Introduction to (randomized) quasi-Monte Carlo

Pierre L’Ecuyer

MCQMC Conference, Stanford University, August 2016
Program

- Monte Carlo, Quasi-Monte Carlo, Randomized quasi-Monte Carlo
- QMC point sets and randomizations
- Error and variance bounds, convergence rates
- Transforming the integrand to make it more QMC friendly (smoother, smaller effective dimension, etc.).
- Numerical illustrations
- RQMC for Markov chains

Focus on ideas, insight, and examples.
Example: A stochastic activity network
Gives precedence relations between activities. Activity $k$ has random duration $Y_k$ (also length of arc $k$) with known cumulative distribution function (cdf) $F_k(y) := \mathbb{P}[Y_k \leq y]$.

Project duration $T = (\text{random})$ length of longest path from source to sink. May want to estimate $\mathbb{E}[T]$, $\mathbb{P}[T > x]$, a quantile, density of $T$, etc.
Monte Carlo (simulation)

**Algorithm: Monte Carlo to estimate** \( \mathbb{E}[T] \)

\[
\text{for } i = 0, \ldots, n - 1 \text{ do} \\
\quad \text{for } k = 0, \ldots, 12 \text{ do} \\
\quad \quad \text{Generate } U_k \sim U(0, 1) \text{ and let } Y_k = F_k^{-1}(U_k) \\
\quad \quad \text{Compute } X_i = T = h(Y_0, \ldots, Y_{12}) = f(U_0, \ldots, U_{12}) \\
\quad \text{Estimate } \mathbb{E}[T] = \int_{(0,1)^s} f(u) du \text{ by } \bar{X}_n = \frac{1}{n} \sum_{i=0}^{n-1} X_i, \text{ etc.}
\]

Can also compute confidence interval on \( \mathbb{E}[T] \), a histogram to estimate the distribution of \( T \), etc.

**Numerical illustration** from Elmaghraby (1977):
\( Y_k \sim N(\mu_k, \sigma_k^2) \) for \( k = 0, 1, 3, 10, 11 \), and \( V_k \sim \text{Expon}(1/\mu_k) \) otherwise.

\( \mu_0, \ldots, \mu_{12}: 13.0, 5.5, 7.0, 5.2, 16.5, 14.7, 10.3, 6.0, 4.0, 20.0, 3.2, 3.2, 16.5. \)

We may pay a penalty if \( T > 90 \), for example.
Naive idea: replace each $Y_k$ by its expectation. Gives $T = 48.2$. 
Naive idea: replace each $Y_k$ by its expectation. Gives $T = 48.2$.

Results of an experiment with $n = 100\,000$. Histogram of values of $T$ gives more information than confidence interval on $\mathbb{E}[T]$ or $\mathbb{P}[T > x]$.

Values from 14.4 to 268.6; 11.57% exceed $x = 90$. 

$\hat{\xi}_{0.99} = 131.8$
When Hurricane Sandy began swirling off the coast of Florida in 2012, the earliest forecasts suggested the gigantic storm was unlikely to hit land. If it wasn’t headed for the coast, everyone could relax. But if landfall was imminent, emergency workers would want as much time as possible to prepare.

Sandy, as we know, pummeled the Eastern Seaboard—especially New York and New Jersey—with damage reaching west all the way to Wisconsin. But thanks to computerized probability simulations, like the ones used for some financial forecasts, meteorologists tracking the storm weren’t caught off guard.

Monte Carlo simulations helped give emergency workers advance warning that Hurricane Sandy would make landfall in New Jersey and New York. Here, an Oct. 31, 2012, file photo of homes in Ortley Beach, N.J., destroyed by the storm.
Sample path of hurricane Sandy for the next 5 days
Monte Carlo to estimate an expectation

Want to estimate \( \mu = \mathbb{E}[X] \) where \( X = f(U) = f(U_0, \ldots, U_{s-1}) \), and the \( U_j \) are i.i.d. \( U(0,1) \) “random numbers.” We have

\[
\mu = \mathbb{E}[X] = \int_{[0,1)^s} f(u)\,du.
\]

Monte Carlo estimator:

\[
\bar{X}_n = \frac{1}{n} \sum_{i=0}^{n-1} X_i
\]

where \( X_i = f(U_i) \) and \( U_0, \ldots, U_{n-1} \) i.i.d. uniform over \([0,1)^s\).

We have \( \mathbb{E}[\bar{X}_n] = \mu \) and \( \text{Var}[\bar{X}_n] = \sigma^2/n = \text{Var}[X]/n. \)
Convergence

**Theorem.** Suppose $\sigma^2 < \infty$. When $n \to \infty$:

(i) **Strong law of large numbers**: $\lim_{n \to \infty} \hat{\mu}_n = \mu$ with probability 1.
Convergence

Theorem. Suppose $\sigma^2 < \infty$. When $n \to \infty$:

(i) Strong law of large numbers: $\lim_{n \to \infty} \hat{\mu}_n = \mu$ with probability 1.

(ii) Central limit theorem (CLT):

$$
\frac{\sqrt{n}(\hat{\mu}_n - \mu)}{S_n} \Rightarrow N(0, 1),
$$

where

$$
S_n^2 = \frac{1}{n-1} \sum_{i=0}^{n-1} (X_i - \bar{X}_n)^2.
$$
Confidence interval at level $\alpha$ (we want $\Phi(x) = 1 - \alpha/2$):

$$(\hat{\mu}_n \pm z_{\alpha/2} S_n / \sqrt{n}), \text{ where } z_{\alpha/2} = \Phi^{-1}(1 - \alpha/2).$$

Example: $z_{\alpha/2} \approx 1.96$ for $\alpha = 0.05$.

The width of the confidence interval is asymptotically proportional to $\sigma / \sqrt{n}$, so it converges as $O(n^{-1/2})$. Relative error: $\sigma / (\mu \sqrt{n})$. For one more decimal digit of accuracy, we must multiply $n$ by 100.
Confidence interval at level $\alpha$ (we want $\Phi(x) = 1 - \alpha/2$):

$$\left(\hat{\mu}_n \pm z_{\alpha/2} \frac{S_n}{\sqrt{n}}\right), \text{ where } z_{\alpha/2} = \Phi^{-1}(1 - \alpha/2).$$

Example: $z_{\alpha/2} \approx 1.96$ for $\alpha = 0.05$.

The width of the confidence interval is asymptotically proportional to $\sigma/\sqrt{n}$, so it converges as $O(n^{-1/2})$. Relative error: $\sigma/(\mu \sqrt{n})$.

For one more decimal digit of accuracy, we must multiply $n$ by 100.

Warning: If the $X_i$ have an asymmetric law, these confidence intervals can have very bad coverage (convergence to normal can be very slow).
Alternative estimator of $P[T > x] = E[I(T > x)]$ for SAN.

Naive estimator: Generate $T$ and compute $X = I[T > x]$. Repeat $n$ times and average.
Conditional Monte Carlo estimator of $\mathbb{P}[T > x]$. Generate the $Y_j$'s only for the 8 arcs that do not belong to the cut $\mathcal{L} = \{4, 5, 6, 8, 9\}$, and replace $\mathbb{I}[T > x]$ by its conditional expectation given those $Y_j$'s,

$$X_e = \mathbb{P}[T > x \mid \{Y_j, j \notin \mathcal{L}\}].$$

This makes the integrand continuous in the $U_j$'s.
Conditional Monte Carlo estimator of $\mathbb{P}[T > x]$. Generate the $Y_j$’s only for the 8 arcs that do not belong to the cut $\mathcal{L} = \{4, 5, 6, 8, 9\}$, and replace $\mathbb{I}[T > x]$ by its conditional expectation given those $Y_j$’s,

$$X_e = \mathbb{P}[T > x \mid \{Y_j, j \notin \mathcal{L}\}].$$

This makes the integrand continuous in the $U_j$’s.

