CREST
CENTER FOR RESEARCH
IN ECONOMICS AND STATISTICS

Série des Documents de Travail

n° 2017-35

Sequential quasi-Monte Carlo: Introduction for Non-Experts, Dimension Reduction, Application to Partly Observed Diffusion Processes

N.CHOPIN[1]
M.GERBER[2]

[1] CREST; ENSAE. E-mail : nicolas.chopin@ensae.fr

[2] School of Mathematics, university of Bristol, University Walk, Clifton, Bristol.
E-mail: Mathieu.gerber@bristol.ac.uk

Sequential quasi-Monte Carlo: Introduction for Non-Experts, Dimension Reduction, Application to Partly Observed Diffusion Processes

Nicolas Chopin and Mathieu Gerber

Abstract SMC (Sequential Monte Carlo) is a class of Monte Carlo algorithms for filtering and related sequential problems. [16] introduced SQMC (Sequential quasi-Monte Carlo), a QMC version of SMC. This paper has two objectives: (a) to introduce Sequential Monte Carlo to the QMC community, whose members are usually less familiar with state-space models and particle filtering; (b) to extend SQMC to the filtering of continuous-time state-space models, where the latent process is a diffusion. A recurring point in the paper will be the notion of dimension reduction, that is how to implement SQMC in such a way that it provides good performance despite the high dimension of the problem.

1 Introduction

SMC (Sequential Monte Carlo) is a class of algorithms that provide Monte Carlo approximations of a sequence of distributions. The main application of SMC is the filtering problem: a phenomenon of interest is modelled as a Markov chain $\{X_t\}$, which is not observed directly; instead one collects sequentially data such as e.g. $Y_t = f(X_t) + V_t$, where V_t is a noise term. Filtering amounts to computing the distribution of X_t given $Y_{0:t} = (Y_0, \ldots, Y_t)$, the data collected up to time t. Filtering and related problems play an important role in target tracking (where X_t is the position of the target, say a ship), robotic mapping (where X_t is the position of the robot), Epidemiology (where X_t is e.g. the number of infected cases), Finance (X_t is the volatility of a given asset) and many other fields. See e.g. the book of [11].

Nicolas Chopin
CREST-ENSAE, 3 Av. Pierre Larousse, 92245 Malakoff, France. e-mail: nicolas.chopin@ensae.fr

Mathieu Gerber
School of Mathematics, University of Bristol, University Walk, Clifton, Bristol BS8 1TW, UK. e-mail: mathieu.gerber@bristol.ac.uk

In [16], we introduced SQMC (Sequential quasi-Monte Carlo), a QMC version of Sequential Monte Carlo. As other types of QMC algorithms, the main advantage of SQMC is the better rate of convergence one may expect, relative to SMC methods.

It is difficult to write a paper that bridges the gap between two scientific communities; in this case, QMC experts on one side, and Statisticians working on Monte Carlo methods (MCMC and SMC) on the other side. We realise now that [16] may be more approachable by the latter than by the former. In particular, that paper spends time explaining basic QMC notions to non-experts, but it does not do the same for SMC.

To address this short-coming, and hopefully generate some interest about SQMC in the QMC community, we decided to devote the first part of this paper to introducing the motivation and basic principles of SMC. We do so using the so-called Feynman-Kac formalism, which is deemed to be abstract, but may be actually more approachable to non-Statisticians.

The second part of this paper discusses how to extend SQMC to the filtering of continuous-time state-space models; i.e. models where the underlying signal is e.g. a diffusion process. These models are popular in Finance and in Biology. What makes this extension interesting is that the inherent dimension of such models is infinity, whereas the performance of SQMC seems to deteriorate with the dimension (according to the numerical studies in [16]). However, by using the Markov property of the latent process, we are able to make some parts of SQMC operate in a low dimension, and, as result, to make it perform well (and significantly better than SMC) despite the infinite dimension of the problem.

2 SMC

2.1 Basic Notions and Definitions

The state space $\mathfrak{X}$ of interest in the paper is always an open subset of $\mathbb{R}^d$, which we equip with the Lebesgue measure.

We use the standard colon short-hand for collections of random variables and related quantities: e.g. $Y_{0:t}$ denote $Y_0, \ldots, Y_t$, $X_t^{1:N}$ denote $X_t^1, \ldots, X_t^N$, and so on. When such variables are vectors, we denote by $X_t(k)$ their $k-$th component.

2.2 Feynman-Kac Formalism

The phrase 'Feynman-Kac model' comes from Probability theory, where 'model' means distributions for variables of interest, and not specifically observed variables (i.e. data, as in Statistics). A Feynman-Kac model consists of:

1. The law of a (discrete-time) Markov process $\{X_t\}$, specified through an initial distribution $\mathbb{M}_0(\mathrm{d}x_0)$, and a sequence of Markov kernels $M_t(x_{t-1},\mathrm{d}x_t)$; i.e. $M_t(x_{t-1},\mathrm{d}x_t)$ is the distribution of X_t, conditional on $X_{t-1}=x_{t-1}$;
2. A sequence of so-called potential (measurable) functions, $G_0 : \mathfrak{X} \to \mathbb{R}^+$, $G_t : \mathfrak{X}\times\mathfrak{X} \to \mathbb{R}^+$. ($\mathbb{R}^+ = [0,+\infty)$.)

From these objects, one defines the following sequence of probability distributions:

$$\mathbb{Q}_t(\mathrm{d}x_{0:t}) = \frac{1}{L_t}\left\{G_0(x_0)\prod_{s=1}^{t} G_s(x_{s-1},x_s)\right\}\mathbb{M}_0(\mathrm{d}x_0)\prod_{s=1}^{t} M_s(x_{s-1},\mathrm{d}x_s)$$

where L_t is simply the normalising constant:

$$L_t = \int_{\mathfrak{X}^{T+1}}\left\{G_0(x_0)\prod_{s=1}^{t} G_s(x_{s-1},x_s)\right\}\mathbb{M}_0(\mathrm{d}x_0)\prod_{s=1}^{t} M_s(x_{s-1},\mathrm{d}x_s).$$

(We assume that $0 < L_t < +\infty$.) A good way to think of Feynman-Kac models is that of a sequential change of measure, from the law of the Markov process $\{X_t\}$, to some modified law $\mathbb{Q}_t$, where the modification applied at time t is given by function G_t. In computational terms, one can also think of (sequential) importance sampling: we would like to approximate $\mathbb{Q}_t$ by simulating process $\{X_t\}$, and re-weight realisations at time t by function G_t. Unfortunately the performance of this basic approach would quickly deteriorate with time.

Example 1. Consider a Gaussian auto-regressive process, $X_0 \sim N(0,1)$, $X_t = \phi X_{t-1} + V_t$, $V_t \sim N(0,1)$, for $t \geq 1$, and take $G_t(x_{t-1},x_t) = \mathbb{1}_{\mathbb{R}^+}(x_t)$. Then, if we use sequential importance sampling, the number of simulated trajectories that would get a non-zero weight would decrease quickly with time. In particular, the probability of 'survival' at time t would be $2^{-(t+1)}$ for $\phi = 0$.

The successive distributions $\mathbb{Q}_t$ are related as follows:

$$\mathbb{Q}_t(\mathrm{d}x_{0:t}) = \frac{1}{\ell_t}\mathbb{Q}_{t-1}(\mathrm{d}x_{0:t-1})M_t(x_{t-1},\mathrm{d}x_t)G_t(x_{t-1},x_t) \tag{1}$$

where $\ell_t = L_t/L_{t-1}$. There are many practical settings (as discussed in the next section) where one is interested only in approximating the *marginal* distribution $\mathbb{Q}_t(\mathrm{d}x_t)$, i.e. the marginal distribution of variable X_t relative to the joint distribution $\mathbb{Q}_t(\mathrm{d}x_{0:t})$. One can deduce from (1) the following recursion for these marginals: $\mathbb{Q}_t(\mathrm{d}x_t)$ is the marginal distribution of variable X_t with respect to the bi-variate distribution

$$\mathbb{Q}_t(\mathrm{d}x_{t-1:t}) = \frac{1}{\ell_t}\mathbb{Q}_{t-1}(\mathrm{d}x_{t-1})M_t(x_{t-1},\mathrm{d}x_t)G_t(x_{t-1},x_t). \tag{2}$$

Note the dramatic dimension reduction: the initial definition of $\mathbb{Q}_t$ involved integrals with respect to $\mathfrak{X}^{t+1}$, but with the above recursion one may obtain expectations with respect to $\mathbb{Q}_t(\mathrm{d}x_t)$ by computing $t+1$ integrals with respect to $\mathfrak{X}^2$.

2.3 Feynman-Kac in Practice

The main application of the Feynman-Kac formalism is the filtering of a state-space model (also known as a hidden Markov model). This time, 'model' has its standard (statistical) meaning, i.e. a probability distribution for observed data.

