Sequential Monte Carlo: introduction, recent advances

Nicolas Chopin (ENSAE, Paris)

(partly based on joint work with Mathieu Gerber, Bristol University)

nicolas.chopin@ensae.fr

Motivation: Feynman-Kac models, particle filtering

Feynman-Kac models: definition

A Feynman-Kac model is made of:

- A Markov chain in $\mathcal{X}$: initial law is $m_0(\mathrm{d}x_0)$, Markov kernel at iteration t is $m_t(x_{t-1}, \mathrm{d}x_t)$
- A sequence of potential functions $G_0 : \mathcal{X} \to \mathbb{R}^+$, $G_t : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$

Feynman-Kac models: definition

A Feynman-Kac model is made of:

- A Markov chain in $\mathcal{X}$: initial law is $m_0(\mathrm{d}x_0)$, Markov kernel at iteration t is $m_t(x_{t-1}, \mathrm{d}x_t)$
- A sequence of potential functions $G_0 : \mathcal{X} \to \mathbb{R}^+$, $G_t : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$

Aim is to compute sequentially quantities such as

$$\mathbb{Q}_t(\varphi) = \frac{1}{Z_t} \mathbb{E}\left[\varphi(X_t) G_0(X_0) \prod_{s=1}^{t} G_s(X_{s-1}, X_s)\right],$$

$$\text{with } Z_t = \mathbb{E}\left[G_0(X_0) \prod_{s=1}^{t} G_s(X_{s-1}, X_s)\right].$$

$\Rightarrow$ change of measure.

Application to rare events

Take for instance

$$G_t(x_{t-1}, x_t) = \mathbb{1}_{A_t}(x_t)$$

then Z_t is the probability that the $X_s \in A_s$ for all $s \leq t$, $\mathbb{Q}_t$ is the dist' of X_t conditional on $X_s \in A_s$ for $s \leq t$ and so on.

Application to HMMs (hidden Markov models)

Imagine a model for a Markov chain (X_t) that is not observed directly, but through

$$Y_t = h(X_t) + \text{noise}$$

and let $g(y_t|x_t)$ be the density of Y_t conditional on $X_t = x_t$. Then, taking

$$G_t(x_{t-1}, x_t) = g(y_t|x_t)$$

turns $\mathbb{Q}_t$ into the filtering distribution (the law of X_t conditional on data $y_{0:t}$), and Z_t into the likelihood of the data (the marginal density of $y_{0:t}$).

Application to HMMs (hidden Markov models)

Imagine a model for a Markov chain (X_t) that is not observed directly, but through

$$Y_t = h(X_t) + \text{noise}$$

and let $g(y_t|x_t)$ be the density of Y_t conditional on $X_t = x_t$. Then, taking

$$G_t(x_{t-1}, x_t) = g(y_t|x_t)$$

turns $\mathbb{Q}_t$ into the filtering distribution (the law of X_t conditional on data $y_{0:t}$), and Z_t into the likelihood of the data (the marginal density of $y_{0:t}$).

Applications in signal processing, Ecology, neurosciences...

Example: autonomous positioning

Vehicle moves in 2D space, acquires its speeds every T_s seconds, and receives d_y radio signals. Model is:

$$Y_{ti} = 10 \log_{10} \left(\frac{P_{i0}}{\|r_i - X_t\|^{\alpha_i}} \right) + \nu_{it}, \quad i = 1, \ldots, d_y$$

$$X_t = X_{t-1} + T_s V_t + T_s \epsilon_t$$

and noise terms ϵ_t, ν_t are Laplace-distributed.

Simulated data

$T_s = 1\text{s}$, $d_y = 5$ (5 emiters), $\alpha_i = 0.95$.

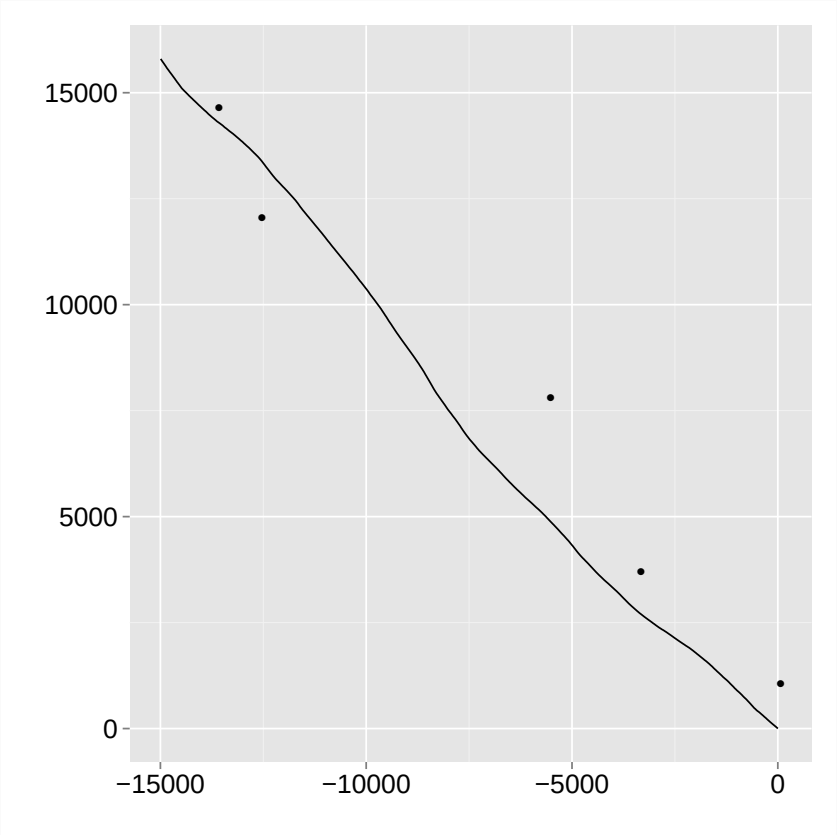


Figure 1: Simulated trajectory (15 min)

Particle Filtering: why?

For a given Feynman-Kac model, a possible approach to approximate $\mathbb{Q}_t$ sequentially would be (sequential) importance sampling:

1. At time t, simulate N copies X_t^n of Markov chain (X_t);
2. reweight according to function G_t.

Particle Filtering: why?

For a given Feynman-Kac model, a possible approach to approximate $\mathbb{Q}_t$ sequentially would be (sequential) importance sampling:

1. At time t, simulate N copies X_t^n of Markov chain (X_t);
2. reweight according to function G_t.

Problem: variance of cumulative weigts:

$$w(x_{0:t}^n) = \prod_{s=0}^{t} G_s(x_{s-1}^n, x_s^n)$$

increases over time (at exponential rate).

