in the normal sense. We sample independently the particles $\{x_1^i\}_{i=1}^N$ from some proposal distribution $r_\theta(x_1)$. To account for the discrepancy between the proposal distribution and the target distribution, the particles are assigned importance weights, given by the ratio between the target and the proposal (up to proportionality), i.e. $w_1^i \propto g_\theta(y_1 \mid x_1^i)\mu_\theta(x_1^i)/r_\theta(x_1^i)$, where the weights are normalised to sum to one.

We proceed in an inductive fashion and assume that we have an empirical approximation of the filtering distribution at time $t-1$ according to

$$\widehat{p}_\theta(x_{t-1} \mid y_{1:t-1}) = \sum_{i=1}^N w_{t-1}^i \delta_{x_{t-1}^i}(x_{t-1}). \tag{31}$$

Inserting this into (30b) results in

$$\begin{aligned}\widehat{p}_\theta(x_t \,|\, y_{1:t-1}) &= \int f_\theta(x_t \,|\, x_{t-1}) \sum_{i=1}^{N} w_{t-1}^i \delta_{x_{t-1}^i}(x_{t-1}) \mathrm{d}x_{t-1} \\ &= \sum_{i=1}^{N} w_{t-1}^i f_\theta(x_t \,|\, x_{t-1}^i). \end{aligned} \tag{32}$$

That is, we obtain a *mixture distribution* approximating $p_\theta(x_t \,|\, y_{1:t-1})$, where we have one mixture component for each of the N particles at time $t-1$. Furthermore, inserting (32) into (30a) results in the following approximation of the filtering PDF

$$p_\theta(x_t \,|\, y_{1:t}) \approx \frac{g_\theta(y_t \,|\, x_t)}{p_\theta(y_t \,|\, y_{1:t-1})} \sum_{i=1}^{N} w_{t-1}^i f_\theta(x_t \,|\, x_{t-1}^i). \tag{33}$$

The task is now to approximate (33) using importance sampling. Inspired by the structure of (33) we choose a proposal density (again denoted by r_θ) of the same form, namely as a mixture distribution,

$$r_\theta(x_t \,|\, y_{1:t}) \triangleq \sum_{i=1}^{N} \nu_{t-1}^i r_\theta(x_t \,|\, x_{t-1}^i, y_t), \tag{34}$$

where both the mixture components $r_\theta(x_t \,|\, x_{t-1}, y_t)$ and the mixture weights ν_{t-1} are design choices constituting parts of the proposal distribution.

To generate a sample from the mixture distribution (34) the following two-step procedure is used; first we randomly select one of the components, and then we generate a sample from that particular component. Note that we will sample N particles from the proposal distribution (34). Let us use a_t^i to denote the index of the mixture component selected for the i^{th} particle at time t. Now, since the probability of selecting component $r_\theta(x_t \,|\, x_{t-1}^j, y_t)$ is encoded by its weight ν_{t-1}^j, we have that

$$\mathbb{P}\left(a_t^i = j\right) = \nu_{t-1}^j, \qquad\qquad j = 1, \ldots, N. \tag{35}$$

Subsequently, we can generate a sample from the selected component a_t^i according to $x_t^i \sim r_\theta(x_t \,|\, \bar{x}_{t-1}^i, y_t)$, where $\bar{x}_{t-1}^i \triangleq x_{t-1}^{a_t^i}$. By construction x_t^i is now a sample from the proposal density (34). The particle $\bar{x}_{t-1}^i$ is referred to as the ancestor particle of x_t^i, since x_t^i is generated conditionally on $\bar{x}_{t-1}^i$. This also explains why the index a_t^i is commonly referred to as the *ancestor index*, since it indexes the ancestor of particle x_t^i at time $t-1$.

In practice we sample the N ancestor indices $\{a_t^i\}_{i=1}^N$ according to (35) in one go. This results in a new set of particles $\{\bar{x}_{t-1}^i\}_{i=1}^N$ that are subsequently used to propagate the particles to time t. This procedure, which (randomly) generates $\{\bar{x}_{t-1}^i\}_{i=1}^N$ by selection (sampling with replacement) from among $\{x_{t-1}^i\}_{i=1}^N$ according to some weights, is commonly referred to as *resampling*.

The next step is to assign importance weights to the new particles accounting for the discrepancy between the target distribution $p_\theta(x_t \mid y_{1:t})$ and the proposal distribution $r_\theta(x_t \mid y_{1:t})$. As before, the weights are computed as the ratio between the (unnormalised) target PDF and the proposal PDF. Direct use of (33) and (34) results in

$$\bar{w}_t^i = \frac{g_\theta(y_t \mid x_t^i) \sum_{j=1}^N w_{t-1}^j f_\theta(x_t^i \mid x_{t-1}^j)}{\sum_{j=1}^N \nu_{t-1}^j r_\theta(x_t^i \mid x_{t-1}^j, y_t)}. \tag{36}$$

By evaluating $\bar{w}_t^i$ for $i = 1, \ldots, N$ and normalising the weights, we obtain a new set of weighted particles $\{x_t^i, w_t^i\}_{i=1}^N$, constituting an empirical approximation of $p_\theta(x_t \mid y_{1:t})$. This completes the algorithm, since these weighted particles in turn can be used to approximate the filtering PDF at time $t+1$, then at time $t+2$ and so on.

A problem with the algorithm presented above is that the weight calculation in (36) has a computational complexity of $\mathcal{O}(N)$ for each particle, rendering an overall computational complexity of $\mathcal{O}(N^2)$, since the weights need to be computed for all N particles. A pragmatic solution to this problem is to use the freedom available in the proposal density and select it according to $r_\theta(x_t \mid y_{1:t}) = \sum_{j=1}^N w_{t-1}^j f_\theta(x_t \mid x_{t-1}^j)$. That is, we select ancestor particles with probabilities given by their importance weights and sample new particles by simulating the system dynamics from time $t-1$ to t. Inserting this into (36) results in the simple expression $\bar{w}_t^i = g_\theta(y_t \mid x_t^i)$, which brings the overall computational complexity down to $\mathcal{O}(N)$. The resulting algorithm is referred to as the *bootstrap particle filter* and it is summarised in Algorithm 1.

The bootstrap PF was the first working particle filter, an early and influential derivation is provided by Gordon et al. [1993]. It is arguably the simplest possible implementation of SMC, but nevertheless, it incorporates the essential methodological ideas that underpin the general SMC framework. *Importance sampling* (i.e. propagation and weighting in Algorithm 1 above) and *resampling* are used to sequentially approximate a sequence of probability distributions of interest; here $\{p_\theta(x_t \mid y_{1:t})\}_{t \geq 1}$.

Selecting the dynamics as proposal distribution, as in the bootstrap particle filter, is appealing due to the simplicity of the resulting algorithm. However,

Algorithm 1 Bootstrap particle filter (all operations are for $i = 1, \ldots, N$)

1: **Initialisation** ($t = 1$)**:**
2: Sample $x_1^i \sim \mu_\theta(x_1)$.
3: Compute $\bar{w}_1^i = g_\theta(y_1 \mid x_1^i)$, normalise, $w_1^i = \bar{w}_1^i / \sum_{j=1}^N \bar{w}_1^j$.
4: **for** $t = 2$ **to** T **do**
5: **Resampling:** Sample a_t^i with $\mathbb{P}\left(a_t^i = j\right) = w_t^j$.
6: **Propagation:** Sample $x_t^i \sim f_\theta(x_t \mid x_{t-1}^{a_t^i})$.
7: **Weighting:** Compute $\bar{w}_t^i = g_\theta(y_t \mid x_t^i)$ and normalise, $w_t^i = \bar{w}_t^i / \sum_{j=1}^N \bar{w}_t^j$.
8: **end**

this choice is unfortunately also suboptimal, since the current measurement y_t is not taken into account when simulating the particles $\{x_t^i\}_{i=1}^N$ from the proposal distribution. A better strategy of reducing the computational complexity of the weight computation from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ is to *target the joint distribution* of (x_t, a_t) with an importance sampler, instead of directly targeting the marginal distribution of x_t as was done above. Indeed, by explicitly introducing the ancestor indices as auxiliary variables in the importance sampler, we obtain the weight expression

$$\bar{w}_t^i = \frac{w_{t-1}^{a_t^i} g_\theta(y_t \mid x_t^i) f_\theta(x_t^i \mid x_{t-1}^{a_t^i})}{\nu_{t-1}^{a_t^i} r_\theta(x_t^i \mid x_{t-1}^{a_t^i}, y_t)}, \tag{37}$$

as a more practical alternative to (36). With this approach we have the possibility of freely selecting the mixture weights ν_{t-1} and mixture components $r_\theta(x_t \mid x_{t-1}, y_t)$ of the proposal, while still enjoying an overall linear computational complexity. The resulting algorithm is referred to as the auxiliary particle filter (APF). Rather than providing the details of the derivation we simply refer to the original paper by Pitt and Shephard [1999] or our complete derivation in Schön and Lindsten [2015].