A state-space model involves two discrete-time processes $\{X_t\}$ and $\{Y_t\}$; $\{X_t\}$ is Markov, and unobserved, $\{Y_t\}$ is observed, and is such that variable Y_t conditional on X_t and all (X_s, Y_s), $s \neq t$ depends only on X_t. The standard way to specify this model is through:

1. The initial distribution $\mathbb{P}_0(\mathrm{d}x_0)$ and the Markov kernels $P_t(x_{t-1}, \mathrm{d}x_t)$ that define the law of the process $\{X_t\}$;
2. The probability density $f_t(y_t|x_t)$ of $Y_t|X_t = x_t$.

Example 2. The stochastic volatility model is a state-space model popular in Finance (e.g., [19]). One observes the log-return Y_t of a given asset, which is distributed according to $Y_t|X_t = x_t \sim N(0, e^{x_t})$. The quantity X_t represents the (unobserved) market volatility, and evolves according to an auto-regressive process:

$$X_t - \mu = \phi(X_{t-1} - \mu) + \sigma V_t, \qquad V_t \sim N(0,1).$$

For X_0, one may take $X_0 \sim N\left(\mu, \sigma^2/(1-\phi^2)\right)$ to make the process $\{X_t\}$ stationary.

Example 3. The bearings-only model is a basic model in target tracking, where X_t represents the current position (in $\mathbb{R}^2$) of a target, and Y_t is a noisy angular measurement obtained by some device (such as a radar):

$$Y_t = \arctan\left(\frac{X_t(2)}{X_t(1)}\right) + V_t, \quad V_t \sim N(0, \sigma^2),$$

where $X_t(1), X_t(2)$ denote the two components of vector X_t. There are several standard ways to model the motion of the target; the most basic one is that of a random walk. See e.g. [2] for more background on target tracking.

Filtering is the task of computing the distribution of variable X_t, conditional on the data acquired until time t, $Y_{0:t}$. It is easy to check that, by taking a Feynman-Kac model such that

- the process $\{X_t\}$ has the same distribution as in the considered model; i.e. $\mathbb{M}_0(\mathrm{d}x_0) = \mathbb{P}_0(\mathrm{d}x_0)$, $M_t(x_{t-1}, \mathrm{d}x_t) = P_t(x_{t-1}, \mathrm{d}x_t)$ for any $x_{t-1} \in \mathfrak{X}$;
- the potential functions are set to $G_t(x_{t-1}, x_t) = f_t(y_t|x_t)$;

then one recovers as $\mathbb{Q}_t(\mathrm{d}x_{0:t})$ the distribution of variables $X_{0:t}$, conditional on $Y_{0:t} = y_{0:t}$; in particular $\mathbb{Q}_t(\mathrm{d}x_t)$ is the filtering distribution of the model.

We call this particular Feynman-Kac representation of the filtering problem the bootstrap model. Consider now a Feynman-Kac model with an *arbitrary* distribution for the Markov process $\{X_t\}$, and with potential

$$G_t(x_{t-1},x_t) = \frac{P_t(x_{t-1},\mathrm{d}x_t) f_t(y_t|x_t)}{M_t(x_{t-1},\mathrm{d}x_t)},$$

the Radon-Nikodym derivative of $P_t(x_{t-1},\mathrm{d}x_t) f_t(y_t|x_t)$ with respect to $M_t(x_{t-1},\mathrm{d}x_t)$ (assuming the latter dominates the former). Whenever kernels P_t and M_t admit conditional probability densities (with respect to a common dominating measure), this expression simplifies to:

$$G_t(x_{t-1},x_t) = \frac{p_t(x_t|x_{t-1}) f_t(y_t|x_t)}{m_t(x_t|x_{t-1})}. \tag{3}$$

Then again it is a simple exercise to check that one recovers as $\mathbb{Q}_t(\mathrm{d}x_t)$ the filtering distribution of the considered model. We call any Feynman-Kac model of this form a *guided* model. The bootstrap model corresponds to the special case where $P_t = M_t$.

We shall see in the following section that each Feynman-Kac model generates a different SMC algorithm. Thus, for a given state-space model, we have potentially an infinite number of SMC algorithms that may be used to approximate its sequence of filtering distributions. Which one to choose? We return to this point in Section 2.5.

2.4 Sequential Monte Carlo

Consider a given Feynman-Kac model. Sequential Monte Carlo amounts to compute recursive Monte Carlo approximations to the marginal distributions $\mathbb{Q}_t(\mathrm{d}x_t)$ of that model. At time 0, we simulate $X_0^n \sim \mathbb{M}_0(\mathrm{d}x_0)$ for $n = 1,\ldots,N$, and weight these 'particles' according to function G_0. Then

$$\mathbb{Q}_0^N(\mathrm{d}x_0) = \sum_{n=1}^N W_0^n \delta_{X_0^n}(\mathrm{d}x_0), \qquad W_0^n = \frac{G_0(X_0^n)}{\sum_{m=1}^N G_0(X_0^m)}$$

is an importance sampling approximation of $\mathbb{Q}_0(\mathrm{d}x_0)$, in the sense that

$$\mathbb{Q}_0^N(\varphi) = \sum_{n=1}^N W_0^n \varphi(X_0^n) \approx \mathbb{Q}_0(\varphi)$$

for any suitable test function φ.

To progress to time 1, recall from (2) that

$$\mathbb{Q}_1(\mathrm{d}x_{0:1}) = \frac{1}{\ell_1} \mathbb{Q}_0(\mathrm{d}x_0) M_1(x_0,\mathrm{d}x_1) G_1(x_0,x_1)$$

which suggests to perform importance sampling, with proposal $\mathbb{Q}_0(\mathrm{d}x_0)M_1(x_0,\mathrm{d}x_1)$, and weight function G_1. But since $\mathbb{Q}_0(\mathrm{d}x_0)$ is not available, we use instead $\mathbb{Q}_0^N$: that is, we sample N times from

$$\sum_{n=1}^{N} W_0^n \delta_{X_0^n}(\mathrm{d}x_0) M_1(X_0^n, \mathrm{d}x_1).$$

To do so, for each n, we draw $A_1^n \sim \mathscr{M}(W_0^{1:N})$, the multinomial distribution which generates value m with probability W_0^m; then we sample $X_1^n \sim M_1(X_0^{A_1^n}, \mathrm{d}x_1)$. We obtain in this way N pairs $(X_0^{A_1^n}, X_1^n)$, and we re-weight them according to function G_1. In particular

$$\mathbb{Q}_1^N(\mathrm{d}x_1) = \sum_{n=1}^{N} W_1^n \delta_{X_1^n}(\mathrm{d}x_1), \qquad W_1^n = \frac{G_1(X_0^{A_1^n}, X_1^n)}{\sum_{m=1}^N G_1(X_0^{A_1^m}, X_1^m)}$$

is our approximation of $\mathbb{Q}_1(\mathrm{d}x_1)$.

We proceed similarly at times 2, 3, ...; see Algorithm 1. At every time t, we sample N points from

$$\sum_{n=1}^{N} W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}x_{t-1}) M_t(X_{t-1}^n, \mathrm{d}x_t)$$

and assign weights $W_t^n \propto G_t(X_{t-1}^{A_t^n}, X_t^n)$ to the so-obtained pairs $(X_{t-1}^{A_t^n}, X_t^n)$. Then we may use

$$\sum_{n=1}^{N} W_t^n \varphi(X_t^n)$$

as an approximation of $\mathbb{Q}_t(\varphi)$, for any test function $\varphi : \mathfrak{X} \to \mathbb{R}$. The approximation error of $\mathbb{Q}_t(\varphi)$ converges to zero at rate $\mathscr{O}_P(N^{-1/2})$, under appropriate conditions [10, 7].

Algorithm 1 Generic SMC sampler, for a given Feynman-Kac model

Step 0:

(a) Sample $X_0^n \sim \mathbb{M}_0(\mathrm{d}x_0)$ for $n = 1, \ldots, N$.
(b) Compute weight $W_0^n = G_0(X_0^n) / \sum_{m=1}^N G_0(X_0^m)$ for $n = 1, \ldots, N$.

Recursively, for $t = 1, \ldots, T$:

(a) Sample $A_t^{1:N} \sim \mathscr{M}(W_{t-1}^{1:N})$; see Appendix A.
(b) Sample $X_t^n \sim M_t(X_{t-1}^{A_t^n}, \mathrm{d}x_t)$ for $n = 1, \ldots, N$.
(c) Compute weight $W_t^n = G_t(X_{t-1}^{A_t^n}, X_t^n) / \sum_{m=1}^N G_t(X_{t-1}^{A_t^m}, X_t^m)$ for $n = 1, \ldots, N$.

2.5 Back to State-Space Models

We have explained in Section 2.3 that, for a given state-space model, there is an infinite number of Feynman-Kac models such that $\mathbb{Q}_t(\mathrm{d}x_t)$ is the filtering distribution. Thus, there is also an infinite number of SMC algorithms that may be used to approximate this filtering distribution.

