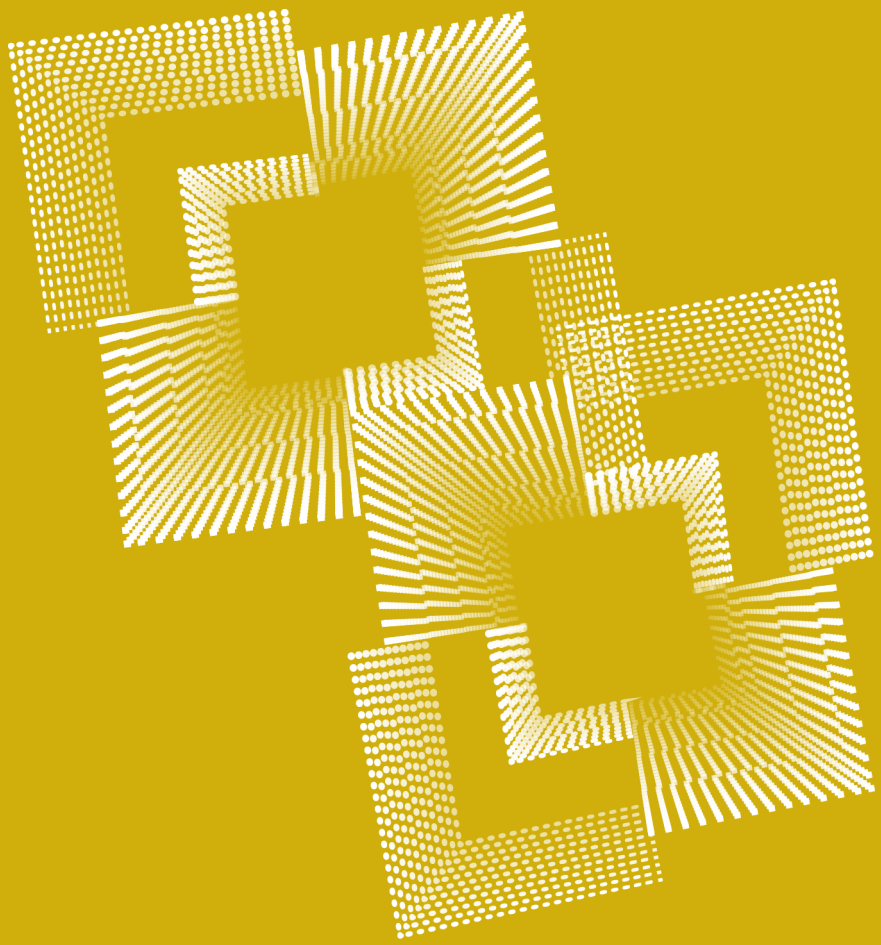
Les Cahiers de la Chaire / N°A-18

Application of Sequential Quasi-Monte Carlo to Autonomous Positioning

Nicolas Chopin & Mathieu Gerber



CHAIRE HAVAS-DAUPHINE
ÉCONOMIE ET GESTION DES NOUVELLES DONNÉES

CONNECTING THE DOTS

APPLICATION OF SEQUENTIAL QUASI-MONTE CARLO TO AUTONOMOUS POSITIONING

Nicolas Chopin *

CREST-ENSAE
92 245 Malakoff
France

Mathieu Gerber †

Harvard University
Department of Statistics
Cambridge, MA

ABSTRACT

Sequential Monte Carlo algorithms (also known as particle filters) are popular methods to approximate filtering (and related) distributions of state-space models. However, they converge at the slow $1/\sqrt{N}$ rate, which may be an issue in real-time data-intensive scenarios. We give a brief outline of SQMC (Sequential Quasi-Monte Carlo), a variant of SMC based on low-discrepancy point sets proposed by [1], which converges at a faster rate, and we illustrate the greater performance of SQMC on autonomous positioning problems.