5.2 Some useful properties of the PF

As any Monte Carlo algorithm, the PF can be interpreted as a *random number generator*. Indeed, the particles and the ancestor indices used within the algorithm are random variables, and executing the algorithm corresponds to simulating a realisation of these variables. It can be useful, both for understanding the properties of the PF and for developing more advanced algorithms around SMC, to make this more explicit. Let

$$\mathbf{x}_t \triangleq \{x_t^1, \ldots, x_t^N\}, \qquad \text{and} \qquad \mathbf{a}_t \triangleq \{a_t^1, \ldots, a_t^N\}, \tag{38}$$

refer to all the particles and ancestor indices, respectively, generated by the PF at time t. The PF in Algorithm 1 then generates a *single realisation* of a collection of random variables $\{\mathbf{x}_{1:T}, \mathbf{a}_{2:T}\} \in \mathsf{X}^{NT} \times \{1, \ldots, N\}^{N(T-1)}$. Furthermore, since we know how these variables are generated, we can directly write down their joint PDF[1] as,

$$\psi_\theta(\mathbf{x}_{1:T}, \mathbf{a}_{2:T} \mid y_{1:T}) \triangleq \prod_{i=1}^N \mu_\theta(x_1^i) \prod_{t=2}^T \left\{ \prod_{i=1}^N w_t^{a_t^i} f_\theta(x_t^i \mid x_{t-1}^{a_t^i}) \right\}. \tag{39}$$

Naturally, any estimator derived from the PF will also be a random variable. From (39) we note that the distribution of this random variable will depend on the number of particles N, and *convergence* of the algorithm can be identified with convergence, in some sense, of this random variable. Specifically, let $\varphi : \mathsf{X} \mapsto \mathbb{R}$ be some *test function* of interest. The posterior expectation

[1] w.r.t. to a natural product of Lebesgue and counting measure.

$\mathrm{E}_\theta\left[\varphi(x_t) \,|\, y_{1:t}\right] = \int \varphi(x_t) p_\theta(x_t \,|\, y_{1:t}) \mathrm{d}x_t$, can be estimated by the PF by computing (cf., (16)),

$$\widehat{\varphi}_t^N \triangleq \sum_{i=1}^N w_t^i \varphi(x_t^i). \tag{40}$$

There is a solid theoretical foundation for SMC, e.g., investigating the convergence of (40) to the true expectation as $N \to \infty$ and establishing non-asymptotic bounds on the approximation error. The (types of) existing theoretical results are too numerous to be mentioned here, and we refer to the book by Del Moral [2004] for a comprehensive treatment. However, to give a flavour of the type of results that can be obtained we state a central limit theorem (CLT) for the estimator (40). Under weak regularity assumptions it holds that [Del Moral and Miclo, 2000, Del Moral, 2004, Chopin, 2004],

$$\sqrt{N}\left(\widehat{\varphi}_t^N - \mathrm{E}_\theta\left[\varphi(x_t) \,|\, y_{1:t}\right]\right) \xrightarrow{\mathrm{d}} \mathcal{N}(0, \sigma_t^2(\varphi)), \tag{41}$$

as $N \to \infty$ where $\xrightarrow{\mathrm{d}}$ denotes convergence in distribution. The *asymptotic estimator variance* $\sigma_t^2(\varphi)$ depends on the test function φ, the specific PF implementation that is used and, importantly, the properties of the state space model (an explicit expression for $\sigma_t^2(\varphi)$ is given by Doucet and Johansen [2011]).

The CLT in (41) is reassuring since it reveals that the estimator converges at a rate $\sqrt{N}$, which is the same rate as for independent and identically distributed (i.i.d.) Monte Carlo estimators. An interesting question to ask, however, is how the asymptotic variance depends on t. In particular, recall from (33) that we use the approximation of the filtering distribution at time $t-1$, in order to construct the *target distribution*, which in turn is approximated by the particles at time t. This "approximation of an approximation" interpretation of the PF may, rightfully, lead to doubts about the stability of the approximation. In other words, will the asymptotic variance $\sigma_t^2(\varphi)$ grow exponentially with t?

Fortunately, in many realistic scenarios, the answer to this question is *no*. The key to this result is that the model exhibits some type of *forgetting*, essentially meaning that the dependence between states x_s and x_t diminishes (fast enough) as $|t-s|$ gets large. If this is the case, we can bound $\sigma_t^2(\varphi) \leq C$ for some constant C which is independent of t, ensuring the stability of the PF approximation. We refer to Del Moral and Guionnet [2001], Chopin [2004] for more precise results in this direction.

In analogy with the Kalman filter, the PF does not only compute the filtering distribution, but it also provides (an approximation of) the likelihood $p_\theta(y_{1:t})$, which is central to the system identification problem. For the bootstrap PF in Algorithm 1, this is given by,

$$\widehat{p}_\theta(y_{1:t}) = \prod_{s=1}^t \left\{ \frac{1}{N} \sum_{i=1}^N \bar{w}_s^i \right\}. \tag{42}$$

Note that the approximation is computed using the *unnormalised* importance weights $\{\bar{w}_s^i\}_{i=1}^N$. The expression (42) can be understood by considering the

factorisation (4) and noting that the one-step predictive likelihood, by (7), can be approximated by,

$$\begin{aligned}\widehat{p}_\theta(y_s \mid y_{1:s-1}) &= \int g_\theta(y_s \mid x_s)\widehat{p}_\theta(x_s \mid y_{1:s-1})\mathrm{d}x_s \\ &= \frac{1}{N}\sum_{i=1}^{N} g_\theta(y_s \mid x_s^i) = \frac{1}{N}\sum_{i=1}^{N} \bar{w}_s^i,\end{aligned}$$

where $\{x_s^i\}_{i=1}^N$ are simulated from the bootstrap proposal given by $r_\theta(x_s \mid y_{1:s}) = \widehat{p}_\theta(x_s \mid y_{1:s-1})$ (a similar likelihood estimator can be defined also for the general APF).

Sharp convergence results are available also for the likelihood estimator (42). First of all, the estimator is *unbiased*, i.e. $\mathrm{E}_{\psi_\theta}\left[\widehat{p}_\theta(y_{1:t})\right] = p_\theta(y_{1:t})$ for any value of N, where the expectation is w.r.t. the randomness of the PF [Pitt et al., 2012, Del Moral, 2004]. We will make use of this result in the sequel. Furthermore, the estimator is convergent as $N \to \infty$. In particular, under similar regularity and forgetting conditions as mentioned above, it is possible to establish a CLT at rate $\sqrt{N}$ also for (42). Furthermore, the asymptotic variance for the normalised likelihood estimator can be bounded by $D \cdot t$ for some constant D. Hence, in contrast with the filter estimator (40), the asymptotic variance for (42) will grow with t, albeit only linearly. However, the growth can be controlled by selecting $N \propto t$, which provides a useful insight into the tuning of the algorithm if it is to be used for likelihood estimation.

5.3 Particle smoother

The PF was derived as a means of approximating the sequence of filtering densities $\{p_\theta(x_t \mid y_{1:t})\}_{t\geq 1}$. We can also start from the forward smoothing relation

$$p_\theta(x_{1:t} \mid y_{1:t}) = p_\theta(x_{1:t-1} \mid y_{1:t-1})\frac{f_\theta(x_t \mid x_{t-1})g_\theta(y_t \mid x_t)}{p_\theta(y_t \mid y_{1:t-1})}, \tag{43}$$

and derive the particle filter as a means of approximating the sequence of *joint smoothing densities* $\{p_\theta(x_{1:t} \mid y_{1:t})\}_{t\geq 1}$. Interestingly, the resulting algorithm is equivalent to the PF that we have already seen. Indeed, by using the ancestor indices we can trace the genealogy of the filter particles to get full state trajectories, resulting in the approximation

$$\widehat{p}_\theta(x_{1:t} \mid y_{1:t}) = \sum_{i=1}^{N} w_t^i \delta_{x_{1:t}^i}(x_{1:t}). \tag{44}$$

However, there is a serious limitation in using the PF as a solution to the smoothing problem, known as *path degeneracy*. It arises due to the fact that the resampling step, by construction, will remove particles with small weights and duplicate particles with high weight. Hence, each resampling step will typically reduce the number of unique particles. An inevitable results of this is

that for any given time s there exists $t > s$ such that the PF approximation of $p_\theta(x_{1:t} \mid y_{1:t})$ collapses to a single particle at time s.

One solution to the path degeneracy problem is to propagate information backwards in time, using a forward/backward smoothing technique. The joint smoothing distribution can be factorised as in (26) where each factor depends only on the *filtering distribution* (cf. (27)). Since the filter can be approximated without (directly) suffering from path degeneracy, this opens up for a solution to the path degeneracy problem. An important step in this direction was provided by Godsill et al. [2004], who made use of *backward simulation* to simulate complete state trajectories $\widetilde{x}_{1:T}$, approximately distributed according to the joint smoothing distribution $p_\theta(x_{1:T} \mid y_{1:T})$. The idea has since then been refined, see e.g. Douc et al. [2011], Bunch and Godsill [2013]. Algorithms based on the combination of MCMC and SMC introduced by Andrieu et al. [2010], resulting in the particle MCMC (PMCMC) methods, also offer promising solutions to the nonlinear state smoothing problem. For a self-contained introduction to particle smoothers, see Lindsten and Schön [2013].

6 Marginalisation in the nonlinear SSM

Now that we have seen how SMC can be used to approximate the filtering distribution, as well as the predictive and smoothing distributions and the likelihood, we are in the position of applying the general identification strategies outlined in the previous sections to identify nonlinear/non-Gaussian state space models.