Particle Filtering: Basic idea

At time 0, use importance sampling, to go from $m_0(\mathrm{d}x_0)$ to $\mathbb{Q}_0(\mathrm{d}x_0) \propto m_0(\mathrm{d}x_0)G_0(x_0)$. We thus obtain the following approximation of $\mathbb{Q}_0$:

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

Particle Filtering: Basic idea

At time 0, use importance sampling, to go from $m_0(\mathrm{d}x_0)$ to $\mathbb{Q}_0(\mathrm{d}x_0) \propto m_0(\mathrm{d}x_0)G_0(x_0)$. We thus obtain the following approximation of $\mathbb{Q}_0$:

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

To progress to time 1:

1. Choose one 'ancestor' X_0^n with probability $\propto G_0(X_0^n)$; call A_0^n the index of the selected ancestor.
2. Simulate $X_1^n \sim m_1(X_0^{A_0^n}, \mathrm{d}x_1)$
3. Reweight, with weight $G_1(X_0^{A_0^n}, X_1^n)$

Particle filtering: the algorithm

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

At time 0,

(a) Generate $X_0^n \sim m_0(\mathrm{d}x_0)$.

(b) Compute $W_0^n = G_0(X_0^n) / \sum_{m=1}^N G_0(X_0^m)$.

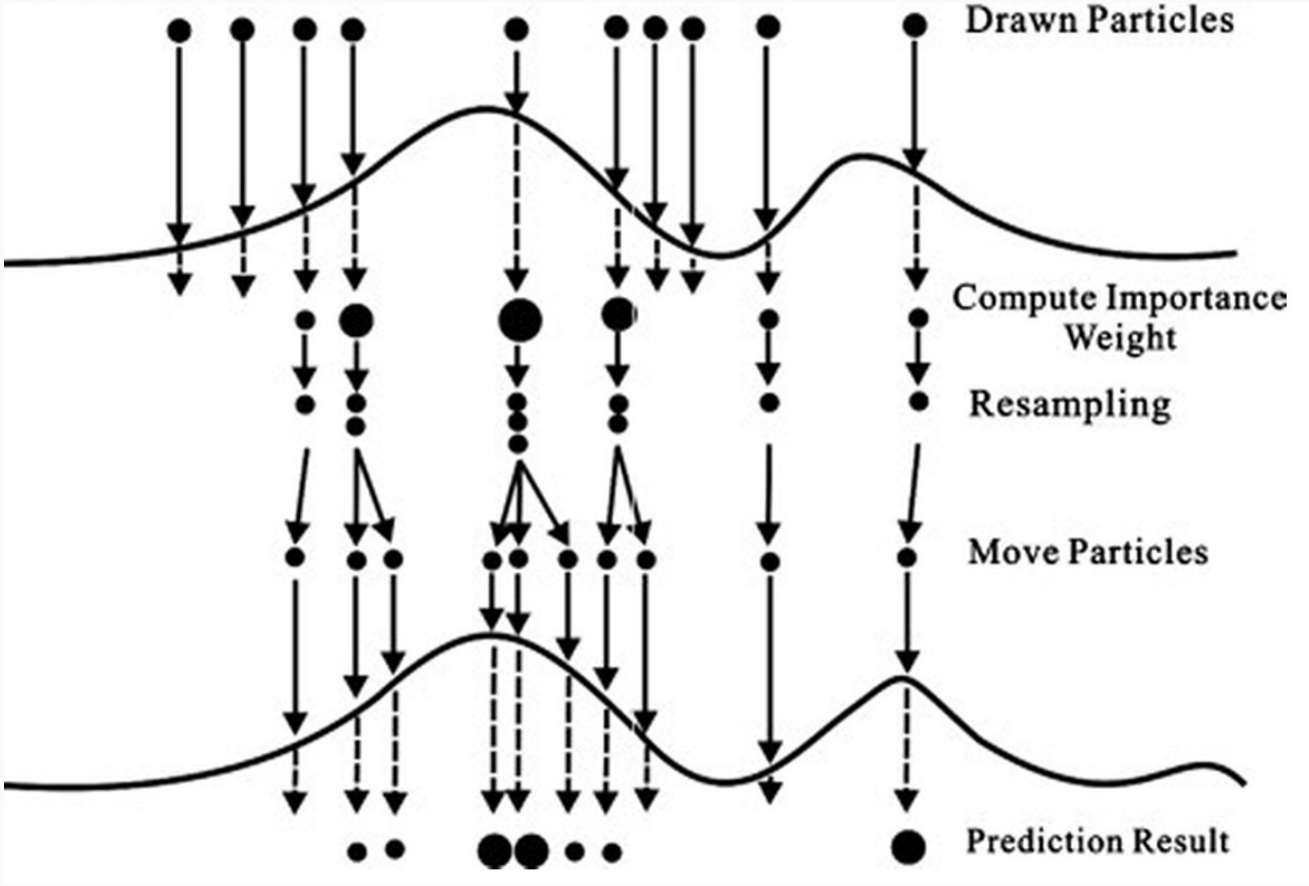
Recursively, for time $t = 1 : T$,

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

(b) Generate $X_t^n \sim m_t(X_{t-1}^{A_{t-1}^n}, \mathrm{d}x_t)$.

(c) Compute
$W_t^n = G_t(X_{t-1}^{A_{t-1}^n}, X_t^n) / \sum_{m=1}^N G_t(X_{t-1}^{A_{t-1}^m}, X_t^m)$.

Cartoon representation



Source: Chris Steinruecken

PF output

At iteration t, compute

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

to approximate $\mathbb{Q}_t(\varphi)$ (the filtering expectation of φ).

PF output

At iteration t, compute

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

to approximate $\mathbb{Q}_t(\varphi)$ (the filtering expectation of φ).

In addition, compute

$$Z_t^N = \prod_{s=0}^{t} \left\{ \frac{1}{N} \sum_{n=1}^{N} G_t(X_{t-1}^{A_{t-1}^n}, X_t^n) \right\}$$

as an approximation of Z_t (the likelihood of the data in a HMM).

Beyond boostrap filters: guided proposals

A PF such that $m_t(x_{t-1}, \mathrm{d}x_t)$ (the kernel used to simulate particles) matches $f_t(x_{t-1}, \mathrm{d}x_t)$ (the Markov kernel of (X_t) for the considered HMM) is called a bootstrap filter. However, it is possible, and often useful, to take $m_t \neq f_t$. Provided

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

$\mathbb{Q}_t$ still matches the filtering distribution.

Beyond boostrap filters: guided proposals

A PF such that $m_t(x_{t-1}, \mathrm{d}x_t)$ (the kernel used to simulate particles) matches $f_t(x_{t-1}, \mathrm{d}x_t)$ (the Markov kernel of (X_t) for the considered HMM) is called a bootstrap filter. However, it is possible, and often useful, to take $m_t \neq f_t$. Provided

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

$\mathbb{Q}_t$ still matches the filtering distribution.