Index Terms— Low-discrepancy point sets; Particle filtering; Quasi-Monte Carlo

1. INTRODUCTION

Many problems in signal processing (and related fields) can be formalised as the filtering of data $(\mathbf{y}_t)$ to recover an unobserved signal $(\mathbf{x}_t)$ that follows a state-space model. For non-linear and/or non-Gaussian state-space models, particle filtering [2,3], also known as Sequential Monte Carlo (SMC), is now the standard approach to perform filtering; see e.g. [4]. However, a potential drawback of SMC for real time applications is its slow $1/\sqrt{N}$ convergence rate (based on N simulations, or 'particles'). In real time problems, the running time per iteration of the filtering algorithm is bounded by the time interval between successive observations and, consequently, this slow convergence rate implies that in some settings the approximation error of SMC might be non negligible.

Recently, [1] proposed and studied the sequential quasi-Monte Carlo (SQMC) algorithm, which is a quasi-Monte Carlo (QMC) version of particle filtering. Based on N particles, SQMC has the advantage to converge at rate $o(1/\sqrt{N})$, i.e. at a faster rate than SMC; see Theorem 7 of [1]. On the other hand, SQMC requires $\mathcal{O}(N \log N)$ operations and is thus slower than SMC, which has complexity $\mathcal{O}(N)$. But [1] show that, in several scenarios, the faster convergence of SQMC does more than compensate its slower running time and, consequently, for a given computational budget, SQMC typically achieves a significantly smaller error size than SMC.

In this paper we propose to apply SQMC to the problem of autonomous positioning of a vehicle moving along a two dimensional space where, following [5], we assume that the Markov transition is non Gaussian. Our numerical study show that for this real time application SQMC provides a much more accurate estimation of the position of the vehicle than SMC.

2. SEQUENTIAL QUASI-MONTE CARLO

2.1. Background on sequential Monte Carlo

To introduce SMC we consider the following generic state-space model, described in term of probability density functions:

$$\begin{cases} \mathbf{y}_t|\mathbf{x}_t \sim f^Y(\mathbf{y}_t|\mathbf{x}_t), & t \geq 0 \\ \mathbf{x}_t|\mathbf{x}_{t-1} \sim f^X(\mathbf{x}_t|\mathbf{x}_{t-1}), & t \geq 1 \\ \mathbf{x}_0 \sim f_0^X(\mathbf{x}_0) \end{cases} \tag{1}$$

where $(\mathbf{x}_t)_{t\geq 0}$ is the unobservable Markov process on $\mathcal{X} \subseteq \mathbb{R}^d$ and $(\mathbf{y}_t)_{t\geq 0}$ is the observation process.

The typical quantity of interest in state-space models is the filtering distribution, that is, the distribution of $\mathbf{x}_t$ given all the available observations at time t, which is given by

$$p(\mathbf{x}_t|\mathbf{y}_{0:t}) = \frac{1}{Z_t} \times \int_{\mathcal{X}^t} f_0^X(\mathbf{x}_0) \prod_{s=1}^{t} f^X(\mathbf{x}_s|\mathbf{x}_{s-1}) \prod_{s=0}^{t} f^Y(\mathbf{y}_s|\mathbf{x}_s) \mathrm{d}\mathbf{x}_{0:t-1} \tag{2}$$

where Z_t is a normalising constant. Except in linear Gaussian models, the integrals in (2) are not tractable, but one may instead run a particle filter to sequentially approximate $p(\mathbf{x}_t|\mathbf{y}_{0:t})$.

The basic idea of particle filtering is to use the Markov transition $f^X(\mathbf{x}_t|\mathbf{x}_{t-1})$ to propagate the discrete approxima-

*The first author is partially supported by grant ANR-11-IDEX-0003/Labex Ecodec/ANR-11-LABX-0047 from the French National Research Agency (ANR) as part of the "Investissements d'Avenir" program.