Example 4. The Feynman-Kac model defined in Example 1 is such that $\mathbb{Q}_t(\mathrm{d}x_t)$ is the distribution of X_t conditional on $X_s \geq 0$ for all $0 \leq s \leq t$, where $\{X_t\}$ is a Gaussian auto-regressive process: $X_t = \phi X_{t-1} + V_t$, $V_t \sim N(0,1)$. We may interpret $\mathbb{Q}_t(\mathrm{d}x_t)$ as the filtering distribution of a state-space model, where $\{X_t\}$ is the same auto-regressive process, $Y_t = \mathbb{1}_{\mathbb{R}^+}(X_t)$, and $y_t = 1$ for all t. Consider now the following alternative Feynman-Kac model: $M_t(x_{t-1}, \mathrm{d}x_t)$ is the Normal distribution $N(\phi x_{t-1}, 1)$ truncated to $\mathbb{R}^+$, i.e. the distribution with probability density

$$m_t(x_t|x_{t-1}) = \frac{\varphi(x_t - \phi x_{t-1})}{\Phi(\phi x_{t-1})} \mathbb{1}_{\mathbb{R}^+}(x_t)$$

where φ and Φ are respectively the PDF and CDF of a $N(0,1)$ distribution; and $G_t(x_{t-1}, x_t) = \Phi(\phi x_{t-1})$, as per (3). Again, quick calculations show that we recover exactly the same distributions $\mathbb{Q}_t(\mathrm{d}x_t)$. Hence we have two SMC algorithms that approximate the same sequence of distributions (one for each Feynman-Kac model). Observe however that the latter SMC algorithm simulates all particles directly inside the region of interest ($\mathbb{R}^+$), while the former (bootstrap) algorithm simulates particles 'blindly', and assigns zero weight to those particles that fall outside $\mathbb{R}^+$. As a result, the latter algorithm tends to perform better. Note also that, under both Feynman-Kac formulations, L_t is the probability that $X_s \geq 0$ for all $0 \leq s \leq t$, hence both algorithms may be used to approximate this rare-event probability (see Section 2.8 below), but again the latter algorithm should typically give lower variance estimates for L_t.

Of course, the previous example is a bit simplistic, as far as state-space models are concerned. Recall from Section 2.3 that, for a given state-space model, any Feynman-Kac model such that G_t is set to (3) recovers the filtering distribution of that model for $\mathbb{Q}_t$. The usual recommendation is to choose one such Feynman-Kac model in a way that the variance of the weights of the corresponding SMC algorithm is low. To minimise the variance of the weights at iteration t, one should take [12] the guided Feynman-Kac model such that

$$M_t^{\mathrm{opt}}(x_{t-1}, \mathrm{d}x_t) \propto P_t(x_{t-1}, \mathrm{d}x_t) f_t(y_t|x_t),$$

the distribution of $X_t | \big(X_{t-1} = x_{t-1}, Y_t = y_t\big)$. In words, one should *guide* particles to a part of space $\mathfrak{X}$ where likelihood $x_t \to f_t(y_t|x_t)$ is high.

In fact, in the previous example, the second Feynman-Kac model corresponds precisely to this optimal kernel. Unfortunately, for most models sampling from the optimal kernel is not easy. One may instead derive an easy-to-sample kernel M_t that approximates M_t^{opt} in some way. Again, provided G_t is set to (3), one will recover the exact filtering distribution as $\mathbb{Q}_t$.

Example 5. In Example 2, [25] observed that the bootstrap filter performs poorly at iterations t where the data-point y_t is an outlier (i.e. takes a large absolute value). A potential remedy is to take into account y_t in some way when simulating X_t. To simplify the discussion, take $\mu = 0$, and consider the probability density of $X_t|X_{t-1}, Y_t$:

$$p_t(x_t|x_{t-1}, y_t) \propto \varphi((x_t - \phi x_{t-1})/\sigma)\varphi(y_t; 0, e^{x_t}) \propto \exp\left\{-\frac{1}{2\sigma^2}(x_t - \phi x_{t-1})^2 - \frac{x_t}{2} - \frac{y_t^2}{2e^{x_t}}\right\}.$$

It is not easy to simulate from this density, but [25] suggested to approximate it by linearizing $\exp(-x_t)$ around $x_t = \phi x_{t-1}$: $\exp(-x_t) \approx \exp(-\phi x_{t-1})(1 + \phi x_{t-1} - x_t)$. This leads to proposal density

$$m_t(x_t|x_{t-1}) \propto \exp\left\{-\frac{1}{2\sigma^2}(x_t - \phi x_{t-1})^2 - \frac{x_t}{2} - \frac{y_t^2}{2e^{\phi x_{t-1}}}(1 + \phi x_{t-1} - x_t)\right\}$$

which is clearly Gaussian (and hence easy to simulate from). Note that this linear 'approximation' does not imply that the resulting SMC algorithm is approximate in some way: provided G_t is set to (3), the resulting algorithm targets exactly the filtering distribution of the model, as we have already discussed.

2.6 Sequential quasi-Monte Carlo

2.6.1 QMC Basics

As mentioned in the introduction, we assume that the reader is already familiar with QMC and RQMC (randomised QMC); otherwise see e.g. the books of [21] and [22]. We only recall briefly the gist of QMC. Consider an expectation with respect to $\mathscr{U}\left([0,1]^d\right)$, and its standard Monte Carlo approximation:

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

where the U^n are IID variables. QMC amounts to replacing the U^n by N deterministic points $u^{n,N}$ that have low discrepancy. The resulting error converges faster than with Monte Carlo under certain conditions, in particular regarding the *regularity* of function φ. This is an important point when it comes to apply QMC in practice: rewriting a given algorithm as a deterministic function of uniforms, and replacing these uniforms by a QMC point set, may not warrant better performance. One has also to make sure that this deterministic function is indeed regular, and maintain low discrepancy in some sense.

2.6.2 SQMC when $d = 1$

We explained in Section 2.4 that SMC amounts to a sequence of importance sampling steps, with proposal distribution

$$\sum_{n=1}^{N} W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}x_{t-1}) M_t(x_{t-1}, \mathrm{d}x_t) \tag{4}$$

at time t. To derive a QMC version of this algorithm, we must find a way to generate a low-discrepancy sequence with respect to this distribution. The difficulty lies in the fact that the support of (4) is partly discrete (the choice of the ancestor X_{t-1}^n), partly continuous (the kernel $M_t(x_{t-1}, \mathrm{d}x_t)$). We focus on the discrete part below. For the continuous part, we assume that $\mathfrak{X} \subset \mathbb{R}^d$, and that we know of a function $\Gamma_t : \mathfrak{X} \times [0,1]^d \to \mathfrak{X}$ such that, for any $x_{t-1} \in \mathfrak{X}$, $\Gamma_t(x_{t-1}, U)$, $U \sim \mathscr{U}\left([0,1]^d\right)$, has the same distribution as $M_t(x_{t-1}, \mathrm{d}x_t)$. The choice of Γ_t is model-dependent, and is often easy; the default choice would be the Rosenblatt transform associated to $M_t(x_{t-1}, \mathrm{d}x_t)$ (the multivariate inverse CDF).

Example 6. Consider a state-space model with latent process $X_t = \phi X_{t-1} + V_t$, $V_t \sim N(0, \sigma^2)$. Then one would take typically $\Gamma_t(x_{t-1}, u) = \phi x_{t-1} + \sigma \Phi^{-1}(u)$, where Φ is the CDF of a $N(0,1)$ distribution. In dimension $d > 1$, such a process would take the form $X_t = AX_{t-1} + V_t$, $V_t \sim N(0, \Sigma)$, where A is a $d \times d$ matrix. Then one would define $\Gamma_t(x_{t-1}, u) = Ax_{t-1} + \Pi_\Sigma(u)$, where the second term may be defined in several ways; e.g. (a) $\Pi_\Sigma(u)$ is the Rosenblatt transform of $N(0, \Sigma)$, i.e. first component of $\Pi_\Sigma(u)$ is $\Sigma_{11}^{1/2} \Phi^{-1}(u_1)$ and so on; or (b) $\Pi_\Sigma(u) = C\boldsymbol{\Phi}^{-1}(u)$, where C is the Cholesky lower triangle of Σ, $CC^T = \Sigma$, and $\boldsymbol{\Phi}^{-1}$ is the function which assigns to vector u the vector $\left(\Phi^{-1}(u(1)), \ldots, \Phi^{-1}(u(d))\right)^T$. In both cases, function Γ_t depends on the order of the components of X_t.