6.1 Direct optimisation using Fisher's identity

Consider the maximum likelihood problem in (2). The objective function, i.e. the log-likelihood, can be approximated by SMC by using (42). However, many standard optimisation methods requires not only evaluation of the cost function, but also the gradient and possibly the Hessian, in solving (2). SMC can be used to compute the gradient via the use of Fisher's identity,

$$\nabla_\theta \log p_\theta(y_{1:T})\big|_{\theta=\theta_k} = \nabla_\theta \mathcal{Q}(\theta, \theta_k)\big|_{\theta=\theta_k}, \tag{45}$$

where the intermediate quantity $\mathcal{Q}$ was defined in (21). It follows that

$$\nabla_\theta \log p_\theta(y_{1:T}) = \mathrm{E}_\theta \left[\nabla_\theta \log p_\theta(x_{1:T}, y_{1:T}) \mid y_{1:T}\right]. \tag{46}$$

That is, the gradient of the log-likelihood can be computed by solving a smoothing problem. This opens up for gradient approximations via a particle smoother, as discussed in Section 5.3; see e.g. Poyiadjis et al. [2011] for further details. The Hessian can also be approximated using, for example, Louis' identity [e.g., Cappé et al., 2005].

Note that the gradient computed in this way will be stochastic, since it is approximated by an SMC method. It is therefore common to choose a diminishing step-size sequence of the gradient ascent method according to standard

stochastic approximation rules; see e.g., Kushner and Yin [1997], Benveniste et al. [1990]. However, it should be noted that the approximation of the gradient of the log-likelihood will be *biased* for a finite number of particles N, and the identification method therefore relies on asymptotics in N for convergence to a maximiser of (2).

6.2 Using unbiased likelihoods within MH

We can make use of the likelihood estimator (42) also for Bayesian identification of nonlinear SSMs via the MH algorithm. Indeed, an intuitive idea is to simply replace the intractable likelihood in the acceptance probability (18) by the (unbiased) estimate $\widehat{p}_\theta(y_{1:T})$. What is maybe less intuitive is that this simple idea does in fact result in a valid (in the sense that it has $p(\theta \mid y_{1:T})$ as its stationary distribution) MH algorithm, for any number of particles $N \geq 1$. Let us now sketch why this is the case.

We start by introducing a (high-dimensional) auxiliary variable u constituted by all the random quantities generated by the PF, i.e. $u \triangleq \{\boldsymbol{x}_{1:T}, \boldsymbol{a}_{2:T}\}$ distributed according to $\psi_\theta(u \mid y_{1:T})$ defined in (39). Note that the joint distribution of the parameters θ and the auxiliary variables u,

$$p(\theta, u \mid y_{1:T}) = \psi_\theta(u \mid y_{1:T}) p(\theta \mid y_{1:T}) \tag{47a}$$

$$= \frac{p_\theta(y_{1:T}) \psi_\theta(u \mid y_{1:T}) \pi(\theta)}{p(y_{1:T})}, \tag{47b}$$

has the original target distribution $p(\theta \mid y_{1:T})$ as one of its marginals. Inspired by (47b), consider the following *extended target* distribution

$$\phi(\theta, u \mid y_{1:T}) = \frac{\widehat{p}_{\theta,u}(y_{1:T}) \psi_\theta(u \mid y_{1:T}) \pi(\theta)}{p(y_{1:T})}, \tag{48}$$

where we have made use of the unbiased likelihood estimate $\widehat{p}_{\theta,u}(y_{1:T})$ from the PF (and indicate explicitly the dependence on u in the notation for clarity). We can now set up a *standard MH algorithm* that operates in the (huge) *non-standard extended space* $\Theta \times \mathsf{X}^{NT} \times \{1, \ldots, N\}^{N(T-1)}$ approximating the extended target distribution (48). The resulting algorithm will generate samples asymptotically from $p(\theta \mid y_{1:T})$ despite the fact that we employ an *approximate* likelihood in (48)! To understand why this is the case, let us marginalise (48) w.r.t. the auxiliary variable u:

$$\int \phi(\theta, u \mid y_{1:T}) \mathrm{d}u = \frac{\pi(\theta)}{p(y_{1:T})} \int \widehat{p}_{\theta,u}(y_{1:T}) \psi_\theta(u \mid y_{1:T}) \mathrm{d}u. \tag{49}$$

The fact that the likelihood estimate $\widehat{p}_{\theta,u}(y_{1:T})$ produced by the PF is unbiased means that

$$\mathrm{E}_{u|\theta}\left[\widehat{p}_{\theta,u}(y_{1:T})\right] = \int \widehat{p}_{\theta,u}(y_{1:T}) \psi_\theta(u \mid y_{1:T}) \mathrm{d}u = p_\theta(y_{1:T}). \tag{50}$$

Algorithm 2 Particle Metropolis Hastings (PMH) for Bayesian system identification of nonlinear SSMs

1: Run a PF (Algorithm 1) targeting $p(x_{1:T} \mid \theta[1])$ to obtain $u' \sim \psi_{\theta[1]}(u \mid y_{1:T})$ and $\widehat{p}_{\theta[1],u'}(y_{1:T})$ according to (42).
2: **for** $m = 1$ to M **do**
3: Sample $\theta' \sim q(\cdot \mid \theta[m])$.
4: Run a PF (Algorithm 1) targeting $p(x_{1:T} \mid \theta')$ to obtain $u' \sim \psi_{\theta'}(u \mid y_{1:T})$ and $\widehat{p}_{\theta',u'}(y_{1:T})$ according to (42).
5: Sample $d_m \sim \mathcal{U}[0, 1]$.
6: Compute the acceptance probability α by (52).
7: **if** $d_m < \alpha$ **then**
8: $\theta[m + 1] \leftarrow \theta'$ and $\widehat{p}_{\theta[m+1]}(y_{1:T}) \leftarrow \widehat{p}_{\theta'}(y_{1:T})$.
9: **else**
10: $\theta[m + 1] \leftarrow \theta[m]$ and $\widehat{p}_{\theta[m+1]}(y_{1:T}) \leftarrow \widehat{p}_{\theta[m]}(y_{1:T})$.
11: **end if**
12: **end for**

The marginalisation in (49) can now be finalised, resulting in $\int \phi(\theta, u \mid y_{1:T})\mathrm{d}u = p(\theta \mid y_{1:T})$, proving that $p(\theta \mid y_{1:T})$ is recovered *exactly* as the marginal of the extended target distribution (48), despite the fact that we employed a PF *approximation* of the likelihood using a finite number of particles N. This explains why it is sometimes referred to as an *exact approximation*. An interpretation is that using the likelihood estimate from the PF does change the marginal distribution w.r.t. u in (47), but it does *not* change the marginal w.r.t. θ.

Based on the current sample $(\theta[m], u[m])$ a new sample (θ', u') is proposed according to

$$\theta' \sim q(\cdot \mid \theta[m]), \qquad u' \sim \psi_{\theta'}(\cdot \mid y_{1:T}). \tag{51}$$

We emphasise that simulation of u' corresponds to running a PF with the model parameterised by θ'. The probability of accepting the sample proposed in (51) as the next sample $(\theta[m + 1], u[m + 1])$ is given by

$$\alpha = 1 \wedge \frac{\widehat{p}_{\theta',u'}(y_{1:T})\pi(\theta')}{\widehat{p}_{\theta[m],u[m]}(y_{1:T})\pi(\theta[m])} \frac{q(\theta[m] \mid \theta')}{q(\theta' \mid \theta[m])}, \tag{52}$$

which was obtained by inserting (48) and (51) into (18). In practice it is sufficient to keep track of the likelihood estimates $\{\widehat{p}_{\theta[m],u[m]}\}_{m\geq 1}$, and we do not need to store the complete auxiliary variable $\{u[m]\}_{m\geq 1}$. The above development is summarised in Algorithm 2. It can be further improved by incorporating gradient and Hessian information about the posterior into the proposal (51), resulting in more efficient use of the generated particles [Dahlin et al., 2015].

The *particle Metropolis Hastings* algorithm constitutes one member of the *particle MCMC (PMCMC)* family of algorithms introduced in the seminal paper by Andrieu et al. [2010]. The derivation above is along the lines of the pseudo-marginal approach due to Andrieu and Roberts [2009]. The extended target construction ϕ, however, is the core of all PMCMC methods and they differ in

that different (more or less standard) MCMC samplers are used for this (non-standard) target distribution. They also have in common that SMC is used as a proposal mechanism on the space of state trajectories X^T.

Example 7: PMH applied to the NL-SSM

We make use of Algorithm 2 to estimate the parameters in (9) together with a simple Gaussian random walk,

$$\theta' \sim q(\cdot \mid \theta[m]) = \mathcal{N}(\theta[m], 2.562^2\Sigma/2),$$

where Σ denotes an estimate of the posterior covariance matrix. This choice is optimal for some target distributions as is discussed by Sherlock et al. [2013]. The posterior covariance estimate is obtained as

$$\Sigma = 10^{-5} \begin{bmatrix} 22.51 & -4.53 \\ -4.53 & 2.57 \end{bmatrix}$$

using a pilot run of the algorithm. In the upper part of Figure 2, we present the resulting marginal posterior estimates. The posterior means $\widehat{\theta}_{\text{PMH}} = \{0.95, 51.05\}$ are indicated by dotted lines.

7 Data augmentation in nonlinear SSM

Algorithms implementing the data augmentation strategy treats the states as auxiliary variables that are estimated along with the parameters, rather than integrating them out. Intuitively this results in algorithms that alterate between updating θ and $x_{1:T}$.

7.1 Expectation maximisation

The expectation maximisation algorithm introduced in Section 4.1 separates the maximum likelihood problem (2) into two closely linked problems, namely the computation of the intermediate quantity $\mathcal{Q}(\theta, \theta[k])$ and its maximisation w.r.t. θ. As previously discussed, computing the intermediate quantity corresponds to solving a smoothing problem. Hence, for a nonlinear/non-Gaussian SSM, a natural idea is to use a particle smoother, as discussed in Section 5.3, to solve this subproblem. The details of the algorithm are provided by Cappé et al. [2005], Olsson et al. [2008], Schön et al. [2011], whereas the general idea of making use of Monte Carlo integration to approximate the E-step dates back to Wei and Tanner [1990].

