Conditional Sampling for Option Pricing under the LT Method

Nico Achtsis,
Dept. of Computer Science, K.U.Leuven, Belgium

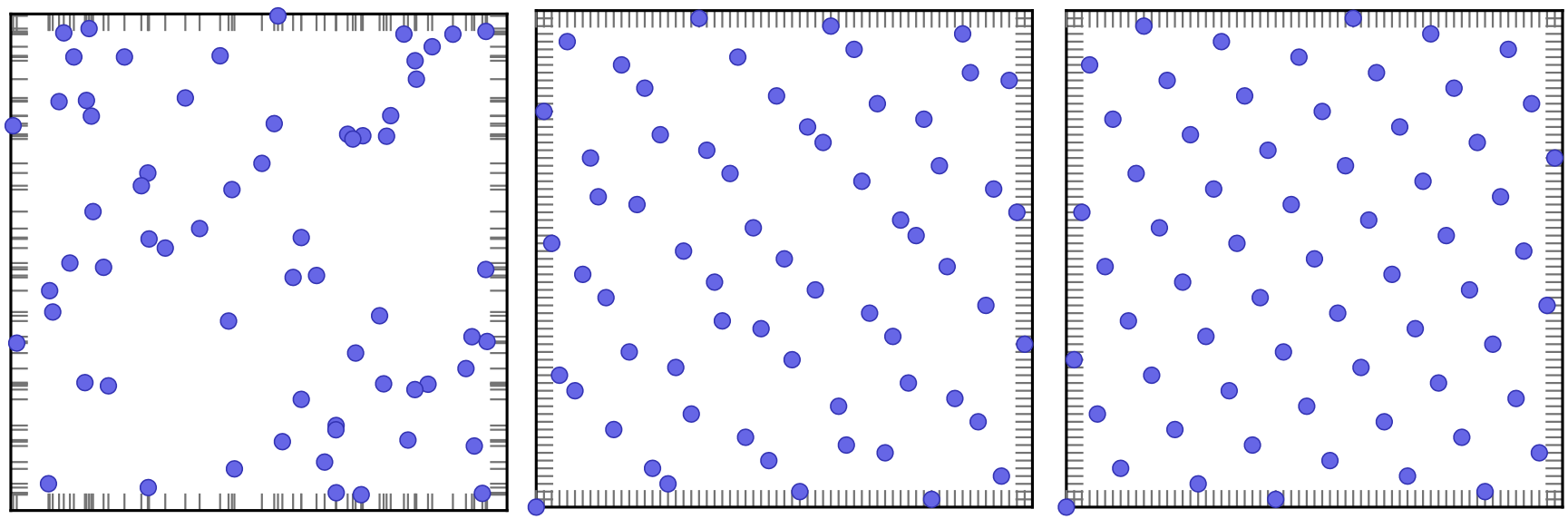
June 9th, 2011
(Joint work with Ronald Cools & Dirk Nuyens)

3rd International Conference on Numerical Methods for Finance (KBS, Limerick, Ireland)

- Monte Carlo versus quasi-Monte Carlo.
- The Linear Transformation (LT) algorithm.
- Conditional sampling.
- Results.

