Randomized Quasi Sequential Markov Chain Monte Carlo[2]

Fabian Goessling[†]

70/2018

[†] Department of Economics, University of Münster, Germany

Randomized Quasi Sequential Markov Chain Monte Carlo^2

Fabian Goessling[a]

[a]*Department of Economics, Am Stadtgraben 9, University of Münster, 48143 Münster, Germany*

Abstract

Sequential Monte Carlo and Markov Chain Monte Carlo methods are combined into a unifying framework for Bayesian parameter inference in non-linear, non-Gaussian state space models. A variety of tuning approaches are suggested to boost convergence: likelihood tempering, data tempering, adaptive proposals, random blocking, and randomized Quasi Monte Carlo numbers. The methods are illustrated and compared by running eight variants of the algorithm to estimate the parameters of a standard stochastic volatility model.

Keywords: SMC, MCMC, Bayesian Estimation, Filtering

JEL: C11, C13, C32

1. Introduction

Bayesian estimation approaches for non-linear, non-Gaussian state space models are – seemingly – divided into two methods: Sequential Monte Carlo (SMC) and Markov Chain Monte Carlo (MCMC). While SMC can be interpreted as an approximation based on a cross-section of samples, MCMC constitutes an approximation based on trajectories of samples. Building upon the work of Chopin et al. (2005), Andrieu et al. (2010), Deligiannidis et al. (2016), and in particular Gerber and Chopin (2015), this paper merges both viewpoints into a single, unifying framework for parameter inference. The resulting randomized Quasi Sequential Markov Chain Monte Carlo^2 (rQSMCMC^2) algorithm is an extended version of the algorithms presented by Chopin et al. (2005), Herbst and Schorfheide (2014) and Duan and Fulop (2015). Augmenting the latter work, I give a theoretic, general presentation of the full algorithm that covers a plethora of variants and special cases. Nevertheless, the presentation is application-oriented to bridge the gap between theory and applied econometrics.

Email address: Fabian.Goessling@uni-muenster.de (Fabian Goessling)

The rQSMCMC^2 algorithm is based on a particle filter for the latent state variables, which itself is used to form a second, outer particle filter on an extended parameter space. The outer SMC approximation is mutated using blocked, correlated pseudo-marginal MCMC steps. Slight variations in the parametrization shift the algorithm from pseudo-marginal MCMC to SMC^2 and vice versa, such that the presented approach nests standard (correlated) pseudo-marginal MCMC, parallel pseudo-marginal MCMC and SMC^2 as special cases, blurring the borders between the methods.

Besides the theoretic justification, which is related to Gerber and Chopin (2015), another contribution is the seamless integration of a variety of tuning approaches, e.g. likelihood tempering, data tempering, adaptive proposals, random blocking and randomized Quasi Monte Carlo (rQMC) numbers.

The paper proceeds as follows. Section 2 introduces the notation, the general setting and briefly reviews importance sampling. Section 3 covers the SMC steps. Section 4 describes several tuning-approaches, extensions and limiting cases as well as a comment on resampling. As an empirical illustration, a standard stochastic volatility model is estimated by eight different versions of the algorithm in section 5. Section 6 concludes.

2. General setting

Throughout the paper, consider a state space model with latent variables $\mathbf{x}_t \in \mathcal{X} \subseteq \mathbb{R}^d$ which are observed through variables $\mathbf{y}_t \in \mathcal{Y} \subseteq \mathbb{R}^l$ with parameter vector $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p$. The objective is to obtain an unbiased estimator of the high-dimensional integral

$$\int_{\Theta}\int_{\mathcal{X}}\int_{\mathcal{X}}\cdots\int_{\mathcal{X}} \varphi(\boldsymbol{\theta})p(\boldsymbol{\theta},\mathbf{x}_0,\mathbf{x}_1,\ldots,\mathbf{x}_T|\mathbf{y}_{1:T})\mathrm{d}\mathbf{x}_0\mathrm{d}\mathbf{x}_1\ldots\mathrm{d}\mathbf{x}_T\mathrm{d}\boldsymbol{\theta} \tag{1}$$

where $\varphi : \Theta \to \mathbb{R}$. Naive SMC methods approximate (1) by an appropriately weighted set of particles of dimension $\dim(\Theta) \times (T+1) \times \dim(\mathcal{X})$. Clearly, such a direct approach, which necessitates approximations of the marginals $\mathbf{x}_t|\mathbf{y}_{1:T}$ for all $t \in \{0,\ldots,T\}$, is needlessly complex if one is exclusively interested in $\boldsymbol{\theta}|\mathbf{y}_{1:T}$.

SMC^2 methods reduce complexity by splitting up the problem. The first step is a pointwise approximation of the likelihood $p(\mathbf{y}_{1:T}|\boldsymbol{\theta})$. In the second step, it is used to construct an approximation of the posterior distribution with density $p(\boldsymbol{\theta}|\mathbf{y}_{1:T})$. Given samples from the posterior distribution, it is straightforward to approximate integrals of the form

$$\int_\Theta \varphi(\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathbf{y}_{1:T}) \mathrm{d}\boldsymbol{\theta}.$$

As importance sampling is the key element in both steps, I proceed with a brief review. Importance sampling allows to sample from a target distribution $\pi(\mathrm{d}\mathbf{x})$ by means of an importance (or proposal) distribution $p(\mathrm{d}\mathbf{x})$.[1] The general idea of importance sampling is based on the identity

$$\pi(\mathrm{d}\mathbf{x}) = \frac{G(\mathbf{x})p(\mathrm{d}\mathbf{x})}{\int_{\mathcal{X}} G(\mathbf{x})p(\mathrm{d}\mathbf{x})} \tag{2}$$

with the Radon-Nikodym derivative

$$G(\mathbf{x}) = \frac{\pi(\mathrm{d}\mathbf{x})}{p(\mathrm{d}\mathbf{x})}.$$

Importance sampling approximates (2) by the discrete distribution

$$\pi^N(\mathrm{d}\mathbf{x}) \propto \sum_{n=1}^{N} G(\mathbf{x}^n)\delta_{\mathbf{x}^n}(\mathrm{d}\mathbf{x}),$$

where $\mathbf{x}^n$ denotes a sample from the importance distribution $p(\mathrm{d}\mathbf{x})$, $\delta_{\mathbf{x}^n}(\mathrm{d}\mathbf{x})$ is the Dirac measure in $\mathbf{x}^n$, and $G(\cdot)$ evaluated at $\mathbf{x}^n$ are unnormalized importance weights. This approach allows to approximate an integral over a function $\varphi : \mathcal{X} \to \mathbb{R}$ as

$$\pi^N(\varphi) = \int_{\mathcal{X}} \varphi(\mathbf{x})\pi^N(\mathrm{d}\mathbf{x}),$$

where $\pi^N(\varphi)$ denotes the expectation of $\varphi(\mathbf{x})$ under the approximate measure $\pi^N(\mathrm{d}\mathbf{x})$. This general idea carries over to Section 3, where sequential iterations of importance sampling steps are used to approximate the likelihood and the posterior distribution of a general state-space model, respectively.