The idea is to choose m_t so that the variance of G_t is as small as possible. The 'optimal' choice is the distribution of X_t conditional on X_{t-1} and Y_t (usually intractable).

Beyond boostrap filters: guided proposals

A PF such that $m_t(x_{t-1}, \mathrm{d}x_t)$ (the kernel used to simulate particles) matches $f_t(x_{t-1}, \mathrm{d}x_t)$ (the Markov kernel of (X_t) for the considered HMM) is called a bootstrap filter. However, it is possible, and often useful, to take $m_t \neq f_t$. Provided

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

$\mathbb{Q}_t$ still matches the filtering distribution.

The idea is to choose m_t so that the variance of G_t is as small as possible. The 'optimal' choice is the distribution of X_t conditional on X_{t-1} and Y_t (usually intractable).

Notice how G_t depends on both x_{t-1} and x_t in this case.

Simple example of a guided proposal

$$X_t | X_{t-1} = x_{t-1} \sim N(x_{t-1}, 1)$$
$$Y_t = \mathbb{1}_{[0,\varepsilon]}(X_t)$$

and assume data such that $y_t = 1$ for all t.

Simple example of a guided proposal

$$X_t | X_{t-1} = x_{t-1} \sim N(x_{t-1}, 1)$$
$$Y_t = \mathbb{1}_{[0,\varepsilon]}(X_t)$$

and assume data such that $y_t = 1$ for all t.

The bootstrap PF simulates particles from $N(x_{t-1}, 1)$, and kill particles that fall outside $[0, \varepsilon]$.

Simple example of a guided proposal

$$X_t | X_{t-1} = x_{t-1} \sim N(x_{t-1}, 1)$$
$$Y_t = \mathbb{1}_{[0,\varepsilon]}(X_t)$$

and assume data such that $y_t = 1$ for all t.

The bootstrap PF simulates particles from $N(x_{t-1}, 1)$, and kill particles that fall outside $[0, \varepsilon]$.

Instead, take $m_t(x_{t-1}, \mathrm{d}x_t)$ to be $N(x_{t-1}, 1)$ conditional on $x_t \in [0, \varepsilon]$. Then

$$G_t(x_{t-1}, x_t) = \Phi(\varepsilon - x_{t-1}) - \Phi(-x_{t-1})$$

and all particles fall in $[0, \varepsilon]$.

Resampling

Resampling schemes: Informal definition

A resampling scheme is a randomization procedure that takes as an input a weighted sample $\left\{(X^n, W^n)\right\}_{n=1}^N$ and returns as an output resampled variables $\left\{X^{A^n}\right\}_{n=1}^N$, where A^n is a random index in $\{1, \ldots, N\}$.

A good resampling scheme should be such that

$$\frac{1}{N}\sum_{n=1}^{N}\delta(X^{A^n}) \approx \sum_{n=1}^{N} W^n \delta(X^n)$$

or, in words, the empirical probability measure of the resampled variables should remain close (in some sense) to the weighted empirical measure of the input variables.

Motivation: Resampling, a key element of particle filtering

It is well known that particle filters 'collapse' if the particles are not resampled from time to time.

(Other applications of resampling algorithms include e.g. survey sampling and weighted bootstrap.)

Most commonly used resampling methods (in PF)

- **Multinomial resampling:**

$$A^n = F_N^-(U^n), \quad n = 1, \ldots, N, \quad F_N(x) = \sum_{n=1}^{N} W^n \mathbb{I}(n \leq x)$$

  where $\{U^n\}_{n=1}^N$ are i.i.d. $\mathcal{U}(0,1)$ random variables.

- **Stratified resampling** (Kitagawa, 1996):

$$A^n = F_N^-\Big(\frac{n-1+U^n}{N}\Big), \quad n = 1, \ldots, N$$

  where $\{U^n\}_{n=1}^N$ are i.i.d. $\mathcal{U}(0,1)$ random variables.

- **Systematic resampling** (Carpenter et al., 1999):

$$A^n = F_N^-\Big(\frac{n-1+U}{N}\Big), \quad n = 1, \ldots, N$$

  where $U \sim \mathcal{U}(0,1)$.