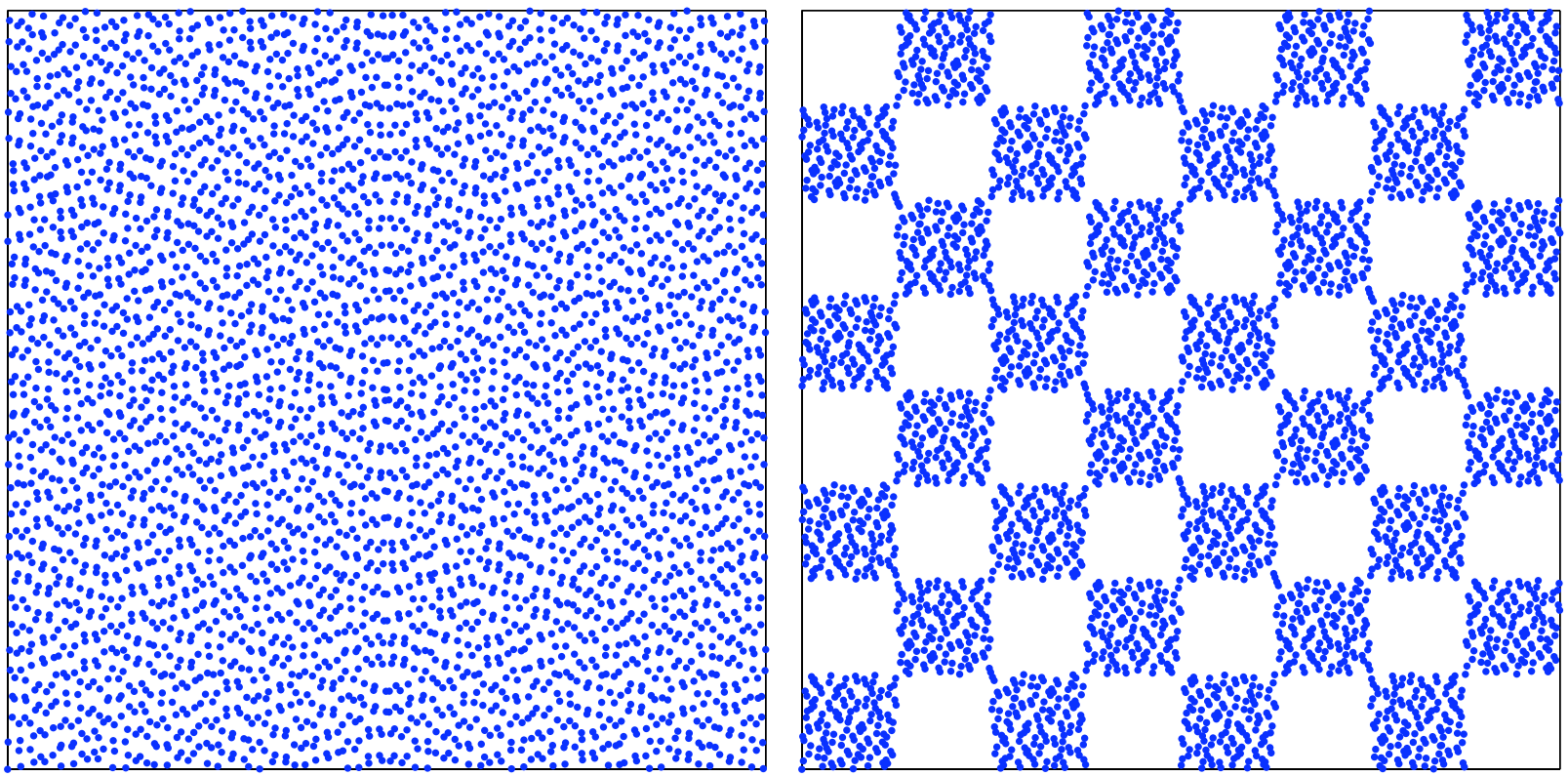
Quasi-Monte Carlo versus Monte Carlo point sets

- QMC point sets have the property that they are well distributed (middle: 64 Sobol' points and right: 64 lattice points).
- In contrast MC shows large gaps and clusters (left: 64 mt19937 points).



Opening dimensions are better

A typical property of QMC point sets is that opening dimensions are better distributed than the latter. This is typically illustrated with 2D-projections.
E.g., left: dimension 2 versus 9, right: dimension 20 versus 29.



We use scrambling to counter this effect.

Assumptions

We assume:

- n assets, m time steps;
- Black-Scholes dynamics, i.e.,

$$S_i(t) = S_i(0)e^{(r-\sigma_i^2/2)t+\sigma_i W_i(t)}.$$

- A European payoff represented as

$$V(S_1(t_1), \ldots, S_n(t_m)) = \max\left[g(S_1(t_1), \ldots, S_n(t_m)), 0\right].$$

Covariance matrix

Write $\mathbf{W} = (\sigma_1 W_1(t_1), \ldots, \sigma_1 W_1(t_m), \sigma_2 W_2(t_1), \ldots, \sigma_n W_n(t_m))'$.
$\mathbf{W}$ is multivariate normally distributed, with covariance matrix