[1]As suggested by Andrieu et al. (2002), the following notation is adopted: Distributions are denoted as $p(\mathrm{d}\mathbf{x})$, and the corresponding density with respect to an underlying measure is denoted as $p(\mathbf{x})$.

3. Sequential Monte Carlo

3.1. SMC for latent variables

The dynamics of the possibly non-linear, non-Gaussian, Markovian state space model considered in Section 2 are

$$\begin{aligned} \mathbf{x}_0 &\sim f_0^X(\mathrm{d}\mathbf{x}_0|\boldsymbol{\theta}), && (3)\\ \mathbf{x}_t|\mathbf{x}_{t-1} &\sim f^X(\mathrm{d}\mathbf{x}_t|\mathbf{x}_{t-1},\boldsymbol{\theta}), && (4)\\ \mathbf{y}_t|\mathbf{x}_t &\sim f^Y(\mathrm{d}\mathbf{y}_t|\mathbf{x}_t,\boldsymbol{\theta}), && (5) \end{aligned}$$

where for the time being, $\boldsymbol{\theta}$ is considered a fixed parameter vector. Let $f(\mathrm{d}\cdot)$ denote known distributions, e.g. implied by an underlying economic model, while unknown generic distributions that have to be approximated are denoted as $p(\mathrm{d}\cdot)$. Following Gerber and Chopin (2015), all probability measures are assumed to be elements of the set $\mathcal{P}(\mathcal{X})$ of probability measures dominated by the Lebesgue measure, and $\pi(\varphi)$ denotes the expectation of a function $\varphi : \mathcal{X} \to \mathbb{R}$ with respect to $\pi \in \mathcal{P}(\mathcal{X})$. For ease of notation, I omit $\boldsymbol{\theta}$ from the conditioning set for the remainder of this subsection.

Due to the Markov property of $\mathbf{x}_t$ and conditional independence of $\mathbf{y}_t$ given $\mathbf{x}_t$, it follows that

$$p(\mathrm{d}\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int_{\mathcal{X}} f^X(\mathrm{d}\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathrm{d}\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}), \tag{6}$$

hence the incremental likelihood contributions are given as

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \int_{\mathcal{X}} f^Y(\mathbf{y}_t|\mathbf{x}_t)p(\mathrm{d}\mathbf{x}_t|\mathbf{y}_{1:t-1}), \tag{7}$$

implying the likelihood

$$p(\mathbf{y}_{1:t}) = p(\mathbf{y}_1)\prod_{\tau=2}^{t} p(\mathbf{y}_\tau|\mathbf{y}_{\tau-1}). \tag{8}$$

The distribution of the latent states in period t conditioned on observations up to t is

$$p(\mathrm{d}\mathbf{x}_t|\mathbf{y}_{1:t}) = p(\mathbf{y}_t|\mathbf{y}_{1:t-1})^{-1} f^Y(\mathbf{y}_t|\mathbf{x}_t)p(\mathrm{d}\mathbf{x}_t|\mathbf{y}_{1:t-1}), \tag{9}$$

which can be used to construct $p(\mathrm{d}\mathbf{x}_{t+1}|\mathbf{y}_{1:t})$ and so on and so forth in a recursive manner. Note that the eventual objective of the current subsection is to get an estimator for the likelihood (8). To do so the normalizing constant $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ in equation (9) has to be approximated as the predictive and filtering distributions in (6) and (9) are in general analytically unknown.

The rigorous approach of Gerber and Chopin (2015) facilitates a deeper understanding of the algorithm's underlying mechanics, at the cost of a slightly more complex notation. Define the functions

$$\begin{aligned} G_0(\mathbf{x}_0) &= \frac{f^Y(\mathbf{y}_0|\mathbf{x}_0)f_0^X(\mathrm{d}\mathbf{x}_0)}{m_0(\mathrm{d}\mathbf{x}_0)}, \\ G_t(\mathbf{x}_{t-1},\mathbf{x}_t) &= \frac{f^Y(\mathbf{y}_t|\mathbf{x}_t)f^X(\mathrm{d}\mathbf{x}_t|\mathbf{x}_{t-1})}{m_t(\mathbf{x}_{t-1},\mathrm{d}\mathbf{x}_t)}, \end{aligned}$$

with

$$\begin{aligned} G_0 &: \mathcal{X} \to \mathbb{R}^+, \\ G_t &: \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+, \\ \varphi &: \mathcal{X} \to \mathbb{R}, \end{aligned}$$

an initial importance distribution $m_0(\mathrm{d}\mathbf{x}_0)$ and a set of Markov transition kernels $m_t(\mathbf{x}_{t-1},\mathrm{d}\mathbf{x})$ for $t \geq 1$. Sampling $\mathbf{x}_0 \sim m_0(\mathrm{d}\mathbf{x}_0)$ yields the importance sampled distribution

$$\begin{aligned} \mathbb{Q}_0(\mathrm{d}\mathbf{x}_0) &= m_0(\mathrm{d}\mathbf{x}_0)G_0(\mathbf{x}_0)/Z_0, \\ Z_0 &= \int_{\mathcal{X}} G_0(\mathbf{x}_0)m_0(\mathrm{d}\mathbf{x}_0), \end{aligned}$$

for $t = 0$, corresponding to $p(\mathrm{d}\mathbf{x}_0|\mathbf{y}_0)$. Combining the latter distribution with the kernel $m_1(\mathbf{x}_0,\mathrm{d}\mathbf{x}_1)$ constitutes the importance distribution for $t = 1$, i.e.

$$\mathbb{Q}_1(\mathrm{d}(\mathbf{x}_0,\mathbf{x}_1)) = \frac{G_1(\mathbf{x}_0,\mathbf{x}_1)\mathbb{Q}_0(\mathrm{d}\mathbf{x}_0)m_1(\mathbf{x}_0,\mathrm{d}\mathbf{x}_1)}{\int_{\mathcal{X}}\int_{\mathcal{X}} G_1(\mathbf{x}_0,\mathbf{x}_1)\mathbb{Q}_0(\mathrm{d}\mathbf{x}_0)m_1(\mathbf{x}_0,\mathrm{d}\mathbf{x}_1)}. \tag{10}$$

As an intermediate measure, define the predictive distribution

$$\bar{\mathbb{Q}}_1(\mathrm{d}(\mathbf{x}_0,\mathbf{x}_1)) = \mathbb{Q}_0(\mathrm{d}\mathbf{x}_0)m_1(\mathbf{x}_0,\mathrm{d}\mathbf{x}_1).$$