†The second author is supported by DARPA under Grant No. FA8750-14-2-0117.

tion (for $t \geq 1$)

$$p^N(\mathbf{x}_{t-1}|\mathbf{y}_{0:t-1}) = \sum_{n=1}^N W_{t-1}^n \delta_{\mathbf{x}_{t-1}^n}(\mathrm{d}\mathbf{x}_{t-1}),$$
$$\text{with} \sum_{n=1}^N W_{t-1}^n = 1,\ W_{t-1}^n \geq 0$$

of the filtering distribution at time $t-1$ to the approximation

$$p^N(\mathbf{x}_{t-1:t}|\mathbf{y}_{0:t-1}) = p^N(\mathbf{x}_{t-1}|\mathbf{y}_{0:t-1}) f^X(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad (3)$$

of $p(\mathbf{x}_{t-1:t}|\mathbf{y}_{0:t-1})$. Then, the marginal distribution of $\mathbf{x}_t$ with respect to

$$\tilde{p}^N(\mathbf{x}_{t-1:t}|\mathbf{y}_{0:t}) \propto p^N(\mathbf{x}_{t-1:t}|\mathbf{y}_{0:t-1}) f^Y(\mathbf{y}_t|\mathbf{x}_t) \quad (4)$$

may be used as an approximation of the filtering distribution at time t. Thus, one can perform an importance sampling step, with proposal distribution (3) and target distribution (4), to get a weighted particle system $\{W_t^n, \mathbf{x}_t^n\}_{n=1}^N$ which is approximately distributed from $p(\mathbf{x}_t|\mathbf{y}_{0:t})$; see Algorithm 1 for a more precise description of particle filtering.

Algorithm 1 SMC Algorithm (Boostrap filter)

Operations must be performed for all $n \in 1:N$
Sample $\mathbf{x}_0^n$ from $f_0^X(\mathbf{x}_0)$ and compute $W_0^n = f^Y(\mathbf{y}_0|\mathbf{x}_0^n)/\sum_{m=1}^N f^Y(\mathbf{y}_0|\mathbf{x}_0^m)$
for $t = 1, \ldots, T$ **do**
  Sample a_{t-1}^n from $\mathcal{M}(W_{t-1}^{1:N})$, the multinomial distribution that produces outcome m with probability W_{t-1}^m
  Sample $\mathbf{x}_t^n$ from $f^X(\mathbf{x}_t|\mathbf{x}_{t-1}^{a_{t-1}^n})$ and compute $W_t^n = f^Y(\mathbf{y}_t|\mathbf{x}_t^n)/\sum_{m=1}^N f^Y(\mathbf{y}_t|\mathbf{x}_t^m)$
end for

2.2. Background on quasi-Monte Carlo

Loosely speaking, a QMC point set $\mathbf{u}^{1:N}$ in $[0,1)^d$ is a set of (deterministic) points which are "more uniformly" distributed than uniform random variates. The most classical measure of uniformity in the QMC literature is the so called star discrepancy, defined by

$$D^\star(\mathbf{u}^{1:N}) = \sup_{\boldsymbol{b} \in (0,1)^d} \left| \frac{1}{N} \sum_{n=1}^N \mathbb{I}\left(\mathbf{u}^n \in [\mathbf{0}, \boldsymbol{b}]\right) - \prod_{i=1}^d b_i \right|,$$

where $\boldsymbol{b} = (b_1, \ldots, b_d)$. We say that $\mathbf{u}^{1:N}$ is a QMC point set if $D^\star(\mathbf{u}^{1:N}) = \mathcal{O}(N^{-1}(\log N)^d)$.

The main motivation for using low discrepancy point sets in numerical integration is 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 explicitly links the integration error and the equidistribution property of the point set at hand, because the quantity $V(\varphi)$ only depends on the integrand φ; see e.g. Chap. 5 of [6] for a definition of $V(\varphi)$.

A useful variant to QMC is randomised QMC (RQMC), which combines the advantages of random sampling and of QMC strategies. A RQMC point set $\mathbf{u}^{1:N}$ is such that $\mathbf{u}^n \sim \mathcal{U}([0,1)^d)$ for all $n \in 1:N$ and $D^\star(\mathbf{u}^{1:N}) = \mathcal{O}(N^{-1}(\log N)^d)$ with probability one. A particularly interesting construction of RQMC point sets is the nested scrambled method for (t,m,s)-nets (see e.g. [7], Chap. 4, for a definition) proposed by [8], which allows to approximate the integral of smooth functions with an error of size $\mathcal{O}(N^{-1.5+\epsilon})$ for any $\epsilon > 0$ [9]. In addition, and contrary to plain QMC, no smoothness assumptions on the integrand φ are needed for scrambled net quadrature rules to outperform Monte Carlo integration [9]. This last point is particularly important in the context of SMC because the resampling step (Step 4 of Algorithm 1) introduces discontinuities which can not be efficiently handled by deterministic QMC strategies.