Inverse CDF plot

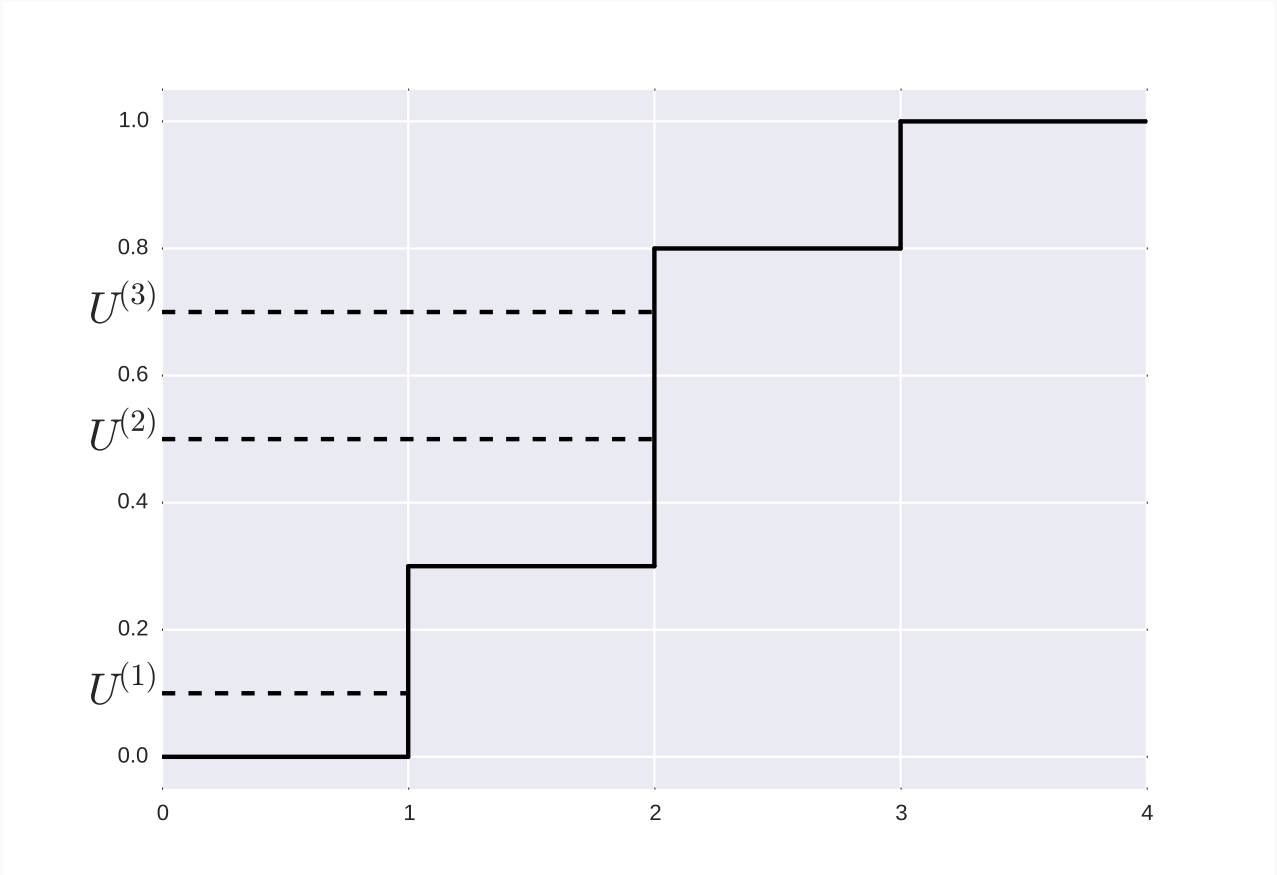


Figure 2: Function $F_N(x) = \sum_{n=1}^{N} W^n \mathbb{I}(n \leq x)$

Schizophrenic Monte Carlo

In practice, We use stratified/systematic (rather than multinomial) resampling, because these schemes are (a) a bit faster, and (b) leads to lower-variance estimates numerically. (See next slide)

In theory, we only consider multinomial resampling, as it is so much easier to study; indeed, resampled particles are IID, from distribution

$$\sum_{n=1}^{N} W^n \delta(X^n).$$

As a result, little is known about stratified/systematic; even wether they are consistent or not.

Numerical comparison of resampling schemes

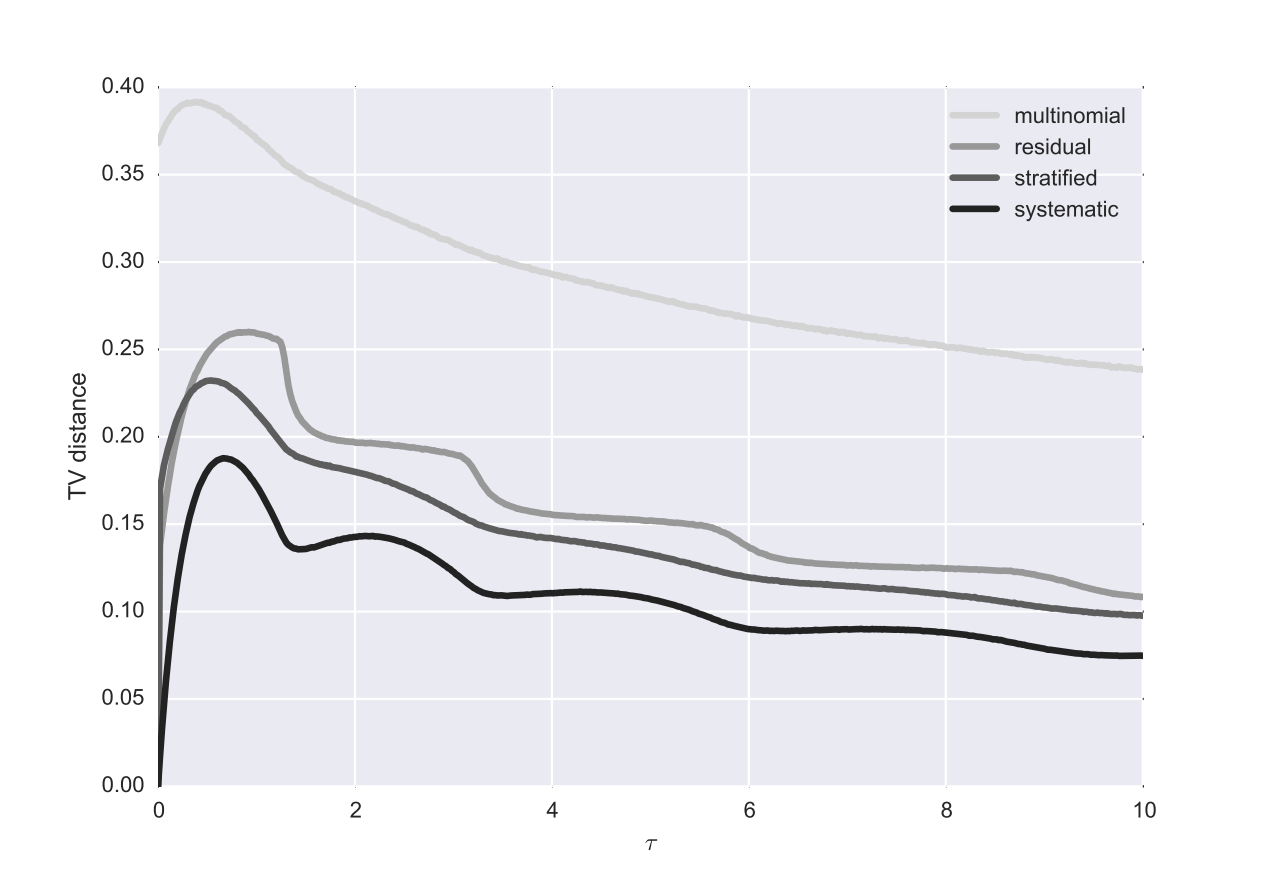


Figure 3: TV distance between empirical distributions of weighted particles, and resampled particles as a function of τ; particles are $\sim N(0,1)$, weight function is $w(x) = exp(-\tau x^2/2)$.

Consistency results for unordered resampling schemes

Preliminaries

- We say that a resampling scheme is consistent if it preserves weak convergence.

Preliminaries

- We say that a resampling scheme is consistent if it preserves weak convergence.
- Random variables $(Z^n)_{n=1}^N$ are negatively associated (NA) if, for every pair of disjoint subsets I_1 and I_2 of $\{1, \dots N\}$,

$$\operatorname{Cov}\Big(\varphi_1\big(Z^n, n \in I_1\big),\, \varphi_2\big(Z^n, n \in I_2\big)\Big) \leq 0$$

for all coordinatewise non-decreasing functions φ_1 and φ_2, such that for $k \in \{1, 2\}$, $\varphi_k : \mathbb{R}^{|I_k|} \to \mathbb{R}$ and such that the covariance is well-defined.

Main result

We have a theorem that says that an unbiased resampling scheme is consistent if the collection of variables

$$\left\{ \#^n_{\rho,z} := \sum_{m=1}^{N} \mathbb{I}(A^m = n) \right\}_{n=1}^{N}$$

is negatively associated (plus other conditions).

Applications

From the previous theorem we deduce that the following resampling schemes are consistent:

- Multinomial resampling (not new);
- Residual resampling (not new):
- Stratified resampling (new);
- Residual/Stratified resampling (new)
- SSP resampling (new, see next slide)

Unfortunately, the theorem does not apply to systematic resampling.