Then equation (10) can be rewritten as

$$\mathbb{Q}_1(\mathrm{d}(\mathbf{x}_0,\mathbf{x}_1)) = \frac{\bar{\mathbb{Q}}_1(\mathrm{d}(\mathbf{x}_0,\mathbf{x}_1))G_1(\mathbf{x}_0,\mathbf{x}_1)}{\bar{\mathbb{Q}}_1(G_1)},$$

which yields the well-known prediction-update recursion of filtering algorithms for $t > 1$ as

$$\begin{aligned}
\bar{\mathbb{Q}}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t)) &= \mathbb{Q}_{t-1}(\mathrm{d}(\mathbf{x}_{t-2},\mathbf{x}_{t-1}))m_t(\mathbf{x}_{t-1},\mathrm{d}\mathbf{x}_t),\\
\mathbb{Q}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t)) &= \frac{\bar{\mathbb{Q}}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t))G_t(\mathbf{x}_{t-1},\mathbf{x}_t)}{\bar{\mathbb{Q}}_t(G_t)},
\end{aligned}$$

where $\bar{\mathbb{Q}}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t))$ and $\mathbb{Q}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t))$ are measures on $\mathcal{X}\times\mathcal{X}$. The marginal distributions of $\bar{\mathbb{Q}}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t))$ and $\mathbb{Q}_t(\mathrm{d}(\mathbf{x}_{t-1},\mathbf{x}_t))$ with respect to $\mathbf{x}_t$ correspond to (6) and (9), respectively. For any function $\varphi : \mathcal{X} \to \mathbb{R}$,

$$\begin{aligned}
\bar{\mathbb{Q}}_t(\varphi) &= \frac{1}{Z_{t-1}}\mathbb{E}\left[\varphi(\mathbf{x}_t)G_0(\mathbf{x}_0)\prod_{\tau=1}^{t-1}G_\tau(\mathbf{x}_{\tau-1},\mathbf{x}_\tau)\right],\\
\mathbb{Q}_t(\varphi) &= \frac{1}{Z_t}\mathbb{E}\left[\varphi(\mathbf{x}_t)G_0(\mathbf{x}_0)\prod_{\tau=1}^{t}G_\tau(\mathbf{x}_{\tau-1},\mathbf{x}_\tau)\right],
\end{aligned}$$

where expectations are taken with respect to the Markov Chain defined by $m_t(\mathbf{x}_{t-1},\mathrm{d}\mathbf{x}_t)$. The normalizing constant

$$Z_t = \int_{\mathcal{X}}\cdots\int_{\mathcal{X}} G_0(\mathbf{x}_0)\prod_{\tau=1}^{t}G_\tau(\mathbf{x}_{\tau-1},\mathbf{x}_\tau)m_0(\mathrm{d}\mathbf{x}_0)\prod_{\tau=1}^{t}m_\tau(\mathbf{x}_{\tau-1},\mathrm{d}\mathbf{x}_\tau) \tag{11}$$

is the likelihood up to time t, corresponding to equation (8).

Having established the theoretical iterations, I turn to their Monte Carlo approximations. Define the normalized importance weights

$$w_0^n = \frac{G_0(\mathbf{x}_0^n)}{\sum_{i=1}^N G_0(\mathbf{x}_0^i)}, \tag{12}$$

$$w_t^n = \frac{G_t(\mathbf{x}_{t-1}^n,\mathbf{x}_t^n)}{\sum_{i=1}^N G_t(\mathbf{x}_{t-1}^i,\mathbf{x}_t^i)} \tag{13}$$

and denote particles by $\mathbf{x}_t^n$ with $n \in \{1,\dots,N\}$. Thus, the estimator of $\mathbb{Q}_0(\varphi)$ is $\sum_{n=1}^N w_0^n\varphi(\mathbf{x}_0^n)$, where the particles $\mathbf{x}_0^n$ are generated by $m_0(\mathrm{d}\mathbf{x}_0)$. Consequently, the approximated *prediction* probability measure is

$$\bar{\mathbb{Q}}_t^N(\mathrm{d}(\tilde{\mathbf{x}}_{t-1},\mathbf{x}_t)) = \sum_{n=1}^N w_{t-1}^n\delta_{\mathbf{x}_{t-1}^n}(\mathrm{d}\tilde{\mathbf{x}}_{t-1})m_t(\mathbf{x}_{t-1}^n,\mathrm{d}\mathbf{x}_t)$$

and the approximated *update* probability measure is

$$\mathbb{Q}_t^N(\mathrm{d}(\tilde{\mathbf{x}}_{t-1},\mathbf{x}_t)) = \frac{\bar{\mathbb{Q}}_t^N(\mathrm{d}(\tilde{\mathbf{x}}_{t-1},\mathbf{x}_t))G_t(\tilde{\mathbf{x}}_{t-1},\mathbf{x}_t)}{\bar{\mathbb{Q}}_t^N(G_t)}.$$

The marginal likelihood (11) can easily be approximated by the product of the average unnormalized importance weights, i.e.

$$Z_t^N = \left(\frac{1}{N} \sum_{n=1}^{N} G_0(\mathbf{x}_0^n) \right) \prod_{\tau=1}^{t} \left(\frac{1}{N} \sum_{n=1}^{N} G_\tau(\tilde{\mathbf{x}}_{\tau-1}^n, \mathbf{x}_\tau^n) \right),$$

and resembles an unbiased estimator of the true likelihood as shown by Andrieu et al. (2010).

Note that the SMC presentation above possibly includes *resampling steps* as the measures are defined on $\mathcal{X} \times \mathcal{X}$ and $\mathbf{x}_t^n \neq \tilde{\mathbf{x}}_t^n$.

3.2. SMC for parameter estimation

The iteration steps of the SMC for latent states yield an unbiased estimator Z_T^N for the likelihood. This estimator now acts as a stepping stone for parameter estimation in the second step which combines SMC and the correlated pseudo-marginal MCMC of Deligiannidis et al. (2016). To economize on notation, many variable names are re-used, and new symbols are only introduced sparingly.

Consider an extended space $\Theta \times \mathcal{U}$ with parameters $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p$ and auxiliary variables $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^{N \times (T+1) \times (d+1)}$. In contrast to Section 3.1, the auxiliary random variables $\mathbf{u}$ which are required for purely technical reasons, are explicitly taken into account. Typically, the auxiliary variables are necessary to draw from the distribution $\mathbb{Q}_t^N(\mathrm{d}(\tilde{\mathbf{x}}_{t-1}, \mathbf{x}_t))$. Denoting the unbiased estimator $\hat{p}(\mathbf{y}_{1:T}|\boldsymbol{\theta}, \mathbf{U}) = Z_T^N$ where $\mathbf{U} \sim p(\mathrm{d}\mathbf{u})$ and defining $\pi(\cdot) = p(\cdot|\mathbf{y}_{1:T})$, the joint posterior density of parameters and auxiliary variables on $\Theta \times \mathcal{U}$ is

$$\pi(\boldsymbol{\theta}, \mathbf{u}) = \frac{\pi(\boldsymbol{\theta}) p(\mathbf{u}) \hat{p}(\mathbf{y}_{1:T}|\boldsymbol{\theta}, \mathbf{u})}{p(\mathbf{y}_{1:T}|\boldsymbol{\theta})},$$

which serves as target distribution for the second stage of SMC iterations. By construction the posterior density

$$\pi(\boldsymbol{\theta}) \propto p(\mathbf{y}_{1:T}|\boldsymbol{\theta}) p(\boldsymbol{\theta}),$$

is the marginal density of $\pi(\boldsymbol{\theta}, \mathbf{u})$ when $\mathbf{u}$ is integrated out.