2.3. Sequential quasi-Monte Carlo

The basic idea of SQMC is to replace the sampling step from the proposal distribution (3) by a low discrepancy point set with respect to the same distribution.

The classical way to transform a low discrepancy point set with respect to the uniform distribution (i.e. a QMC point set) into a low discrepancy point set with respect to a non-uniform distribution $\pi(\mathbf{x})$ on $\mathcal{X} \subset \mathbb{R}^d$ is to use the inverse of the Rosenblatt transformation of π, defined by

$$F_\pi(\mathbf{x}) = (u_1, \ldots, u_d)^T, \quad \mathbf{x} = (x_1, \ldots, x_d)^T \in \mathcal{X},$$

where, $u_1 = F_{\pi,1}(x_1)$, $F_{\pi,1}$ being the CDF of the marginal distribution of the first component (relative to π), and for $i \geq 2$, $u_i = F_{\pi,i}(x_i|x_{1:i-1})$, $F_{\pi,i}(\cdot|x_{1:i-1})$ being the CDF of component x_i, conditional on $(x_1, \ldots, x_{i-1})$, relative to π.

Following this idea, and assuming for the moment that the state variable $\mathbf{x}_t$ is univariate, one can generate a low discrepancy point set $(\hat{\mathbf{x}}_{t-1}^{1:N}, \mathbf{x}_t^{1:N})$ from (3) as follows: let $\mathbf{u}_t^{1:N}$ be a (R)QMC point set in $[0,1)^2$, with $\mathbf{u}^n = (u_t^n, v_t^n)$, and compute

$$\hat{\mathbf{x}}_{t-1}^n = F^{-1}_{p^N(\mathbf{x}_{t-1}|\mathbf{y}_{0:t})}(u_t^n), \quad \mathbf{x}_t^n = F^{-1}_{f^X(\cdot|\hat{x}_{t-1}^n)}(\boldsymbol{v}_t^n).$$

However, the extension of this approach to $d > 1$ is not trivial because the distribution $p^N(\mathbf{x}_t|\mathbf{y}_{0:t})\mathrm{d}\mathbf{x}_t$ is then a (weighted) sum of Dirac measures over $\mathbb{R}^d$.

To overcome this difficulty, [1] proposes to transform the multivariate (discrete) distribution $p^N(\mathbf{x}_{t-1}|\mathbf{y}_{0:t-1})\mathrm{d}\mathbf{x}_t$ into a univariate (discrete) distribution $p_h^N(h_t|\mathbf{y}_{0:t})\mathrm{d}h_t$ on $[0,1)$ using the following change of variable

$$\mathbf{x} \in \mathcal{X} \mapsto h \circ \psi_t(\mathbf{x}) \in [0,1],$$

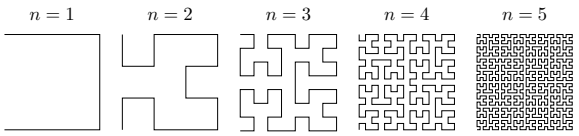


Fig. 1. The Hilbert curve is a $[0,1] \to [0,1]^d$ continuous fractal map, which is obtained as the limit of sequence (H_n), the first elements of which are represented above (for $d = 2$). Source: Marc van Dongen

where $h : [0,1]^d \to [0,1]$ is the (generalised) inverse of the Hilbert space filling curve $H : [0,1] \to [0,1]^d$, and $\psi_t : \mathcal{X} \to [0,1]^d$ is some user-chosen discrepancy-preserving bijection between $\mathcal{X}$ and $\psi_t(\mathcal{X}) \subset [0,1]^d$. See [1] and Section 3.2 below for more details on how to choose ψ_t, and see Figure 2.3 for a depiction of the Hilbert curve in two dimensions.