Basic idea behind of SSP

Start with $Y^n = NW^n$ for $n = 1, \ldots, N$. Take a pair, say $Y^1 = 3.7$, $Y^2 = 2.2$.

- With probability p, increase both, by amount 0.3: then Y^1 is 4.
- With probablity $(1 - p)$, decrease both, by amount 0.2; then Y^2 is 2.

(Choose p so that the scheme remains unbiased: $p = 2/5$.)

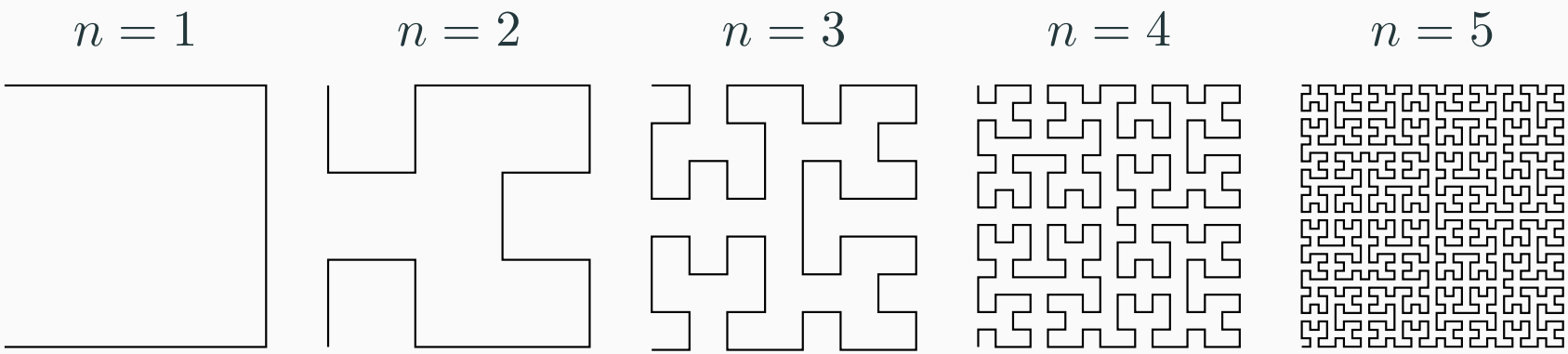
Pair the particle with a fractional weight with another particle, and start over.

Very vague sketch of the proof

In a first step, we show that consistency is equivalent to a certain condition on the set of points, when ordered through the Hilbert curve.

In a second step, we use the NA condition to show that the same technical condition holds whatever the order of the input points.

Hilbert curve



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 the corresponding transformed points remains close in $[0, 1]^d$. (Source for the plot: Marc van Dongen)

Analysis of Hilbert-ordered resampling schemes

Motivation: Kitagawa's Conjecture

Using simulations, Kitagawa (1996) noticed the following.
[Kitagawa, 1996] Assume $\mathcal{X} = \mathbb{R}$ (i.e. $d = 1$). Then, if the X_t^n's are ordered before the resampling, The approximation error of stratified resampling is of size $\mathcal{O}_{\mathbb{P}}(N^{-1})$;

Is this true? does it generalize to $d > 1$?

Short version

- Yes.
- Use the Hilbert curve.

CLT

$$N^{1/2}\left(\sum_{n=1}^{N} W_t^n \varphi(X_t^n) - \mathbb{Q}_t(\varphi)\right) \Rightarrow N\left(0, \mathcal{V}_t(\varphi)\right)$$

where the asymptotic variances are defined recursively:

$$\mathcal{V}_t\left[\varphi\right] = \frac{1}{\ell_t^2}\widetilde{\mathcal{V}}_t\left[G_t\left\{\varphi - \pi_t(\varphi)\right\}\right]$$

$$\widehat{\mathcal{V}}_t\left[\varphi\right] = \mathcal{V}_t\left[\varphi\right] + R_t\left(\rho, \varphi\right)$$

$$\widetilde{\mathcal{V}}_{t+1}\left[\varphi\right] = \widehat{\mathcal{V}}_t\left[M_{t+1}(\varphi)\right] + \pi_t\left[V_{t+1}(\varphi)\right]$$

We proved that $R_t(\varphi) = 0$ for the Hilbert-ordered version of stratified resampling. (It is > 0 for multinomial and residual, see C, 2004).

Note: also optimality results for the auxiliary weight function of the APF, where the optimal function depends on the resampling scheme.

Quick introduction to QMC

QMC basics

Consider the standard MC approximation

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

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

QMC basics

Consider the standard MC approximation

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

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

QMC replaces $\mathbf{u}^{1:N}$ 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:

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

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

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

Discrepancy

Koksma–Hlawka inequality:

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

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

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

There are various ways to construct point sets $P_N = \left\{ \mathbf{u}^{1:N} \right\}$ so that $D^\star(\mathbf{u}^{1:N}) = \mathcal{O}(N^{-1+\epsilon})$.

SQMC

Formalisation

We can formalise the succession of Steps (a), (b) and (c) at iteration t as an importance sampling step from random probability measure

$$\sum_{n=1}^{N} W_{t-1}^{n} \delta_{X_{t-1}^{n}}(\mathrm{d}\widetilde{X}_{t-1}) m_t(\widetilde{X}_{t-1}, \mathrm{d}x_t) \tag{1}$$

to

$$\{\text{same thing}\} \times G_t(\widetilde{x}_{t-1}, x_t).$$

Formalisation

We can formalise the succession of Steps (a), (b) and (c) at iteration t as an importance sampling step from random probability measure

$$\sum_{n=1}^{N} W_{t-1}^{n} \delta_{X_{t-1}^{n}}(\mathrm{d}\widetilde{X}_{t-1}) m_t(\widetilde{X}_{t-1}, \mathrm{d}x_t) \tag{1}$$

to

$$\{\text{same thing}\} \times G_t(\widetilde{x}_{t-1}, x_t).$$

Idea: use QMC instead of MC to sample N points from (1); i.e. rewrite sampling from (1) this as a function of uniform variables, and use low-discrepancy sequences instead.

Intermediate step

More precisely, we are going to write the simulation from

$$\sum_{n=1}^{N} W_{t-1}^{n} \delta_{X_{t-1}^{n}}(\mathrm{d}\widetilde{X}_{t-1}) m_t(\widetilde{X}_{t-1}, \mathrm{d}x_t)$$

as a function of $\mathbf{u}_t^n = (u_t^n, \mathbf{v}_t^n)$, $u_t^n \in [0,1]$, $\mathbf{v}_t^n \in [0,1]^d$, such that:

1. We will use the scalar u_t^n to choose the ancestor $\widetilde{X}_{t-1}$.
2. We will use $\mathbf{v}_t^n$ to generate X_t^n as

$$X_t^n = \Gamma_t(\widetilde{X}_{t-1}, \mathbf{v}_t^n)$$

where Γ_t is a deterministic function such that, for $\mathbf{v}_t^n \sim \mathcal{U}[0,1]^d$, $\Gamma_t(\widetilde{X}_{t-1}, \mathbf{v}_t^n) \sim m_t(\widetilde{X}_{t-1}, \mathrm{d}x_t)$.