We now focus on the discrete component of (4). The standard approach to sample from such a finite distribution is the inverse CDF method: define $F_{t-1}^N(x) = \sum_{n=1}^N W_{t-1}^n \mathbb{1}\{n \le x\}$, and set $\hat{X}_{t-1}^n = X_{t-1}^{A_t^n}$ with $A_t^n = \left(F_{t-1}^N\right)^{-1}(U_t^n)$, where $U_t^n \sim \mathscr{U}([0,1])$ and $\left(F_t^N\right)^{-1}$ is the generalised inverse of F_t^N. This is precisely how resampling is implemented in a standard particle filter. See Appendix A for a description of the standard algorithm to evaluate in $\mathscr{O}(N)$ time function $\left(F_{t-1}^N\right)^{-1}$ for N inputs.

A first attempt at introducing a QMC point set would be to set again $A_t^n = \left(F_t^N\right)^{-1}(U_t^n)$, but taking this time for U_t^n the first component of a QMC point set (of dimension $d+1$). The problem with this approach is that this defines a transformation, from the initial uniforms to the points, which is quite irregular. In fact, since the labels of the N particles are arbitrary, this distribution somehow involves a *random permutation* of the N initial points. In other terms, we add some noise in our transformation, which is not a good idea in any type of QMC procedure.

Now consider the special case $\mathfrak{X} \subset \mathbb{R}$, and let $\sigma_{t-1} = \mathrm{argsort}(X_{t-1}^{1:N})$, i.e. σ_{t-1} is a permutation of the N first integers such that:

$$X_{t-1}^{\sigma_{t-1}(1)} \le \ldots \le X_{t-1}^{\sigma_{t-1}(N)}$$

and, for $x \in \mathfrak{X}$, let

$$\hat{F}_{t-1}^N(x) = \sum_{n=1}^N W_{t-1}^n \mathbb{1}\left\{X_{t-1}^n \leq x\right\} = \sum_{n=1}^N W_{t-1}^{\sigma_{t-1}(n)} \mathbb{1}\left\{X_{t-1}^{\sigma_{t-1}(n)} \leq x\right\}.$$

Note that $\hat{F}_{t-1}^N$ does not depend on the *labels* of the N ancestors (like F_{t-1}^N does); for instance, the smallest x such that $\hat{F}_{t-1}^N(x) > 0$ is $X_{t-1}^{\sigma_{t-1}(1)}$, the smallest ancestor (whatever its label).

The first main idea in SQMC is to choose A_t^n such that $X_{t-1}^{A_t^n} = \hat{F}_{t-1}^N(U_t^n)$, where U_t^n is the first component of some QMC or RQMC point set. In this way, the resampled ancestors, i.e. the points $X_{t-1}^{A_t^n}$, may be viewed as a low-discrepancy point set with respect to the marginal distribution of component x_{t-1} in distribution (4). In practice, computing A_t^n amounts to (a) sort the N ancestors; and (b) apply the inverse CDF algorithm of Appendix A to these N sorted ancestors.

2.6.3 SQMC for $d > 1$

When $\mathfrak{X} \subset \mathbb{R}^d$, with $d > 1$, it is less clear how to invert the empirical CDF of the ancestors

$$\hat{F}_{t-1}^N(x) = \sum_{n=1}^N W_{t-1}^n \mathbb{1}\left\{X_t^n \leq x\right\}$$

as this function is $\mathbb{R}^d \to [0,1]$.

The second main idea in SQMC is to transform the N ancestors X_{t-1}^n into N scalars Z_{t-1}^n, in a certain way that maintains the low discrepancy of the N initial points. Then we may construct a QMC point relative to

$$\hat{F}_{t-1,h}^N(z) = \sum_{n=1}^N W_{t-1}^n \delta_{Z_{t-1}^n}(\mathrm{d}z), \quad z \in [0,1]$$

in the same way as described in the previous section.

To do so, we take $Z_{t-1}^n = h \circ \psi(X_{t-1}^n)$, where $h : [0,1]^d \to [0,1]$ is the inverse of the Hilbert curve, see below, and $\psi : \mathfrak{X} \to [0,1]^d$ is model-dependent. (For instance, if $\mathfrak{X} = \mathbb{R}^d$, we may apply a component-wise version of the logistic transform.)

The Hilbert curve is a space-filling curve, that is a function $H : [0,1] \to [0,1]^d$ with the following properties: it is defined as the limit of the process depicted in Figure 1; it is Hölder with coefficient $1/d$ (in particular it is continuous); it 'fills' entirely $[0,1]^d$; the set of points in $[0,1]^d$ that admit more than one pre-image is of measure 0. Thanks to these properties, it is possible to define a pseudo-inverse $h : [0,1] \to [0,1]^d$, such that $H \circ h(u) = u$ for $u \in [0,1]$.

In addition, the pseudo-inverse h maintains low-discrepancy in the following sense: if the N ancestors X_{t-1}^n are such that $\|\pi^N - \pi\|_E \to 0$ where $\pi^N(\mathrm{d}x) = \sum_{n=1}^N W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}x)$, and π is some limiting probability distribution, then (under

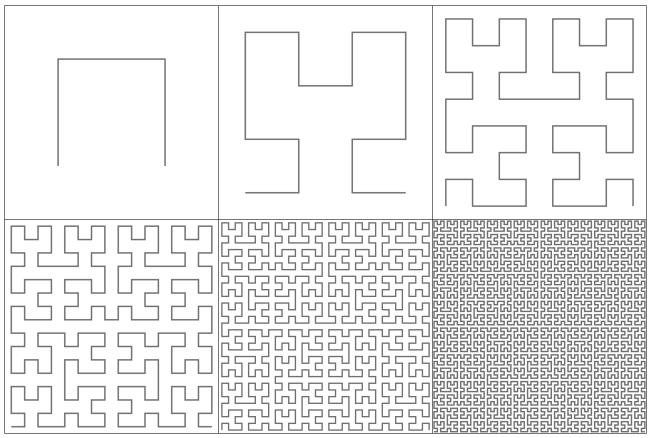

Fig. 1 Sequence of curves of which the Hilbert curve is the limit, for $d = 2$ (Source: Wikipedia)

appropriate conditions, see Theorem 3 in [16]), $\|\pi_h^N - \pi_h\|_E \to 0$, where π_h^N and π_h are the images of π^N and π through h. The extreme norm $\|\cdot\|_E$ in this theorem is some generalisation of the QMC concept of extreme discrepancy; again see [16] for more details.

We note that other functions $[0,1]^d \to [0,1]$ (e.g. pseudo-inverse of other space-filling curves, such as the Lebesgue curve) could be used in lieu of the inverse of the Hilbert curve. However, our impression is that other choices would not necessarily share the same property of "maintaining low discrepancy". At the very least, our proofs in [16] rely on properties that are specific to the Hilbert curve, and would not be easily extended to other functions.

Algorithm 2 summarises the operations performed in SQMC.

Algorithm 2 SQMC algorithm

At time 0,

(a) Generate a QMC point set $u_0^{1:N}$ of dimension d.
(b) Compute $X_0^n = \Gamma_0(u_0^n)$ for all $n \in 1:N$.
(b) Compute $W_0^n = G_0(X_0^n)/\sum_{m=1}^N G_0(X_0^m)$ for all $n \in 1:N$.

Recursively, for time $t = 1:T$,

(a) Generate a QMC or RQMC point set $(u_t^{1:N}, v_t^{1:N})$ of dimension $d+1$ (u_t^n being the first component, and v_t^n the vector of the d remaining components, of point n).
(b) Hilbert sort: find permutation σ_t such that $h \circ \psi(X_{t-1}^{\sigma_t(1)}) \leq \ldots \leq h \circ \psi(X_{t-1}^{\sigma_t(N)})$ if $d \geq 2$, or $X_{t-1}^{\sigma(1)} \leq \ldots \leq X_{t-1}^{\sigma(N)}$ if $d = 1$.
(c) Generate $A_t^{1:N}$ using Algorithm 3, with inputs $\text{sort}(u_t^{1:N})$ and $W_t^{\sigma(1:N)}$, and compute $X_t^n = \Gamma_t(X_{t-1}^{\sigma_t(A_t^n)}, v_t^n)$.
(e) Compute $W_t^n = G_t(X_{t-1}^{\sigma(A_t^n)}, X_t^n)/\sum_{m=1}^N G_t(X_{t-1}^{\sigma(A_t^m)}, X_t^m)$ for all $n \in 1:N$.

2.7 Connection to Array-RQMC

In the Feynman-Kac formalism, taking $G_0(x_0) = 1$, $G_t(x_{t-1}, x_t) = 1$ for all $t \geq 1$, makes $\mathbb{Q}_t$ the distribution of the Markov chain $\{X_t\}$. In that case, SQMC may be used to approximate expectations with respect to the distribution of that Markov chain. In fact, such a SQMC algorithm may be seen as a certain version of the array-RQMC algorithm of [20], where the particles are ordered at every iteration using the inverse of the Hilbert curve. In return, the convergence results established in [16] apply to that particular version of array-RQMC.

Although designed initially for a smaller class of problems, array-RQMC is built on the same insight as SQMC of viewing the problem of interest not a single Monte Carlo exercise, of dimension $d(T+1)$ (e.g. simulating a Markov chain in $\mathfrak{X} \subset R^d$ over $T+1$ time steps), but as $T+1$ exercises of dimension $d+1$. See also [14] for a related idea in the filtering literature.