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{21} & \cdots & \Sigma_{n1} \\ \Sigma_{12} & \Sigma_{22} & \cdots & \Sigma_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{1n} & \Sigma_{2n} & \cdots & \Sigma_{nn} \end{pmatrix},$$

where

$$\Sigma_{ij} = \begin{pmatrix} \rho_{ij}\sigma_i\sigma_j\Delta t & \rho_{ij}\sigma_i\sigma_j\Delta t & \cdots & \rho_{ij}\sigma_i\sigma_j\Delta t \\ \rho_{ij}\sigma_i\sigma_j\Delta t & 2\rho_{ij}\sigma_i\sigma_j\Delta t & \cdots & 2\rho_{ij}\sigma_i\sigma_j\Delta t \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{ij}\sigma_i\sigma_j\Delta t & 2\rho_{ij}\sigma_i\sigma_j\Delta t & \cdots & m\rho_{ij}\sigma_i\sigma_j\Delta t \end{pmatrix}.$$

Covariance matrix

- Under the LT method, instead of simulating g directly from $\mathbf{z}$, first an orthogonal transformation is applied, i.e., g is simulated from

$$\mathbf{W} = A\mathbf{z} = CQ\mathbf{z}$$

where CC' is the Cholesky decomposition of Σ and
Q is a carefully chosen orthogonal matrix.

- The matrix Q is chosen according to the following optimization problem:

$$\underset{Q_{\cdot k} \in \mathbb{R}^{mn}}{\text{maximize}} \quad \text{variance contribution of } g \text{ due to } k\text{th dimension}$$

$$\text{subject to} Q^T Q = 1.$$

- Note that introducing Q would not change the behaviour of ordinary Monte Carlo, but it makes a lot of difference to quasi-Monte Carlo methods.

LT algorithm

Therefore, Imai & Tan (2006) proposed to approximate the objective function by linearizing it using a first-order Taylor expansion around a point $\mathbf{z} = \hat{\mathbf{z}} + \Delta\mathbf{z}$,

$$g(\mathbf{z}) \approx g(\hat{\mathbf{z}}) + \sum_{\ell=1}^{mn} \left.\frac{\partial g}{\partial z_\ell}\right|_{\mathbf{z}=\hat{\mathbf{z}}} \Delta z_\ell.$$

Using this expansion, the variance contributed to the kth component is

$$\left(\left.\frac{\partial g}{\partial z_k}\right|_{\mathbf{z}=\hat{\mathbf{z}}} \right)^2.$$

The expansion points are chosen as $\hat{\mathbf{z}}_k = (1, \ldots, 1, 0, \ldots, 0)$, the vector with $k-1$ leading ones. The optimization problem becomes

$$\begin{aligned} \underset{Q_{\cdot k} \in \mathbb{R}^{mn}}{\text{maximize}} \quad & \left(\left.\frac{\partial g}{\partial z_\ell}\right|_{\mathbf{z}=\hat{\mathbf{z}}_k} \right)^2 \\ \text{subject to} \quad & Q^T Q = 1. \end{aligned}$$

LT algorithm

- If V is an arithmetic basket Asian option then there exists an analytical solution.
- For $k = 1$, this solution is found as

$$Q^*_{\cdot 1} = \pm \frac{\sum_{i=1}^{mn} e^{\mu_i} C_{i\cdot}}{\| \sum_{i=1}^{mn} e^{\mu_i} C_{i\cdot} \|}$$

- For subsequent k the solution is

$$Q^*_{\cdot k} = \pm \frac{\sum_{i=1}^{mn} e^{\mu_i + \sum_{j=1}^{k-1} \langle C_{i\cdot}, Q^*_{\cdot j} \rangle} \langle C_{i\cdot}, Q_{\cdot k} \rangle}{\| \sum_{i=1}^{mn} e^{\mu_i + \sum_{j=1}^{k-1} \langle C_{i\cdot}, Q^*_{\cdot j} \rangle} \langle C_{i\cdot}, Q_{\cdot k} \rangle \|}$$

Problem outline

- Suppose we are interested in pricing an up-&-out option with payoff

$$g(S_1(t_1), \ldots, S_n(t_m)) = f(S_1(t_1), \ldots, S_n(t_m))\, \mathbb{I}\left\{ \max_j S_1(t_j) < B \right\}.$$

- When the probability of survival becomes low (e.g., if $B - S_1(t_0)$ is small) a lot of the generated paths will result in a knock-out.
- This can make the variance among all paths quite large relative to the price.
- Remedy this problem by using conditional sampling.