Let the index j denote the iteration stage of the algorithm, i.e. the j-th *bridge distribution* is $\pi_j(\mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j))$. Similar to the previous section let

$$G_0((\boldsymbol{\theta}_0, \mathbf{u}_0)) = \frac{\pi_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0))}{k_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0))}, \tag{14}$$

$$G_j((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)) = \frac{\pi_j(\mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j))}{k_j\left((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), \mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j)\right)}, \tag{15}$$

where

$$\begin{aligned}
&G_0 : (\Theta \times \mathcal{U}) \to \mathbb{R}^+,\\
&G_j : (\Theta \times \mathcal{U}) \times (\Theta \times \mathcal{U}) \to \mathbb{R}^+.
\end{aligned}$$

Let the initial target distribution $\pi_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0))$ equal the prior $p(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0))$, and define Markov kernels $k_j\left((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), \mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j)\right)$ for $j = 1, \ldots, J$. The prediction and update measures are then

$$\begin{aligned}
\mathbb{Q}_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0)) &= \frac{G_0(\boldsymbol{\theta}_0, \mathbf{u}_0) k_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0))}{Z_0},\\
Z_0 &= \int_{\Theta \times \mathcal{U}} G_0(\boldsymbol{\theta}_0, \mathbf{u}_0) k_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0)),\\
\bar{\mathbb{Q}}_1(\mathrm{d}((\boldsymbol{\theta}_0, \mathbf{u}_0), (\boldsymbol{\theta}_1, \mathbf{u}_1))) &= \mathbb{Q}_0(\mathrm{d}(\boldsymbol{\theta}_0, \mathbf{u}_0)) k_1((\boldsymbol{\theta}_0, \mathbf{u}_0), \mathrm{d}(\boldsymbol{\theta}_1, \mathbf{u}_1)),\\
\mathbb{Q}_1(\mathrm{d}((\boldsymbol{\theta}_0, \mathbf{u}_0), (\boldsymbol{\theta}_1, \mathbf{u}_1))) &= \frac{\bar{\mathbb{Q}}_1(\mathrm{d}((\boldsymbol{\theta}_0, \mathbf{u}_0), (\boldsymbol{\theta}_1, \mathbf{u}_1))) G_1((\boldsymbol{\theta}_0, \mathbf{u}_0), (\boldsymbol{\theta}_1, \mathbf{u}_1))}{\bar{\mathbb{Q}}_1(G_1)}.
\end{aligned}$$

and so on for all $j = 2, \ldots, J$.

The Monte Carlo implementation closely follows Section 3.1, apart from from indexing the particles $\{\boldsymbol{\theta}^m, \boldsymbol{u}^m\}$ by $m \in \{1, \ldots, M\}$, where M is the number of particles in the $\boldsymbol{\theta}$ dimension. The approximated measures are

$$\begin{aligned}
&\bar{\mathbb{Q}}_j^M \left(\mathrm{d}((\tilde{\boldsymbol{\theta}}_{j-1}, \tilde{\mathbf{u}}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j))\right)\\
&\quad = \sum_{m=1}^{M} w_{j-1}^m \delta_{(\boldsymbol{\theta}_{j-1}^m, \mathbf{u}_{j-1}^m)}(\mathrm{d}(\tilde{\boldsymbol{\theta}}_{j-1}, \tilde{\mathbf{u}}_{j-1})) k_j\left((\boldsymbol{\theta}_{j-1}^m, \mathbf{u}_{j-1}^m), \mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j)\right),\\
&\mathbb{Q}_j^M \left(\mathrm{d}((\tilde{\boldsymbol{\theta}}_{j-1}, \tilde{\mathbf{u}}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j))\right)\\
&\quad = \frac{1}{\bar{\mathbb{Q}}_j^M(G_j)} \bar{\mathbb{Q}}_j^M \left(\mathrm{d}((\tilde{\boldsymbol{\theta}}_{j-1}, \tilde{\mathbf{u}}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j))\right) G_j((\tilde{\boldsymbol{\theta}}_{j-1}, \tilde{\mathbf{u}}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)),
\end{aligned}$$

where the normalized weights $w_{j-1}^{1:M}$ are constructed using (14) and (15).

4. Implementation details and extensions

4.1. Auxiliary Markov kernel

An auxiliary Markov transition kernel can serve as a powerful tool to refine the samples from $\mathbb{Q}_j(\mathrm{d}((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)))$. Consider an index $s = 1, \ldots, S$ which denotes auxiliary steps in iteration j and denote the corresponding kernel $\tilde{k}_j((\boldsymbol{\theta}_{j,s-1}, \mathbf{u}_{j,s-1}), \mathrm{d}(\boldsymbol{\theta}_{j,s}, \mathbf{u}_{j,s}))$.

If $\tilde{k}_j(\cdot, \mathrm{d}\cdot)$ has invariant distribution $\pi_j(\mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j))$ the invariance property (Andrieu et al. (2002))

$$\begin{aligned}\int_{\Theta\times\mathcal{U}} \mathbb{Q}_j(\mathrm{d}((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)))\tilde{k}((\boldsymbol{\theta}_{j,1}, \mathbf{u}_{j,1}), \mathrm{d}(\boldsymbol{\theta}_{j,2}, \mathbf{u}_{j,2}))\\ = \mathbb{Q}_j(\mathrm{d}((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)))\end{aligned}$$

holds. Thus, applying the kernel $\tilde{k}(\cdot, \mathrm{d}\cdot)$ for S iterations preserves the target distribution, and defining

$$\begin{aligned}&\mathbb{Q}_j(\mathrm{d}((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_{j,s}, \mathbf{u}_{j,s})))\\ &\quad = \mathbb{Q}_j(\mathrm{d}((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)))\prod_{r=2}^{s}\tilde{k}_j((\boldsymbol{\theta}_{j,r-1}, \mathbf{u}_{j,r-1}), \mathrm{d}(\boldsymbol{\theta}_{j,r}, \mathbf{u}_{j,r})),\end{aligned}$$

allows to generate S random realizations from the approximate measure of interest.

