Sequential Quasi Monte Carlo

N. Chopin (CREST-ENSAE)

nicolas.chopin@ensae.fr

joint work with Mathieu Gerber (CREST, Université de Lausanne)
Particle filtering (a.k.a. Sequential Monte Carlo) is a set of Monte Carlo techniques for sequential inference in state-space models. The error rate of PF is therefore $O_P(N^{-1/2})$. 
Outline

Particle filtering (a.k.a. Sequential Monte Carlo) is a set of Monte Carlo techniques for sequential inference in state-space models. The error rate of PF is therefore $O_P(N^{-1/2})$.

Quasi Monte Carlo (QMC) is a substitute for standard Monte Carlo (MC), which typically converges at the faster rate $O(N^{-1+\epsilon})$. However, standard QMC is usually defined for IID problems.
Particle filtering (a.k.a. Sequential Monte Carlo) is a set of Monte Carlo techniques for sequential inference in state-space models. The error rate of PF is therefore $O_P(N^{-1/2})$.

Quasi Monte Carlo (QMC) is a substitute for standard Monte Carlo (MC), which typically converges at the faster rate $O(N^{-1+\epsilon})$. However, standard QMC is usually defined for IID problems.

The purpose of this work is to derive a QMC version of PF, which we call SQMC (Sequential Quasi Monte Carlo).
Consider the standard MC approximation

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

where the \( N \) vectors \( u^n \) are IID variables simulated from \( U([0,1]^d) \).
Consider the standard MC approximation

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

where the $N$ vectors $u^n$ are IID variables simulated from $\mathcal{U}([0,1]^d)$.

QMC replaces $u^{1:N}$ by a set of $N$ points that are more evenly distributed on the hyper-cube $[0,1]^d$. This idea is formalised through the notion of discrepancy.
QMC vs MC in one plot

QMC versus MC: \( N = 256 \) points sampled independently and uniformly in \([0, 1]^2\) (left); QMC sequence (Sobol) in \([0, 1]^2\) of the same length (right)
Discrepancy

Koksma–Hlawka inequality:

\[
\left| \frac{1}{N} \sum_{n=1}^{N} \varphi(u^n) - \int_{[0,1]^d} \varphi(u) \, du \right| \leq V(\varphi) D^*(u^{1:N})
\]

where \( V(\varphi) \) depends only on \( \varphi \), and the star discrepancy is defined as:

\[
D^*(u^{1:N}) = \sup_{[0,b]} \left| \frac{1}{N} \sum_{n=1}^{N} 1(u^n \in [0,b]) - \prod_{i=1}^{d} b_i \right|.
\]
Koksma–Hlawka inequality:

\[
\left| \frac{1}{N} \sum_{n=1}^{N} \varphi(u^n) - \int_{[0,1]^d} \varphi(u) \, du \right| \leq V(\varphi) D^*(u^{1:N})
\]

where \( V(\varphi) \) depends only on \( \varphi \), and the star discrepancy is defined as:

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

There are various ways to construct point sets \( P_N = \{u^{1:N}\} \) so that \( D^*(u^{1:N}) = O(N^{-1+\epsilon}) \). (Describing these different constructions is beyond the scope of this talk.)
RQMC (randomised QMC)

RQMC randomises QMC so that each $u^n \sim \mathcal{U}([0, 1]^d)$ marginally. In this way

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

and one may evaluate the MSE through independent runs.
**RQMC (randomised QMC)**

RQMC randomises QMC so that each $u^n \sim U([0, 1]^d)$ marginally. In this way

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

and one may evaluate the MSE through independent runs.

A simple way to generate a RQMC sequence is to take $u^n = w + v^n \equiv 1$, where $w \sim U([0,1]^d)$ and $v^{1:N}$ is a QMC point set.
RQMC (randomised QMC)

RQMC randomises QMC so that each $u^n \sim \mathcal{U}([0, 1]^d)$ marginally. In this way

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

and one may evaluate the MSE through independent runs.

A simple way to generate a RQMC sequence is to take $u^n = w + v^n \equiv 1$, where $w \sim \mathcal{U}([0, 1]^d)$ and $v^{1:N}$ is a QMC point set.

