The Tenth International Symposium on Operations Research and Its Applications (ISORA 2011)
Dunhuang, China, August 28–31, 2011
Copyright © 2011 ORSC & APORC, pp. 83–88

Hybrid Quasi-Monte Carlo Method for the Simulation of State Space Models

Hozumi Morohosi

National Graduate Institute for Policy Studies
7-22-1 Roppongi, Minato-ku, Tokyo, Japan 106-8677

Abstract We propose a simulation method for state space models which utilizes quasi-Monte Carlo method with randomization aiming at good approximation of the probability distribution at each time step. Numerical experiments shows our proposed method outperforms a traditional Monte Carlo simulation method in the estimation of model likelihood.

Keywords Quasi-Monte Carlo method; State space model; Simulation

1 Introduction

State space modeling increases its popularity in wide range of applications. It is used for several purposes, such as forecasting, smoothing, or parameter estimation. A traditional approach is known as Kalman filter, and several variants of it. On the other hand, more computer intensive approach, so-called Monte Carlo filer, is also now available. See, e.g. [4]. This work proposes an alternative for Monte Carlo filter by introducing quasirandom point set, with aiming at obtaining a good approximation of the distribution with less computational cost. This is probably most meaningful in parameter estimation problem. When we try to find the estimate of some parameters in the model by maximum likelihood method, we necessarily run the optimization algorithm repeatedly, which requires computation of objective function, i.e. likelihood function. Hence we need fast and accurate computation of the likelihood for the estimation of parameters. In this experimental study we focus on the convergence speed of the likelihood computation.

In the following, Sec. 2 gives a short introduction to state space modeling and Monte Carlo filter method. Sec. 3 proposes hybrid quasi-Monte Carlo method for state space model, and Sec. 4 shows a numerical experiment result. Final section summaries the work and gives some remarks.

2 State Space Model and Monte Carlo Filter

We consider a discrete time dynamical system described by equations:

$$\mathbf{X}_j = A(\mathbf{X}_{j-1}, \mathbf{v}_j), \tag{1}$$
$$\mathbf{Y}_j = B(\mathbf{X}_j, \mathbf{w}_j), \tag{2}$$

for $j = 1,2,\ldots$, where $\mathbf{X}_j$ is an unobservable state variable taking a value on (a subspace of) $\mathscr{R}^l$, and $\mathbf{Y}_j$ is an observation on $\mathscr{R}^k$. Random vectors $\mathbf{v}_j \in \mathscr{R}^d$ and $\mathbf{w}_j \in \mathscr{R}^c$ are state noise and observation noise, respectively, and they are mutually independent. The evolution of system is governed by the function $A : \mathscr{R}^l \times \mathscr{R}^d \to \mathscr{R}^l$, while the observation of system is done by the function $B : \mathscr{R}^l \times \mathscr{R}^c \to \mathscr{R}^k$. Another possible interpretation of eq. (1) is to think of it as a kind of Markov chain whose transition probability is given by

$$\mathsf{P}\{\mathbf{X}_j = \mathbf{x}_j | \mathbf{X}_{j-1} = \mathbf{x}_{j-1}\} = \mathsf{P}\{A(\mathbf{x}_{j-1}, \mathbf{v}_j) = \mathbf{x}_j\}. \tag{3}$$

This interpretation leads to a Monte Carlo (MC) simulation algorithm, referred to such as Monte Carlo filter, particle filter, or sequential Monte Carlo, for the system. More specifically, we approximate the distribution $F_j(\mathbf{x})$ of $\mathbf{X}_j$ at step j by an empirical distribution $\hat{F}_j(\mathbf{x})$ of $\{\mathbf{X}_j^{(1)}, \ldots, \mathbf{X}_j^{(n)}\}$. The sample $\{\mathbf{X}_j^{(1)}, \ldots, \mathbf{X}_j^{(n)}\}$ is generated by eq. (1) with random vectors $\{\mathbf{v}_j^{(1)}, \ldots, \mathbf{v}_j^{(n)}\}$, given the point set $\{\mathbf{X}_{j-1}^{(1)}, \ldots, \mathbf{X}_{j-1}^{(n)}\}$ at step $j-1$. It is necessary to give the initial point set $\{\mathbf{X}_0^{(1)}, \ldots, \mathbf{X}_0^{(n)}\}$ following an appropriate initial distribution at the first stage of simulation