A possible auxiliary Markov kernel is the commonly used Metropolis Hastings (MH) kernel. Omitting the fixed index j for parsimony the MH kernel is

$$\begin{aligned}\tilde{k}((\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), \mathrm{d}(\boldsymbol{\theta}_s, \mathbf{u}_s)) = \alpha\{(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), (\boldsymbol{\theta}_s, \mathbf{u}_s)\}q((\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), \mathrm{d}(\boldsymbol{\theta}_s, \mathbf{u}_s))\\ + r(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1})\delta_{(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1})}\mathrm{d}(\boldsymbol{\theta}_s, \mathbf{u}_s),\end{aligned}$$

with acceptance probability

$$\alpha\{(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), (\boldsymbol{\theta}_s, \mathbf{u}_s)\} = \min\left\{1, \frac{\pi_j(\boldsymbol{\theta}_s, \mathbf{u}_s)q(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}|\boldsymbol{\theta}_s, \mathbf{u}_s)}{\pi_j(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1})q(\boldsymbol{\theta}_s, \mathbf{u}_s|\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1})}\right\}, \tag{16}$$

and rejection probability

$$r(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}) = 1 - \int_{\Theta\times\mathcal{U}} \alpha\{(\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), (\boldsymbol{\theta}_s, \mathbf{u}_s)\}q((\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), \mathrm{d}(\boldsymbol{\theta}_s, \mathbf{u}_s))$$

where $q(\cdot, \mathrm{d})$ denotes the proposal distribution and $q(\cdot|\cdot)$ the corresponding density. Instead of using a standard kernel, I propose the correlated pseudo marginal kernel of Deligiannidis et al. (2016). It is a Metropolis-Hastings kernel on $\Theta \times \mathcal{U}$ with cunning proposals and priors for the auxiliary variables $\mathbf{u}$. The corresponding proposal distribution is split up into two separate kernels, i.e.

$$q((\boldsymbol{\theta}_{s-1}, \mathbf{u}_{s-1}), \mathrm{d}(\boldsymbol{\theta}_s, \mathbf{u}_s)) = q^1(\boldsymbol{\theta}_{s-1}, \mathrm{d}\boldsymbol{\theta}_s)q^2(\mathbf{u}_{s-1}, \mathrm{d}\mathbf{u}_s),$$

and the prior for $\mathbf{u}$ is constructed, such that

$$p(\mathbf{u}_{s-1})q^2(\mathbf{u}_s|\mathbf{u}_{s-1}) = p(\mathbf{u}_s)q^2(\mathbf{u}_{s-1}|\mathbf{u}_s) \tag{17}$$

holds. Thus the calculation of equation (16) simplifies noticeably, as the proposal density and the prior for the auxiliary variables drop out of the calculation of the acceptance probability. Moreover, correlating the auxiliary variables decreases the Monte Carlo noise in the acceptance probability, and thus the algorithm can be implemented with a significantly smaller number N of particles in the $\mathbf{x}_t$ dimension.

Note that the kernels $k_j(\cdot, \mathrm{d}\cdot)$ and $\tilde{k}_j(\cdot, \mathrm{d}\cdot)$ are subject to different requirements as they serve different purposes. The former kernel is required to propagate the $\boldsymbol{\theta}$ particles and even though an MCMC step preserving the target distribution is a sensible choice, this is not mandatory. By contrast, the latter kernel is used to refine the sample and has to have the correct invariant distribution. In the following section, I discuss the proposal strategy with regards to $\tilde{k}_j(\cdot, \mathrm{d}\cdot)$.

4.2. Proposal distribution

Since the success of SMC hinges on suitable proposals for the $\boldsymbol{\theta}$ particles, I suggest to use the tailored proposal kernel $q^1(\boldsymbol{\theta}_{s-1}, \mathrm{d}\boldsymbol{\theta}_s)$ of Herbst and Schorfheide (2014). Further, I adopt their Gibbs strategy, which randomly partitions the parameter space into a predefined number of subspaces $\Theta = \prod_{g=1}^{G} \Theta_g$. While constructing the proposal I make use of the full set of particles in the parameter dimension (Andrieu et al. (2002), Herbst and Schorfheide (2014) and Chopin et al. (2005)) and use the mixture distribution

$$\begin{aligned}
&(\boldsymbol{\theta}_s^m, \mathbf{u}_s^m)|\boldsymbol{\theta}_{s-1}^{1:M}, \mathbf{u}_{s-1}^{1:M} \\
&\sim \mathcal{N}(\rho \mathbf{u}_{s-1}^m, (1-\rho^2)) \times \begin{cases} \mathcal{N}(\boldsymbol{\theta}_{s-1}^m, c_{s-1}^2 \boldsymbol{\Sigma}_{s-1}) & \textit{with probability } p \\ \mathcal{N}(\boldsymbol{\theta}_{s-1}^m, c_{s-1}^2 \mathrm{diag}(\boldsymbol{\Sigma}_{s-1})) & \textit{with probability } \frac{1-p}{2} \\ \mathcal{N}(\bar{\boldsymbol{\theta}}_{s-1}, c_{s-1}^2 \boldsymbol{\Sigma}_{s-1}) & \textit{with probability } \frac{1-p}{2} \end{cases}
\end{aligned}$$

where $\boldsymbol{\Sigma}_{s-1}$ and $\bar{\boldsymbol{\theta}}_{s-1}$ are the empirical covariance matrix and the mean of the particle set $\boldsymbol{\theta}_{s-1}^{1:M}$, while diag$(\cdot)$ sets all non-diagonal elements of its input matrix to zero. Furthermore, the scaling variable c is adapted after each iteration in order to achieve a pre-specified acceptance ratio $\tilde{\alpha}^*$. Note that the acceptance ratio $\tilde{\alpha}_s$ measures the proportion of accepted moves at step s, while the acceptance probability α is the probability of moving given the current state of the Markov chain. In particular, I adopt the specification of Herbst and Schorfheide (2014) and use the recursion

$$c_s = c_{s-1} f(1 - \tilde{\alpha}_{s-1}),$$

where

$$f(x) = 0.95 + 0.1 \frac{\exp(16(x - \tilde{\alpha}^*))}{1 + \exp(16(x - \tilde{\alpha}^*))}.$$

Note that the reversibility of the auxiliary distributions in (17) is left untouched by this proposal distribution. Thus, calculating the acceptance probability (16) only requires to evaluate the density of the normal mixture distribution at $\boldsymbol{\theta}_s$. Furthermore, one could dampen the Monte Carlo noise in the likelihood estimation by using an additional Gibbs block for the auxiliary variables. An alternative approach is presented by Tran et al. (2017), who propose to update random partitions of the $\mathbf{u}_s$ vector in a pseudo-marginal MCMC framework.

4.3. Tempering strategies

This section discusses different ways to construct bridge densities, in particular the three obvious approaches: *likelihood tempering*, *data tempering* and a *combination* of both. Each of these approaches applies a tempering schedule $\Phi(j)$ to the target distribution.