Owen (1995, 1997a, 1997b, 1998) developed RQMC strategies such that (for a certain class of smooth functions $\varphi$):

$$\text{Var} \left\{ \frac{1}{N} \sum_{n=1}^{N} \varphi(u^n) \right\} = O(N^{-3+\epsilon})$$
Consider an unobserved Markov chain \((x_t), x_0 \sim m_0(dx_0)\) and
\[
x_t|\mathbf{x}_{t-1} = x_{t-1} \sim m_t(x_{t-1}, dx_t)
\]
Taking values in \(\mathcal{X} \subset \mathbb{R}^d\), and an observed process \((y_t)\),
\[
y_t|x_t \sim g(y_t|x_t).
\]
Consider an unobserved Markov chain \((x_t), x_0 \sim m_0(dx_0)\) and
\[x_t|x_{t-1} = x_{t-1} \sim m_t(x_{t-1}, dx_t)\]
taking values in \(\mathcal{X} \subset \mathbb{R}^d\), and an observed process \((y_t)\),
\[y_t|x_t \sim g(y_t|x_t)\].

Sequential analysis of HMMs amounts to recover quantities such as
\(p(x_t|y_{0:t})\) (filtering), \(p(x_{t+1}|y_{0:t})\) (prediction), \(p(y_{0:t})\) (marginal likelihood), etc., recursively in time. Many applications in engineering (tracking), finance (stochastic volatility), epidemiology, ecology, neurosciences, etc.
Feynman-Kac formalism

Taking $G_t(x_{t-1}, x_t) := g_t(y_t|x_t)$, we see that sequential analysis of a HMM may be cast into a Feynman-Kac model. In particular, filtering amounts to computing

$$Q_t(\varphi) = \frac{1}{Z_t} \mathbb{E}\left[ \varphi(x_t) G_0(x_0) \prod_{s=1}^{t} G_s(x_{s-1}, x_s) \right],$$

with $Z_t = \mathbb{E}\left[ G_0(x_0) \prod_{s=1}^{t} G_s(x_{s-1}, x_s) \right]$ and expectations are wrt the law of the Markov chain $(x_t)$. 
Feynman-Kac formalism

Taking \( G_t(x_{t-1}, x_t) := g_t(y_t|x_t) \), we see that sequential analysis of a HMM may be cast into a Feynman-Kac model. In particular, filtering amounts to computing

\[
Q_t(\varphi) = \frac{1}{Z_t} \mathbb{E} \left[ \varphi(x_t) G_0(x_0) \prod_{s=1}^{t} G_s(x_{s-1}, x_s) \right],
\]

with \( Z_t = \mathbb{E} \left[ G_0(x_0) \prod_{s=1}^{t} G_s(x_{s-1}, x_s) \right] \)

and expectations are wrt the law of the Markov chain \((x_t)\).

Note: FK formalism has other applications that sequential analysis of HMM. In addition, for a given HMM, there is a more than one way to define a Feynman-Kac formulation of that model.
Particle filtering: the algorithm

Operations must be performed for all $n \in 1 : N$.

At time 0,

(a) Generate $x^n_0 \sim m_0(dx_0)$.

(b) Compute $W^n_0 = G_0(x^n_0)/\sum_{m=1}^N G_0(x^m_0)$ and $Z^N_0 = N^{-1} \sum_{n=1}^N G_0(x^n_0)$.

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

(a) Generate $a^n_{t-1} \sim \mathcal{M}(W_{t-1}^{1:N})$.

(b) Generate $x^n_t \sim m_t(x^a^n_{t-1}, dx_t)$.

(c) Compute $W^n_t = G_t(x^n_{t-1}, x^n_t)/\sum_{m=1}^N G_t(x^a^n_{t-1}, x^m_t)$ and $Z^N_t = Z^N_{t-1} \left\{ N^{-1} \sum_{n=1}^N G_t(x^a^n_{t-1}, x^n_t) \right\}$. 
At iteration $t$, compute

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

to approximate $Q_t(\varphi)$ (the filtering expectation of $\varphi$). In addition, compute

$$Z_t^N$$

as an approximation of $Z_t$ (the likelihood of the data).
Cartoon representation