Using this change of variable, we can see iteration t of SMC as an importance sampling step form

$$p_h^N(h_{t-1}, \mathbf{x}_t | \mathbf{y}_{0:t-1}) = p_h^N(h_{t-1} | \mathbf{y}_{0:t-1}) f^X(\mathbf{x}_t | H(h_{t-1})) \quad (5)$$

to

$$p_h^N(h_{t-1}, \mathbf{x}_t | \mathbf{y}_{0:t}) \propto p_h^N(h_{t-1}, \mathbf{x}_t | \mathbf{y}_{0:t-1}) f^Y(\mathbf{y}_t | \mathbf{x}_t)$$

and we can therefore generate a low discrepancy point set $(\hat{h}_{t-1}^{1:N}, \mathbf{x}_t^{1:N})$ from (5) as follows: let $\mathbf{u}_t^{1:N}$ be a (R)QMC point set in $[0,1)^{d+1}$, with $\mathbf{u}^n = (u_t^n, \boldsymbol{v}_t^n)$, and compute

$$\hat{h}_{t-1} = F^{-1}_{p_h^N(\cdot|\mathbf{y}_{0:t-1})}(u_t^n), \quad \hat{\mathbf{x}}_{t-1}^n = H(h_{t-1}^n),$$

$$\mathbf{x}_t^n = F^{-1}_{f^X(\cdot|\hat{\mathbf{x}}_{t-1}^n)}(\boldsymbol{v}_t^n).$$

See Algorithm 2.3 for a pseudo-code description of SQMC.

2.4. Practical implementation

The complexity of Algorithm 2.3 is $\mathcal{O}(N \log N)$, because it performs two sorting steps at each iteration. Regarding the practical implementation of Algorithm 2.3, note that: (a) QMC generation (Steps 2 and 5) routines are available in most software (e.g. package `randtoolbox` in `R`, or class `qrandset` in the Statistics toolbox of `Matlab`); to compute the a_{t-1}^n's (Step 8), one may use the standard approach based on sorted uniforms for resampling; and (c) in order to compute h, see e.g. [10], and Chris Hamilton's C++ program available at `https://web.cs.dal.ca/~chamilto/hilbert/index.html`.

Our SQMC implementation is available at `https://bitbucket.org/mgerber/sqmc`. We shall use RQMC (randomised QMC) point sets in our simulations (more precisely scrambled Sobol' sequences; see [9, 11, 12] for more details on scrambling), as this makes it possible to evaluate the numerical error through repeated runs.

Algorithm 2 SQMC Algorithm (Boostrap filter)

1: Operations must be performed for all $n \in 1:N$
2: Generate a QMC point set $\mathbf{u}_0^{1:N}$ in $[0,1)^d$
3: Compute $\mathbf{x}_0^n = F^{-1}_{f_0^X}(\mathbf{u}_0^n)$ and $W_0^n = f^Y(\mathbf{y}_0|\mathbf{x}_0^n) / \sum_{m=1}^N f^Y(\mathbf{y}_0|\mathbf{x}_0^m)$
4: **for** $t = 1, \ldots, T$ **do**
5: Generate a QMC point set $\mathbf{u}_t^{1:N}$ in $[0,1)^{d+1}$, let $\mathbf{u}_t^n = (u_t^n, \mathbf{v}_t^n)$, with $u_t^n \in [0,1)$, $\mathbf{v}_t^n \in [0,1)^d$
6: Find permutation τ such that $u_t^{\tau(1)} \leq \ldots \leq u_t^{\tau(N)}$
7: Hilbert sort: find permutation σ_{t-1} such that
$$h \circ \psi_{t-1}(\mathbf{x}_{t-1}^{\sigma_{t-1}(1)}) \leq \ldots \leq h \circ \psi_{t-1}(\mathbf{x}_{t-1}^{\sigma_{t-1}(N)})$$
8: Compute $a_{t-1}^n = F_{t,N}^{-1}(u_t^{\tau(n)})$ where $F_{t,N}(m) = \sum_{n=1}^N W_{t-1}^{\sigma_{t-1}(n)} \mathbb{I}(n \leq m)$
9: Compute $\mathbf{x}_t^n = F^{-1}_{f^X(\cdot|\mathbf{x}_{t-1}^{a_{t-1}^n})}(\mathbf{v}_t^{\tau(n)})$ and $W_t^n = f^Y(\mathbf{y}_t|\mathbf{x}_t^n) / \sum_{m=1}^N f^Y(\mathbf{y}_t|\mathbf{x}_t^m)$
10: **end for**

Finally, while we presented SQMC in this specific case where particles are mutated according to $f^X(\mathbf{x}_t|\mathbf{x}_{t-1})$, the Markov transition of the considered model, it of course extends directly to situations where particles are mutated according to some other kernel $q_t(\mathbf{x}_t|\mathbf{x}_{t-1})$ (assuming that the particles are reweighted accordingly, as in standard SMC).