To compute $X_e$: for each $l \in \mathcal{L}$, say from $a_l$ to $b_l$, compute the length $\alpha_l$ of the longest path from 1 to $a_l$, and the length $\beta_l$ of the longest path from $b_l$ to the destination.

The longest path that passes through link $l$ does not exceed $x$ iff $\alpha_l + Y_l + \beta_l \leq x$, which occurs with probability $\mathbb{P}[Y_l \leq x - \alpha_l - \beta_l] = F_l[x - \alpha_l - \beta_l]$. 

Can be faster to compute than $X$, and always has less variance.
Conditional Monte Carlo estimator of $\mathbb{P}[T > x]$. Generate the $Y_j$'s only for the 8 arcs that do not belong to the cut $\mathcal{L} = \{4, 5, 6, 8, 9\}$, and replace $\mathbb{I}[T > x]$ by its conditional expectation given those $Y_j$'s,

$$X_e = \mathbb{P}[T > x \mid \{Y_j, j \not\in \mathcal{L}\}].$$

This makes the integrand continuous in the $U_j$'s.

To compute $X_e$: for each $l \in \mathcal{L}$, say from $a_l$ to $b_l$, compute the length $\alpha_l$ of the longest path from 1 to $a_l$, and the length $\beta_l$ of the longest path from $b_l$ to the destination.

The longest path that passes through link $l$ does not exceed $x$ iff $\alpha_l + Y_l + \beta_l \leq x$, which occurs with probability $\mathbb{P}[Y_l \leq x - \alpha_l - \beta_l] = F_l[x - \alpha_l - \beta_l]$.

Since the $Y_l$ are independent, we obtain

$$X_e = 1 - \prod_{l \in \mathcal{L}} F_l[x - \alpha_l - \beta_l].$$

Can be faster to compute than $X$, and always has less variance.
Example: Pricing a financial derivative.

Market price of some asset (e.g., one share of a stock) evolves in time as stochastic process \( \{S(t), t \geq 0\} \) with (supposedly) known probability law (estimated from data).

A financial contract gives owner net payoff \( g(S(t_1), \ldots, S(t_d)) \) at time \( T = t_d \), where \( g : \mathbb{R}^d \to \mathbb{R} \), and \( 0 \leq t_1 < \cdots < t_d \) are fixed observation times.

Under a no-arbitrage assumption, present value (fair price) of contract at time 0, when \( S(0) = s_0 \), can be written as

\[
\nu(s_0, T) = \mathbb{E}^* \left[ e^{-rT} g(S(t_1), \ldots, S(t_d)) \right],
\]

where \( \mathbb{E}^* \) is under a risk-neutral measure and \( e^{-rT} \) is the discount factor.

This expectation can be written as an integral over \([0, 1)^s\) and estimated by the average of \( n \) i.i.d. replicates of \( X = e^{-rT} g(S(t_1), \ldots, S(t_d)) \).
A simple model for $S$: geometric Brownian motion (GBM):

$$S(t) = s_0 e^{(r - \sigma^2/2)t + \sigma B(t)}$$

where $r$ is the interest rate, $\sigma$ is the volatility, and $B(\cdot)$ is a standard Brownian motion: for any $t_2 > t_1 \geq 0$, $B(t_2) - B(t_1) \sim N(0, t_2 - t_1)$, and the increments over disjoint intervals are independent.
A simple model for $S$: geometric Brownian motion (GBM):

$$S(t) = s_0 e^{(r-\sigma^2/2)t+\sigma B(t)}$$

where $r$ is the interest rate, $\sigma$ is the volatility, and $B(\cdot)$ is a standard Brownian motion: for any $t_2 > t_1 \geq 0$, $B(t_2) - B(t_1) \sim \mathcal{N}(0, t_2 - t_1)$, and the increments over disjoint intervals are independent.

**Algorithm: Option pricing under GBM model**

```plaintext
for $i = 0, \ldots, n-1$ do
    Let $t_0 = 0$ and $B(t_0) = 0$
    for $j = 1, \ldots, d$ do
        Generate $U_j \sim U(0, 1)$ and let $Z_j = \Phi^{-1}(U_j)$
        Let $B(t_j) = B(t_{j-1}) + \sqrt{t_j - t_{j-1}} Z_j$
        Let $S(t_j) = s_0 \exp \left[(r - \sigma^2/2)t_j + \sigma B(t_j)\right]$
        Compute $X_i = e^{-rT} g(S(t_1), \ldots, S(t_d))$
    Return $\bar{X}_n = \frac{1}{n} \sum_{i=0}^{n-1} X_i$, estimator of $\nu(s_0, T)$.
```
Example of contract: Discretely-monitored Asian call option:

\[ g(S(t_1), \ldots, S(t_d)) = \max \left( 0, \frac{1}{d} \sum_{j=1}^{d} S(t_j) - K \right). \]

Option price written as an integral over the unit hypercube:

Let \( Z_j = \Phi^{-1}(U_j) \) where the \( U_j \) are i.i.d. \( U(0,1) \). Here we have \( s = d \) and

\[
\nu(s_0, T) = \int_{[0,1]^s} e^{-rT} \max \left( 0, \frac{1}{s} \sum_{i=1}^{s} s_0 \cdot \exp \left[ (r - \sigma^2/2)t_i + \sigma \sum_{j=1}^{i} \sqrt{t_j - t_{j-1}} \Phi^{-1}(u_j) \right] - K \right) du_1 \ldots du_s \\
= \int_{[0,1]^s} f(u_1, \ldots, u_s) du_1 \ldots du_s.
\]
Numerical illustration: Bermudean Asian option with $d = 12$, $T = 1$ (one year), $t_j = j/12$ for $j = 0, \ldots, 12$, $K = 100$, $s_0 = 100$, $r = 0.05$, $\sigma = 0.5$.

We performed $n = 10^6$ independent simulation runs. In 53.47% of cases, the payoff is 0. Mean: 13.1. Max = 390.8

Histogram of the 46.53% positive values:
Reducing the variance by changing $f$

If we replace the arithmetic average by a geometric average in the payoff, we obtain

$$ C = e^{-rT} \max \left( 0, \prod_{j=1}^{d} (S(t_j))^{1/d} - K \right), $$

whose expectation $\nu = \mathbb{E}[C]$ has a closed-form formula.

When estimating the mean $\mathbb{E}[X] = \nu(s_0, T)$, we can then use $C$ as a control variate (CV): Replace the estimator $X$ by the “corrected” version

$$ X_c = X - \beta (C - \nu) $$

for some well-chosen constant $\beta$. Optimal $\beta$ is $\beta^* = \text{Cov}[C, X]/\text{Var}[C]$. Using a CV makes the integrand $f$ smoother. Can provide a huge variance reduction, e.g., by a factor of over a million in some examples.
Quasi-Monte Carlo (QMC)

Replace the independent random points $U_i$ by a set of deterministic points $P_n = \{u_0, \ldots, u_{n-1}\}$ that cover $[0, 1)^s$ more evenly.

Estimate

$$\mu = \int_{[0,1)^s} f(u) \, du \quad \text{by} \quad \bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(u_i).$$

Integration error $E_n = \bar{\mu} - \mu$.