Source for image: some dark corner of the Internet.
We can formalise the succession of the resampling step (a) and the mutation step (b) at iteration $t$ as an importance sampling step from random probability measure $Q_N^t$:

$$Q_N^t (d(\tilde{x}_{t-1}, x_t)) = \sum_{n=1}^{N} W_{t-1}^n \delta_{x_{t-1}^n} (d\tilde{x}_{t-1}) m_t (\tilde{x}_{t-1}, dx_t)$$

to

$$Q_N^t (d(\tilde{x}_{t-1}, x_t)) \propto \bar{Q}_t^N (d(\tilde{x}_{t-1}, x_t)) G_t (\tilde{x}_{t-1}, x_t).$$
We can formalise the succession of the resampling step (a) and the mutation step (b) at iteration $t$ as an importance sampling step from random probability measure

$$
\overline{Q}_t^N (\text{d}(\tilde{x}_{t-1}, x_t)) = \sum_{n=1}^{N} W_{t-1}^n \delta_{x_{t-1}^n} (\text{d}\tilde{x}_{t-1}) m_t (\tilde{x}_{t-1}, \text{d}x_t)
$$

to

$$
Q_t^N (\text{d}(\tilde{x}_{t-1}, x_t)) \propto \overline{Q}_t^N (\text{d}(\tilde{x}_{t-1}, x_t)) G_t (\tilde{x}_{t-1}, x_t).
$$

\textbf{Idea:} use QMC instead of MC to sample $N$ points from $\overline{Q}_t^N (\text{d}(\tilde{x}_{t-1}, x_t))$. The main difficulty is that this distribution is partly discrete, partly continuous.
Case $d = 1$

Let $u^n_t = (v^n_t, w^n_t)$ be uniform variates in $[0, 1]^2$. Then

1. Use the inverse transform to obtain $\tilde{x}^n_{t-1} = \tilde{F}^{-1}(v^n_t)$, where $\tilde{F}$ is the empirical cdf of $\sum_{n=1}^{N} W^n_{t-1} \delta_{x^n_{t-1}}(dx_{t-1})$.

2. Sample $x^n_t \sim m_t(\tilde{x}^n_{t-1}, dx_t)$ as: $x^n_t = \Gamma_t(\tilde{x}^n_{t-1}, w^n_t)$, where $\Gamma_t$ is e.g. the inverse CDF of $m_t(\tilde{x}^n_{t-1}, dx_t)$ (or some other appropriate deterministic function)
From $d = 1$ to $d > 1$

When $d > 1$, we cannot use the inverse CDF method to sample from the empirical distribution

$$\sum_{n=1}^{N} W_{t-1}^{n} \delta_{x_{t-1}^{n}} (d\tilde{x}_{t-1}).$$

Idea: we “project” the $x_{t-1}^{n}$’s into $[0, 1]$ through the (generalised) inverse of the Hilbert curve, which is a fractal, space-filling curve $H : [0, 1] \rightarrow [0, 1]^d$. 
From $d = 1$ to $d > 1$

When $d > 1$, we cannot use the inverse CDF method to sample from the empirical distribution

$$\sum_{n=1}^{N} W_{t-1}^{n} \delta_{x_{t-1}^{n}} (d\tilde{x}_{t-1}).$$

Idea: we “project” the $x_{t-1}^{n}$’s into $[0, 1]$ through the (generalised) inverse of the Hilbert curve, which is a fractal, space-filling curve $H : [0, 1] \rightarrow [0, 1]^d$.

More precisely, we transform $X$ into $[0, 1]^d$ through some function $\psi$, then we transform $[0, 1]^d$ into $[0, 1]$ through $h = H^{-1}$. 

The Hilbert curve is the limit of this sequence. Note the locality property of the Hilbert curve: if two points are close in $[0, 1]$, then the corresponding transformed points remains close in $[0, 1]^d$. (Source for the plot: Wikipedia)
SQMC Algorithm

At time 0,

(a) Generate a QMC point set $\mathbf{u}^{1:N}_0$ in $[0, 1]^d$, and compute $\mathbf{x}^n_0 = \Gamma_0(\mathbf{u}^n_0)$. (e.g. $\Gamma_0 = F_{m_0}^{-1}$)

(b) Compute $W^n_0 = G_0(\mathbf{x}^n_0) / \sum_{m=1}^N G_0(\mathbf{x}^m_0)$.