After obtaining sample $\{\mathbf{X}_j^{(1)}, \ldots, \mathbf{X}_j^{(n)}\}$, we perform resampling to find the posterior distribution given an observation $\mathbf{y}_j$. The weight of each point $\mathbf{X}_j^{(i)}$ is given by its likelihood

$$\alpha_j^{(i)} = r(K(\mathbf{y}_j, \mathbf{X}_j^{(i)})) \left| \frac{\partial K}{\partial \mathbf{y}_j} \right|, \tag{4}$$

where K is the inverse function of B with respect to the second argument $\mathbf{w}$, i.e., $K(\mathbf{y}, \mathbf{X}) = B^{-1}(\mathbf{y}, \mathbf{X})$, and $r(\mathbf{w})$ is the probability density of $\mathbf{w}_j$ (cf. [4]). We choose n points, each $\mathbf{X}_j^{(i)}$ with probability $\alpha_j^{(i)} / \sum_{i=1}^n \alpha_j^{(i)}$, from $\{\mathbf{X}_j^{(1)}, \ldots, \mathbf{X}_j^{(n)}\}$, and such chosen point set is used as sample at step j.

Among various applications of state space model (1, 2), it is of great interest to estimate the model parameters, to say θ in general, which specify the probability distribution of $\mathbf{v}_j$ and $\mathbf{w}_j$ from the observations $\mathbf{y}_j$, $j = 1,2,\ldots$. A common approach to this end is to utilize likelihood method. The log-likelihood $l(\theta)$ of the state space model (1, 2) is approximated by MC as

$$l(\theta) \approx \sum_j \log \left(\frac{1}{n} \sum_{i=1}^n \alpha_j^{(i)} \right). \tag{5}$$

The maximum likelihood estimate (MLE) can be found as the parameter value $\hat{\theta}$ which maximizes (5). Since we apply an iterative optimization algorithm to find $\hat{\theta}$ maximizing $l(\theta)$, we have to compute (5) with different θ values so many times, and often experiences usual MC method is not sufficiently fast for our purpose. We propose a new method based on Quasi-Monte Carlo integration in the next section.

3 Hybrid Quasi-Monte Carlo Methods

Quasi-Monte Carlo (QMC) method [1, 8, 9] has been recognized as a promising alternative to MC in high-dimensional numerical integration, and begin to spread over new application areas. Attempts to simulate a Markov chain in terms of quasirandom sequence are made by several authors [3, 5, 6, 7]. Among them we explore the algorithm named *array-RQMC* (Randomized Quasi-Monte Carlo) in [5, 6] and propose a simplified one combined with the idea of resampling for quasirandom points by [10] for the state space model simulation.

The algorithm aims to give a good approximation for the (unknown) distribution at each step. Denote the distribution of the state variable $\mathbf{X}_j$ by $F_j(\mathbf{x})$. MC method gives an approximation $\hat{F}_j(\mathbf{x})$ for $F_j(\mathbf{x})$ by the empirical distribution of the point set $\{\mathbf{X}_j^{(1)},\dots,\mathbf{X}_j^{(n)}\}$ generated by (1) with random vectors $\{\mathbf{v}_j^{(1)},\dots,\mathbf{v}_j^{(n)}\}$. Applying QM-C to the problem begins with replacing random vectors by uniformly distributed points (quasirandom points) $\{\mathbf{u}_j^{(1)},\dots,\mathbf{u}_j^{(n)}\}$. with special care in order to achieve high uniformity of point set $\{\mathbf{X}_j^{(i)}\}_{i=1}^n$.