By this approach, a completely new set of simulated particles has to be generated at each iteration of the algorithm, since we continuously update the value of θ. Once an approximation of $\mathcal{Q}(\theta, \theta[k])$ has been computed, the current particles are discarded and an entirely new set has to be generated at the next iteration. While it does indeed result in a working algorithm it makes for an

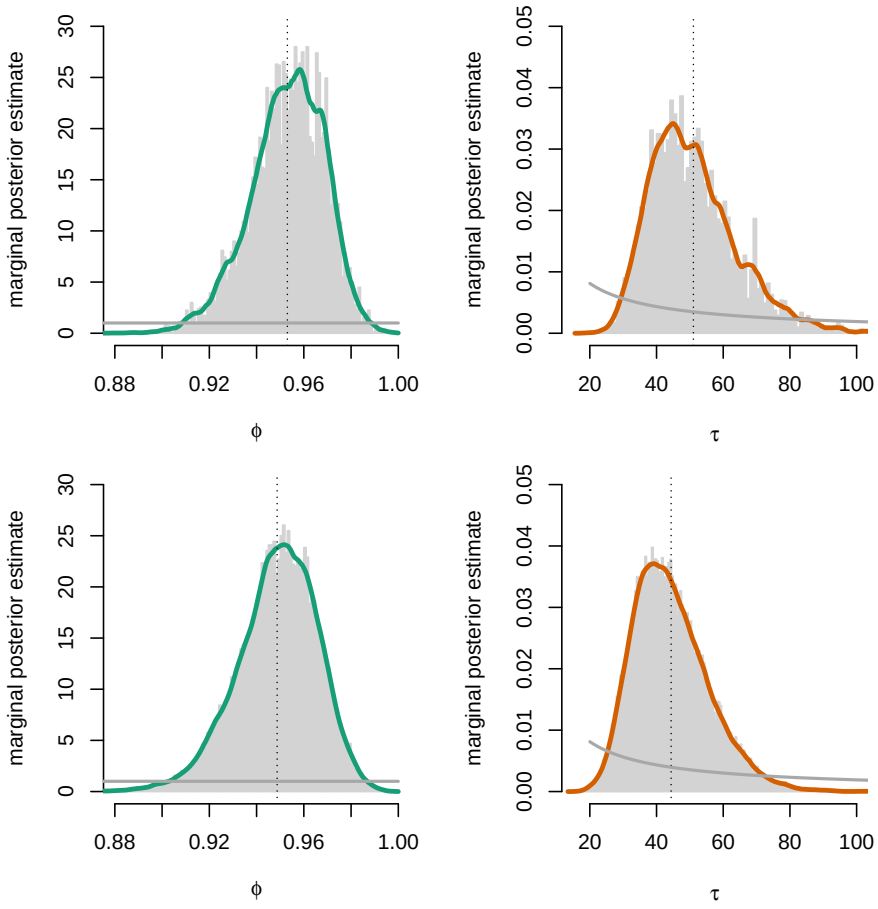

Figure 2: The marginal posterior estimates for ϕ (left) and τ (right) using the PMH algorithm (upper) and the PGAS algorithm (lower). The dotted vertical and the dark grey lines indicate the estimated posterior mean and the prior densities, respectively.

inefficient use of the particles. The PMCMC family of algorithms opens up for the construction of Markov kernels that can be used to generate samples of the state trajectory (to be used in the approximation of $\mathcal{Q}(\theta, \theta[k])$) in a computationally more efficient fashion, which serves as one (of several) motivation of the subsequent development.

7.2 Sampling state trajectories using Markov kernels

We now introduce another member of the PMCMC family of algorithms (recall PMH from Section 6.2) that can be used whenever we are faced with the problem of sampling from an intractable joint smoothing distribution $p_\theta(x_{1:T} \mid y_{1:T})$. In those situations an exact sample can be replaced with a draw from an MCMC kernel with stationary distribution $p_\theta(x_{1:T} \mid y_{1:T})$, without introducing any systematic error, and PMCMC opens up for using SMC to construct such MCMC kernels.

Here, we review a method denoted as particle Gibbs with ancestor sampling (PGAS), introduced by Lindsten et al. [2014]. To construct the aforementioned Markov kernel, PGAS makes use of a procedure reminiscent of the PF in Algorithm 1. The only difference is that in PGAS we condition on the event that an *a priori* specified state x'_t is always present in the particle system, for each time t. Hence, the states $(x'_1, \ldots, x'_T)$ must be retained throughout the sampling prodecure. To accomplish this we sample x_t^i according to the bootstrap PF only for $i = 1, \ldots, N-1$. The remaining N^{th} particle x_t^N is then set deterministically as $x_t^N = x'_t$. It is often the case that we are interested in complete particles trajectories; cf., (44). To generate a genealogical trajectory for x'_t, it is possible to connect it to one of the particles at time $t-1$, $\{x_{t-1}^i\}_{i=1}^N$ by sampling a value for the corresponding ancestor index a_t^N from its conditional distribution. This is referred to as *ancestor sampling*, see Algorithm 3.

Note that Algorithm 3 takes as input a state trajectory $x'_{1:T} = (x'_1, \ldots, x'_T)$ and returns another state trajectory $x^\star_{1:T}$, which is simulated randomly accord-

Algorithm 3 PGAS kernel (with a bootstrap PF)

1: **Initialisation ($t = 1$):** Draw $x_1^i \sim \mu(x_1)$ for $i = 1, \ldots, N-1$ and set $x_1^N = x'_1$. Compute $\bar{w}_1^i = g_\theta(y_t \mid x_1^i)$ for $i = 1, \ldots, N$.
2: **for** $t = 2$ to T **do**
3: Sample a_t^i with $\mathbb{P}\left(a_t^i = j\right) = w_{t-1}^j$ for $i = 1, \ldots, N-1$.
4: Sample $x_t^i \sim f_\theta(x_t \mid x_{t-1}^{a_t^i})$ for $i = 1, \ldots, N-1$.
5: Set $x_t^N = x'_t$.
6: Draw a_t^N with $\mathbb{P}\left(a_t^N = j\right) \propto \bar{w}_{t-1}^j f_\theta(x'_t \mid x_{t-1}^j)$.
7: Set $x_{1:t}^i = \{x_{1:t-1}^{a_t^i}, x_t^i\}$ for $i = 1, \ldots, N$.
8: Compute $\bar{w}_t^i = g_\theta(y_t \mid x_t^i)$ for $i = 1, \ldots, N$.
9: **end for**
10: Draw k with $\mathbb{P}(k = i) \propto \bar{w}_T^i$.
11: **Return** $x^\star_{1:T} = x^k_{1:T}$.

ing to *some* distribution (which, however, cannot be written on closed form). Hence, we can view Algorithm 3 as sampling from a Markov kernel defined on the space of state trajectories X^T. This Markov kernel is referred to as the PGAS kernel. The usefulness of the method comes from the fact that the PGAS kernel is a valid MCMC kernel for the joint smoothing distribution $p_\theta(x_{1:T} \mid y_{1:T})$ for any number of particles $N \geq 2$! A detailed derivation is provided by Lindsten et al. [2014], who show that the PGAS kernel is ergodic and that it admits the joint smoothing distribution as its unique stationary distribution. This implies that the state trajectories generated by PGAS can be used as samples from the joint smoothing distribution. Hence, the method is indeed an interesting alternative to other particle smoothers. Moreover, the PGAS kernel can be used as a component in any (standard) MCMC method. In the subsequent section we will make explicit use of this, both for ML and Bayesian identification.

8 Identification using Markov kernels

8.1 Expectation maximisation revisited

In Section 7.1 we made use of particle smoothers to approximate the intractable integral defining the intermediate quantity $\mathcal{Q}(\theta, \theta[m])$. However, it is possible to make more efficient use of the simulated variables by using the PGAS Algorithm 3 and employing a *stochastic approximation* update of the intermediate quantity $\mathcal{Q}$,

$$\widehat{\mathcal{Q}}_k(\theta) = (1-\alpha_k)\widehat{\mathcal{Q}}_{k-1}(\theta) + \alpha_k \sum_{i=1}^{N} w_T^i \log p_\theta(x_{1:T}^i, y_{1:T}), \tag{53}$$

where α_k is the step size and $\{w_T^i, x_{1:T}^i\}_{i=1}^N$ is generated by Algorithm 3. Stochastic approximation EM (SAEM) was introduced and analysed by Delyon et al. [1999] and it was later realised that it is possible to use MCMC kernels within SAEM [Andrieu et al., 2005] (see also Benveniste et al. [1990]). The aforementioned *particle SAEM* algorithm for nonlinear system identification was presented by Lindsten [2013] and it is summarised in Algorithm 4.