$P_n$ is called a highly-uniform point set or low-discrepancy point set if some measure of discrepancy between the empirical distribution of $P_n$ and the uniform distribution converges to 0 faster than $O(n^{-1/2})$ (the typical rate for independent random points).
Quasi-Monte Carlo (QMC)

Replace the independent random points $U_i$ by a set of deterministic points $P_n = \{u_0, \ldots, u_{n-1}\}$ that cover $[0, 1)^s$ more evenly.

Estimate

$$\mu = \int_{[0,1)^s} f(u) du \quad \text{by} \quad \bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(u_i).$$

Integration error $E_n = \bar{\mu} - \mu$.

$P_n$ is called a highly-uniform point set or low-discrepancy point set if some measure of discrepancy between the empirical distribution of $P_n$ and the uniform distribution converges to 0 faster than $O(n^{-1/2})$ (the typical rate for independent random points).

Main construction methods: lattice rules and digital nets
(Korobov, Hammersley, Halton, Sobol’, Faure, Niederreiter, etc.)
Simple case: one dimension \((s = 1)\)

Obvious solutions:

\[ P_n = \mathbb{Z}_n / n = \{0, 1/n, \ldots, (n - 1)/n\} \]  
(left Riemann sum):

\[
\begin{align*}
0 & \quad 0.5 & \quad 1 \\
\bullet & \quad \bullet & \quad \bullet & \quad \bullet & \quad \bullet
\end{align*}
\]

which gives

\[
\bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(i/n), \quad \text{and} \quad E_n = O(n^{-1}) \quad \text{if} \quad f' \quad \text{is bounded},
\]
Simple case: one dimension \((s = 1)\)

Obvious solutions:

\[
P_n = \mathbb{Z}_n/n = \{0, 1/n, \ldots, (n - 1)/n\} \text{ (left Riemann sum)}:
\]

\[
\begin{align*}
0 & \quad 0.5 & \quad 1 \\
\end{align*}
\]

which gives \(\bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(i/n)\), and \(E_n = \mathcal{O}(n^{-1})\) if \(f'\) is bounded,

or \(P'_n = \{1/(2n), 3/(2n), \ldots, (2n - 1)/(2n)\} \text{ (midpoint rule)}:
\]

\[
\begin{align*}
0 & \quad 0.5 & \quad 1 \\
\end{align*}
\]

for which \(E_n = \mathcal{O}(n^{-2})\) if \(f''\) is bounded.
If we allow different weights on the $f(u_i)$, we have the trapezoidal rule:

$$
\frac{1}{n} \left[ \frac{f(0) + f(1)}{2} + \sum_{i=1}^{n-1} f(i/n) \right],
$$

for which $|E_n| = O(n^{-2})$ if $f''$ is bounded,
If we allow different weights on the $f(u_i)$, we have the trapezoidal rule:

$$
\frac{1}{n} \left[ \frac{f(0) + f(1)}{2} + \sum_{i=1}^{n-1} f(i/n) \right],
$$

for which $|E_n| = O(n^{-2})$ if $f''$ is bounded, or the Simpson rule,

$$
\frac{f(0) + 4f(1/n) + 2f(2/n) + \cdots + 2f((n-2)/n) + 4f((n-1)/n) + f(1)}{3n},
$$

which gives $|E_n| = O(n^{-4})$ if $f^{(4)}$ is bounded, etc.
If we allow different weights on the $f(u_i)$, we have the trapezoidal rule:

$$\frac{1}{n} \left[ \frac{f(0) + f(1)}{2} + \sum_{i=1}^{n-1} f(i/n) \right],$$

for which $|E_n| = O(n^{-2})$ if $f''$ is bounded, or the Simpson rule,

$$\frac{f(0) + 4f(1/n) + 2f(2/n) + \cdots + 2f((n-2)/n) + 4f((n-1)/n) + f(1)}{3n},$$

which gives $|E_n| = O(n^{-4})$ if $f^{(4)}$ is bounded, etc.

Here, for QMC and RQMC, we restrict ourselves to equal weight rules. For the RQMC points that we will examine, one can prove that equal weights are optimal.
Simplistic solution for $s > 1$: rectangular grid

$P_n = \{(i_1/d, \ldots, i_s/d) \text{ such that } 0 \leq i_j < d \ \forall j\}$ where $n = d^s$. 

Midpoint rule in $s$ dimensions. Quickly becomes impractical when $s$ increases. Moreover, each one-dimensional projection has only $d$ distinct points, each two-dimensional projections has only $d^2$ distinct points, etc.
Simplistic solution for $s > 1$: rectangular grid

$P_n = \{(i_1/d, \ldots, i_s/d) \text{ such that } 0 \leq i_j < d \ \forall j \}$ where $n = d^s$.

Midpoint rule in $s$ dimensions.
Quickly becomes impractical when $s$ increases.
Moreover, each one-dimensional projection has only $d$ distinct points, each two-dimensional projections has only $d^2$ distinct points, etc.
Lattice rules (Korobov, Sloan, etc.)

Integration lattice:

\[
L_s = \left\{ \mathbf{v} = \sum_{j=1}^{s} z_j \mathbf{v}_j \text{ such that each } z_j \in \mathbb{Z} \right\},
\]

where \( \mathbf{v}_1, \ldots, \mathbf{v}_s \in \mathbb{R}^s \) are linearly independent over \( \mathbb{R} \) and where \( L_s \) contains \( \mathbb{Z}^s \). Lattice rule: Take \( P_n = \{ \mathbf{u}_0, \ldots, \mathbf{u}_{n-1} \} = L_s \cap [0, 1)^s \).
Lattice rules (Korobov, Sloan, etc.)

Integration lattice:

\[ L_s = \left\{ v = \sum_{j=1}^{s} z_j v_j \text{ such that each } z_j \in \mathbb{Z} \right\} , \]

where \( v_1, \ldots, v_s \in \mathbb{R}^s \) are linearly independent over \( \mathbb{R} \) and where \( L_s \) contains \( \mathbb{Z}^s \). Lattice rule: Take \( P_n = \{u_0, \ldots, u_{n-1}\} = L_s \cap [0,1)^s \).

Lattice rule of rank 1: \( u_i = iv_1 \mod 1 \) for \( i = 0, \ldots, n-1 \), where \( nv_1 = a = (a_1, \ldots, a_s) \in \{0,1,\ldots,n-1\}^s \).

Korobov rule: \( a = (1, a, a^2 \mod n, \ldots) \).
Lattice rules (Korobov, Sloan, etc.)

Integration lattice:

$$L_s = \left\{ \mathbf{v} = \sum_{j=1}^{s} z_j \mathbf{v}_j \text{ such that each } z_j \in \mathbb{Z} \right\},$$

where $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \mathbb{R}^s$ are linearly independent over $\mathbb{R}$ and where $L_s$ contains $\mathbb{Z}^s$. Lattice rule: Take $P_n = \{\mathbf{u}_0, \ldots, \mathbf{u}_{n-1}\} = L_s \cap [0,1)^s$.

Lattice rule of rank 1: $\mathbf{u}_i = i \mathbf{v}_1 \mod 1$ for $i = 0, \ldots, n - 1$, where $n \mathbf{v}_1 = \mathbf{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n - 1\}^s$.

Korobov rule: $\mathbf{a} = (1, a, a^2 \mod n, \ldots)$.