3. APPLICATION: AUTONOMOUS POSITIONING

3.1. Model description

We consider the problem of autonomous positioning of a vehicle moving in a two dimensional space. To determine its position, the vehicle estimates its speed every $T_s > 0$ seconds and measures the power of $d_y \geq 1$ radio signals. We suppose that the radio signals are emitted from known locations $\boldsymbol{r}_i \in \mathbb{R}^2$, $i = 1, \ldots, d_y$, and that the corresponding attenuation coefficients α_i are known as well. This positioning problem admits the following state space representation (see [13] and [5])

$$\begin{cases} y_{ti} = 10 \log_{10}\left(\frac{P_{i0}}{\|r_i - \mathbf{x}_t\|^{\alpha_i}}\right) + \nu_{it}, & t \geq 0 \\ \mathbf{x}_t = \mathbf{x}_{t-1} + T_s \boldsymbol{v}_t + T_s \boldsymbol{\epsilon}_t, & t \geq 1 \\ \mathbf{x}_0 \sim \mathcal{N}_2(\mathbf{0}, \boldsymbol{I}_2) \end{cases} \quad (6)$$

where $i \in 1:d_y$, $\mathbf{x}_t \in \mathbb{R}^2$ is the position of the vehicle at time t, $\boldsymbol{v}_t$ is a measure of its speed, which is assumed to be constant over successive time intervals of T_s seconds, $\boldsymbol{\epsilon}_t$ and $\boldsymbol{\nu}_y = (\nu_{1t}, \ldots, \nu_{d_y t})$ represent measurement errors while y_{it} is the power received at time t by emitter i. In the sequel, P_{0i}

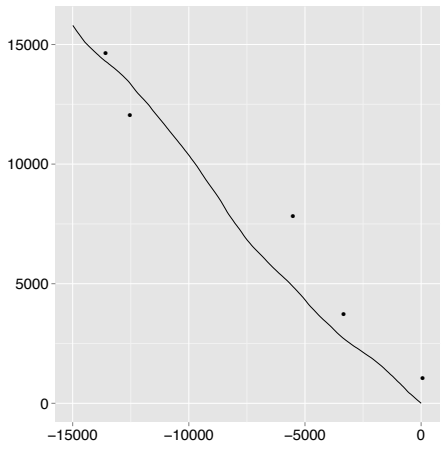


Fig. 2. Trajectory of a vehicle evolving for 15 minutes and starting at a location close to $(0, 0)$. The dots show the locations of the 5 emitters.

is the initial signal from emitter i and, following [5], we suppose that all the error terms are independent and distributed according to a Laplace distribution with parameter 0.5.

3.2. Simulation set-up

To compare the performance of SQMC and SMC for this tracking problem we simulate the trajectory of a vehicle evolving during 15 minutes according to (6). We assume that the sample period is $T_s = 1$ second, that $d_y = 5$ (5 emitters) and that $\alpha_i = 0.95$ for all $i = 1, \ldots, d_y$. The resulting trajectory and the locations of the emitters are shown in Figure 2.

The SMC algorithm is implemented using systematic resampling [14], which is usually recognised as being the most efficient resampling strategy.