Intermediate step

More precisely, we are going to write the simulation from

$$\sum_{n=1}^{N} W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}\widetilde{X}_{t-1}) m_t(\widetilde{X}_{t-1}, \mathrm{d}x_t)$$

as a function of $\mathbf{u}_t^n = (u_t^n, \mathbf{v}_t^n)$, $u_t^n \in [0,1]$, $\mathbf{v}_t^n \in [0,1]^d$, such that:

1. We will use the scalar u_t^n to choose the ancestor $\widetilde{X}_{t-1}$.
2. We will use $\mathbf{v}_t^n$ to generate X_t^n as

$$X_t^n = \Gamma_t(\widetilde{X}_{t-1}, \mathbf{v}_t^n)$$

where Γ_t is a deterministic function such that, for $\mathbf{v}_t^n \sim \mathcal{U}[0,1]^d$, $\Gamma_t(\widetilde{X}_{t-1}, \mathbf{v}_t^n) \sim m_t(\widetilde{X}_{t-1}, \mathrm{d}x_t)$.

The main problem is point 1.

Case $d = 1$

Simply use the inverse transform method: $\tilde{X}_{t-1}^n = \hat{F}^{-1}(u_t^n)$, where $\hat{F}$ is the empirical cdf of

$$\sum_{n=1}^{N} W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}\widetilde{X}_{t-1}).$$

From $d = 1$ to $d > 1$

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

$$\sum_{n=1}^{N} W_{t-1}^{n} \delta_{X_{t-1}^{n}}(\mathrm{d}\widetilde{X}_{t-1}).$$

Idea: we "project" the X_{t-1}^{n}'s into $[0, 1]$ through the (generalised) inverse of the Hilbert curve.

SQMC Algorithm

At time 0,

- **(a)** Generate a QMC point set $\mathbf{u}_0^{1:N}$ in $[0,1]^d$, and compute $X_0^n = \Gamma_0(\mathbf{u}_0^n)$. (e.g. $\Gamma_0 = F_{m_0}^{-1}$)
- **(b)** Compute $W_0^n = G_0(X_0^n) / \sum_{m=1}^N G_0(X_0^m)$.

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

- **(a)** Generate QMC ps $\mathbf{u}_t^{1:N}$ in $[0,1]^{d+1}$; let $\mathbf{u}_t^n = (u_t^n, \mathbf{v}_t^n)$.
- **(b)** Hilbert sort: find permutation σ such that $h \circ \psi(X_{t-1}^{\sigma(1)}) \leq \ldots \leq h \circ \psi(X_{t-1}^{\sigma(N)})$.
- **(c)** Generate $a_{t-1}^{1:N}$ using inverse CDF Algorithm, with inputs $\mathrm{sort}(u_t^{1:N})$ and $W_{t-1}^{\sigma(1:N)}$, and compute $X_t^n = \Gamma_t(X_{t-1}^{\sigma(a_{t-1}^n)}, \mathbf{v}_t^{\sigma(n)})$. (e.g. $\Gamma_t = F_{m_t}^{-1}$)
- **(e)** Compute $W_t^n \propto G_t(X_{t-1}^{\sigma(a_{t-1}^n)}, X_t^n)$.

Some remarks

- Related to array-RQMC (L'Ecuyer et al, 2006): same idea of using $(T+1)$ RQMC sequences of dimension $d+1$, rather than a single sequence of dimension $(T+1)d$.
- Because two sort operations are performed, the complexity of SQMC is $\mathcal{O}(N \log N)$ (while SMC is $\mathcal{O}(N)$).

Some remarks

- Related to array-RQMC (L'Ecuyer et al, 2006): same idea of using $(T+1)$ RQMC sequences of dimension $d+1$, rather than a single sequence of dimension $(T+1)d$.
- Because two sort operations are performed, the complexity of SQMC is $\mathcal{O}(N \log N)$ (while SMC is $\mathcal{O}(N)$).
- The main requirement to implement SQMC is that one may simulate from Markov kernel $m_t(x_{t-1}, \mathrm{d}x_t)$ by computing $X_t = \Gamma_t(X_{t-1}, \mathbf{u}_t)$, where $\mathbf{u}_t \sim \mathcal{U}[0,1]^d$, for some deterministic function Γ_t (e.g. multivariate inverse CDF).

Some remarks

- Related to array-RQMC (L'Ecuyer et al, 2006): same idea of using $(T+1)$ RQMC sequences of dimension $d+1$, rather than a single sequence of dimension $(T+1)d$.
- Because two sort operations are performed, the complexity of SQMC is $\mathcal{O}(N \log N)$ (while SMC is $\mathcal{O}(N)$).
- The main requirement to implement SQMC is that one may simulate from Markov kernel $m_t(x_{t-1}, \mathrm{d}x_t)$ by computing $X_t = \Gamma_t(X_{t-1}, \mathbf{u}_t)$, where $\mathbf{u}_t \sim \mathcal{U}[0,1]^d$, for some deterministic function Γ_t (e.g. multivariate inverse CDF).
- The dimension of the point sets $\mathbf{u}_t^{1:N}$ is $1+d$: first component is for selecting the parent particle, the d remaining components is for sampling X_t^n given $X_{t-1}^{a_{t-1}^n}$.

Extensions

- If we use RQMC (randomised QMC) point sets $\mathbf{u}_t^{1:N}$, then SQMC generates an unbiased estimate of the marginal likelihood Z_t.

Extensions

- If we use RQMC (randomised QMC) point sets $\mathbf{u}_t^{1:N}$, 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.)

Extensions

- If we use RQMC (randomised QMC) point sets $\mathbf{u}_t^{1:N}$, 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 develop QMC versions of particle smoothing algorithms: forward smoothing, backward smoothing, two-filter smoothing.

Extensions

- If we use RQMC (randomised QMC) point sets $\mathbf{u}_t^{1:N}$, 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 develop QMC versions of particle smoothing algorithms: forward smoothing, backward smoothing, two-filter smoothing.

Main results

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

$$\mathbb{Q}_t^N(\varphi) - \mathbb{Q}_t(\varphi) \to 0, \quad \text{as } N \to +\infty$$

for certain functions φ, and rate of convergence

$$\mathrm{MSE}\left[\mathbb{Q}_t^N(\varphi)\right] = o(N^{-1})$$

(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

$$\|\mathbb{Q}_t^N - \mathbb{Q}_t\|_\star = \sup_{[\mathbf{0},\boldsymbol{b}]\subset[0,1)^d} \left| \left(\mathbb{Q}_t^N - \mathbb{Q}_t\right)(B) \right| \to 0.$$

This implies consistency for bounded functions φ,
$\mathbb{Q}_t^N(\varphi) - \mathbb{Q}_t(\varphi) \to 0$.