MC+CS

- Suppose we are interested in pricing an up-&-out option with payoff

$$g(S_1(t_1), \ldots, S_n(t_m)) = f(S_1(t_1), \ldots, S_n(t_m)) \, \mathbb{I}\left\{ \max_j S_1(t_j) < B \right\}.$$

- Consider first an ordinary Monte Carlo estimation of the price
- Glasserman & Staum (2001) proposed the following estimator:

$$\mathbb{E}\left[e^{-rT} V\right] = \mathbb{E}\left[e^{-rT} \max(f, 0) \prod_{j=0}^{m-1} \mathbb{P}\left[S_1(t_{j+1}) < B \middle| S_1(t_j) \right] \right]$$

- Using this estimator there are no more paths that cross the barrier.

LT+CS

- The aim is to have a similar construction for the LT algorithm.
- For the option to stay alive we have the restriction

$$\sigma_1 W_1(t_j) < \underbrace{\log\left(\frac{B}{S_1(0)}\right) - \left(r - \frac{\sigma_1^2}{2}\right) t_j}_{=b(t_j)}, \quad j = 1, \ldots, m.$$

- Under the LT method,

$$\sigma_1 W_1(t_j) = \left(A\Phi^{-1}(\mathbf{u})\right)_j,$$

  where the vector $\mathbf{u}$ consists of mn uniform random variables.
- Rewriting, we need

$$\sum_{i=1}^{mn} a_{j,i}\Phi^{-1}(u_i) < b(t_j), \quad j = 1, \ldots, m.$$

LT+CS

- Assuming C has only non-negative elements we can modify the inequalities so that there is only a constraint on the first uniform variable:

$$\Phi^{-1}(u_1) < \frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}, \qquad j = 1, \ldots, m.$$

- For $u_2, \ldots, u_{mn}$ fixed, the restriction on u_1 can be written as

$$u_1 < \Phi\left(\min_j \left[\frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}\right]\right).$$

- Our algorithm thus samples u_1 to u_{mn}, then calculates the upper bound on u_1 using $u_2, \ldots, u_{mn}$, and afterwards rescales u_1 to satisfy the barrier condition.
- For a down-&-out option, a similar analysis can be done, leading to the condition

$$u_1 > \Phi\left(\max_j \left[\frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}\right]\right).$$

LT+CS

Consider now an up-&-in condition.

- For some t_j we must have

$$\sigma_1 W_1(t_j) \geq b(t_j).$$

- The difference with a knock-out type option is that now the above condition does not need to hold for all t_j: for an up-&-in option the condition on u_1 is

$$\Phi^{-1}(u_1) > \frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}$$

  for at least one j.
- We can essentially "force" the paths to cross the barrier level by imposing

$$u_1 > \Phi\left(\min_j \left[\frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}\right]\right),$$

  which lowers the minimum of the path (if necessary).
- For a down-&-in option the condition on u_1 is analoguously

$$u_1 < \Phi\left(\max_j \left[\frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}\right]\right).$$

LT+CS

In case C has negative elements, it is possible to have negative values for $a_{\cdot,1}$. Our algorithm can be slightly modified to work around this problem:

- suppose $a_{j,1} > 0$ for $j = j_1, \ldots, j_p \in \{1, \ldots, mn\}$ and $a_{j,1} < 0$ for all other j in $\{1, \ldots, mn\}$. Then we find the conditions on u_1 as

$$\Phi^{-1}(u_1) < \frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}, \qquad j \in \{j_1, \ldots, j_p\}$$

and

$$\Phi^{-1}(u_1) > \frac{b(t_j) - a_{j,2}\Phi^{-1}(u_2) - \ldots - a_{j,mn}\Phi^{-1}(u_{mn})}{a_{j,1}}, \qquad j \text{ elsewhere.}$$

We see that this case is in fact a combination of the previous up-&-out and down-&-out cases.