SQMC is implemented using nested scrambled Sobol' sequences for the point sets $\mathbf{u}_t^{1:N}$. As described above, we need to use a mapping ψ_t to map the particles generated at iteration t of SQMC into the unit square before performing the Hilbert sort. Following [1], we chose for ψ_t a component-wise (re-scaled) logistic transform; that is, $\psi_t(\mathbf{x}) = (\psi_{t1}(x_1), \psi_{t2}(x_2))$ with

$$\psi_{ti}(x_i) = \left[1 + \exp\left(-\frac{x_i - \underline{x}_{ti}}{\bar{x}_{ti} - \underline{x}_{ti}}\right)\right]^{-1}, \quad i = 1, 2.$$

and where the time varying constants $\bar{x}_{ti}$ and $\underline{x}_{ti}$ are used to solve numerical problems due to high values of $|x_i|$. More precisely, these constants should be chosen such that, with high probability, $x_{ti} \in [\underline{x}_{ti}, \bar{x}_{ti}]$. To this aims, note that

$$\mathrm{Var}(x_{ti}) = \mathrm{Var}(\mathbf{x}_{0i}) + t\, T_s^2\, \mathrm{Var}(\epsilon_{1i}).$$

and thus, a reasonable choice for $\underline{x}_{ti}$ and $\bar{x}_{ti}$ is

$$\underline{x}_{ti},\, \bar{x}_{ti} = \sum_{s=0}^{t} \boldsymbol{v}_s \pm 2\sqrt{\mathrm{Var}(\mathbf{x}_{0i}) + t\, T_s^2\, \mathrm{Var}(\epsilon_{1i})}.$$

Simulation results are presented for $N \in \{2^8, \ldots, 2^{16}\}$, where 2 is the base of the Sobol' sequence. Taking a power of 2 for the number of simulations is the standard approach in QMC integration based on Sobol' sequence because both good theoretical and empirical results are obtained for this choice of N. However, this restriction is non necessary for QMC to outperform Monte Carlo methods and little gain may be expected in the context of SQMC, see [15] for more details on this point.

3.3. Results

In Figure 3 we compare the mean square error (MSE) of the filtering expectation estimate obtained from SQMC and SMC, as a function of t, for $N \in \{2^8, 2^{10}, 2^{16}\}$. To save space, only the results for the first component of $\mathbf{x}_t$ are presented; the results for the second component are essentially the same. One observes that the performance gain of SQMC (relative to standard particle filtering) increases quickly with N.

We now study the amount of CPU time required to have a "reasonable" Monte Carlo error using both SMC and SQMC. Letting $\hat{\mathbf{x}}_t$ be an estimate of the filtering expectation $\mathbb{E}[\mathbf{x}_t|\mathbf{y}_{0:t}]$, we consider the Monte Carlo error to be reasonable if it is small compared to the posterior variance, that is, if $\mathrm{MSE}(\hat{x}_{it}) \leq \delta^2 \mathrm{Var}(x_{it}|\mathbf{y}_{0:t})$ for $i = 1, 2$ and where we set $\delta = 0.01$.

Figure 4 shows the number of time steps $t \in \{0, \ldots, 899\}$ for which this criterion is not met, as a function of the CPU budget (i.e. CPU time per iteration). To increase the CPU budget, we simply increase N. We observe that much better results are achieved using SQMC. Indeed, when the CPU budget is 0.05s per iteration, the SMC error is too large for more than 600 time steps, while a CPU budget of 0.07s is enough to estimate both coordinates of $\mathbf{x}_t$ for all iterations with SQMC.

4. CONCLUSION

In this paper we have illustrated the potential of sequential quasi-Monte Carlo for real time signal processing processing problems with a non-linear and non-Gaussian state-space model for autonomous positioning. Compared to Monte Carlo particle filtering, dramatic variance reductions are observed when SQMC is used, both as a function of the number of particles and of CPU time. In real time application, the running time of the filtering algorithm is a crucial element and, concerning this point, we believe that significant improvement can be achieved for SQMC, notably concerning the Hilbert sort step. For instance, the computations of the Hilbert indices involve only bits operations and therefore GPU computing may allow for dramatic cost reductions.

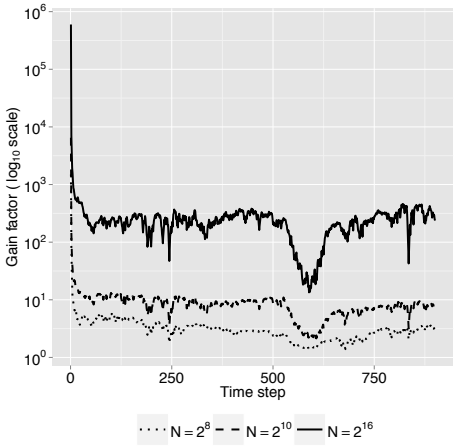


Fig. 3. Filtering of the state-space model (6): The plot gives the gain factor, defined as the MSE(SMC) over MSE(SQMC), as a function of t for the estimation of $\mathbb{E}[x_{1t}|\mathbf{y}_{0:t})]$. The results are obtained from 100 independent runs of SMC and SQMC.