We employ two kinds of technique to this end. First one is *sorting* introduced in array-RQMC [6], which proposes to sort the $(j-1)$-th step points $\{\mathbf{X}_{j-1}^{(1)},\dots,\mathbf{X}_{j-1}^{(n)}\}$ in terms of an appropriate ordering function $h(\mathbf{x})$, i.e., to hold $h(\mathbf{X}_{j-1}^{(1)}) \le \cdots \le h(\mathbf{X}_{j-1}^{(n)})$. The ordering function plays a crucial role in making the algorithm efficiently work. Although it is difficult to give general guidelines how to choose a good ordering function, we can often obtain a function showing good performance in practice. Second one is *allocation* of quasirandom points to random vectors. Quasirandom point set is originally introduced to compute a high-dimensional integral efficiently and have to be appropriately assigned to integration variables. For instance, log-likelihood (5) contains the average of weights $\alpha_j^{(i)}$ and this average can be considered as the expectation of them with respect to $\mathbf{X}_j$, more specifically, some relevant components of $\mathbf{X}_j$. Since it is generated by $\mathbf{v}_1,\dots,\mathbf{v}_j$, we have to find which components of them directly generate the relevant components of $\mathbf{X}_j$. The allocation have to be done by carefully tracing the relation between variables. An example is shown in next section.

Another ingredient of our filter algorithm is resampling from the point set $\{\mathbf{X}_j^{(1)},\dots,\mathbf{X}_j^{(n)}\}$ based on the likelihood $\alpha_j^{(i)}$. Resampling on a quasirandom point set is studied in [10], and shown to give the convergence to target distribution. Eventually, our algorithm is given in the following.

HQMC filter algorithm

```
1. Generate $\mathbf{X}_0^{(1)},\dots,X_0^{(n)}$ following an initial distribution.
2. Sort $\mathbf{X}_0^{(1)},\dots,X_0^{(n)}$ by increasing order of the value $h(\mathbf{X}_0^{(i)})$ and renumber them.
3. Generate randomized uniform quasirandom point set $\{\mathbf{u}_j^{(1)},\dots,\mathbf{u}_j^{(n)}\}_{j=1}^t$, and allocate them in appropriate way to $\{\mathbf{v}_j^{(1)},\dots,\mathbf{v}_j^{(n)}\}_{j=1}^t$, after transformation if necessarily.
4. For $j=1,\dots,t$ do the following computation.
    (a) Compute $\mathbf{X}_j^{(i)} = A(\mathbf{X}_{j-1}^{(i)},\mathbf{v}_j^{(i)})$.
    (b) Calculate the weight $\alpha_j^{(i)}$, then resampling each $\mathbf{X}_j^{(i)}$ with probability
        $\alpha_j^{(i)}/\sum_{i=1}^n \alpha_j^{(i)}$ to obtain $\{\mathbf{X}_j^{(1)'},\dots,\mathbf{X}_j^{(n)'}\}$.
    (c) Sort $\{\mathbf{X}_j^{(1)'},\dots,\mathbf{X}_j^{(n)'}\}$ by increasing order of the value $h(\mathbf{X}_j^{(i)'})$ and renumber
```

them to eventually obtain $\{\mathbf{X}_j^{(1)},\ldots,\mathbf{X}_j^{(n)}\}$

After running the algorithm, we compute the log-likelihood (5). We usually compute it several times using independently randomized quasirandom point sets to obtain the error estimate. This procedure is called randomized QMC (see [8] for details).