For any $u \subset \{1, \ldots, s\}$, the projection $L_s(u)$ of $L_s$ is also a lattice.
Example: lattice with $s = 2$, $n = 101$, $\mathbf{v}_1 = (1, 12)/n$

$$P_n = \{ u_i = i\mathbf{v}_1 \mod 1 : i = 0, \ldots, n-1 \}$$
$$= \{(0, 0), (1/101, 12/101), (2/101, 43/101), \ldots \}.$$ 

Here, each one-dimensional projection is \{0, 1/n, \ldots, (n-1)/n\}. 
Example: lattice with $s = 2$, $n = 101$, $\mathbf{v}_1 = (1, 12)/n$

$$P_n = \{ u_i = i\mathbf{v}_1 \mod 1 : i = 0, \ldots, n-1 \}$$
$$= \{(0, 0), (1/101, 12/101), (2/101, 43/101), \ldots \}.$$

Here, each one-dimensional projection is $\{0, 1/n, \ldots, (n-1)/n\}$. 
Example: lattice with \( s = 2, \ n = 101, \ \mathbf{v}_1 = (1, 12)/n \)

\[
P_n = \{u_i = iv_1 \mod 1 : i = 0, \ldots, n-1\} \\
= \{(0,0), (1/101, 12/101), (2/101, 43/101), \ldots \}.
\]

Here, each one-dimensional projection is \{0, 1/n, \ldots, (n-1)/n\}.
Example: lattice with $s = 2$, $n = 101$, $\mathbf{v}_1 = (1, 12)/n$

\[
P_n = \{\mathbf{u}_i = i\mathbf{v}_1 \mod 1 : i = 0, \ldots, n-1\}
= \{(0, 0), (1/101, 12/101), (2/101, 43/101), \ldots \}.
\]

Here, each one-dimensional projection is $\{0, 1/n, \ldots, (n-1)/n\}$. 
Example: lattice with $s = 2$, $n = 101$, $\mathbf{v}_1 = (1, 12)/n$

\[ P_n = \{ \mathbf{u}_i = i\mathbf{v}_1 \mod 1 : i = 0, \ldots, n-1 \} \]
\[ = \{(0,0), (1/101, 12/101), (2/101, 43/101), \ldots \}. \]

Here, each one-dimensional projection is \{0, 1/n, \ldots, (n−1)/n\}. 
Another example: \( s = 2, \ n = 1021, \ v_1 = (1, 90)/n \)

\[
P_n = \{ u_i = i v_1 \mod 1 : i = 0, \ldots, n - 1 \} = \{ (i/1021, (90i/1021) \mod 1) : i = 0, \ldots, 1020 \}.
\]
A bad lattice: $s = 2$, $n = 101$, $v_1 = (1, 51)/n$

Good uniformity in one dimension, but not in two!
Digital net in base $b$ (Niederreiter)

Gives $n = b^k$ points. For $i = 0, \ldots, b^k - 1$ and $j = 1, \ldots, s$:

$$i = a_{i,0} + a_{i,1}b + \cdots + a_{i,k-1}b^{k-1} = a_{i,k-1} \cdots a_{i,1}a_{i,0},$$

$$\begin{pmatrix} u_{i,j,1} \\ \vdots \\ u_{i,j,w} \end{pmatrix} = C_j \begin{pmatrix} a_{i,0} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \mod b,$$

$$u_{i,j} = \sum_{\ell=1}^{w} u_{i,j,\ell}b^{-\ell}, \quad u_i = (u_{i,1}, \ldots, u_{i,s}),$$

where the generating matrices $C_j$ are $w \times k$ with elements in $\mathbb{Z}_b$.

In practice, $w$ and $k$ are finite, but there is no limit.

**Digital sequence:** infinite sequence. Can stop at $n = b^k$ for any $k$. 

Digital net in base $b$ (Niederreiter)

Gives $n = b^k$ points. For $i = 0, \ldots, b^k - 1$ and $j = 1, \ldots, s$: 

$$i = a_{i,0} + a_{i,1}b + \cdots + a_{i,k-1}b^{k-1} = a_{i,0}a_{i,1}a_{i,0},$$

$$\begin{pmatrix}
  u_{i,j,1} \\
  \vdots \\
  u_{i,j,w}
\end{pmatrix} = C_j \begin{pmatrix}
  a_{i,0} \\
  \vdots \\
  a_{i,k-1}
\end{pmatrix} \mod b,$$

$$u_{i,j} = \sum_{\ell=1}^{w} u_{i,j,\ell}b^{-\ell}, \quad u_i = (u_{i,1}, \ldots, u_{i,s}),$$

where the generating matrices $C_j$ are $w \times k$ with elements in $\mathbb{Z}_b$.

In practice, $w$ and $k$ are finite, but there is no limit.

**Digital sequence:** infinite sequence. Can stop at $n = b^k$ for any $k$.

Can also multiply in some ring $R$, with bijections between $\mathbb{Z}_b$ and $R$.

Each one-dim projection truncated to first $k$ digits is

$$\mathbb{Z}_n/n = \{0, 1/n, \ldots, (n-1)/n\}.$$ 

Each $C_j$ defines a permutation of $\mathbb{Z}_n/n$. 
Small example: Hammersley in two dimensions

Let \( n = 2^8 = 256 \) and \( s = 2 \). Take the points (in binary):

<table>
<thead>
<tr>
<th>( i )</th>
<th>( u_{1,i} )</th>
<th>( u_{2,i} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.00000000</td>
<td>.0</td>
</tr>
<tr>
<td>1</td>
<td>.00000001</td>
<td>.1</td>
</tr>
<tr>
<td>2</td>
<td>.00000010</td>
<td>.01</td>
</tr>
<tr>
<td>3</td>
<td>.00000011</td>
<td>.11</td>
</tr>
<tr>
<td>4</td>
<td>.00000100</td>
<td>.001</td>
</tr>
<tr>
<td>5</td>
<td>.00000101</td>
<td>.101</td>
</tr>
<tr>
<td>6</td>
<td>.00000110</td>
<td>.011</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>254</td>
<td>.11111110</td>
<td>.01111111</td>
</tr>
<tr>
<td>255</td>
<td>.11111111</td>
<td>.11111111</td>
</tr>
</tbody>
</table>

Right side: van der Corput sequence in base 2.
Hammersley point set, \( n = 2^8 = 256, \ s = 2. \)
Hammersley point set, \( n = 2^8 = 256, \ s = 2. \)
Hammersley point set, $n = 2^8 = 256$, $s = 2$. 
Hammersley point set, $n = 2^8 = 256$, $s = 2$. 
Hammersley point set, \( n = 2^8 = 256, \ s = 2 \).
In general, can take $n = 2^k$ points.

If we partition $[0, 1)^2$ in rectangles of sizes $2^{-k_1}$ by $2^{-k_2}$ where $k_1 + k_2 \leq k$, each rectangle will contain exactly the same number of points. We say that the points are **equidistributed** for this partition.
In general, can take $n = 2^k$ points.

If we partition $[0, 1)^2$ in rectangles of sizes $2^{-k_1}$ by $2^{-k_2}$ where $k_1 + k_2 \leq k$, each rectangle will contain exactly the same number of points. We say that the points are equidistributed for this partition.

For a digital net in base $b$ in $s$ dimensions, we choose $s$ permutations of \{0, 1, \ldots, 2^b - 1\}, then divide each coordinate by $b^k$.

Can also have $s = \infty$ and/or $n = \infty$ (infinite sequence of points).
Suppose we divide axis $j$ in $b^{q_j}$ equal parts, for each $j$. This determines a partition of $[0, 1)^s$ into $2^{q_1+\cdots+q_s}$ rectangles of equal sizes. If each rectangle contains exactly the same number of points, we say that the point set $P_n$ is $(q_1, \ldots, q_s)$-equidistributed in base $b$.

This occurs iff the matrix formed by the first $q_1$ rows of $C_1$, the first $q_2$ rows of $C_2$, \ldots, the first $q_s$ rows of $C_s$, is of full rank (mod $b$). To verify equidistribution, we can construct these matrices and compute their rank.