For likelihood tempering this schedule is defined such that $\Phi(1) = 0$ and $\Phi(J) = 1$, and the bridge distributions are constructed as

$$\pi_j(\mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j)) \propto p(\mathbf{y}_{1:T} | \boldsymbol{\theta}_j, \mathbf{u}_j)^{\Phi(j)} p((\mathrm{d}\boldsymbol{\theta}_j, \mathbf{u}_j))$$

for $j = 1, \ldots, J$ and $\Phi(\cdot)$ is increasing in j. Furthermore, setting the initial target distribution equal to the prior and defining the particle generating kernel

$$k_j\left((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), \mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j)\right) = \pi_{j-1}(\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}) \delta_{(\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1})}(\mathrm{d}(\boldsymbol{\theta}_j, \mathbf{u}_j)), \tag{18}$$

simplifies the algorithm as the ratios (14) and (15) reduce to

$$\begin{aligned} G_0((\boldsymbol{\theta}_0, \mathbf{u}_0)) &= 1 \\ G_j((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)) &= p(\mathbf{y}_{1:T} | \boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1})^{\Phi(j) - \Phi(j-1)}. \end{aligned}$$

In contrast, the data tempering approach adds observations step by step to the conditioning set of the target distribution, such that $\pi_j(\boldsymbol{\theta}_j, \mathbf{u}_j) \propto p(\mathbf{y}_{1:\tilde{\Phi}(j)}|\boldsymbol{\theta}_j, \mathbf{u}_j)p(\boldsymbol{\theta}_j, \mathbf{u}_j)$ where e.g. $\tilde{\Phi}(1) = 1$ and $\tilde{\Phi}(J) = T$. Thus, while maintaining kernel (18), the corresponding weights simplify to

$$G_0((\boldsymbol{\theta}_0, \mathbf{u}_0)) = 1$$
$$G_j((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)) = p(\mathbf{y}_{\tilde{\Phi}(j)}|\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}, \mathbf{y}_{1:\tilde{\Phi}(j-1)}).$$

Both approaches can be combined to construct doubly-tempered bridge distributions. For each data tempering step one runs a full likelihood tempering schedule. Such a combined approach could be termed *incremental likelihood tempering* and the weights are

$$G_0((\boldsymbol{\theta}_0, \mathbf{u}_0)) = 1$$
$$G_j((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_j, \mathbf{u}_j)) = p(\mathbf{y}_{\tilde{\Phi}(j)}|\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}, \mathbf{y}_{1:\tilde{\Phi}(j-1)})^{\Phi(j)-\Phi(j-1)},$$

where $\tilde{\Phi}(j)$ is constant for a sweep of the likelihood tempering schedule. The combined tempered approach might be particularly useful for very high dimensional posterior distributions, as it ensures that successive bridge distributions are even more alike than in the standard tempering cases.

All approaches use the approximation of stage $j-1$ as importance distribution, and thus the particles of the initial stage are carried through all stages, leading to a deteriorating variety of the particle swarm over iterations. Therefore auxiliary mutation steps (i.e. $S \geq 1$) are necessary to rejuvenate the swarms. Note that data tempering allows to interpret the bridge densities, while likelihood tempering does not. Even though the intermediate distributions are not unbiased estimators of the "true" transformed posterior, this is not a concern as only the final approximation is used for inference.

4.4. (Randomized) Quasi Monte Carlo

As additional extension randomized Quasi Monte Carlo (rQMC) numbers instead of pseudo-random numbers are used (Gerber and Chopin (2015)). This choice is motivated by the Koksma-Hlawka inequality

$$\left|\frac{1}{N}\sum_{n=1}^{N}\varphi(\mathbf{u}^n) - \int_{[0,1)^d}\varphi(\mathbf{u})\,\mathrm{d}\mathbf{u}\right| \leq V(\varphi)D^{\star}(\mathbf{u}^{1:N}),$$

which states that the approximation error is bounded by a measure of variation $V(\cdot)$ of a function $\varphi : \mathcal{U} \to \mathbb{R}$ and a discrepancy measure $D^\star(\mathbf{u}^{1:N})$[2]. For parsimony, consider integration in the unit cube $[0,1)^d$, i.e. approximating an integral by

$$\frac{1}{N}\sum_{n=1}^{N}\varphi(\mathbf{u}^n) \approx \int_{[0,1)^d}\varphi(\mathbf{u})\mathrm{d}\mathbf{u},$$

where the discrepancy measure quantifies the uniformity of the integration points $\mathbf{u}^{1:N}$ in the unit cube. QMC provides are low discrepancy set by construction and is thus a natural choice for constructing $\mathbf{u}^{1:N}$. Nevertheless, directly using a QMC sequence $\mathbf{v}^{1:N}$ or a deterministic (sparse) grid as integration points $\mathbf{u}^{1:N}$ is not adequate as the resulting estimator is deterministic, and hence biased. Therefore a randomization has to be applied which transforms the QMC point set into a rQMC point set that, on the one hand, retains the low discrepancy property, but on the other hand, ensures that marginally $\mathbf{u}^n \sim \mathbf{U}([0,1)^d)$. A simple method is the Cranley-Patterson shift,

$$\begin{aligned}\mathbf{u}^n &= (\mathbf{v}^n + \mathbf{w}) \mod 1,\\ \mathbf{w} &\sim \mathbf{U}([0,1)^d).\end{aligned}$$

Estimators based on the shifted points are unbiased,

$$\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^{N}\varphi(\mathbf{u}^n)\right] = \int_{[0,1)^d}\varphi(\mathbf{u})\mathrm{d}\mathbf{u}.$$

As the state variables in equation (3) are transformations of uniform auxiliary variables by assumption, the rQMC points imply the set of integration points $\mathbf{x}_t^{1:N}$, i.e. the particle swarm. Implementing resampling steps (discussed in the next section) requires additional random variates $u_t^{r,n} \sim \mathbf{U}([0,1))$ for each particle at each time step t. Overall, I thus consider a QMC sequence $\mathbf{v}^{1:N}$ where $\mathbf{v}^n \in [0,1)^{(d+1)(T+1)}$ and apply a random shift for $n = 1,\dots,N$, such that

$$\mathbf{u}^n = (\mathbf{v}^n + \mathbf{w}) \mod 1$$

[2]See Gerber and Chopin (2015) for more details on discrepancy measures and variation.

is marginally $\mathbf{u}^n \sim \mathbf{U}([0,1)^{(d+1)(T+1)})$ if $\mathbf{w} \sim \mathbf{U}([0,1)^{(d+1)(T+1)})$. Using this construction with M different $\mathbf{w}$ generates M random replications of an integration grid of dimension $N \times (d+1)(T+1)$, which I use to run the M inner particle filters.

Using the rQMC sequence $\mathbf{u}^{1:N}$ has a welcome side effect when computing the correlated pseudo-marginal kernel of Deligiannidis et al. (2016). Instead of keeping track of $M \times N \times (d+1)(T+1)$ uniform random numbers, there are only M random variables $\mathbf{w}$, i.e. $M \times (d+1)(T+1)$ one dimensional uniform random numbers, and the deterministic sequence $\mathbf{v}^{1:N}$. Note that while my implementation relies on the Sobol set, any kind of integration lattice or (sparse) grid could be used to construct $\mathbf{v}^{1:N}$.