The Hilbert curve conserves discrepancy:

$$\|\pi^N - \pi\|_\star \to 0 \quad \Rightarrow \quad \|\pi_h^N - \pi_h\|_\star \to 0$$

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

Examples

Examples: Kitagawa ($d = 1$)

Well known toy example (Kitagawa, 1998):

$$\begin{cases} Y_t = \frac{X_t^2}{a} + \epsilon_t \\ X_t = b_1 X_{t-1} + b_2 \frac{X_{t-1}}{1+X_{t-1}^2} + b_3 \cos(b_4 t) + \sigma \nu_t \end{cases}$$

No paramater 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.

Kitagawa ($d = 1$)

Log-likelihood evaluation (based on $T = 100$ data point and 500 independent SMC and SQMC runs).

Kitagawa ($d = 1$)

Filtering: computing $\mathbb{E}(X_t|\mathbf{y}_{0:t})$ at each iteration t. Gain factor is MSE(SMC)/MSE(SQMC).

Autonomous positioning: results

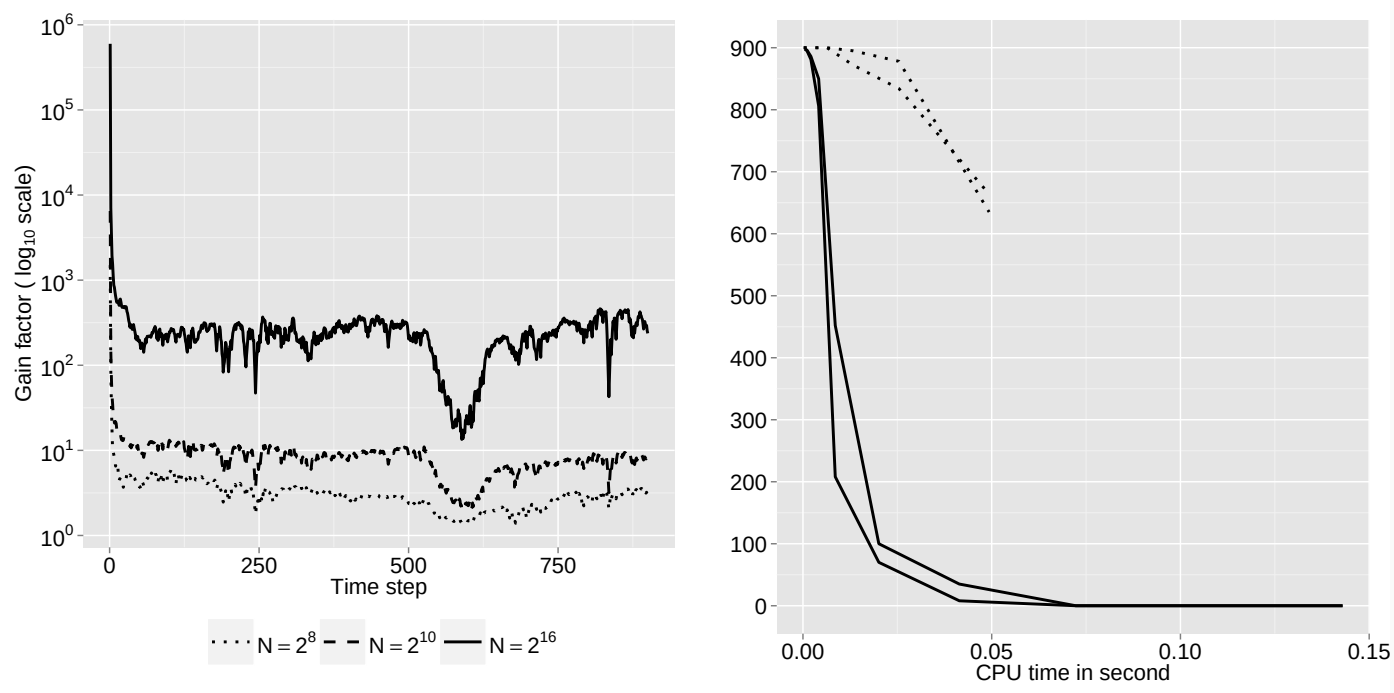


Figure 4: Left: Gain factor vs time (PF MSE/SQMC MSE); Right: number of time steps such that $\mathrm{MSE}(\hat{x}_{t1}) > 0.01\mathrm{Var}(x_{t1}|y_{0:t})$, as a function of CPU time

▸ Model

Multivariate Stochastic Volatility

Model is

$$\begin{cases} \mathbf{y}_t = S_t^{\frac{1}{2}} \boldsymbol{\epsilon}_t \\ X_t = \boldsymbol{\mu} + \Phi(X_{t-1} - \boldsymbol{\mu}) + \Psi^{\frac{1}{2}} \boldsymbol{\nu}_t \end{cases}$$

where $S_t = \mathrm{diag}(e^{x_{t1}}, \ldots, e^{x_{td}})$, with correlated noise terms: $(\boldsymbol{\epsilon}_t, \boldsymbol{\nu}_t) \sim N_{2d}(\mathbf{0}, \boldsymbol{C})$.

Multivariate Stochastic Volatility ($d = 2$)

Log-likelihood evaluation (based on $T = 400$ data points and 200 independent runs).

Multivariate Stochastic Volatility ($d = 2$)

Log-likelihood evaluation (left) and filtering (right) as a function of t.

Multivariate Stochastic Volatility ($d = 4$)

Log-likelihood estimation (based on $T = 400$ data points and 200 independent runs)

Multivariate Stochastic volatility: dimension comparison

Log-likelihood estimation (based on $T = 400$ data points and 200 independent runs)

Remark on dimension

Increasing the dimension has two effects:

1. The Hilbert curve is less and less smooth (See also He and Owen, 2015)
2. The proportion of particles with non-negligible weights get small.

Remark on dimension

Increasing the dimension has two effects:

1. The Hilbert curve is less and less smooth (See also He and Owen, 2015)
2. The proportion of particles with non-negligible weights get small.

We managed recently to obtain reasonable gains even for $d = 10$ for a multivariate linear Gaussian model (using the optimal proposal).

Diffusion-driven SV model ($d = \infty$)

$$\begin{cases} \mathrm{d}Y_t = \{\mu_P + \beta e^{X_t}\}\mathrm{d}t + e^{X_t/2}\mathrm{d}B_t \\ \mathrm{d}\tilde{X}_t = \mu(\tilde{X}_t)\mathrm{d}t + \omega(\tilde{X}_t)\mathrm{d}W_t \end{cases}$$

where $(B_t)_{t\geq 0}$ and $(W_t)_{t\geq 0}$ are Brownian motions with correlation coefficient $\rho \in (-1, 1)$ and

$$\mu(x) = \kappa(\mu - e^x)e^{-x} - 0.5\omega^2 e^{-x}$$
$$\omega(x) = \omega e^{-x/2}$$

The Y_t are observed at times $t = 0, 1, \ldots, T$.