$P_n$ is a $(t, k, s)$-net iff it is $(q_1, \ldots, q_s)$-equidistributed whenever $q_1 + \cdots + q_s = k - t$. This is possible for $t = 0$ only if $b \geq s - 1$.

t-value of a net: smallest $t$ for which it is a $(t, k, s)$-net.
Suppose we divide axis $j$ in $b^{q_j}$ equal parts, for each $j$. This determines a partition of $[0, 1)^s$ into $2^{q_1 + \cdots + q_s}$ rectangles of equal sizes. If each rectangle contains exactly the same number of points, we say that the point set $P_n$ is $(q_1, \ldots, q_s)$-equidistributed in base $b$.

This occurs iff the matrix formed by the first $q_1$ rows of $C_1$, the first $q_2$ rows of $C_2$, \ldots, the first $q_s$ rows of $C_s$, is of full rank (mod $b$). To verify equidistribution, we can construct these matrices and compute their rank.

$P_n$ is a $(t, k, s)$-net iff it is $(q_1, \ldots, q_s)$-equidistributed whenever $q_1 + \cdots + q_s = k - t$. This is possible for $t = 0$ only if $b \geq s - 1$.

$t$-value of a net: smallest $t$ for which it is a $(t, k, s)$-net.

An infinite sequence $\{u_0, u_1, \ldots, \}$ in $[0, 1)^s$ is a $(t, s)$-sequence in base $b$ if for all $k > 0$ and $\nu \geq 0$, $Q(k, \nu) = \{u_i : i = \nu b^k, \ldots, (\nu + 1)b^k - 1\}$, is a $(t, k, s)$-net in base $b$. 
Suppose we divide axis \( j \) in \( b^{q_j} \) equal parts, for each \( j \). This determines a partition of \([0, 1)^s\) into \( 2^{q_1} + \cdots + q_s \) rectangles of equal sizes. If each rectangle contains exactly the same number of points, we say that the point set \( P_n \) is \((q_1, \ldots, q_s)\)-equidistributed in base \( b \).

This occurs iff the matrix formed by the first \( q_1 \) rows of \( C_1 \), the first \( q_2 \) rows of \( C_2 \), \ldots, the first \( q_s \) rows of \( C_s \), is of full rank (mod \( b \)). To verify equidistribution, we can construct these matrices and compute their rank.

\( P_n \) is a \((t, k, s)\)-net iff it is \((q_1, \ldots, q_s)\)-equidistributed whenever \( q_1 + \cdots + q_s = k - t \). This is possible for \( t = 0 \) only if \( b \geq s - 1 \).

\( t \)-value of a net: smallest \( t \) for which it is a \((t, k, s)\)-net.

An infinite sequence \( \{u_0, u_1, \ldots, \} \) in \([0, 1)^s\) is a \((t, s)\)-sequence in base \( b \) if for all \( k > 0 \) and \( \nu \geq 0 \), \( Q(k, \nu) = \{u_i : i = \nu b^k, \ldots, (\nu + 1)b^k - 1\} \), is a \((t, k, s)\)-net in base \( b \). This is possible for \( t = 0 \) only if \( b \geq s \).
Sobol’ nets and sequences

Sobol’ (1967) proposed a digital net in base $b = 2$ where

$$
C_j = \begin{pmatrix}
1 & v_{j,2,1} & \cdots & v_{j,c,1} & \cdots \\
0 & 1 & \cdots & v_{j,c,2} & \cdots \\
\vdots & 0 & \ddots & \vdots & \ddots \\
\vdots & & \ddots & 1 & \ddots \\
\end{pmatrix}.
$$

The integers $m_{j,c}$ are selected as follows.
Sobol’ nets and sequences

Sobol’ (1967) proposed a digital net in base $b = 2$ where

$$C_j = \begin{pmatrix}
1 & v_{j,2,1} & \cdots & v_{j,c,1} & \cdots \\
0 & 1 & \cdots & v_{j,c,2} & \cdots \\
\vdots & 0 & \ddots & \vdots \\
\vdots & \vdots & \ddots & 1
\end{pmatrix}.$$ 

Column $c$ of $C_j$ is represented by an odd integer

$$m_{j,c} = \sum_{l=1}^{c} v_{j,c,l}2^{c-l} = v_{j,c,1}2^{c-1} + \cdots + v_{j,c,c-1}2 + 1 < 2^c.$$ 

The integers $m_{j,c}$ are selected as follows.
For each \( j \), we choose a primitive polynomial over \( \mathbb{F}_2 \),

\[
f_j(z) = z^{d_j} + a_{j,1}z^{d_j-1} + \cdots + a_{j,d_j},
\]

and we choose \( d_j \) integers \( m_{j,0}, \ldots, m_{j,d_j-1} \) (the first \( d_j \) columns).
For each $j$, we choose a primitive polynomial over $\mathbb{F}_2$,

$$f_j(z) = z^{d_j} + a_{j,1}z^{d_j-1} + \cdots + a_{j,d_j},$$

and we choose $d_j$ integers $m_{j,0}, \ldots, m_{j,d_j-1}$ (the first $d_j$ columns). Then, $m_{j,d_j}, m_{j,d_j+1}, \ldots$ are determined by the recurrence

$$m_{j,c} = 2a_{j,1}m_{j,c-1} + \cdots + 2^{d_j-1}a_{j,d_j-1}m_{j,c-d_j+1} + 2^{d_j}m_{j,c-d_j}$$

**Proposition.** If the polynomials $f_j(z)$ are all distinct, we obtain a $(t,s)$-sequence with $t \leq d_0 + \cdots + d_{s-1} + 1 - s$. 
For each $j$, we choose a primitive polynomial over $\mathbb{F}_2$,

$$f_j(z) = z^{d_j} + a_{j,1}z^{d_j-1} + \cdots + a_{j,d_j},$$

and we choose $d_j$ integers $m_{j,0},\ldots,m_{j,d_j-1}$ (the first $d_j$ columns).

Then, $m_{j,d_j},m_{j,d_j+1},\ldots$ are determined by the recurrence

$$m_{j,c} = 2a_{j,1}m_{j,c-1} \oplus \cdots \oplus 2^{d_j-1}a_{j,d_j-1}m_{j,c-d_j+1} \oplus 2^{d_j}m_{j,c-d_j} \oplus m_{j,c-d_j}$$

**Proposition.** If the polynomials $f_j(z)$ are all distinct, we obtain a $(t,s)$-sequence with $t \leq d_0 + \cdots + d_{s-1} + 1 - s$.

Sobol’ suggests to list all primitive polynomials over $\mathbb{F}_2$ by increasing order of degree, starting with $f_0(z) \equiv 1$ (which gives $C_0 = I$), and to take $f_j(z)$ as the $(j+1)$-th polynomial in the list.

There are many ways of selecting the first $m_{j,c}$’s, which are called the **direction numbers**. They can be selected to minimize some discrepancy (or figure of merit). The values proposed by Sobol’ give an $(s,\ell)$-equidistribution for $\ell = 1$ and $\ell = 2$ (only the first two bits).

For $n = 2^k$ fixed, we can gain one dimension as for the Faure sequence.