2.8 Extensions

In state-space modelling, one may be interested in computing other quantities than the filtering distributions: in particular the likelihood of the data up to t, $p_t(y_{0:t})$, and the smoothing distribution, i.e. the joint law of the states $X_{0:T}$, given some complete dataset $Y_{0:T}$.

The likelihood of the data $p_t(y_{0:t})$ equals the normalising constant L_t in any guided Feynman-Kac model. This quantity may be estimated at iteration t as follows:

$$L_t^N = \left(\frac{1}{N} \sum_{n=1}^{N} G_0(X_0^n) \right) \prod_{s=1}^{t} \left(\frac{1}{N} \sum_{n=1}^{N} G_s(X_{s-1}^{A_s^n}, X_s^n) \right).$$

A non-trivial property of SMC algorithms is that this quantity is an unbiased estimate of L_t [8]. This makes it possible to develop MCMC algorithms for parameter estimation of state-space models which (a) runs at each MCMC iteration a particle filter to approximate the likelihood at given value of the parameter; and yet (b) targets the exact posterior distribution of the parameters, despite the fact the likelihood is computed only approximately. The corresponding PMCMC (particle MCMC) algorithms have been proposed in the influential paper of [1]. If we use RQMC (randomised QMC) point steps within SQMC, then L_t^N remains an unbiased estimate of L_t. Thus, SQMC is compatible with PMCMC (meaning that one may use SQMC instead of SMC at every iteration of a PMCMC algorithm), and in fact one may improve the performance of PMCMC in this way; see [16] for more details.

Smoothing is significantly more difficult than filtering. Smoothing algorithms usually amount to (a) run a standard particle filter, forward in time; (b) run a second algorithm, which performs some operations on the output of the first algorithm, backward in time. Such algorithms have complexity $O(N^2)$ in general. We refer the readers to [3], [13] for a general presentation of smoothing algorithms, and to [15]

for how to derive QMC smoothing algorithms that offer better performance than standard (Monte Carlo-based) smoothing algorithms.

Finally, we mention that SMC algorithms may also be used in other contexts that the sequential inference of state-space models. Say we wish to approximate expectations with respect to some distribution of interest π, but it is is difficult to sample directly from π (e.g. the density π is strongly multimodal). One may define a geometric bridge between some easy to sample distribution π_0 and π as follows: $\pi_t(x) \propto \pi_0(x)^{1-\gamma_t}\pi(x)^{\gamma_t}$ where $0 = \gamma_0 < \ldots < \gamma_T = 1$. Then one may apply SMC to the sequence (π_t), and use the output of the final iteration to approximate π. Other sequence of distributions may be considered as well. For more background on such applications of SMC see e.g. [23], [6], and [9]. The usefulness of SQMC for such problems remains to be explored.

2.9 A Note on the Impact of the Dimension

[16] include a numerical study of the impact of the dimension on the performance of SQMC. It is observed that the extra performance of SQMC (relative to standard SMC) quickly decreases with the dimension.

Three factors may explain this curse of dimensionality:

1. The inherent curse of dimensionality of QMC: the standard discrepancy bounds invoked as a formal justification of QMC deteriorate with the dimension.
2. Regularity of the Hilbert curve: the Hilbert curve is Hölder with coefficient $1/d$. Consequently, the mapping $u_t^n \mapsto X_{t-1}^{\sigma_t(A_{t-1}^n)}$ induced by steps (a) and (b) of Algorithm 2 for time $t \geq 1$ is less and less regular as the dimension increases. (We however believe that this property is not specific to the use of the Hilbert curve but is due to the resampling mechanism itself, where a single point in $u_t^n \in [0,1]$ is used to select the d-dimensional ancestor $X_{t-1}^{A_t^n}$.)
3. SMC curse of dimensionality: SMC methods also suffer from the curse of dimensionality, for the simple reason that they rely on importance sampling: the larger the dimension, the greater the discrepancy between the proposal distribution and the target distribution. In practice, one observes in high-dimensional filtering problem that, at each iteration, only a small proportion of the particles get a non-negligible weight.

We thought earlier that factor 2 was the 'main culprit'. However, factor 3 seems to play an important part as well. To see this, we compare below the relative performance of SQMC and SMC for the filtering of the following class of linear Gaussian state-space models (as in [17]): $X_0 \sim N_d(0, I_d)$, and

$$\begin{aligned} X_t &= FX_{t-1} + V_t, \qquad V_t \sim N_d(0, I_d), \\ Y_t &= X_t + W_t, \qquad W_t \sim N_d(0, I_d), \end{aligned}$$

with $F = (\alpha^{|i-j|})_{i,j=1:d}$, and $\alpha = 0.4$. For such models, the filtering distribution may be computed exactly using the Kalman filter [18]. We consider two Feynman-Kac formalisms of that problem:

- The bootstrap formalism, where M_t is set to $N_d(FX_{t-1}, I_d)$, the distribution of $X_t|X_{t-1}$ according to the model, and $G_t(x_{t-1}, x_t) = f_t(y_t|x_t) = N_d(y_t; x_t, I_d)$, the probability density at point y_t of distribution $N_d(x_{t-1}, I_d)$.
- The 'optimal' guided formalism where

$$M_t(x_{t-1}, \mathrm{d}x_t) \propto P_t(x_{t-1}, \mathrm{d}x_t) f_t(y_t|x_t) \sim N_d\left(\frac{Y_t + FX_{t-1}}{2}, \frac{1}{2} I_d\right)$$

  and, by (3),

$$G_t(x_{t-1}, x_t) = N_d(y_t; Fx_{t-1}, 2I_d)$$

  the probability density at point y_t of distribution $N_d(Fx_{t-1}, 2I_d)$.

In both cases, as already explained, we recover the filtering distribution as $\mathbb{Q}_t$. But the latter formalism is chosen so as to minimise the variance of the weights at each iteration.

We simulate $T = 50$ data-points from the model, for $d = 5$, 10, 15 and 20. Figure 2 compares the following four algorithms: SMC-bootstrap, SQMC-bootstrap, SMC-guided, and SQMC-guided. The comparison is in terms of the MSE (mean square error) of the estimate of the filtering expectation of the first component of X_t, i.e. $\mathbb{E}[X_t(1)|Y_{0:t} = y_{0:t}]$. We use SMC-guided as the reference algorithm, and we plot for each of the three other algorithms the variations of the gain (MSE of reference algorithm divided by MSE of considered algorithm) for the T estimates. (We use violin plots, which are similar to box-plots, except that the box is replaced by kernel density estimates.) A gain g means that the considered algorithm would need g times less particles (roughly) to provide an estimate with a similar variance (to that of the reference algorithm). Each algorithm was run with $N = 10^4$.

First, we observe that guided algorithms outperforms bootstrap algorithms more and more as the dimension increases. Second, for bootstrap algorithms, the performance between SMC and SQMC is on par as soon as $d \geq 10$. (In fact, the performance is rather bad in both cases, owning to the aforementioned curse of dimensionality.) On the other hand, for guided formalisms we still observe a gain of order $\mathscr{O}(10^1)$ (resp. $10^{0.5}$) for $d = 10$ (resp. $d = 20$).

The bottom line is that the amount of extra performance brought by SQMC (relative to SMC) depends strongly on the chosen Feynman-Kac formalism. If one is able to construct a Feynman-Kac formalism (for the considered problem) that leads to good performance for the corresponding SMC algorithm (meaning that the variance of the weights is low at each iteration), then one may expect significant extra performance from SQMC, even in high dimension.

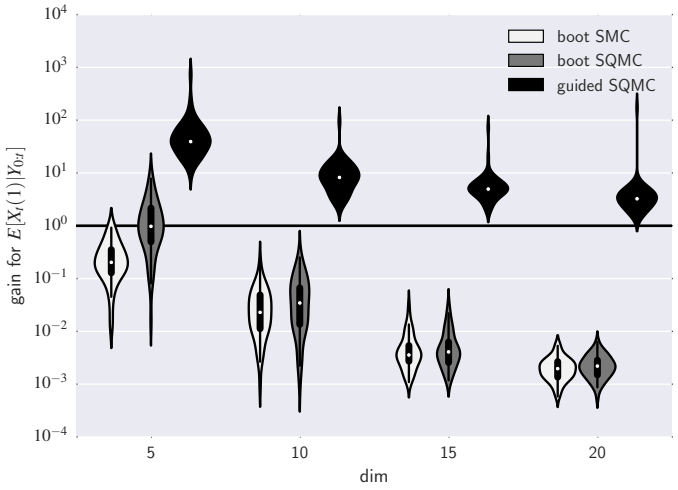

Fig. 2 Violin plots of the gains of the considered algorithms when estimating the filtering expectations $\mathbb{E}[X_t(1)|Y_{0:t}]$ for $t = 0,\ldots,T = 50$. (Each violin plot represents the variability of the T gains for these T estimates.) Gain is MSE (mean square error) of reference algorithm (guided-SMC) divided by MSE of considered algorithm.