4 Numerical Example

A target tracking problem (cf. [2]) is utilized to illustrate how to apply HQMC method to state space model simulation. The model is described by following equations:

$$\begin{aligned}
\mathbf{X}_j &= A\mathbf{X}_{j-1}+\mathbf{v}_j = \begin{pmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x_{j-1} \\ y_{j-1} \\ \dot{x}_{j-1} \\ \dot{y}_{j-1} \end{pmatrix} + \mathbf{v}_j, && (6)\\
\mathbf{Y}_j &= b(\mathbf{X}_j)+\mathbf{w}_j = \begin{pmatrix} \sqrt{x_j^2+y_j^2} \\ \tan^{-1}(y_j/x_j) \end{pmatrix} + \mathbf{w}_j, && (7)
\end{aligned}$$

where $\mathbf{X}_j = (x_j, y_j, \dot{x}_j, \dot{y}_j)^\top$ is the actual but unobserved position (x_j and y_j) and velocity ($\dot{x}_j$ and $\dot{y}_j$) of the target object on the two-dimensional plane, $\mathbf{Y}_j = (r_j, \delta_j)^\top$ is observed position in polar coordinates at time j, see Fig. 1. In state equation (6) the matrix A evolves the the state $\mathbf{X}_{j-1}$ to $\mathbf{X}_j$ by time step Δ with additive Gaussian noise $\mathbf{v}_j \sim \mathcal{N}(0,\Sigma_v)$, while the observation is done by nonlinear transformation $b(\mathbf{x})$ with additive Gaussian noise $\mathbf{w}_j \sim \mathcal{N}(0,\Sigma_w)$ in (7).

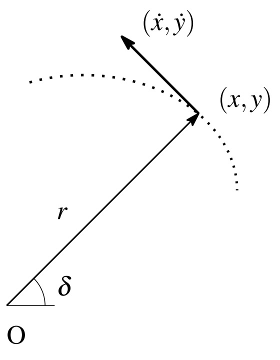

Figure 1: Illustration of target tracking problem.

Our experiment focuses on comparing MC and QMC computation of likelihood from the viewpoint of convergence speed. First artificial observations, $\mathbf{y}_1,\ldots,\mathbf{y}_t$, are generated as test data set. They are produced following (6) with fixed model parameters, $\Sigma_v = \mathrm{diag}(\sigma_p^2,\sigma_p^2,\sigma_v^2,\sigma_v^2)$, and $\Sigma_w = \mathrm{diag}(\sigma_r^2,\sigma_\delta^2)$.

The computation of the weight $\alpha_j^{(i)}$ in (4) of each sample point $\mathbf{X}_j^{(i)} = (x_{ij}, y_{ij}, \dot{x}_{ij}, \dot{y}_{ij})^\top$, at each step j has an explicit form for our model:

$$\alpha_j^{(i)} = \frac{1}{2\pi\sigma'_r\sigma'_\delta} \exp\left\{ -\frac{\left(r_j - \sqrt{x_{ij}^2 + y_{ij}^2}\right)^2}{2\sigma_r'^2} - \frac{(\delta_j - \tan^{-1}(y_{ij}/x_{ij})^2}{2\sigma_\delta'^2} \right\}. \tag{8}$$

It should be noticed this α_j contains only (x_j, y_j). Since (x_j, y_j) is generated by $(x_j, y_j, \dot{x}_{j-1}, \dot{y}_{j-1})$, it depends directly on the random variables $(v_{1j}, v_{2j}, v_{3,j-1}, v_{4,j-1})$. Base on this relation, we allocate four-dimensional quasirandom point $\mathbf{u}_j$ to $(v_{1j}, v_{2j}, v_{3,j-1}, v_{4,j-1})$. We choose the ordering function as $h(\mathbf{x}) = x + y$.

For the sake of error estimation we carry out the computation of it several times to obtain estimates, $l_1(\theta'), \dots, l_m(\theta')$. Then calculate the standard deviation of them as follows.

$$s(l) = \left(\frac{1}{m-1} \sum_{k=1}^{m} (l_k(\theta') - \bar{l}(\theta'))^2 \right)^{1/2}, \qquad \text{where } \bar{l}(\theta') = \frac{1}{m} \sum_{k=1}^{m} l_k(\theta'). \tag{9}$$

The artificial observations are generated after real parameters $\sigma_p = \sqrt{0.05}$, $\sigma_v = \sqrt{5}$, $\sigma_r = 2$, and $\sigma_\delta = 2\pi/180$. We run HQMC filter for them with the model parameters set to the same as real parameters. Fig. 2 shows the convergence speed of standard error for MC and HQMC, where horizontal axis shows the number of points n at each time step, and vertical axis shows the standard error $s(l)/\bar{l}$. Comparing MC and HQMC, we observe several times faster convergence of HQMC.