Joe and Kuo (2008) tabulated direction numbers giving the best $t$-value for the two-dimensional projections, for given $s$ and $k$. 
Other constructions

Faure nets and sequences
Niederreiter-Xing point sets and sequences
Polynomial lattice rules (special case of digital nets)
Halton sequence
Etc.
Worst-case error bounds

Koksma-Hlawka-type inequalities (Koksma, Hlawka, Hickernell, etc.):

\[ |\hat{\mu}_{n,\text{rqmc}} - \mu| \leq V(f) \cdot D(P_n) \]

for all \( f \) in some Hilbert space or Banach space \( \mathcal{H} \), where
\( V(f) = \| f - \mu \|_\mathcal{H} \) is the variation of \( f \), and \( D(P_n) \) is the discrepancy of \( P_n \).
Worst-case error bounds

Koksma-Hlawka-type inequalities (Koksma, Hlawka, Hickernell, etc.):

\[ |\hat{\mu}_{n,\text{rqmc}} - \mu| \leq V(f) \cdot D(P_n) \]

for all \( f \) in some Hilbert space or Banach space \( \mathcal{H} \), where \( V(f) = \|f - \mu\|_\mathcal{H} \) is the variation of \( f \), and \( D(P_n) \) is the discrepancy of \( P_n \).

**Lattice rules:** For certain Hilbert spaces of smooth periodic functions \( f \) with square-integrable partial derivatives of order up to \( \alpha \):

\[ D(P_n) = O(n^{-\alpha + \epsilon}) \]

for arbitrary small \( \epsilon \).

**Digital nets:** “Classical” Koksma-Hlawka inequality for QMC: \( f \) must have finite variation in the sense of Hardy and Krause (implies no discontinuity not aligned with the axes). Popular constructions achieve

\[ D(P_n) = O(n^{-1} (\ln n)^s) = O(n^{-1+\epsilon}) \]

for arbitrary small \( \epsilon \).

More recent constructions offer better rates for smooth functions.
Worst-case error bounds
Koksma-Hlawka-type inequalities (Koksma, Hlawka, Hickernell, etc.):

\[ |\hat{\mu}_{n,\text{rqmc}} - \mu| \leq V(f) \cdot D(P_n) \]

for all \( f \) in some Hilbert space or Banach space \( \mathcal{H} \), where \( V(f) = \|f - \mu\|_{\mathcal{H}} \) is the variation of \( f \), and \( D(P_n) \) is the discrepancy of \( P_n \).

**Lattice rules:** For certain Hilbert spaces of smooth periodic functions \( f \) with square-integrable partial derivatives of order up to \( \alpha \):

\[ D(P_n) = O(n^{-\alpha+\epsilon}) \]

for arbitrary small \( \epsilon \).

**Digital nets:** “Classical” Koksma-Hlawka inequality for QMC: \( f \) must have finite variation in the sense of Hardy and Krause (implies no discontinuity not aligned with the axes). Popular constructions achieve

\[ D(P_n) = O(n^{-1}(\ln n)^s) = O(n^{-1+\epsilon}) \]

for arbitrary small \( \epsilon \). More recent constructions offer better rates for smooth functions.

Bounds are conservative and too hard to compute in practice.
Randomized quasi-Monte Carlo (RQMC)

\[ \hat{\mu}_{n,\text{rqmc}} = \frac{1}{n} \sum_{i=0}^{n-1} f(U_i), \]

with \( P_n = \{ U_0, \ldots, U_{n-1} \} \subset (0, 1)^s \) an RQMC point set:

(i) each point \( U_i \) has the uniform distribution over \((0, 1)^s\);
(ii) \( P_n \) as a whole is a low-discrepancy point set.

\[ \mathbb{E}[\hat{\mu}_{n,\text{rqmc}}] = \mu \quad \text{(unbiased)}. \]

\[ \text{Var}[\hat{\mu}_{n,\text{rqmc}}] = \frac{\text{Var}[f(U_i)]}{n} + \frac{2}{n^2} \sum_{i<j} \text{Cov}[f(U_i), f(U_j)]. \]

We want to make the last sum as negative as possible.
Randomized quasi-Monte Carlo (RQMC)

\[ \hat{\mu}_{n,\text{rqmc}} = \frac{1}{n} \sum_{i=0}^{n-1} f(U_i), \]

with \( P_n = \{U_0, \ldots, U_{n-1}\} \subset (0, 1)^s \) an RQMC point set:

(i) each point \( U_i \) has the uniform distribution over \((0, 1)^s\);
(ii) \( P_n \) as a whole is a low-discrepancy point set.

\[ \mathbb{E}[\hat{\mu}_{n,\text{rqmc}}] = \mu \quad \text{(unbiased)}. \]

\[ \text{Var}[\hat{\mu}_{n,\text{rqmc}}] = \frac{\text{Var}[f(U_i)]}{n} + \frac{2}{n^2} \sum_{i<j} \text{Cov}[f(U_i), f(U_j)]. \]

We want to make the last sum as negative as possible.
Weaker attempts to do the same: antithetic variates \((n = 2)\), Latin hypercube sampling (LHS), stratification, ...
Variance estimation:

Can compute \( m \) independent realizations \( X_1, \ldots, X_m \) of \( \hat{\mu}_{n,\text{rqmc}} \), then estimate \( \mu \) and \( \text{Var}[\hat{\mu}_{n,\text{rqmc}}] \) by their sample mean \( \bar{X}_m \) and sample variance \( S_m^2 \). Could be used to compute a confidence interval.

Temptation: assume that \( \bar{X}_m \) has the normal distribution.
Beware: usually wrong unless \( m \to \infty \).
Stratification of the unit hypercube

Partition axis $j$ in $k_j \geq 1$ equal parts, for $j = 1, \ldots, s$.
Draw $n = k_1 \cdots k_s$ random points, one per box, independently.

**Example**, $s = 2$, $k_1 = 12$, $k_2 = 8$, $n = 12 \times 8 = 96$. 
Stratification of the unit hypercube

Example, $s = 2$, $k_1 = 24$, $k_2 = 16$, $n = 384$. 
Stratified estimator:

\[ X_{s,n} = \frac{1}{n} \sum_{j=0}^{n-1} f(U_j). \]

The crude MC variance with \( n \) points can be decomposed as

\[ \text{Var}[\bar{X}_n] = \text{Var}[X_{s,n}] + \frac{1}{n} \sum_{j=0}^{n-1} (\mu_j - \mu)^2 \]

where \( \mu_j \) is the mean over box \( j \).

The more the \( \mu_j \) differ, the more the variance is reduced.
Stratified estimator:

\[ X_{s,n} = \frac{1}{n} \sum_{j=0}^{n-1} f(U_j). \]

The crude MC variance with \( n \) points can be decomposed as

\[ \text{Var}[\bar{X}_n] = \text{Var}[X_{s,n}] + \frac{1}{n} \sum_{j=0}^{n-1} (\mu_j - \mu)^2 \]

where \( \mu_j \) is the mean over box \( j \).

The more the \( \mu_j \) differ, the more the variance is reduced.

If \( f' \) is continuous and bounded, and all \( k_j \) are equal, then

\[ \text{Var}[X_{s,n}] = O(n^{-1-2/s}). \]
Stratified estimator:

\[
X_{s,n} = \frac{1}{n} \sum_{j=0}^{n-1} f(U_j).
\]

The crude MC variance with \( n \) points can be decomposed as

\[
\text{Var}[\bar{X}_n] = \text{Var}[X_{s,n}] + \frac{1}{n} \sum_{j=0}^{n-1} (\mu_j - \mu)^2
\]

where \( \mu_j \) is the mean over box \( j \).

The more the \( \mu_j \) differ, the more the variance is reduced.

If \( f' \) is continuous and bounded, and all \( k_j \) are equal, then

\[
\text{Var}[X_{s,n}] = O(n^{-1-2/s}).
\]

For large \( s \), not practical. For small \( s \), not really better than midpoint rule with a grid when \( f \) is smooth. But can still be applied to a few important random variables.