4.5. Resampling

As stated before, the resampling step is implicit in Section 3, but its computational implementation requires some comments. For the sake of brevity, I only discuss how to resample the particles $\mathbf{x}_t^{1:N}$. All arguments carry over to the particles $\boldsymbol{\theta}_j^{1:M}$.

Resampling the particle set $\mathbf{x}_{t-1}^{1:N}$ to obtain a new set $\tilde{\mathbf{x}}_{t-1}^{1:N}$ is necessary to avoid dominating single weights. It is straightforward to implement a resampling step by using a projection $h : \mathcal{X} \to [0,1)$ which is given by an absolute ordering $\sigma(\cdot)$ of $\mathbf{x}_{t-1}^{1:N}$. In combination with the normalized weights (12) and (13), the ordering allows to construct an empirical cumulative distribution function

$$F^N(n) = \sum_{i=1}^{N} w_{t-1}^{\sigma(i)} \mathbf{1}_{\{\sigma(i) \leq n\}},$$

such that the index a_{t-1}^n of the ancestor of particle $\mathbf{x}_t^n$ can be sampled by inversion methods. This requires additional N uniform random variables $u_t^{r,n} \sim \mathbf{U}([0,1))$ (as mentioned in Section 4.4) which are used to set $a_{t-1}^n = F_N^{-1}(u_t^{r,n})$ and $\mathbf{x}_{t-1}^n = \mathbf{x}_{t-1}^{a_{t-1}}$.

The crucial choice is the ordering $\sigma(\cdot)$. Obviously, the simplest approach is to use the index n as ordering criterion, i.e. $\sigma(n) = n$, which corresponds to standard multinomial resampling. There are, however, better ways to resample (in terms of the variance of the estimator based on the resampled particles). In the one-dimensional case an intuitive and natural choice is to sample a set of quantiles of x_t that uniformly cover $[0,1)$. To do so, the particles x_t have to be sorted ascendingly. For higher dimensions, such a natural ordering does no longer exist. Gerber and Chopin (2015) propose a sorting mechanism that is based on Hilbert's space-filling curve. This method constructs $\sigma(\cdot)$ such that after rearranging, successive particles are "similar" in some sense and increasingly ordered. Thus, by using a quasi-random number set to construct $u_t^{r,n}$ the resampled particles fill the space $\mathcal{X}$ evenly. Samples obtained from $\bar{\mathbb{Q}}^N(\mathrm{d}(\tilde{\mathbf{x}}_{t-1}, \mathbf{x}_t))$ are a low discrepancy point set and the integration error can be kept small.

Note that Deligiannidis et al. (2016) propose to use the Hilbert sort with a different intention. They correlate the pseudo-random numbers $u_t^{r,n}$ and adopt the Hilbert sort in order to assure that slightly changing $u_t^{r,n}$ to $u'^{r,n}_t$ results in $a'^n_{t-1} = F_N^{-1}(u'^{r,n}_t)$ such that $\mathbf{x}_{t-1}^{a_{t-1}}$ is close to $\mathbf{x}_{t-1}^{a'_{t-1}}$. Then, the likelihood approximation $\hat{p}'(\mathbf{y}_{1:T})$ is (hoped to be) close to $\hat{p}(\mathbf{y}_{1:T})$, i.e. the acceptance probability of a MH step is less distorted by noise induced by the filter. As an alternative, I propose to use the previous weights directly, such that $w_{t-1}^{\sigma(1)} \leq \ldots \leq w_{t-1}^{\sigma(N)}$. Then, after resampling the likelihood approximation $\hat{p}'(\mathbf{y}_{1:T})$ is close to $\hat{p}(\mathbf{y}_{1:T})$ as well.

4.6. Parallel MCMC

In principle, SMC algorithms are intended to approximate a distribution by a cross-section of samples. However, a special case of the rQSMCMC^2 nests a (parallel) MCMC method. Therefore only a single bridge distribution is required, i.e.

$$\pi_0(\boldsymbol{\theta}_0, \mathbf{u}_0) = p(\boldsymbol{\theta}, \mathbf{u}),$$
$$\pi_1(\boldsymbol{\theta}_1, \mathbf{u}_1) = \pi(\boldsymbol{\theta}, \mathbf{u}),$$

and S iterations of the MH-kernel $\tilde{k}((\boldsymbol{\theta}_{1,s-1}, \mathbf{u}_{1,s-1}), \mathrm{d}(\boldsymbol{\theta}_{1,s}, \mathbf{u}_{1,s}))$ are carried out. This corresponds to a single importance sampling step, that is passed on to M "embarrassingly parallel" MCMC chains. The proposal distribution is constructed using the pooled information across all chains. Whereas MCMC requires ergodic chains and a burn-in phase, the SMC framework is less demanding, i.e. the pooled set $\{\mathbb{Q}_1^M(\mathrm{d}((\boldsymbol{\theta}_{j-1}, \mathbf{u}_{j-1}), (\boldsymbol{\theta}_{j,s}, \mathbf{u}_{j,s})))\}_{s\in\{1,\ldots,S\}}$ resembles draws from the posterior distribution of $\boldsymbol{\theta}$.

Algo- rithm	Data tempering	Likelihood tempering	S	M	T
1	-	100	5	20000	80
2	80	5	5	5000	80
3	80	-	10	12500	80
4	-	40	5000	50	80
5	-	300	5	10000	300
6	300	-	5	10000	300
7	300	4	2	6250	300
8	-	20	12500	60	300

Table 1: This table specifies eight different variants of the estimation algorithm. Each variant uses a fixed number of 20 000 000 (30 000 000) (incremental) likelihood evaluations. Columns 2 to 5 report the number of data tempering/likelihood tempering steps, the number of MCMC mutation steps (S), the number of $\boldsymbol{\theta}$ particles (M) and the length of the data sample (T).

5. Application

As an empirical application consider the simple stochastic volatility model

$$\begin{aligned} y_t &\sim \mathcal{N}\left(0, \exp\left(\frac{x_t}{2}\right)\sigma_y^2\right), \\ x_t &\sim \mathcal{N}(\rho x_{t-1}, \sigma_x^2), \end{aligned}$$

with parameter vector $\theta = (\sigma_y, \rho, \sigma_x)'$. Given this setting, I compare the eight versions of the algorithm listed in Table 1. In particular, I fix the number of (incremental) likelihood evaluations to 20 000 000 (30 000 000) in order to assess the performance under a shared constraint for a small (medium) sample size of 80 (300) observations.

I use the same tailored proposal distribution (10) and priors across all versions, randomized Sobol sequences, two random Gibbs blocks and a fixed number of particles $N = 500$ in the x_t dimensions. The data set is simulated using $\sigma_y = 1.2$, $\rho = 0.95$ and $\sigma_x = 0.2$ and depicted in Figure 1.