3 Application to Diffusions

3.1 Dimension Reduction in SQMC

We start this section by a basic remark, which makes it possible to improve the performance of SQMC when applied to models having a certain structure. We explained in Section 2.4 that SMC amounts to performing importance sampling at every step, using as a proposal distribution:

$$\sum_{n=1}^{N} W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}x_{t-1}) M_t(x_{t-1},\mathrm{d}x_t) \tag{5}$$

and as a target distribution, the same distribution times $G_t(x_{t-1},x_t)$ (up to a constant). We used this remark to derive SQMC as an algorithm that constructs a low-discrepancy point-set with respect to the distribution above; i.e. to construct N points $\left(X_{t-1}^{A_t^n}, X_t^n\right)$, the empirical distribution of which approximates well (5).

Now consider a situation where we know of a function $\Lambda : \mathcal{X} \to \mathbb{R}^k$, with $k < d$, such that (a) G_t depends only on X_t and $\Lambda(X_{t-1})$; and Markov kernel $M_t(x_{t-1},\mathrm{d}x_t)$ also depends only on $\Lambda(x_{t-1})$. (In particular, it is possible to simulate X_t conditional on X_{t-1}, knowing only $\Lambda(X_{t-1})$.) In that case, one may define the same importance

sampling operation on a lower-dimensional space. In particular, the new proposal distribution would be:

$$\sum_{n=1}^{N} W_{t-1}^n \delta_{\Lambda(X_{t-1}^n)}(\mathrm{d}\lambda_{t-1}) M_t^{\Lambda}(\lambda_{t-1}, \mathrm{d}x_t)$$

where $M_t^{\Lambda}(\lambda_{t-1}, \mathrm{d}x_t)$ is simply the Markov kernel which associates distribution $M_t(x_{t-1}, \mathrm{d}x_t)$ to any x_{t-1} such that $\Lambda(x_{t-1}) = \lambda_{t-1}$. We may use exactly the same ideas as before, i.e. generate a QMC point of dimension $d+1$, and use the first component to pick the ancestor. However, the Hilbert sorting is now applied to the N points $\Lambda(X_{t-1}^n)$, and therefore operates in a smaller dimension. Thus one may expect better performance, compared to the standard version of SQMC.

This remark is related somehow to the QMC notion of "effective dimension": the performance of QMC may remain good in high-dimensional problems, if one is able to reformulate the problem in such a way that it depends "mostly" (or in our case, "only") on a few dimensions of the state-space.

3.2 Filtering of Diffusion Processes

We now consider the general class of diffusion-driven state-space models:

$$\begin{aligned} \mathrm{d}\widetilde{X}_t &= \mu_X(\widetilde{X}_t) + \sigma_X(\widetilde{X}_t)\mathrm{d}W_t^X \\ \mathrm{d}\widetilde{Y}_t &= \mu_Y(\widetilde{X}_t) + \sigma_Y(\widetilde{X}_t)\mathrm{d}W_t^Y \end{aligned}$$

where $(W_t^X)_{t\geq 0}$ and $(W_t^Y)_{t\geq 0}$ are possibly correlated Wiener processes. Functions μ_X, μ_Y, σ_X and σ_Y may also depend on t, and μ_Y, σ_Y may also depend on Y_t, but for the sake of exposition we stick to the simple notations above.

Filtering in continuous time amounts to recover the distribution of $\widetilde{X}_t$ conditional on trajectory $y_{[0:t]}$ (i.e. the observation of process $\{\widetilde{Y}_t\}$ over interval $[0,t]$). However, in most practical situations, one does not observe process $\{\widetilde{Y}_t\}$ continuously, but on a grid. To simplify, we assume henceforth that process $\{\widetilde{Y}_t\}$ is observed at times $t \in \mathbb{N}$ and we rewrite the above model as

$$\begin{aligned} \mathrm{d}\widetilde{X}_t &= \mu_X(\widetilde{X}_t) + \sigma_X(\widetilde{X}_t)\mathrm{d}W_t^X \\ \widetilde{Y}_{t+1} &= \widetilde{Y}_t + \int_t^{t+1} \mu_Y(\widetilde{X}_s)\mathrm{d}s + \int_t^{t+1} \sigma_Y(\widetilde{X}_s)\mathrm{d}W_s^Y . \end{aligned} \tag{6}$$

It is typically too difficult to work directly in continuous time. Thus, as standardly done when dealing with such processes, we replace the initial process $(\widetilde{X}_t)$ by its (Euler-) discretized version $\{X_t\}$, with discretisation step $\delta = 1/M$, $M \geq 1$. That is, $\{X_t\}$ is a $\mathbb{R}^M-$valued process, where X_t is a M-dimensional vector representing the original process at times t, $t+1/M$, ..., $t+1-1/M$, which is defined as:

$$\begin{aligned} X_t(1) &= X_{t-1}(M) + \delta\mu_X\left(X_{t-1}(M)\right) + \sigma_X(X_{t-1}(M))\left\{W^X_{t+\delta} - W^X_t\right\} \\ &\vdots \\ X_t(M) &= X_t(M-1) + \delta\mu_X\left(X_t(M-1)\right) + \sigma_X(X_t(M-1))\left\{W^X_{t+1} - W^X_{t+1-\delta}\right\} \end{aligned} \tag{7}$$

and the resulting dicretization of (6) is given by

$$Y_{t+1} = Y_t + \delta\sum_{m=1}^{M}\mu_Y\left(X_t(m)\right) + \sum_{m=1}^{M}\sigma_Y\left(X_t(m)\right)\left\{W^Y_{t+\delta m} - W^Y_{t+\delta(m-1)}\right\}. \tag{8}$$

SQMC may be applied straightforwardly to the filtering of the discretized model defined by (7) and (8). However, the choice of the $\delta = 1/M$ becomes problematic. We would like to take M large, to reduce the discretization bias. But M is also the dimension of the state-space, so a large M may mean a degradation of performance for SQMC (relative to SMC).

Fortunately, the dimension reduction trick of the previous section applies here. For simplicity, consider the bootstrap Feynman-Kac formalism of this particular state-space model:

- $M_t(x_{t-1}, \mathrm{d}x_t)$ is the distribution of $X_t|X_{t-1}$ defined by (7); observe that it only depends on $X_{t-1}(M)$, the last component of X_{t-1};
- $G_t(x_{t-1}, x_t)$ is the probability density of datapoint y_t given $X_t = x_t$ and $Y_{t-1} = y_{t-1}$, induced by (7)-(8); observe that it does not depend on x_{t-1} when (W^X_t) and (W^Y_t) are uncorrelated and that it depends on x_{t-1} only through $x_{t-1}(M)$ when these two processes are correlated (see the next subsection).

Hence we may define $\Lambda(x_{t-1}) = x_{t-1}(M) \in \mathbb{R}$. The Hilbert ordering step may be applied to the values $Z^n_t = X^n_{t-1}(M)$. In fact, since these values are scalars, there is no need to implement any Hilbert ordering, a standard sorting is enough.

3.3 QMC and Brownian Motion

We now briefly discuss how to choose Γ_t, the deterministic function such that $\Gamma_t(x_{t-1}, v)$, for $x_{t-1} \in \mathfrak{X}$ and $v \in [0,1]^d$, returns a variate from kernel $M_t(x_{t-1}, \mathrm{d}x_t)$.

The distribution of $X_t|X_{t-1}$ defined in the previous section is a simple linear transform of the distribution of a Brownian path on a regular grid. Thus, defining function Γ_t amounts to constructing a certain function $[0,1]^M \to \mathbb{R}^M$ that transforms $\mathscr{U}\left([0,1]^M\right)$ into the joint distribution of $(W^X_{t+\delta}, \ldots, W^X_{t+1})$, conditional on W^X_t.

It is well known in the QMC literature (e.g. Section 8.2 of [22]) that there is more than one way to write the simulation of a Brownian path as a function of uniforms, and that the most obvious way may perform poorly when applied in conjunction with QMC. More precisely, consider the following two approaches:

1. Forward construction: simulate independently the increments $W^X_{t+\delta m} - W^X_{t+\delta(m-1)}$ from a $N(0,\delta)$ distribution.

2. Brownian bridge construction [4]: Simulate $(W^X_{t+\delta},\ldots,W^X_{t+1})$ given W^X_t sequentially according to the Van der Corput sequence: $W^X_{t+\delta\lceil M/2\rceil}$, $W^X_{t+\delta\lceil M/4\rceil}$, $W^X_{t+\delta\lceil 3M/4\rceil}$ until all the components of vector $(W^X_{t+\delta},\ldots,W^X_{t+1})$ are simulated. For instance, for $s<t'<u$, we use

$$W^X_{t'}|W^X_s,W^X_u \sim N_1\left(\frac{u-t'}{u-s}W^X_s+\frac{t'-s}{u-s}W^X_u,\frac{(u-t')(t'-s)}{u-s}\right)$$

and the fact that (W^X_t) is a Markov process (i.e. $W^X_t|W^X_s$ does not depend on $W^X_{s'}$ for $s'<s$).