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

(a) Generate a QMC point set $\mathbf{u}^{1:N}_t$ in $[0, 1]^{d+1}$; let $\mathbf{u}^n_t = (u^n_t, v^n_t)$.

(b) Hilbert sort: find permutation $\sigma$ such that $h \circ \psi(\mathbf{x}^{\sigma(1)}_{t-1}) \leq \ldots \leq h \circ \psi(\mathbf{x}^{\sigma(N)}_{t-1})$.

(c) Generate $a^{1:N}_{t-1}$ using inverse CDF Algorithm, with inputs $\text{sort}(\mathbf{u}^{1:N}_t)$ and $W^{\sigma(1:N)}_{t-1}$, and compute $\mathbf{x}^n_t = \Gamma_t(\mathbf{x}^{\sigma(a^n_{t-1})}_{t-1}, v^{\sigma(n)}_t)$. (e.g. $\Gamma_t = F_{m_t}^{-1}$)

(e) Compute

$$W^n_t = G_t(\mathbf{x}^{\sigma(a^n_{t-1})}_{t-1}, \mathbf{x}^n_t) / \sum_{m=1}^N G_t(\mathbf{x}^{\sigma(a^m_{t-1})}_{t-1}, \mathbf{x}^m_t).$$
• Because two sort operations are performed, the complexity of SQMC is $O(N \log N)$. (Compare with $O(N)$ for SMC.)
Some remarks

- Because two sort operations are performed, the complexity of SQMC is $O(N \log N)$. (Compare with $O(N)$ for SMC.)
- The main requirement to implement SQMC is that one may simulate from Markov kernel $m_t(x_{t-1}, dx_t)$ by computing $x_t = \Gamma_t(x_{t-1}, u_t)$, where $u_t \sim \mathcal{U}[0, 1]^d$, for some deterministic function $\Gamma_t$ (e.g. multivariate inverse CDF).
Some remarks

- Because two sort operations are performed, the complexity of SQMC is $O(N \log N)$. (Compare with $O(N)$ for SMC.)
- The main requirement to implement SQMC is that one may simulate from Markov kernel $m_t(x_{t-1}, dx_t)$ by computing $x_t = \Gamma_t(x_{t-1}, u_t)$, where $u_t \sim U[0, 1]^d$, for some deterministic function $\Gamma_t$ (e.g. multivariate inverse CDF).
- The dimension of the point sets $u_{t}^{1:N}$ is $1 + d$: first component is for selecting the parent particle, the $d$ remaining components is for sampling $x^n_t$ given $x^n_{t-1}$.
Extensions

• If we use RQMC (randomised QMC) point sets $u_{1:t}$, then SQMC generates an unbiased estimate of the marginal likelihood $Z_t$.
• If we use RQMC (randomised QMC) point sets $u_t^{1:N}$, then SQMC generates an unbiased estimate of the marginal likelihood $Z_t$.

• This means we can use SQMC within the PMCMC framework. (More precisely, we can run e.g. a PMMH algorithm, where the likelihood of the data is computed via SQMC instead of SMC.)
• If we use RQMC (randomised QMC) point sets \( u_t^{1:N} \), then SQMC generates an unbiased estimate of the marginal likelihood \( Z_t \).

• This means we can use SQMC within the PMCMC framework. (More precisely, we can run e.g. a PMMH algorithm, where the likelihood of the data is computed via SQMC instead of SMC.)

• We can also adapt quite easily the different particle smoothing algorithms: forward smoothing, backward smoothing, two-filter smoothing.
• If we use RQMC (randomised QMC) point sets $u_1^{1:N}$, then
SQMC generates an unbiased estimate of the marginal likelihood $Z_t$.
• This means we can use SQMC within the PMCMC framework.
  (More precisely, we can run e.g. a PMMH algorithm, where the
  likelihood of the data is computed via SQMC instead of SMC.)
• We can also adapt quite easily the different particle smoothing
  algorithms: forward smoothing, backward smoothing, two-filter
  smoothing.
Main results

We were able to establish the following types of results: consistency

\[ Q_t^N(\varphi) - Q_t(\varphi) \to 0, \quad \text{as } N \to +\infty \]

for certain functions \( \varphi \), and rate of convergence

\[ \text{MSE} \left[ Q_t^N(\varphi) \right] = o(N^{-1}) \]

(under technical conditions, and for certain types of RQMC point sets).