Table 2 summarizes the key estimation results. Judging from a single simulation run, Algorithms 1, 2 and 4 essentially yield equivalent point estimates and intervals, while Algorithm 3 slightly deviates. This is understandable if one recalls that the data tempering approach adds new information step by step, while likelihood tempering uses all available

	Alg.	ρ	σ_x	σ_y
Panel A: Mean				
	1	0.7973	0.2700	1.5779
	2	0.7971	0.2697	1.5773
	3	0.7861	0.2991	1.5689
	4	0.7973	0.2681	1.5774
	5	0.9126	0.2523	1.4692
	6	0.9040	0.2659	1.4721
	7	0.9098	0.2590	1.4690
	8	0.9165	0.2478	1.4656
Panel B: Median				
	1	0.7999	0.2483	1.5815
	2	0.7998	0.2465	1.5801
	3	0.7908	0.2695	1.5738
	4	0.7996	0.2473	1.5816
	5	0.9327	0.2294	1.4737
	6	0.9258	0.2421	1.4785
	7	0.9297	0.2363	1.4764
	8	0.9361	0.2253	1.4714
Panel C: Interval [0.05,0.95]				
	1	[0.6369, 0.9491]	[0.0370, 0.5783]	[1.2717, 1.8777]
	2	[0.6370, 0.9512]	[0.0386, 0.5809]	[1.2747, 1.8789]
	3	[0.5974, 0.9565]	[0.0375, 0.6664]	[1.2277, 1.8929]
	4	[0.6367, 0.9508]	[0.0377, 0.5741]	[1.2721, 1.8728]
	5	[0.7672, 0.9870]	[0.1293, 0.4550]	[1.1767, 1.7461]
	6	[0.7489, 0.9877]	[0.1273, 0.4840]	[1.1609, 1.7593]
	7	[0.7607, 0.9881]	[0.1296, 0.4676]	[1.1644, 1.7531]
	8	[0.7775, 0.9878]	[0.1268, 0.4448]	[1.1614, 1.7481]

Table 2: This table reports estimation results for the variants of the algorithm listed in Table 1. Panel A reports the posterior mean, Panel B reports the median and Panel C a 90% Bayesian interval constructed from the 0.05 and 0.95 quantile of the posterior distribution.

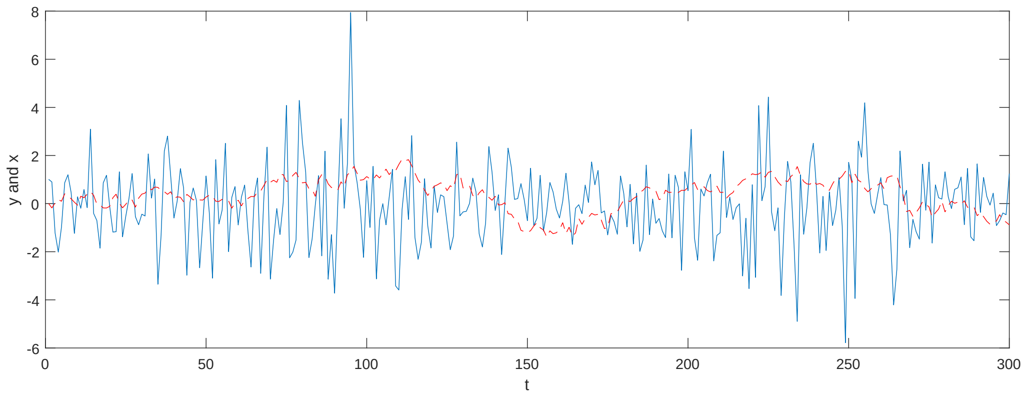

Figure 1: Simulated stochastic volatility model. This figure shows the simulated data set: the observable return series y_t (solid blue line) and the latent log-volatility x_t (dashed red line).

information right from the start. While the bridge distributions for data tempering are sensitive to outliers in the data and thus relatively different from each other, the likelihood tempered bridge distributions are by construction more alike. The downside is that the intermediate distributions have no direct interpretation, while data tempering allows to track the evolution of the posterior distribution over time.

As the results for Algorithm 2 show, combining both tempering approaches solves the problem of data tempering. Moreover, note that the results of the parallel MCMC-style Algorithm 4 are equivalent to the SMC-style Algorithms 1 and 2. For the medium sample size of 300 observations and more tempering steps, Algorithms 5 to 8 result in roughly the same parameter estimates. The sightly more disperse estimates of Algorithms 5 to 8 as compared to Algorithms 1 to 4 can be explained by the fact that the dimension of the integral (1) is more than 3 times higher if $T = 300$. Still, due to more data points and better identification, the estimates approach the "true" values used for the simulation.

Summing up, the results in Table 2 are in favour of likelihood tempering with a balanced choice between the number of MCMC steps S and the number of $\boldsymbol{\theta}$ particles M.

6. Conclusion

The rQSMCMC^2 algorithm is a highly versatile and powerful tool for parameter estimation of non-linear, non-Gaussian state space models and nests a considerable variety of conventional algorithms. In particular, it merges SMC with MCMC methods into a unifying, flexible framework. It is straightforward to exploit parallel computing architectures. This is especially helpful for estimating complex economic models, e.g. non-linear Dynamic Stochastic General Equilibrium models, when it is necessary – but very expensive in terms of computing time – to solve the model for each draw of $\boldsymbol{\theta}$.

A moot question, subject to future research, is how an optimal trade-off between SMC and MCMC can be achieved, and how specific (model) criteria can be used to automatize tuning of the algorithm. Incorporating the Hilbert sort also constitutes a promising extension to reduce the Monte Carlo integration errors.

Literature

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

Andrieu, C., A. Doucet, and E. Punskaya (2002). Sequential Monte Carlo Methods for Optimal Filtering. In *Sequential Monte Carlo Methods in Pratice*, pp. 79–95.

Chopin, N., P. Jacob, and O. Papaspiliopoulos (2005). SMC^2 : An Efficient Algorithm for Sequential Analysis of State-Space Models. *Working Paper* (1), 1–27.

Deligiannidis, G., A. Doucet, and M. K. Pitt (2016). The Correlated Pseudo-Marginal Method. pp. 1–61.

Duan, J. and A. Fulop (2015). Density-Tempered Marginalized Sequential Monte Carlo Samplers. *Journal of Business & Economic Statistics 33*(2), 192–202.

Gerber, M. and N. Chopin (2015). Sequential Quasi Monte Carlo. *Journal of the Royal Statistical Society Series B - Statistical Methodology 77*(3), 509–579.

Herbst, E. and F. Schorfheide (2014). Sequential Monte Carlo Sampling for DSGE Models. *Journal of Applied Econometrics 1098*(July), 1073–1098.

Tran, M.-N., R. Kohn, M. Quiroz, and M. Villani (2017). The Block Pseudo-Marginal Sampler. *Working Paper*.