Results

Consider the following option payoff on one asset:

$$V = \mathbb{I}\left\{\frac{1}{60}\sum_{i=1}^{60} S(t_i) \geq 100\right\}\mathbb{I}\left\{\min_i S(t_i) \geq B^L\right\}\mathbb{I}\left\{\max_i S(t_i) \leq B^U\right\}.$$

The model parameters are $S(0) = 100$, $\sigma = 30\%$, $T = 3\text{m}$, $r = 0\%$ and $q = 0\%$.

(B^L, B^U)	LT+CS	LT	MC+CS
$(1, 1000)$	8.20×10^{-5}	8.20×10^{-5}	1.61×10^{-3}
$(50, 150)$	9.00×10^{-5}	1.48×10^{-4}	1.68×10^{-3}
$(90, 110)$	1.57×10^{-4}	5.00×10^{-4}	4.09×10^{-4}

Table: Results of the three methods for valuing a binary Asian option with an up and down barrier. The values are standard deviations based on 8192 samples and 10 independent shifts (except for the MC+CS method, which is based on 81920 samples).

Results

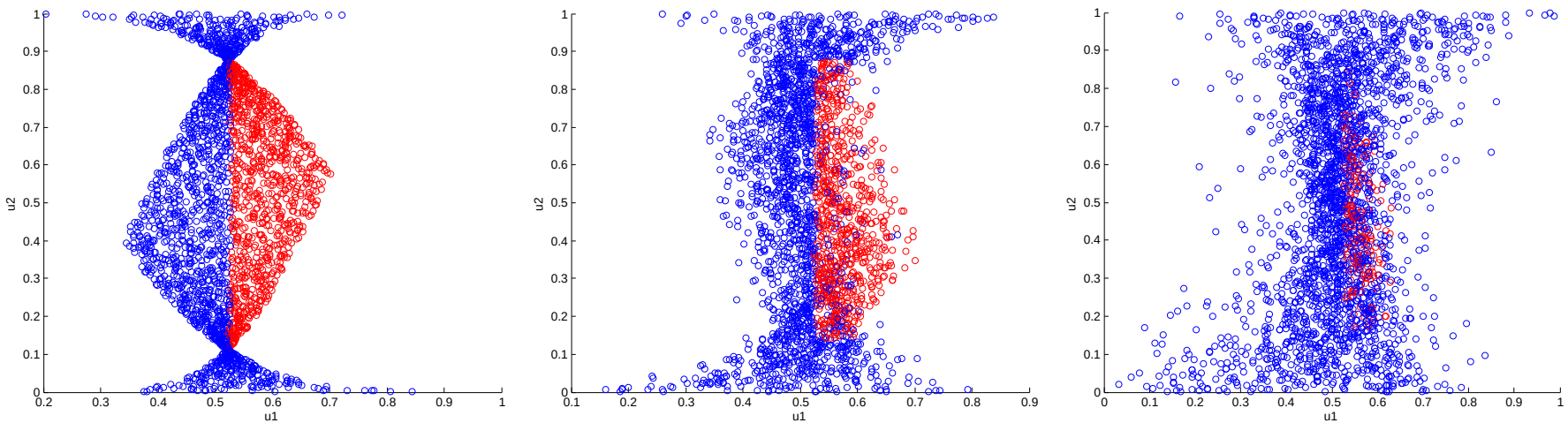
An extreme case would be taking $(B^L, B^U) = (98, 102)$.
We obtain the following results (in function of the number of observations m):

m	LT+CS	LT	MC+CS
2	6.84×10^{-6}	7.80×10^{-5}	4.16×10^{-5}
3	8.41×10^{-6}	8.55×10^{-5}	7.22×10^{-6}
4	9.30×10^{-6}	6.44×10^{-5}	4.01×10^{-6}

We see that the standard deviation for $m = 4$ for our new LT+CS method is higher than that of the MC+CS method! So what is happening here?

Results

We can plot the projection onto the first two dimensions:



Left: $m = 2$; middle: $m = 3$; right: $m = 16$.
We see that the LT algorithm is not able to separate the positive payoffs from the zeroes for increasing m.

Conclusion

- We constructed conditional (unbiased) sampling under the LT algorithm
- Significant variance reduction is achieved
- While the redundant paths are dealt with, new problems arise
- Currently we are working on an improvement of the algorithm