Algorithm 4 PGAS for ML sys. id. of nonlinear SSMs

1: **Initialisation:** Set $\theta[0]$ and $x_{1:T}[0]$ arbitrarily. Set $\widehat{\mathcal{Q}}_0 = 0$.
2: **for** $k \geq 1$ **do**
3: Run Algorithm 3 with $x'_{1:T} = x_{1:T}[k-1]$. Set $x_{1:T}[k] = x^\star_{1:T}$.
4: Compute $\widehat{\mathcal{Q}}_k(\theta)$ according to (53).
5: Compute $\theta[k] = \arg\max\, \widehat{\mathcal{Q}}_k(\theta)$.
6: **if** convergence criterion is met **then**
7: **return** $\theta[k]$
8: **end if**
9: **end for**

Algorithm 5 PGAS for Bayesian sys. id. of nonlinear SSMs

1: **Initialisation:** Set $\theta[0]$ and $x_{1:T}[0]$ arbitrarily.
2: **for** $m = 1$ to M **do**
3: Run Algorithm 3 conditionally on $(x_{1:T}[m-1], \theta[m-1])$ and set $x_{1:T}[m] = x^\star_{1:T}$.
4: Draw $\theta[m] \sim p(\theta \,|\, x_{1:T}[m], y_{1:T})$.
5: **end for**

Note the important difference between the SMC-based EM algorithm outlined in Section 7.1 and Algorithm 4. In the former we generate a completely new set of particles at each iteration, whereas in particle SAEM *all* simulated particles contribute, but they are down-weighted using a forgetting factor given by the step size. This approach is more efficient in practice, since we can use much fewer particles at each iteration. In fact, the method can be shown to converge to a maximiser of (2) even when using a fixed number of particles $N \geq 2$ when executing Algorithm 4.

8.2 Bayesian identification

Gibbs sampling can be used to simulate from the posterior distribution (3) or more generally, the joint state and parameter posterior $p(\theta, x_{1:T} \,|\, y_{1:T})$. The PGAS kernel allows us to sample the complete state trajectory $x_{1:T}$ in one block. Due to the invariance and ergodicity properties of the PGAS kernel, the validity of the Gibbs sampler is not violated. We summarise the procedure in Algorithm 5.

Example 8: PGAS applied to (9)

To make use of Algorithm 5 to estimate the parameters in (9), we need to simulate from the conditional distribution $\theta[m] \sim p(\theta \,|\, x_{1:T}[m], y_{1:T})$. This distribution is not available in closed form, however we can generate samples from it by using rejection sampling with the following instrumental distribution

$$
\begin{aligned}
&q(\phi, \tau | x_{1:T}[m], y_{1:T}) = \mathcal{G}\left(\tau; \alpha, \beta\right) \mathcal{N}\left(\phi; \tilde{\mu}, \tilde{\tau}^{-1}\right), \\
&\alpha = 0.01 + \frac{T-1}{2}, \\
&\beta = 0.01 + \frac{1}{2}\sum_{t=1}^{T} x_t[m]^2 - \frac{1}{2}\frac{\left(\sum_{t=1}^{T-1} x_{t+1}[m] x_t[m]\right)^2}{\sum_{t=2}^{T-1} x_t[m]^2}, \\
&\tilde{\mu} = \frac{\sum_{t=1}^{T-1} x_{t+1}[m] x_t[m]}{\sum_{t=2}^{T-1} x_t[m]^2}, \; \tilde{\tau} = \tau \sum_{t=2}^{T-1} x_t[m]^2.
\end{aligned}
$$

In the lower part of Figure 2, we present the resulting marginal posterior estimates. The posterior means $\widehat{\theta}_{\mathrm{PG}} = \{0.953, 44.37\}$ are indicated by dotted lines.

9 Future challenges

We end this tutorial by pointing out directions for future research involving interesting challenges where we believe that SMC can open up for significant developments.

Over two decades ago the SMC development started by providing a solution to the intractable filtering problem inherent in the nonlinear SSM. We have since then seen that SMC in indeed much more widely applicable and we strongly believe that this development will continue, quite possibly at a higher pace. This development opens up entirely new arenas where we can use SMC to solve hard inference problems. To give a few concrete examples of this we have the Bayesian nonparametric models (such as the Dirichlet and the Beta processes) that are extensively used in machine learning. There are also the so-called *spatio-temporal* models, which do not only have structure in time, but also in a spatial dimension, imagine the weather forecasting problem. A known deficiency of the standard (bootstrap) particle filter is its inability to handle high-dimensional variables x_t [Bickel et al., 2008], which is usually the case in for example spatio-temporal models. However, some very recent work has shown promising directions to tackle high-dimensional models in a consistent way using SMC [Naesseth et al., 2014, Beskos et al., 2014, Naesseth et al., 2015].

There is a well-known (but underutilised) duality between the control problem and the model learning problem. Coupled with the various SMC based approximations this opens up for fundamentally new controllers to be learnt by formulating the policy optimisation in control problems as an inference problem. For some work along this direction, see e.g. [Doucet et al., 2010, Hoffman et al., 2009, Toussaint and Storkey, 2006].

The PMCMC family of methods that have been discussed and used throughout this tutorial is a concrete example of another interesting trend, namely that of coupling various sampling methods into more powerful solutions. This is a trend that will continue to evolve. The online (Bayesian) inference problem is also a future challenge where we believe that SMC will play an important role.

A Implementation details

This appendix provides additional implementation details and clarifications about the numerical illustrations given in the paper.

A.1 Linear Gaussian state space model

The LGSS model studied in Example 1 is given by:

$$x_{t+1} = 0.7x_t + v_t, \qquad v_t \sim \mathcal{N}(0, \theta^{-1}), \tag{54a}$$

$$y_t = x_t + e_t, \qquad e_t \sim \mathcal{N}(0, 0.1), \tag{54b}$$

$$(\theta \sim \mathcal{G}(0.01, 0.01)), \tag{54c}$$

where the unknown parameter θ corresponds to the *precision* of the process noise v_t (i.e., θ^{-1} is the process noise variance). Note that the prior for the Bayesian model is chosen as the Gamma ($\mathcal{G}$) distribution with know parameters for reasons of simplicity (it provides positive realizations and it is the conjugate prior). Specifically, $\mathcal{G}(a, b)$ denotes a Gamma distribution with shape a and rate b such that the mean is a/b:

$$\mathcal{G}(\theta; a, b) = \frac{b^a \theta^{a-1} \exp(-b\theta)}{\Gamma(a)}. \tag{55}$$

The state process is assumed to be stationary. This implies that the distribution of the initial state (i.e. the state at time $t = 1$) is given by,

$$p(x_1 \,|\, \theta) = \mathcal{N}(x_1; 0, \{(1 - 0.7^2)\theta\}^{-1}) = \mathcal{N}(0, \{0.51\theta\}^{-1}).$$

Identification of θ is based on a simulated data set consisting of $T = 100$ samples $y_{1:100}$ with true parameter $\theta_0 = 1$.

A.1.1 Marginalization

Direct optimization The log-likelihood for the LGSS model is given by

$$\begin{aligned} V(\theta) &= \log p_\theta(y_{1:T}) = \log \prod_{t=1}^{T} p_\theta(y_t \,|\, y_{1:t-1}) = \sum_{t=1}^{T} \log p_\theta(y_t \,|\, y_{1:t-1}) \\ &= \sum_{t=1}^{T} \log \mathcal{N}(y_t; \widehat{x}_{t\,|\,t-1}, \underbrace{P_{t\,|\,t-1} + 0.1}_{\triangleq \Lambda_t}) \\ &= \sum_{t=1}^{T} \left[-\frac{1}{2} \log 2\pi - \frac{1}{2} \log \Lambda_t - \frac{1}{2\Lambda_t} (y_t - \widehat{x}_{t\,|\,t-1})^2 \right] \\ &= -\frac{T}{2} \log 2\pi - \frac{1}{2} \sum_{t=1}^{T} \left[\log \Lambda_t + \frac{1}{\Lambda_t} (y_t - \widehat{x}_{t\,|\,t-1})^2 \right] \end{aligned} \tag{56}$$

where $\widehat{x}_{t\,|\,t-1}$ is the optimal predictor and $P_{t\,|\,t-1}$ is the covariance of the prediction error. These quantities can be computed by the Kalman filter via the following recursions:

$$\Lambda_t = P_{t\,|\,t-1} + 0.1 \tag{57a}$$

$$K_t = 0.7 P_{t\,|\,t-1} \Lambda_t^{-1} \tag{57b}$$

$$\widehat{x}_{t+1\,|\,t} = 0.7 \widehat{x}_{t\,|\,t-1} + K_t (y_t - \widehat{x}_{t\,|\,t-1}) \tag{57c}$$

$$P_{t+1\,|\,t} = 0.49 P_{t\,|\,t-1} + \theta^{-1} - 0.7 K_t P_{t\,|\,t-1} \tag{57d}$$

initialized with $\widehat{x}_{1\,|\,0} = 0$ and $P_{1\,|\,0} = (0.51\theta)^{-1}$, the mean and covariance of x_1, the state at time $t = 1$.

The gradient of the objective function becomes

$$\begin{aligned}\frac{d}{d\theta}V(\theta) &= -\frac{1}{2}\sum_{t=1}^{T}\left[\frac{d}{d\theta}\log\Lambda_t + \frac{d}{d\theta}\frac{1}{\Lambda_t}(y_t - \widehat{x}_{t\,|\,t-1})^2\right] \\ &= -\frac{1}{2}\sum_{t=1}^{T}\left[\frac{1}{\Lambda_t}\frac{d\Lambda_t}{d\theta} - \frac{2}{\Lambda_t}(y_t - \widehat{x}_{t\,|\,t-1})\frac{d\widehat{x}_{t\,|\,t-1}}{d\theta} - \frac{1}{\Lambda_t^2}(y_t - \widehat{x}_{t\,|\,t-1})^2\frac{d\Lambda_t}{d\theta}\right],\end{aligned}$$

where

$$\frac{d\Lambda_t}{d\theta} = \frac{dP_{t\,|\,t-1}}{d\theta}.$$

In order to compute the gradient, we need to compute $\frac{d}{d\theta}\widehat{x}_{t\,|\,t-1}$ and $\frac{d}{d\theta}P_{t\,|\,t-1}$. This can be done recursively by differentiating (57) with respect to θ. We get

$$\frac{\mathrm{d}K_t}{\mathrm{d}\theta} = \frac{0.7}{\Lambda_t}\left(1 - \frac{P_{t\,|\,t-1}}{\Lambda_t}\right)\frac{\mathrm{d}P_{t\,|\,t-1}}{\mathrm{d}\theta}, \tag{58a}$$

$$\frac{\mathrm{d}\widehat{x}_{t+1\,|\,t}}{\mathrm{d}\theta} = (0.7 - K_t)\frac{\mathrm{d}\widehat{x}_{t\,|\,t-1}}{\mathrm{d}\theta} + (y_t - \widehat{x}_{t\,|\,t-1})\frac{\mathrm{d}K_t}{\mathrm{d}\theta}, \tag{58b}$$

$$\frac{\mathrm{d}P_{t+1\,|\,t}}{\mathrm{d}\theta} = (0.49 - 0.7K_t)\frac{\mathrm{d}P_{t\,|\,t-1}}{\mathrm{d}\theta} - \frac{1}{\theta^2} - 0.7P_{t\,|\,t-1}\frac{\mathrm{d}K_t}{\mathrm{d}\theta}. \tag{58c}$$

For a more complete treatment of this problem, see [Åström, 1980].