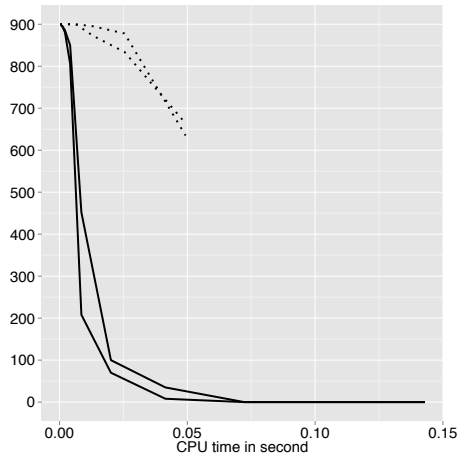


Fig. 4. Filtering of the state-space model (6): The plot gives the number of time steps $t \in \{0, \ldots, 899\}$ such that $\mathrm{MSE}(\hat{x}_{ti}) \geq 0.01^2 \mathrm{Var}(x_{ti}|\mathbf{y}_{0:t})$ as a function of the CPU budget (average CPU time per iteration), where $\hat{x}_{it}$ is either the SQMC (solid lines) or the SMC (dashed lines) estimate of $\mathbb{E}[x_{ti}|\mathbf{y}_{0:t}]$, $i = 1, 2$. The results are obtained from 100 independent runs of SMC and SQMC.

REFERENCES

[1] M. Gerber and N. Chopin, "Sequential quasi-Monte Carlo," *J. R. Statist. Soc. B (to appear)*, 2015.

[2] A. Doucet, N. de Freitas, and N. J. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, New York, 2001.

[3] O. Cappé, E. Moulines, and T. Rydén, *Inference in Hidden Markov Models*, Springer-Verlag, New York, 2005.

[4] P. M. Djuric, S. Godsill, and A. Doucet, "Special issue on particle filtering in signal processing," *EURASIP Journal of Applied Signal Processing*, 2004.

[5] J. Míguez, M. F. Bugallo, and P. M. Djurić, "A new class of particle filters for random dynamic systems with unknown statistics," *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 2278–2294, 2004.

[6] Christiane Lemieux, *Monte Carlo and Quasi-Monte Carlo Sampling (Springer Series in Statistics)*, Springer, February 2009.

[7] J. Dick and F. Pillichshammer, *Digital nets and sequences: discrepancy theory and quasi-Monte Carlo integration*, Cambridge University Press, 2010.

[8] A. B. Owen, "Randomly permuted (t, m, s)-nets and (t, s)-sequences," in *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing. Lecture Notes in Statististics*, vol. 106, pp. 299–317. Springer, New York, 1995.

[9] A. B. Owen, "Scramble net variance for integrals of smooth functions," *Ann. Stat.*, vol. 25, no. 4, pp. 1541–1562, 1997.

[10] Chris H. Hamilton and Andrew Rau-Chaplin, "Compact Hilbert indices: Space-filling curves for domains with unequal side lengths," *Inf. Process. Lett.*, vol. 105, no. 5, pp. 155–163, 2008.

[11] A. B. Owen, "Monte Carlo variance of scrambled net quadrature," *SIAM Journal on Numerical Analysis*, vol. 34, no. 5, pp. 1884–1910, 1997.

[12] A. B. Owen, "Scrambling Sobol' and Niederreiter-Xing points," *Journal of complexity*, vol. 14, no. 4, pp. 466–489, 1998.

[13] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 425–437, 2002.

[14] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Proc. Radar, Sonar Navigation*, vol. 146, no. 1, pp. 2–7, 1999.

[15] M. Gerber, "On integration methods based on scrambled nets of arbitrary size," *arXiv:1408.2773*, 2014.