In both cases, it is easy to write the simulation of $(W^X_{t+\delta},\ldots,W^X_{t+1})$ as a function of M uniform variates. However, in the first case, the obtained function depends in the same way on each of the M variates, while in the second case, the function depends less and less on the successive components. This mitigates the inherent curse of dimensionality of QMC [4].

We shall observe the same phenomenon applies to SQMC; even so for a moderate value of M, interestingly. We also mention briefly the PCA (principal components analysis) construction as another interesting way to construct Brownian paths, and refer again to Section 8.2 [22] for a more in-depth discussion of QMC and Brownian paths.

Lastly, although we focus on univariate diffusion processes in this section for the sake of simplicity, the above considerations also hold for multivariate models. Notably, the Brownian bridge construction is easily generalizable to the case where (W^X_t) is a d-dimensional vector of correlated Wiener processes. The dimension of the QMC point set used as input of SQMC is then of size $dM+1$ and the Hilbert ordering would operate on a d-dimensional space.

3.4 Numerical Experiments

To illustrate the discussion of the previous subsections we consider the following diffusion driven stochastic volatility model (e.g.[5])

$$\begin{aligned}\mathrm{d}\widetilde{X}_t &= \left\{\kappa(\mu^X-e^{\widetilde{X}_t})e^{-\widetilde{X}_t}-0.5\omega^2e^{-X_t}\right\}\mathrm{d}t+\omega e^{-\widetilde{X}_t/2}\mathrm{d}W^X_t\\ \widetilde{Y}_{t+1} &= \widetilde{Y}_t+\int_t^{t+1}\left\{\mu^Y+\beta e^{\widetilde{X}_z}\right\}\mathrm{d}z+\int_t^{t+1}e^{\widetilde{X}_s/2}\mathrm{d}W^Y_s\end{aligned}$$

where (W^X_t) and (W^Y_t) are Wiener processes with correlation coefficient $\rho\in(-1,1)$, $\omega>0$, $\kappa>0$ while the other parameters μ^Y, β are in $\mathbb{R}$.

To fit this model into the bootstrap Feynman-Kac formalism that we consider in this section, note that, for $t\geq 0$,

$$\widetilde{Y}_{t+1}|\widetilde{Y}_t,\widetilde{X}_{[t,t+1]} \sim N\Big(\widetilde{Y}_t+\mu^Y+\beta\sigma^2_{t+1}+\rho Z_{t+1},\ (1-\rho^2)\sigma^2_{t+1}\Big)$$

with $\sigma^2_{t+1}=\int_t^{t+1} e^{\widetilde{X}_s}\mathrm{d}s$ and $Z_{t+1}=\int_t^{t+1} e^{\widetilde{X}_s/2}\mathrm{d}W^X_s$, and thus, as explained in Section 3.2,

$$\begin{aligned}G_t(x_{t-1},x_t) &= \tilde{G}_t(x_{t-1}(M),x_t)\\ &:= N\Big(\widetilde{Y}_{t+1};\widetilde{Y}_t+\mu^Y+\beta\hat{\sigma}^2_{t+1}(x_t)+\rho\hat{Z}_{t+1}(x_{t-1}(M),x_t),\ (1-\rho^2)\hat{\sigma}^2_{t+1}(x_t)\Big)\end{aligned}$$

where

$$\hat{\sigma}^2_{t+1}(x_t)=\frac{1}{M}\sum_{m=1}^M e^{x_t(m)},\quad \hat{Z}_{t+1}(x_{t-1}(M),x_t)=\sum_{m=1}^M e^{\frac{x_t(m)}{2}}\big(W^X_{t+m\delta}-W^X_{t+(m-1)\delta}\big).$$

Note that $W^X_{t+m\delta}-W^X_{t+(m-1)\delta}$ depends on $(x_{t-1}(M),x_t)$ through (7). To complete the model we take for $\mathbb{M}_0(\mathrm{d}x_0)$, the initial distribution of process $\{X_t\}$, the density of the $N\big(\mu^X,\omega^2/(2\kappa)\big)$ distribution.

We set the parameters of the model to their estimated values for the daily return data on the closing price of the S&P 500 index from 5/5/1995 to 4/14/2003 [5] and simulate observations $\{Y_t\}_{t=0}^T$ using the discretized model (7)-(8) with $M=20\,000$. The number of observations T is set to 4 000.

Below we compare SMC with SQMC based on the forward construction and on the Brownian bridge construction of Brownian paths. In both cases, SQMC is implemented using as input a nested scrambled [24] Sobol' sequence. The performance of these three algorithms is compared, for $t=1,\ldots,T$, for the estimation of (1) the filtering expectation $\mathbb{E}[X_t|Y_{0:t}]$ and (2) of the log-likelihood function $\log(L_t)$.

Figure 3 shows the ratio of the SMC variance over the SQMC variance for the two alternative implementations of SQMC. Results are presented for a discretization grid of size $M=5$ and for different number of particles N. Two observations are worth noting from this figure. First, the two versions of SQMC outperform SMC in terms of variance. Second, the variance reduction is much larger with the Brownian bridge construction than with the forward construction of Brownian paths, as expected from the discussion of the previous subsection. Note that for both versions of SQMC the ratio of variances increases with the number of particles, showing that SQMC converges faster than the $N^{-1/2}$ Monte Carlo error rate.

In Figure 4 we perform the same analysis than in Figure 3 but now with $M=10$ and $M=20$ discretization steps. ($M=10$ is considered as sufficient for parameter estimation by [5].) Results are presented only for the Brownian bridge construction. Despite the large dimension of the QMC point set used as input, we observe that SQMC converges much faster than the $N^{-1/2}$ Monte Carlo error rate. In particular, we observe that the gains in term of variance brought by SQMC are roughly similar whatever the choice of M is. As explained above, this observation suggests that the effective dimension of the model remains low (or even constant in the present setting) even when the "true" dimension M increases.

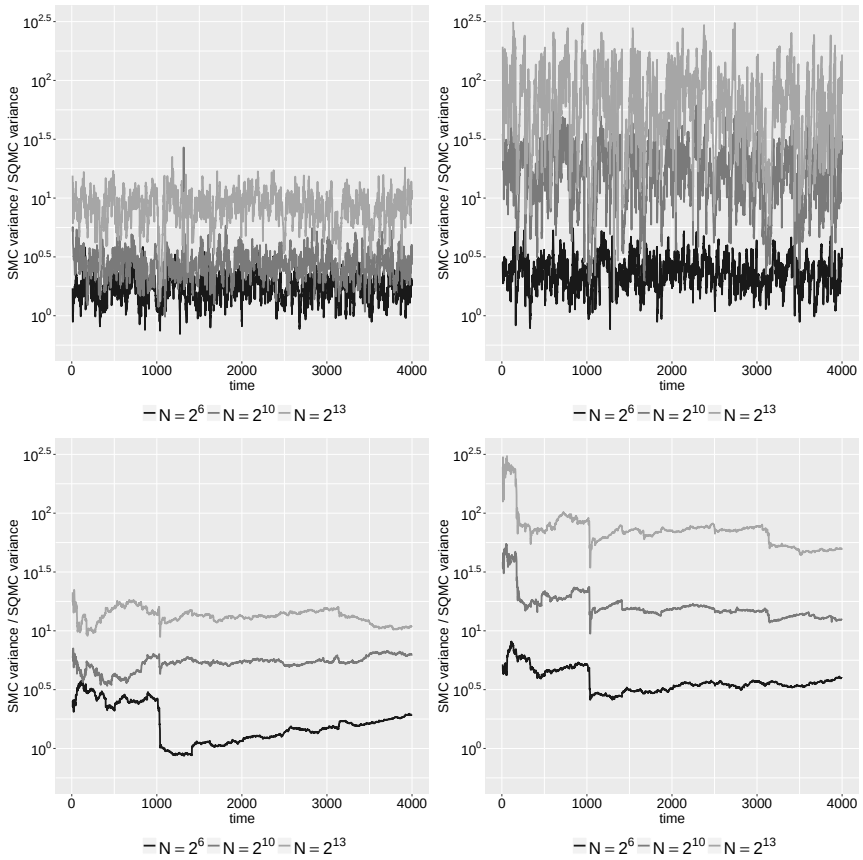

Fig. 3 Estimation of $\mathbb{E}[X_t|Y_{0:t} = y_{0:t}]$ (top plots) and of $\log p(y_{0:t})$ for $t \in \{0, \ldots, T\}$ and for different values of N. SQMC is implemented with the forward construction (left plots) and with the Brownian Bridge construction of Brownian paths (right plots), and $M = 5$