Metropolis Hastings The first task in setting up an MH algorithm to compute $p(\theta \,|\, y_{1:T})$ is to decide on which proposal distribution to use. For simplicity, let us make use of a random walk

$$q(\theta' \,|\, \theta[m]) = \mathcal{N}\left(\theta' \,|\, \theta[m], 0.1\right). \tag{59}$$

Now that we can propose new samples according to (59), the probability of accepting the new sample θ' has to be computed. The prior $\pi(\cdot)$ and the proposal $q(\cdot)$ are given by (54c) and (59), respectively. Hence, it remains to compute the likelihood. The log-likelihood, denoted by $V(\theta)$ (to agree with the previous section), is given by (56) and it can be computed by running the Kalman filter (57). The resulting expression for the acceptance probability is thus given by

$$\begin{aligned}\alpha &= 1 \wedge \frac{p_{\theta'}(y_{1:T})\pi(\theta')q(\theta[m] \,|\, \theta')}{p_{\theta[m]}(y_{1:T})\pi(\theta[m])q(\theta' \,|\, \theta[m])} \\ &= 1 \wedge \frac{p_{\theta'}(y_{1:T})\pi(\theta')}{p_{\theta[m]}(y_{1:T})\pi(\theta[m])} \\ &= 1 \wedge \exp(V(\theta') - V(\theta[m]) - 0.99\log(\theta'/\theta[m]) - 0.01(\theta' - \theta[m])),\end{aligned} \tag{60}$$

where the first equality follows from the fact that the random walk proposal (59) is symmetric in θ' and $\theta[m]$, and the second equality follows from (55) and (56). Note also that the prior π is only supported on positive values on θ, so if a negative value is proposed it is automatically rejected ($\alpha = 0$).

A.1.2 Data augmentation

Expectation Maxmimisation In the problem setting, $x_{1:T}$ act as the latent variables. In order to apply the EM algorithm, we need to calculate the following surrogate function

$$\mathcal{Q}(\theta, \theta[k]) = \mathbb{E}_{\theta[k]} \log p_\theta(x_{1:T}, y_{1:T}). \tag{61}$$

Expanding the right hand side of Eq. (61) gives that

$$\begin{aligned}\mathcal{Q}(\theta, \theta[k]) &= \mathbb{E}_{\theta[k]}\left(\log p_\theta(x_1)p(y_1|x_1)\prod_{t=2}^{t=T} p_\theta(x_t|x_{t-1})p(y_t|x_t)\right) \\ &= \mathbb{E}_{\theta[k]}\left(\log p_\theta(x_1) + \sum_{t=2}^{T}\log p_\theta(x_t|x_{t-1}) + \sum_{t=1}^{T}\log p(y_t|x_t)\right)\end{aligned}$$

In the following, we will drop the terms which are independent of θ and use the linearity of the expectation operator, which gives that

$$\mathcal{Q}(\theta, \theta[k]) = \frac{1}{2}\left\{T\log(\theta) - \theta\left[0.51\mathbb{E}_{\theta[k]}(x_1^2) + \sum_{t=1}^{T-1}\mathbb{E}_{\theta[k]}\big((x_{t+1} - 0.7x_t)^2\big)\right]\right\} + \text{const.}$$

We see that we need to compute expectations w.r.t. the smoothing distribution (for the model parameterised by $\theta[k]$), which can be done by running any convenient Kalman smoother.

Next, the M step amounts to maximising $\mathcal{Q}(\theta, \theta[k])$ with respect to θ. In our case, this maximisation has a closed form solution, given by

$$\theta[k+1] = \frac{T}{0.51\mathbb{E}_{\theta[k]}(x_1^2) + \sum_{t=1}^{T-1}\mathbb{E}_{\theta[k]}((x_{t+1} - 0.7x_t)^2)}. \tag{62}$$

Gibbs The Gibbs sampler iterates between simulating $x_{1:T}$ from $p_\theta(x_{1:T} \,|\, y_{1:T})$ and θ from $p(\theta \,|\, x_{1:T}, y_{1:T})$. Simulating $x_{1:T}$ is done by *backward sampling* as shown in Eq. (27). The filtering densities $p_\theta(x_t \,|\, y_{1:t}) = N(x_t \,|\, \widehat{x}_{t|t}, P_{t|t})$ are computed by running a Kalman filter. We then obtain the following expression for the backward kernel:

$$p_\theta(x_t \,|\, y_{1:t}, \widetilde{x}_{t+1}) = N(x_t \,|\, \mu_t, \Sigma_t),$$

with

$$\begin{aligned}\mu_t &= \widehat{x}_{t|t} + 0.7P_{t|t}\left(\frac{1}{\theta} + 0.49P_{t|t}\right)^{-1}(\widetilde{x}_{t+1} - 0.7\widehat{x}_{t|t}), \\ \Sigma_t &= P_{t|t} - 0.49P_{t|t}^2\left(\frac{1}{\theta} + 0.49P_{t|t}\right)^{-1}.\end{aligned}$$

As for the conditional distribution of θ: due to the fact that the Gamma distribution is the conjugate prior for the precision in a Gaussian model, we obtain a closed form expression for,

$$
\begin{aligned}
p(\theta \,|\, x_{1:T}, y_{1:T}) &= p(\theta \,|\, x_{1:T}) \propto p(x_{1:T} \,|\, \theta) p(\theta) = p(\theta) p(x_1 \,|\, \theta) \prod_{t=1}^{T-1} p(x_{t+1} \,|\, x_t, \theta) \\
&\propto \theta^{a-1} \exp(-b\theta) \sqrt{\theta} \exp\left(-\frac{0.51\theta}{2} x_1^2 \right) \times \prod_{t=1}^{T-1} \sqrt{\theta} \exp\left(-\frac{\theta}{2} (x_{t+1} - 0.7 x_t)^2 \right) \\
&= \theta^{a + \frac{T}{2} - 1} \exp\left(-\left(b + \frac{0.51}{2} x_1^2 + \frac{1}{2} \sum_{t=1}^{T-1} (x_{t+1} - 0.7 x_t)^2 \right) \theta \right) \\
&\propto \mathcal{G}\left(\theta; a + \frac{T}{2}, b + \frac{1}{2} \left(0.51 x_1^2 + \sum_{t=1}^{T-1} (x_{t+1} - 0.7 x_t)^2 \right) \right).
\end{aligned}
$$

Note that we have use proportionality, rather than equality, in several of the steps above. However, since we know that $p(\theta \,|\, x_{1:T}, y_{1:T})$ is a PDF (i.e., it integrates to one), it is sufficient to obtain an expression which is proportional to a Gamma PDF (the penultimate line). By normalisation we then obtain that $p(\theta \,|\, x_{1:T}, y_{1:T})$ is indeed given by the Gamma PDF on the final line.

The above derivation can straightforwardly be generalized to derive a Gibbs sampler for a general LGSS model, the details are provided by Wills et al. [2012].

A.2 Nonlinear example

Example 2 is borrowed from Shumway and Stoffer [2011] (see pages 63, 131, 151, 270, 280). Consider a data set consisting of 634 measurements of the thickness of ice varves (the layers of clay collected in glaciers) formed at a location in Massachusetts between years $9,883$ and $9,250$ BC. The data is modelled using a nonlinear state space model given by,

$$
\begin{aligned}
x_{t+1} | x_t &\sim \mathcal{N}(x_{t+1}; \phi x_t, \tau^{-1}), && \text{(63a)} \\
y_t | x_t &\sim \mathcal{G}(y_t; 6.25, 0.256 \exp(-x_t)), && \text{(63b)}
\end{aligned}
$$

with the parameters $\theta = (\phi, \tau)^\mathsf{T}$. The system is assumed to be stable and the state process stationary. This implies that the distribution of the initial state (i.e. the state at time $t = 1$) is given by,

$$
p(x_1 \,|\, \theta) = \mathcal{N}(x_1; 0, \{(1 - \phi^2)\tau\}^{-1}).
$$

In the Bayesian setting, we use a uniform prior for ϕ to reflect the stability assumption, and a conjugate Gamma prior for τ:

$$
\begin{aligned}
p(\phi) &= \mathcal{U}(\phi; -1, 1), \\
p(\tau) &= \mathcal{G}(\tau; 0.01, 0.01).
\end{aligned}
$$

Identification of θ is based on the measured data set consisting of $T = 634$ samples $y_{1:634}$.