Also, gives an unbiased estimator, and variance can be estimated by replicating \( m \geq 2 \) times.
Randomly-Shifted Lattice

Example: lattice with $s = 2$, $n = 101$, $v_1 = (1, 12)/101$
Randomly-Shifted Lattice

Example: lattice with $s = 2$, $n = 101$, $\mathbf{v}_1 = (1, 12)/101$
Randomly-Shifted Lattice

Example: lattice with $s = 2$, $n = 101$, $v_1 = (1, 12)/101$
Randomly-Shifted Lattice

Example: lattice with $s = 2$, $n = 101$, $\mathbf{v}_1 = (1, 12)/101$
Random digital shift for digital net

Equidistribution in digital boxes is lost with random shift modulo 1, but can be kept with a random digital shift in base $b$.

In base 2: Generate $U \sim U(0,1)^s$ and XOR it bitwise with each $u_i$.

Example for $s = 2$:

$$u_i = (0.01100100..., 0.10011000...)_2$$
$$U = (0.01001010..., 0.11101001...)_2$$
$$u_i \oplus U = (0.00101110..., 0.01110001...)_2.$$

Each point has $U(0,1)$ distribution.

Preservation of the equidistribution ($k_1 = 3$, $k_2 = 5$):

$$u_i = (0.***, 0.******)$$
$$U = (0.010, 0.11101)_2$$
$$u_i \oplus U = (0.***, 0.******)$$
Example with

\[ \mathbf{U} = (0.1270111220, 0.3185275653)_{10} \]
\[ = (0.0010 000100000111100, 0.0101 0001100010111000)_{2}. \]

Changes the bits 3, 9, 15, 16, 17, 18 of \( u_{i,1} \)
and the bits 2, 4, 8, 9, 13, 15, 16 of \( u_{i,2} \).

Red and green squares are permuted (\( k_1 = k_2 = 4 \), first 4 bits of \( \mathbf{U} \)).
Random digital shift in base $b$

We have $u_{i,j} = \sum_{\ell=1}^{w} u_{i,j,\ell} b^{-\ell}$.
Let $U = (U_1, \ldots, U_s) \sim U[0, 1)^s$ where $U_j = \sum_{\ell=1}^{w} U_{j,\ell} b^{-\ell}$.

We replace each $u_{i,j}$ by $\tilde{U}_{i,j} = \sum_{\ell=1}^{w} [(u_{i,j,\ell} + U_{j,\ell}) \mod b] b^{-\ell}$.

**Proposition.** $\tilde{P}_n$ is $(q_1, \ldots, q_s)$-equidistributed in base $b$ iff $P_n$ is.
For $w = \infty$, each point $\tilde{U}_j$ has the uniform distribution over $(0, 1)^s$. 
Other permutations that preserve equidistribution and may help reduce the variance further:

**Linear matrix scrambling (Matoušek, Hickernell et Hong, Tezuka, Owen):**
We left-multiply each matrix $C_j$ by a random $w \times w$ matrix $M_j$, non-singular and lower triangular, mod $b$. Several variants.

We then apply a random digital shift in base $b$ to obtain uniform distribution for each point (unbiasedness).
Other permutations that preserve equidistribution and may help reduce the variance further:

**Linear matrix scrambling** (Matoušek, Hickernell et al., Hong, Tezuka, Owen): We left-multiply each matrix $C_j$ by a random $w \times w$ matrix $M_j$, non-singular and lower triangular, mod $b$. Several variants.

We then apply a random digital shift in base $b$ to obtain uniform distribution for each point (unbiasedness).

**Nested uniform scrambling** (Owen 1995). More costly. But provably reduces the variance to $O(n^{-3}(\log n)^{s})$ when $f$ is sufficiently smooth!
Asian option example

\( T = 1 \) (year), \( t_j = j/d, K = 100, s_0 = 100, r = 0.05, \sigma = 0.5. \)
\( s = d = 2. \) Exact value: \( \mu \approx 17.0958. \) MC Variance: \( 934.0. \)

Lattice: Korobov with \( a \) from old table + random shift.
Sobol: left matrix scramble + random digital shift.

Variance estimated from \( m = 1000 \) indep. randomizations.
\[ VRF = \frac{\text{MC variance}}{\text{nVar}[X_{s,n}]} \]

<table>
<thead>
<tr>
<th>Method</th>
<th>( n )</th>
<th>( \bar{X}_m )</th>
<th>( nS_m^2 )</th>
<th>VRF</th>
</tr>
</thead>
<tbody>
<tr>
<td>stratif.</td>
<td>( 2^{10} )</td>
<td>17.100</td>
<td>232.8</td>
<td>4</td>
</tr>
<tr>
<td>lattice</td>
<td>( 2^{10} )</td>
<td>17.092</td>
<td>20.8</td>
<td>45</td>
</tr>
<tr>
<td>Sobol</td>
<td>( 2^{10} )</td>
<td>17.094</td>
<td>1.66</td>
<td>563</td>
</tr>
<tr>
<td>stratif.</td>
<td>( 2^{16} )</td>
<td>17.046</td>
<td>135.3</td>
<td>7</td>
</tr>
<tr>
<td>lattice</td>
<td>( 2^{16} )</td>
<td>17.096</td>
<td>4.38</td>
<td>213</td>
</tr>
<tr>
<td>Sobol</td>
<td>( 2^{16} )</td>
<td>17.096</td>
<td>0.037</td>
<td>25,330</td>
</tr>
<tr>
<td>stratif.</td>
<td>( 2^{20} )</td>
<td>17.085</td>
<td>117.6</td>
<td>8</td>
</tr>
<tr>
<td>lattice</td>
<td>( 2^{20} )</td>
<td>17.096</td>
<td>0.112</td>
<td>8,318</td>
</tr>
<tr>
<td>Sobol</td>
<td>( 2^{20} )</td>
<td>17.096</td>
<td>0.0026</td>
<td>360,000</td>
</tr>
</tbody>
</table>
$s = d = 12$. $\mu \approx 13.122$. MC variance: \textbf{516.3}.

**Lattice**: Korobov $+$ random shift.

**Sobol**: left matrix scramble $+$ random digital shift.

Variance estimated from $m = 1000$ indep. randomizations.

<table>
<thead>
<tr>
<th>method</th>
<th>$n$</th>
<th>$\tilde{X}_m$</th>
<th>$nS_m^2$</th>
<th>VRF</th>
</tr>
</thead>
<tbody>
<tr>
<td>lattice</td>
<td>$2^{10}$</td>
<td>13.114</td>
<td>39.3</td>
<td>13</td>
</tr>
<tr>
<td>Sobol</td>
<td>$2^{10}$</td>
<td>13.123</td>
<td>5.9</td>
<td>88</td>
</tr>
<tr>
<td>lattice</td>
<td>$2^{16}$</td>
<td>13.122</td>
<td>6.61</td>
<td>78</td>
</tr>
<tr>
<td>Sobol</td>
<td>$2^{16}$</td>
<td>13.122</td>
<td>1.63</td>
<td>317</td>
</tr>
<tr>
<td>lattice</td>
<td>$2^{20}$</td>
<td>13.122</td>
<td>8.59</td>
<td>60</td>
</tr>
<tr>
<td>Sobol</td>
<td>$2^{20}$</td>
<td>13.122</td>
<td>0.89</td>
<td>579</td>
</tr>
</tbody>
</table>
Variance for randomly-shifted lattice rules

Suppose $f$ has Fourier expansion

$$f(u) = \sum_{h \in \mathbb{Z}^s} \hat{f}(h)e^{2\pi \sqrt{-1} h^t u}.$$ 

For a randomly shifted lattice, the exact variance is always

$$\text{Var}[\hat{\mu}_{n, \text{rqmc}}] = \sum_{0 \neq h \in L^*_s} |\hat{f}(h)|^2,$$

where $L^*_s = \{h \in \mathbb{R}^s : h^t v \in \mathbb{Z} \text{ for all } v \in L_s\} \subseteq \mathbb{Z}^s$ is the dual lattice.