Theory is non-standard and borrows heavily from QMC concepts.
Some concepts used in the proofs

Let $\mathcal{X} = [0, 1]^d$. Consistency results are expressed in terms of the star norm

$$\|Q_t^N - Q_t\|_* = \sup_{[0,b] \subset [0,1)^d} \left| \left( Q_t^N - Q_t \right)(B) \right| \to 0.$$ 

This implies consistency for bounded functions $\varphi$,

$$Q_t^N(\varphi) - Q_t(\varphi) \to 0.$$ 

The Hilbert curve conserves discrepancy:

$$\|\pi^N - \pi\|_* \to 0 \implies \|\pi^N_h - \pi_h\|_* \to 0$$

where $\pi \in \mathcal{P}([0, 1]^d)$, $h : [0, 1]^d \to [0, 1]$ is the (pseudo-)inverse of the Hilbert curve, and $\pi_h$ is the image of $\pi$ through $\pi$. 
Examples: Kitagawa \( (d = 1) \)

Well known toy example (Kitagawa, 1998):

\[
\begin{align*}
    y_t &= \frac{x_t^2}{a} + \epsilon_t \\
    x_t &= b_1 x_{t-1} + b_2 \frac{x_{t-1}}{1+x_{t-1}^2} + b_3 \cos(b_4 t) + \sigma \nu_t
\end{align*}
\]

No parameter estimation (parameters are set to their true value). We compare SQMC with SMC (based on systematic resampling) both in terms of \( N \), and in terms of CPU time.
Log-likelihood evaluation (based on $T = 100$ data point and 500 independent SMC and SQMC runs).
Examples: Kitagawa ($d = 1$)

Filtering: computing $\mathbb{E}(x_t | y_{0:t})$ at each iteration $t$. Gain factor is $\text{MSE(SMC)} / \text{MSE(SQMC)}$. 
Model is

\[
\begin{align*}
y_t &= S_t^{\frac{1}{2}} \epsilon_t \\
x_t &= \mu + \Phi(x_{t-1} - \mu) + \Psi^{\frac{1}{2}} \nu_t
\end{align*}
\]

with possibly correlated noise terms: \((\epsilon_t, \nu_t) \sim N_{2d}(0, C)\).

We shall focus on \(d = 2\) and \(d = 4\).
Examples: Multivariate Stochastic Volatility ($d = 2$)

Log-likelihood evaluation (based on $T = 400$ data points and 200 independent SMC and SQMC runs).
Examples: Multivariate Stochastic Volatility \((d = 2)\)

Filtering.
Examples: Multivariate Stochastic Volatility \((d = 4)\)

Log-likelihood estimation.
• Only requirement to replace SMC with SQMC is that the simulation of $x^n_t | x^n_{t-1}$ may be written as a $x^n_t = \Gamma_t(x^n_{t-1}, u^n_t)$ where $u^n_t \sim U[0, 1]^d$.

• We observe very impressive gains in performance (even for small $N$ or $d = 6$).

• Supporting theory.
Further work

- Adaptive resampling (triggers resampling steps when weight degeneracy is too high).
- Adapt SQMC to situations where sampling from $m_t(x^n_{t-1}, dx_t)$ involves some accept/reject mechanism (e.g. Metropolis). In this way, we could develop SQMC counterparts of SMC samplers (Del Moral et al, 2006).
- SQMC$^2$ (QMC version of SMC$^2$, C. et al, 2013)?
Further work

• Adaptive resampling (triggers resampling steps when weight degeneracy is too high).

• Adapt SQMC to situations where sampling from \( m_t(x_{t-1}^n, dx_t) \) involves some accept/reject mechanism (e.g. Metropolis). In this way, we could develop SQMC counterparts of SMC samplers (Del Moral et al, 2006).

• SQMC\(^2\) (QMC version of SMC\(^2\), C. et al, 2013)?

Paper is on Arxiv.