Algorithm 6 Gradient-based maximum likelihood inference in NL-SSMs

INPUTS: K (no. iterations), $y_{1:T}$ (data), θ_0 (initial parameter), γ and α (step length sequence).
OUTPUTS: $\widehat{\theta}$ (est. of parameter).

1: Initialise the parameter estimate $\widehat{\theta}_0 = \theta_0$ and set $k = 1$.
2: **while** $k \leq N$ or until convergence **do**
3: Run the FFBSi smoother at $\widehat{\theta}_{k-1}$ to obtain $\nabla_\theta \log \widehat{p}_\theta(y_{1:T})$.
4: Apply the update $\widehat{\theta}_k = \widehat{\theta}_{k-1} + \gamma \cdot k^{-\alpha} \nabla_\theta \log \widehat{p}_\theta(y_{1:T})$.
5: Set $k = k + 1$.
6: **end while**
7: Set $\widehat{\theta} = \widehat{\theta}_k$.

A.2.1 Marginalization

Direct optimization – particle based gradient ascent For this implementation, we make use of the approach from Poyiadjis et al. [2011], which is summarised in Algorithm 6. To improve the numerical performance, the inference is carried out over the transformed parameters $\tilde{\phi} = \tanh^{-1}(\phi)$ and $\tilde{\tau} = \log(\tau)$. Hence, the two parameters are now unconstrained and these types of transformations can often result in beneficial variance reduction.

The gradient of the log-likelihood $\nabla_\theta \log p_\theta(y_{1:T})_{|\theta=\theta_{k-1}}$ is estimated by the Fisher identity using the fast forward-filtering backward-smoother (FFBSi) with early stopping as discussed by Taghavi et al. [2013]. We make use of 500 forward particles, 100 backward trajectories and rejection sampling for 75 trajectories. For the Fisher identity, we require calculating the gradients of the complete data log-likelihood (with respect to the transformed parameters). Note first that the complete data log-likelihood is given by

$$\begin{aligned}\log p_\theta(x_{1:T}, y_{1:T}) &= \log p_\theta(x_1) + \sum_{t=1}^{T-1} \log p_\theta(x_{t+1} \,|\, x_t) + \text{const.} \\ &= \frac{1}{2}\left\{ \log((1-\phi^2)\tau) - (1-\phi^2)\tau x_1^2 + \sum_{t=1}^{T-1} \left(\log\tau - \tau(x_{t+1} - \phi x_t)^2\right) \right\} + \text{const.}\end{aligned} \tag{64}$$

We thus get,

$$\begin{aligned}\frac{\partial}{\partial \tilde{\phi}} \log p_\theta(x_{1:T}, y_{1:T}) &= \frac{\partial}{\partial \phi} \left\{\log p_\theta(x_{1:T}, y_{1:T})\right\} \left(\frac{d\tilde{\phi}}{d\phi}\right)^{-1} \\ &= -\phi + (1-\phi^2)\tau \left\{ x_1^2 + \sum_{t=1}^{T-1} x_t(x_{t+1} - \phi x_t) \right\},\end{aligned}$$

where we have used the fact that $\frac{d}{d\phi}\tanh^{-1}(\phi) = (1-\phi^2)^{-1}$. Furthermore, we

have,

$$\begin{aligned}\frac{\partial}{\partial \tilde{\tau}} \log p_\theta(x_{1:T}, y_{1:T}) &= \frac{\partial}{\partial \tau} \left\{\log p_\theta(x_{1:T}, y_{1:T})\right\} \left(\frac{d\tilde{\tau}}{d\tau}\right)^{-1} \\ &= \frac{1}{2}\left\{T - \tau(1-\phi^2)x_1^2 - \tau \sum_{t=1}^{T-1}(x_{t+1} - \phi x_t)^2\right\}.\end{aligned}$$

The optimisation is initialised in (untransformed) parameters $\{\phi, \tau\} = \{0.95, 10\}$ with $\alpha = -2/3$, $\gamma = 0.01$ and runs for $K = 250$ iterations.

Metropolis Hastings – PMH The sampler is implemented in two steps. In the first step the smooth particle filter [Malik and Pitt, 2011] is used with 500 particles to get an initialisation of the parameters and to estimate the Hessian of the log-likelihood. The optimisation of the log-likelihood is done using a bounded limited-memory BFGS optimizer and the Hessian is estimated numerically using a central finite difference scheme. The resulting estimates of the parameters and inverse Hessian are

$$\widehat{\theta}_{\mathrm{ML}} = \{0.95, 0.02\} \qquad \widehat{\mathcal{I}}(\widehat{\theta}_{\mathrm{ML}}) = 10^{-5}\begin{bmatrix} 9.30 & 2.96 \\ 2.96 & 1.99 \end{bmatrix}.$$

The PMH0 algorithm is initialised in $\widehat{\theta}_{\mathrm{ML}}$ and makes use of the bootstrap particle filter with 1 000 particles to estimate the log-likelihood. The proposal is selected using the rules-of-thumb in Sherlock et al. [2013] as

$$q(\theta''|\theta') = \mathcal{N}(\theta''; \theta', (2.562^2/2)\widehat{\mathcal{I}}(\widehat{\theta}_{\mathrm{ML}})).$$

We use 15 000 iterations (discarding the first 2 000 iterations as burn-in) to estimate the posteriors and their means.

A.2.2 Data augmentation

Expectation Maximisation – PSAEM We outline the implementation details for the Particle SAEM algorithm (see Section 8.1), but the implementation for the PSEM algorithm (see Section 7.1) follows similarly.

Note that particle SAEM requires us to compute an approximation of the $\mathcal{Q}$-function according to (53). The complete data log-likelihood can be written as (see (64)),

$$\begin{aligned}&\log p_\theta(x_{1:T}, y_{1:T}) \\ &\quad = \frac{1}{2}\left\{\log((1-\phi^2)\tau) - (1-\phi^2)\tau x_1^2 + \sum_{t=1}^{T-1}\left(\log\tau - \tau(x_{t+1} - \phi x_t)^2\right)\right\} + \text{const.}\end{aligned}$$

If we define the complete data *sufficient statistics* as $\mathcal{S} := (\Psi, \Phi, \Sigma, X)^\mathsf{T} \in \mathbb{R}^4$ with

$$\Psi = \frac{1}{T-1}\sum_{t=1}^{T-1} x_{t+1}x_t, \quad \Phi = \frac{1}{T-1}\sum_{t=2}^{T} x_t^2, \quad \Sigma = \frac{1}{T-1}\sum_{t=1}^{T-1} x_t^2, \quad X = x_1^2,$$

we can thus write $\log p_\theta(x_{1:T}, y_{1:T}) = -0.5 f(\theta; \mathcal{S}) + \text{const.}$, where the function f is defined as:

$$\begin{aligned} f(\theta; \mathcal{S}) := &- \log((1-\phi^2)\tau) + X(1-\phi^2)\tau \\ &+ (T-1)\left\{-\log\tau + \tau(\Phi - 2\Psi\phi + \phi^2\Sigma)\right\}. \end{aligned} \tag{65}$$

Expressing the complete data log-likelihood in terms of its sufficient statistics in this way is useful, since it allows us to write the approximation of the $\mathcal{Q}$-function in (53) as:

$$\widehat{\mathcal{Q}}_k(\theta) = -0.5 f(\theta; \widehat{\mathcal{S}}_k) + \text{const.},$$

where $\widehat{\mathcal{S}}_k = (\widehat{\Psi}_k, \widehat{\Phi}_k, \widehat{\Sigma}_k, \widehat{X}_k)^\mathsf{T}$ is a stochastic approximation of the sufficient statistics, computed recursively as

$$\begin{aligned} \widehat{\Psi}_k &= (1-\alpha_k)\widehat{\Psi}_{k-1} + \frac{\alpha_k}{T-1}\sum_{t=1}^{T-1}\left(\sum_{i=1}^N w_T^i[k] x_{t+1}^i[k] x_t^i[k]\right), \\ \widehat{\Phi}_k &= (1-\alpha_k)\widehat{\Phi}_{k-1} + \frac{\alpha_k}{T-1}\sum_{t=2}^{T}\left(\sum_{i=1}^N w_T^i[k] (x_t^i[k])^2\right), \\ \widehat{\Sigma}_k &= (1-\alpha_k)\widehat{\Sigma}_{k-1} + \frac{\alpha_k}{T-1}\sum_{t=1}^{T-1}\left(\sum_{i=1}^N w_T^i[k] (x_t^i[k])^2\right), \\ \widehat{X}_k &= (1-\alpha_k)\widehat{X}_{k-1} + \alpha_k \sum_{i=1}^N w_T^i[k] (x_1^i[k])^2, \end{aligned}$$

where $\{x_{1:T}^i[k], w_T^i[k]\}_{i=1}^N$ are the particle trajectories generated by the PGAS algorithm at iteration k.

Maximising $\widehat{\mathcal{Q}}_k(\theta)$ in the M-step of the algorithm is thus equivalent to minimising $f(\theta; \widehat{\mathcal{S}}_k)$. Let us therefore turn to the problem of minimising $f(\theta; \mathcal{S})$ for an arbitrary (but fixed) value of the sufficient statistics $\mathcal{S}$. First, noting that the leading two terms in (65) originate from the initial condition, which should have a negligible effect on the maximising argument for large T, a good initialisation for the maximisation can be obtained by approximating

$$f(\theta; \mathcal{S}) \approx (T-1)\left\{-\log\tau + \tau(\Phi - 2\Psi\phi + \phi^2\Sigma)\right\}.$$

Indeed, minimising this approximation can be done on closed form, suggesting that

$$\begin{aligned} \phi_{\text{opt.}} &\approx \Psi/\Sigma \\ \tau_{\text{opt.}} &\approx (\Phi - \Psi^2/\Sigma)^{-1}. \end{aligned}$$

This provides us with a good initialisation for a numerical optimisation method which can be used to minimise $f(\theta; \mathcal{S})$ to desired precision.