Euler Discretisation

For $M \geq 1$ (with $\delta = M^{-1}$), let $\boldsymbol{x}_t \in \mathbb{R}^M$ be such that:

$$\begin{cases} Y_t | Y_{t-1}, X_t \sim \mathcal{N}_1\Big(Y_{t-1} + \mu_P + \beta\sigma_t^2 + \rho Z_t,\ (1-\rho^2)\sigma_t^2\Big) \\ X_t(1) = X_{t-1}(M) + \delta\mu(X_{t-1}(M)) + \omega(X_{t-1}(M))(W_{t-1+\delta} - W_{t-1}) \\ \vdots \\ X_t(M) = X_{t-1}(M-1) + \delta\mu(X_{t-1}(M-1)) \\ \qquad\qquad + \omega(X_{t-1}(M-1))(W_t - W_{t-\delta}) \end{cases}$$

and

$$\sigma_t^2 = \frac{1}{M}\sum_{m=1}^{M} e^{X_{t-1}(m)}, \quad Z_t = \sum_{m=1}^{M} e^{X_{t-1}(m)}\big(W_{t+m\delta} - W_{t+(m-1)\delta}\big).$$

Euler Discretisation

For $M \geq 1$ (with $\delta = M^{-1}$), let $\boldsymbol{x}_t \in \mathbb{R}^M$ be such that:

$$\begin{cases} Y_t | Y_{t-1}, X_t \sim \mathcal{N}_1\Big(Y_{t-1} + \mu_P + \beta\sigma_t^2 + \rho Z_t,\ (1-\rho^2)\sigma_t^2\Big) \\ X_t(1) = X_{t-1}(M) + \delta\mu(X_{t-1}(M)) + \omega(X_{t-1}(M))(W_{t-1+\delta} - W_{t-1}) \\ \vdots \\ X_t(M) = X_{t-1}(M-1) + \delta\mu(X_{t-1}(M-1)) \\ \qquad\qquad\qquad\qquad + \omega(X_{t-1}(M-1))(W_t - W_{t-\delta}) \end{cases}$$

and

$$\sigma_t^2 = \frac{1}{M}\sum_{m=1}^{M} e^{X_{t-1}(m)}, \quad Z_t = \sum_{m=1}^{M} e^{X_{t-1}(m)}\big(W_{t+m\delta} - W_{t+(m-1)\delta}\big).$$

For this model, $M = 10$ is a reasonable choice (Chib et al., 2004).

Resampling step collapses to $d = 1$

A naive application of SQMC would imply working in dimension $M = 10$, in particular for the Hilbert ordering.

Resampling step collapses to $d = 1$

A naive application of SQMC would imply working in dimension $M = 10$, in particular for the Hilbert ordering.

However, if we implement a bootstrap filter, we notice that (a) G_t depends only on X_t; and (b) the simulation of $\tilde{X}_t$ depends only on $X_{t-1}(M)$ (the last component of X_{t-1}), because process (X_t) is Markov. Thus we can collapse the choice of the ancestor to the choice of a scalar.

Resampling step collapses to $d = 1$

A naive application of SQMC would imply working in dimension $M = 10$, in particular for the Hilbert ordering.

However, if we implement a bootstrap filter, we notice that (a) G_t depends only on X_t; and (b) the simulation of $\tilde{X}_t$ depends only on $X_{t-1}(M)$ (the last component of X_{t-1}), because process (X_t) is Markov. Thus we can collapse the choice of the ancestor to the choice of a scalar.

More generally, notion of effective dimension for the choice of the ancestors.

Mutation step of SQMC: Choice of Γ_t

To simulate X_t^n given $X_{t-1}^n(M)$, we must simulate the innovation terms $W_{t+m\delta} - W_{t+(m-1)\delta}$.

- Forward approach: simulate the consecutive

$$W_{t+m\delta}^n - W_{t+(m-1)\delta}^n$$

as IID $N(0, \delta)$ variates.

Mutation step of SQMC: Choice of Γ_t

To simulate X_t^n given $X_{t-1}^n(M)$, we must simulate the innovation terms $W_{t+m\delta} - W_{t+(m-1)\delta}$.

- Forward approach: simulate the consecutive

$$W_{t+m\delta}^n - W_{t+(m-1)\delta}^n$$

  as IID $N(0, \delta)$ variates.
- Brownian bridge: Simulate $W_{t+m\delta}$ (cond. on the previous ones) in the following order: $m = M$, $m = M/2$, $m = M/4$, $m = 3M/4$, ...

Diffusion driven SV model: Simulation set-up

The parameters of the model are set 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 (Chib et al., 2004).

Diffusion driven SV model: Simulation Results

Estimation of $\mathbb{E}[X_t|Y_{0:T}]$ for $t \in \{1, \ldots, T\}$ and for different values of N (and based 100 independent SMC and SQMC runs). SQMC is implemented with the forward method (left) and with the Brownian Bridge method (right).

Diffusion driven SV model: Simulation Results

Estimation of the log-likelihood for different values of N (and based 100 independent SMC and SQMC runs). SQMC is implemented with the forward method (left) and with the Brownian Bridge method (right).

Conclusion

SQMC: conclusion

- Only requirement to replace SMC with SQMC is that the simulation of $X_t^n | X_{t-1}^n$ may be written as a $X_t^n = \Gamma_t(X_{t-1}^n, \mathbf{u}_t^n)$ where $\mathbf{u}_t^n \sim U[0,1]^d$.
- impressive gains in performance (even for small N and moderate d).
- Supporting theory.
- C++ package by Mathieu (improved), Python library coming soon.

References

- Gerber, M., and C., N. **Sequential quasi Monte Carlo**. J. R. Stat. Soc. Ser. B. Stat. Methodol. 77, 3 (2015), 509–579.
- Gerber, M., and C., N. **Convergence of sequential quasi-monte carlo smoothing algorithms**. Bernoulli (forthcoming), ArXiv:1506.06117.
- C., N., and Gerber, M. **Application of sequential quasi-Monte Carlo to autonomous positioning** EUSIPCO 2015 proceedings, ArXiv:1503.01631.
- C., N. and Gerber, M. **Sequential quasi-Monte Carlo: Introduction for Non-Experts, Dimension Reduction, Application to Partly Observed Diffusion Processes** MCQMC2016 proceedings (forthcoming).
- Chopin, N. and Gerber, M. and Whiteley, N. **Negative association, ordering and convergence of resampling methods**, arxiv:1707.01845.