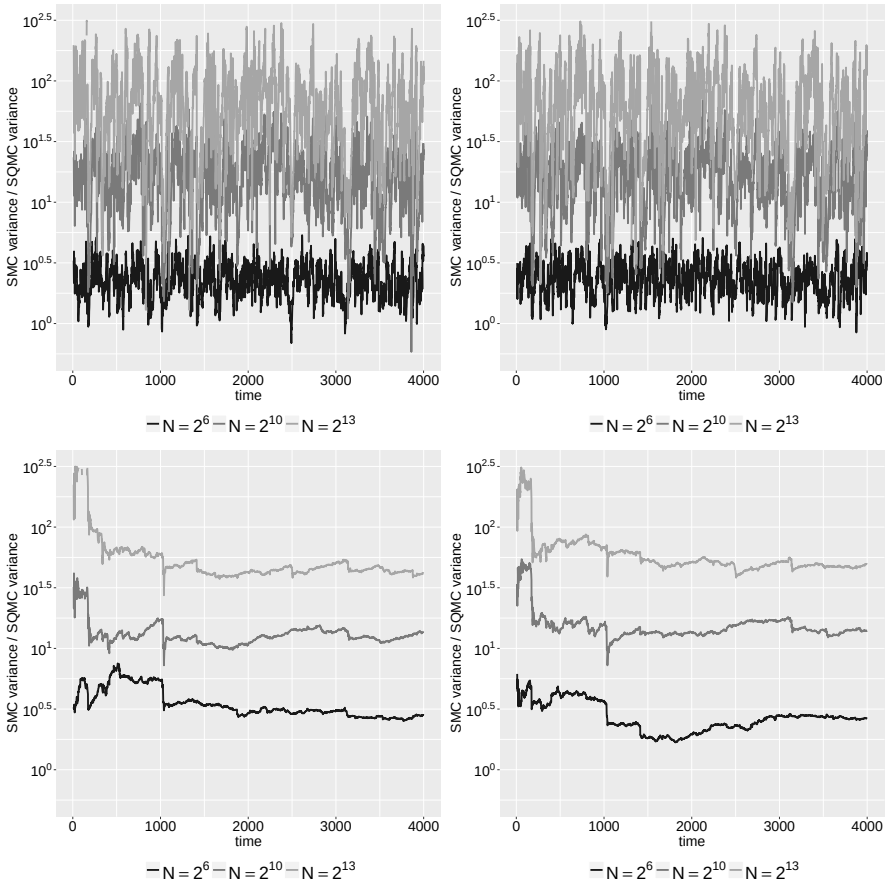

Fig. 4 Estimation of $\mathbb{E}[X_t|Y_{0:t} = y_{0:t}]$ (top) and of $\log p(y_{0:t})$ for $t \in \{0,\ldots,T\}$ and for different values of N. SQMC is implemented with the Brownian Bridge construction of Brownian paths. Results are presented for $M = 10$ (left plots) and for $M = 20$.

References

1. Andrieu, C., Doucet, A., Holenstein, R.: Particle Markov chain Monte Carlo methods. J. R. Stat. Soc. Ser. B Stat. Methodol. **72**(3), 269–342 (2010). DOI 10.1111/j.1467-9868.2009.00736.x
2. Arulampalam, M.S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for online nonlinear/non-gaussian Bayesian tracking. IEEE Transactions on signal processing **50**(2), 174–188 (2002)
3. Briers, M., Doucet, A., Maskell, S.: Smoothing algorithms for state-space models. Ann. Inst. Statist. Math. **62**(1), 61–89 (2010). DOI 10.1007/s10463-009-0236-2
4. Caflisch, R.E., Morokoff, W.J., Owen, A.B.: Valuation of mortgage backed securities using Brownian bridges to reduce effective dimension. Department of Mathematics, University of California, Los Angeles (1997)
5. Chib, S., Pitt, M.K., N., S.: Likelihood-based inference for diffusion models. Tech. rep., Nuffield College, Oxford (2004)
6. Chopin, N.: A sequential particle filter method for static models. Biometrika **89**(3), 539–551 (2002). DOI 10.1093/biomet/89.3.539
7. Chopin, N.: Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. Ann. Statist. **32**(6), 2385–2411 (2004). DOI 10.1214/009053604000000698
8. Del Moral, P.: Non-linear filtering: interacting particle resolution. Markov processes and related fields **2**(4), 555–581 (1996)
9. Del Moral, P., Doucet, A., Jasra, A.: Sequential Monte Carlo samplers. J. R. Stat. Soc. Ser. B Stat. Methodol. **68**(3), 411–436 (2006). DOI 10.1111/j.1467-9868.2006.00553.x
10. Del Moral, P., Guionnet, A.: Central limit theorem for nonlinear filtering and interacting particle systems. Ann. Appl. Probab. **9**(2), 275–297 (1999). DOI 10.1214/aoap/1029962742
11. Doucet, A., de Freitas, N., Gordon, N.J.: Sequential Monte Carlo Methods in Practice. Springer-Verlag, New York (2001)
12. Doucet, A., Godsill, S., Andrieu, C.: On sequential Monte Carlo sampling methods for Bayesian filtering. Statist. Comput. **10**(3), 197–208 (2000). DOI 10.1023/A:1008935410038
13. Doucet, A., Kantas, N., Singh, S.S., Maciejowski, J.M.: An overview of Sequential Monte Carlo methods for parameter estimation in general state-space models. In: Proceedings IFAC System Identification (SySid) Meeting. (2009)
14. Fearnhead, P.: Using random quasi-Monte-Carlo within particle filters, with application to financial time series. J. Comput. Graph. Statist. **14**(4), 751–769 (2005). DOI 10.1198/106186005X77243
15. Gerber, M., Chopin, N.: Convergence of Sequential Quasi-Monte Carlo Smoothing Algorithms. ArXiv preprint 1506.06117 (2015)
16. Gerber, M., Chopin, N.: Sequential quasi Monte Carlo. J. R. Stat. Soc. Ser. B. Stat. Methodol. **77**(3), 509–579 (2015). DOI 10.1111/rssb.12104
17. Guarniero, P., Johansen, A.M., Lee, A.: The iterated auxiliary particle filter. ArXiv e-prints (2015)
18. Kalman, R.E., Bucy, R.S.: New results in linear filtering and prediction theory. Trans. ASME Ser. D. J. Basic Engrg. **83**, 95–108 (1961). DOI 10.1115/1.3658902
19. Kim, S., Shephard, N., Chib, S.: Stochastic volatility: likelihood inference and comparison with arch models. The Review of Economic Studies **65**(3), 361–393 (1998)
20. L'Ecuyer, P., Lécot, C., Tuffin, B.: A randomized quasi-Monte Carlo simulation method for Markov chain. In: Monte-Carlo and quasi Monte-Carlo methods 2004, pp. 331–342. Springer Berlin Heidelberg (2006)
21. Lemieux, C.: Monte Carlo and Quasi-Monte Carlo Sampling (Springer Series in Statistics). Springer (2009)
22. Leobacher, G., Pillichshammer, F.: Introduction to quasi-Monte Carlo integration and applications. Compact Textbook in Mathematics. Birkhäuser/Springer, Cham (2014). DOI 10.1007/978-3-319-03425-6

23. Neal, R.M.: Annealed importance sampling. Stat. Comput. **11**(2), 125–139 (2001). DOI 10.1023/A:1008923215028
24. Owen, A.B.: Randomly permuted (t,m,s)-nets and (t,s)-sequences. In: Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing. Lecture Notes in Statistics, vol. 106, pp. 299–317. Springer, New York (1995)
25. Pitt, M.K., Shephard, N.: Filtering via simulation: auxiliary particle filters. J. Amer. Statist. Assoc. **94**(446), 590–599 (1999). DOI 10.2307/2670179

Resampling

Algorithm 3 below takes as input N sorted points $u^1 \leq \ldots \leq u^n$, and N weights W^n, and return as an output the N values $\left(F^N\right)^{-1}(u^n)$, where $\left(F^N\right)^{-1}$ is the inverse CDF relative to CDF $F^N(z) = \sum_{n=1}^N W^n \mathbb{1}\{n \leq z\}$, $z \in \mathbb{R}$. Its complexity is $O(N)$.

To compute the inverse CDF corresponding to the empirical CDF of N ancestors (as discussed in Section 2.6.2), i.e.

$$F^N(x) = \sum_{n=1}^N W^n \mathbb{1}\{X^n \leq x\}$$

simply order the N ancestors, and apply the same algorithm to the sorted ancestors.

Algorithm 3 Resampling Algorithm (inverse transform method)

Input: $u^{1:N}$ (such that $0 \leq u^1 \leq \ldots \leq u^N \leq 1$, $W^{1:N}$ (normalised weights)
Output: $a^{1:N}$ (labels in $1:N$)

```
s ← 0, m ← 0
for n = 1 → N do
    repeat
        m ← m + 1
        s ← s + W^m
    until s > u^n
    a^n ← m
end for
```