PGAS In the PGAS algorithm for Bayesian inference we employ a Gibbs sampler, iteratively simulating $x_{1:T}$ from the PGAS kernel, and θ from the conditional distribution $p(\theta \mid x_{1:T}, y_{1:T})$. This distribution is given by

$$p(\phi, \tau | x_{1:T}, y_{1:T}) = \frac{1}{Z} \tau^{a+\frac{T}{2}-1} e^{-\tilde{b}\tau} \mathbb{1}_{\{|\phi| \leq 1\}} \sqrt{1-\phi^2} e^{-\frac{\tau}{2}(\phi-\tilde{\mu})^2}, \tag{66}$$

where the constants are given as follows:

$$\tilde{b} = b + \frac{1}{2} \sum_{t=1}^{T} x_t^2 - \frac{1}{2} \frac{\left(\sum_{t=1}^{T-1} x_{t+1} x_t \right)^2}{\sum_{t=2}^{T-1} x_t^2}, \tag{67}$$

$$\tilde{\tau} = \tau \sum_{t=2}^{T-1} x_t^2, \tag{68}$$

$$\tilde{\mu} = \frac{\sum_{t=1}^{T-1} x_{t+1} x_t}{\sum_{t=2}^{T-1} x_t^2}. \tag{69}$$

Simulating from (66) is done by rejection sampling with an instrumental distribution,

$$q(\phi, \tau | x_{1:T}, y_{1:T}) = \mathcal{G}\left(\tau; a + \frac{T-1}{2}, \tilde{b}\right) \mathcal{N}\left(\phi; \tilde{\mu}, \tilde{\tau}^{-1}\right). \tag{70}$$

Specifically, we propose a draw (ϕ', τ') from the instrumental distribution and accept this as a draw from (66) with probability $\mathbb{1}_{\{|\phi| \leq 1\}} \sqrt{1-\phi^2}$.

References

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

C. Andrieu, A. Doucet, S. S. Singh, and V. B. Tadić. Particle methods for change detection, system identification, and contol. *Proceedings of the IEEE*, 92(3):423–438, March 2004.

C. Andrieu, E. Moulines, and P. Priouret. Stability of stochastic approximation under verifiable conditions. *SIAM Journal on Control and Optimization*, 44 (1):283–312, 2005.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72(3):269–342, 2010.

K.J. Åström. Maximum likelihood and prediction error methods. *Automatica*, 16(5):551–574, 1980.

A Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*. Springer-Verlag, New York, USA, 1990.

A. Beskos, D. Crisan, A. Jasra, K. Kamatani, and Y. Zhou. A stable particle filter in high-dimensions. *ArXiv:1412.3501*, December 2014.

P. Bickel, B. Li, and T. Bengtsson. *Sharp failure rates for the bootstrap particle filter in high dimensions*, volume Volume 3 of *Collections*, pages 318–329. Institute of Mathematical Statistics, Beachwood, Ohio, USA, 2008.

P. Bunch and S. Godsill. Improved particle approximations to the joint smoothing distribution using Markov chain Monte Carlo. *IEEE Transactions on Signal Processing*, 61(4):956–963, 2013.

O. Cappé, E. Moulines, and T. Rydén. *Inference in Hidden Markov Models*. Springer, New York, NY, USA, 2005.

N. Chopin. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *The Annals of Statistics*, 32(6):2385–2411, 2004.

J. Dahlin, F. Lindsten, and T. B. Schön. Particle Metropolis Hastings using gradient and Hessian information. *Statistics and Computing*, 25(1):81–92, 2015.

P. Del Moral. *Feynman-Kac Formulae - Genealogical and Interacting Particle Systems with Applications*. Probability and its Applications. Springer, 2004.

P. Del Moral and A. Guionnet. On the stability of interacting processes with applications to filtering and genetic algorithms. *Annales de l'Institut Henri Poincaré*, 37(2):155–194, 2001.

P. Del Moral and L. Miclo. Branching and interacting particle systems approximations of Feynman-Kac formulae with applications to non-linear filtering. In J. Azéma, M. Ledoux, M. Émery, and M. Yor, editors, *Séminaire de Probabilités XXXIV*, Lecture Notes in Mathematics, pages 1–145. Springer, 2000.

B. Delyon, M. Lavielle, and E. Moulines. Convergence of a stochastic approximation version of the EM algorithm. *The Annals of Statistics*, 27(1):94–128, 1999.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

R. Douc, A. Garivier, E. Moulines, and J. Olsson. Sequential Monte Carlo smoothing for general state space hidden Markov models. *Annals of Applied Probability*, 21(6):2109–2145, 2011.

A. Doucet and A. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In D. Crisan and B. Rozovskii, editors, *The Oxford Handbook of Nonlinear Filtering*, pages 656–704. Oxford University Press, Oxford, UK, 2011.

A. Doucet, A. M. Johansen, and V. B. Tadić. On solving integral equations using Markov chain Monte Carlo methods. *Applied Mathematics and Computation*, 216:2869–2880, 2010.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

S. Gibson and B. Ninness. Robust maximum-likelihood estimation of multivariable dynamic systems. *Automatica*, 41(10):1667–1682, 2005.

S. J. Godsill, A. Doucet, and M. West. Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association*, 99(465):156–168, March 2004.

N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proc. on Radar and Sig. Proc.*, volume 140, pages 107–113, 1993.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

M. Hoffman, H. Kueck, N. de Freitas, and A. Doucet. New inference strategies for solving Markov decision processes using reversible jump MCMC. In *Proceedings of the 25th conference on uncertainty in artificial intelligence (UAI)*, Corvallis, OR, USA, June 2009.

N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin. On particle methods for parameter estimation in state-space models. arXiv:1412.8695, submitted to Statistical Science, December 2014.

G. Kitagawa. A Monte Carlo filtering and smoothing method for non-Gaussian nonlinear state space models. In *Proceedings of the 2nd U.S.-Japan joint seminar on statistical time series analysis*, pages 110–131, Honolulu, Hawaii, jan 1993.

H. J. Kushner and G. G. Yin. *Stochastic approximation algorithms and applications*. Springer, 1997.

R. Langrock. Some applications of nonlinear and non-Gaussian state-space modelling by means of hidden Markov models. *Journal of Applied Statistics*, 38(12):2955–2970, 2011.

F. Lindsten. An efficient stochastic approximation EM algorithm using conditional particle filters. In *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.

F. Lindsten and T. B. Schön. Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends in Machine Learning*, 6(1):1–143, 2013.

F. Lindsten, M. I. Jordan, and T. B. Schön. Particle Gibbs with ancestor sampling. *Journal of Machine Learning Research*, 15:2145–2184, 2014.

L. Ljung. *System identification, Theory for the user*. System sciences series. Prentice Hall, Upper Saddle River, NJ, USA, second edition, 1999.

S. Malik and M. K. Pitt. Particle filters for continuous likelihood evaluation and maximisation. *Journal of Econometrics*, 165(2):190–209, 2011. doi: doi: 10.1016/j.jeconom.2011.07.006.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machine. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.

C. A. Naesseth, F. Lindsten, and T. B. Schön. Sequential monte carlo for graphical models. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1862–1870. Curran Associates, Inc., 2014.

C. A. Naesseth, Fl Lindsten, and T. B. Schön. Nested sequential monte carlo methods. *arXiv:1502.02536*, 2015.

B. Ninness and S. Henriksen. Bayesian system identification via Markov chain Monte Carlo techniques. *Automatica*, 46(1):40–51, 2010.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, USA, 2 edition, 2006.

J. Olsson, R. Douc, O. Cappé, and E. Moulines. Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state-space models. *Bernoulli*, 14(1):155–179, 2008.

V. Peterka. Bayesian system identification. *Automatica*, 17(1):41–53, 1981.

M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

M. K. Pitt, R. S. Silva, P. Giordani, and R. Kohn. On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal of Econometrics*, 171:134–151, 2012.

G. Poyiadjis, A. Doucet, and S. S. Singh. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80, 2011.

T. B. Schön and F. Lindsten. *Learning of dynamical systems – Particle filters and Markov chain methods*. 2015. (forthcoming, draft manuscript is available from the authors).

T. B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

C. Sherlock, A. H. Thiery, G. O. Roberts, and J. S. Rosenthal. On the efficency of pseudo-marginal random walk Metropolis algorithms. *Pre-print*, 2013. arXiv:1309.7209v1.

R. H. Shumway and D. S. Stoffer. *Time Series Analysis and Its Applications – with R examples*. Springer Texts in Statistics. Springer, New York, USA, 3 edition, 2011.

T. Söderström and P. Stoica. *System Identification*. Prentice Hall, 1989.

E. Taghavi, F. Lindsten, L. Svensson, and T. B. Schön. Adaptive stopping for fast particle smoothing. In *Proceedings of the 38th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.

M. A. Tanner and W. H. Wong. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82(398): 528–540, June 1987.

M. Toussaint and A. Storkey. Probabilistic inference for solving discrete and continuous state Markov decision processes. In *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, USA, 2006.

D. A. van Dyk and X.-L. Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, March 2001.

G. C. G. Wei and M. A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704, 1990.

A. Wills, T. B. Schön, F. Lindsten, and B. Ninness. Estimation of linear systems using a Gibbs sampler. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, July 2012.