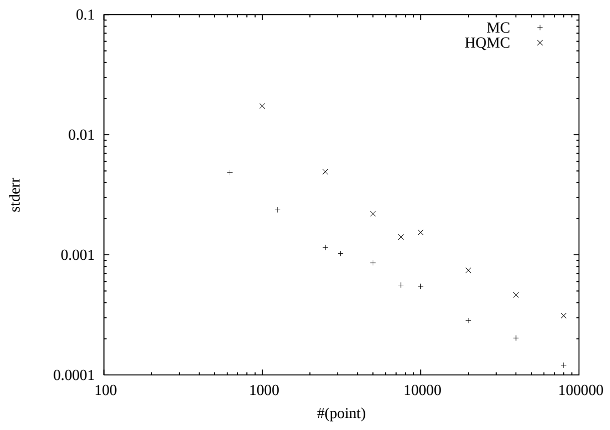

Figure 2: Convergence of standard error $s(l)/\bar{l}$ of MC and HQMC.

5 Concluding Remarks

We proposed a new simulation method using quasirandom point set for state space models. Numerical experiment shows faster convergence of the proposed method than traditional Monte Carlo. One question probably arises about the convergence rate: QMC usually shows the convergence rate of $O(1/n)$, where n is the number of sample points. In our experiments the convergence rate of HQMC is observed to be less than $O(1/n)$, and almost the same rate as MC. A question on the possibility or impossibility to improve the convergence rate is left to be investigated in a future work.

Acknowledgements

This work is supported by Grant-in-Aid for Scientific Research (C) 21510141, and partially (B) 23300104, from Japan Society for the Promotion of Science.

References

[1] Dick, J. and F. Pillichshammer: *Digital Nets and Sequences*, Cambridge UP, 2010.

[2] Doucet, A, N. de Feitas, and N. Gordon (eds.): *Sequential Monte Carlo Methods in Practice*, Springer, 2001.

[3] Guo, D. and X. Wang: Quasi-Monte Carlo Filtering in Nonlinear Dynamic Systems, *IEEE Trans. Signal Process.*, Vol. 54, No. 6, pp. 2087–2098, 2006.

[4] G. Kitagawa: *Introduction to Time Series Modeling*, CRC Press, 2010.

[5] P. L'Ecuyer, C. Lécot and A. L'Archevêque-Gaudet: On Array-RQMC for Markov Chains: Mapping Alternatives and Convergence Rates, Quasirandom Walk Methods, in Monte Carlo and Quasi-Monte Carlo Methods 2008, P. L'Ecuyer and A. B. Owen(eds.), Springer, 2009.

[6] P. L'Ecuyer, C. Lécot and B. Tuffin: A Randomized Quasi-Monte Carlo Simulation Method for Markov Chains, *Operations Research*, Vol. 56, No. 4, pp. 958–975, 2008.

[7] C. Lécot and S. Ogawa: Quasirandom Walk Methods, in Monte Carlo and Quasi-Monte Carlo Methods 2000, K.-T. Fang, F. J. Hickernell, and H. Niederreiter (eds.), Springer, 2002.

[8] C. Lemieux: *Monte Carlo and Quasi-Monte Carlo Sampling*, Springer, 2009.

[9] Niederreiter, H.: *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia, 1992.

[10] B. Vandewoestyne and R. Cools: On the Convergence of Quasi-random Sampling/Importance Resampling, *Mathematics and Computers in Simulation*, Vol. 81, pp. 490–505, 2010.