From the viewpoint of variance reduction, an optimal lattice for $f$ minimizes $\text{Var}[\hat{\mu}_{n, \text{rqmc}}]$. 
\[
\text{Var}[\hat{\mu}_{n,\text{rqmc}}] = \sum_{0 \neq h \in L_s^*} |\hat{f}(h)|^2.
\]

Let \(\alpha > 0\) be an even integer. If \(f\) has square-integrable mixed partial derivatives up to order \(\alpha/2 > 0\), and the periodic continuation of its derivatives up to order \(\alpha/2 - 1\) is continuous across the unit cube boundaries, then

\[
|\hat{f}(h)|^2 = \mathcal{O}((\max(1, h_1) \cdots \max(1, h_s))^{-\alpha}).
\]

Moreover, there is a vector \(v_1 = v_1(n)\) such that

\[
P_\alpha := \sum_{0 \neq h \in L_s^*} (\max(1, h_1) \cdots \max(1, h_s))^{-\alpha} = \mathcal{O}(n^{-\alpha+\epsilon}).
\]

This \(P_\alpha\) has been proposed long ago as a figure of merit, often with \(\alpha = 2\). It is the variance for a worst-case \(f\) having

\[
|\hat{f}(h)|^2 = (\max(1, |h_1|) \cdots \max(1, |h_s|))^{-\alpha}.
\]

A larger \(\alpha\) means a smoother \(f\) and a faster convergence rate.
For even integer $\alpha$, this worst-case $f$ is

$$f^*(u) = \sum_{u \subseteq \{1, \ldots, s\}} \prod_{j \in u} \frac{(2\pi)^{\alpha/2}}{\binom{\alpha/2}{2!}} B_{\alpha/2}(u_j).$$

where $B_{\alpha/2}$ is the Bernoulli polynomial of degree $\alpha/2$.

In particular, $B_1(u) = u - 1/2$ and $B_2(u) = u^2 - u + 1/6$.

Easy to compute $P_\alpha$ and search for good lattices in this case!

**However:** This worst-case function is not necessarily representative of what happens in applications. Also, the hidden factor in $O$ increases quickly with $s$, so this result is not very useful for large $s$.

To get a bound that is uniform in $s$, the Fourier coefficients must decrease faster with the dimension and “size” of vectors $h$; that is, $f$ must be “smoother” in high-dimensional projections. This is typically what happens in applications for which RQMC is effective!
Baker’s (or tent) transformation
To make the periodic continuation of $f$ continuous.

If $f(0) \neq f(1)$, define $\tilde{f}$ by $\tilde{f}(1 - u) = \tilde{f}(u) = f(2u)$ for $0 \leq u \leq 1/2$. This $\tilde{f}$ has the same integral as $f$ and $\tilde{f}(0) = \tilde{f}(1)$. 

\[ \begin{align*}
\text{Diagram showing the transformation with the interval from 0 to 1/2 and 1.}
\end{align*} \]
Baker’s (or tent) transformation
To make the periodic continuation of $f$ continuous.

If $f(0) \neq f(1)$, define $\tilde{f}$ by $\tilde{f}(1-u) = \tilde{f}(u) = f(2u)$ for $0 \leq u \leq 1/2$. This $\tilde{f}$ has the same integral as $f$ and $\tilde{f}(0) = \tilde{f}(1)$.
Baker’s (or tent) transformation
To make the periodic continuation of $f$ continuous.

If $f(0) \neq f(1)$, define $	ilde{f}$ by $	ilde{f}(1 - u) = \tilde{f}(u) = f(2u)$ for $0 \leq u \leq 1/2$. This $	ilde{f}$ has the same integral as $f$ and $	ilde{f}(0) = \tilde{f}(1)$.
**Baker’s (or tent) transformation**

To make the periodic continuation of $f$ **continuous**.

If $f(0) \neq f(1)$, define $\tilde{f}$ by $\tilde{f}(1 - u) = \tilde{f}(u) = f(2u)$ for $0 \leq u \leq 1/2$. This $\tilde{f}$ has the same integral as $f$ and $\tilde{f}(0) = \tilde{f}(1)$.

![Graph of Baker's transformation](image)

For smooth $f$, can reduce the variance to $O(n^{-4+\epsilon})$ (Hickernell 2002). The resulting $\tilde{f}$ is symmetric with respect to $u = 1/2$.

In practice, we transform the points $U_i$ instead of $f$. 
One-dimensional case

Random shift followed by \textit{baker’s transformation}.
Along each coordinate, stretch everything by a factor of 2 and fold.
Same as replacing $U_j$ by $\min[2U_j, 2(1 - U_j)]$. 

\begin{center}
0 \quad 0.5 \quad 1
\end{center}
One-dimensional case

Random shift followed by baker’s transformation. Along each coordinate, stretch everything by a factor of 2 and fold. Same as replacing $U_j$ by $\min[2U_j, 2(1 - U_j)]$.

\[
\begin{array}{c}
0 & 0.5 & 1 \\
\hline
\end{array}
\]

$U/n$
One-dimensional case

Random shift followed by baker’s transformation.
Along each coordinate, stretch everything by a factor of 2 and fold.
Same as replacing $U_j$ by $\min[2U_j, 2(1 - U_j)]$.

![Diagram showing intervals and points](image-url)
One-dimensional case

Random shift followed by baker’s transformation. Along each coordinate, stretch everything by a factor of 2 and fold. Same as replacing $U_j$ by $\min[2U_j, 2(1 - U_j)]$.

\begin{center}
\begin{tikzpicture}
\draw[very thick] (0,0) -- (3,0);
\draw[very thick] (0.4,0.2) -- (0.4,0.4);
\draw[very thick] (1.6,0.2) -- (1.6,0.4);
\draw[very thick] (2.8,0.2) -- (2.8,0.4);
\draw[very thick] (0,0.1) -- (0.4,0.1);
\draw[very thick] (1.6,0.1) -- (2.8,0.1);
\draw[very thick] (0,0.3) -- (0.4,0.3);
\draw[very thick] (1.6,0.3) -- (2.8,0.3);
\draw[very thick] (0,0.2) -- (0.4,0.2);
\draw[very thick] (1.6,0.2) -- (2.8,0.2);
\node at (0,0) {0};
\node at (0.5,0) {0.5};
\node at (1.5,0) {0.5};
\node at (2,0) {1};
\end{tikzpicture}
\end{center}

Gives locally antithetic points in intervals of size $2/n$. This implies that linear pieces over these intervals are integrated exactly. Intuition: when $f$ is smooth, it is well-approximated by a piecewise linear function, which is integrated exactly, so the error is small.
ANOVA decomposition

The Fourier expansion has too many terms to handle. As a cruder expansion, we can write $f(u) = f(u_1, \ldots, u_s)$ as:

$$f(u) = \sum_{u \subseteq \{1, \ldots, s\}} f_u(u) = \mu + \sum_{i=1}^{s} f_{\{i\}}(u_i) + \sum_{i,j=1}^{s} f_{\{i,j\}}(u_i, u_j) + \cdots$$

where

$$f_u(u) = \int_{[0,1]^{|\bar{u}|}} f(u) \, du_{\bar{u}} - \sum_{v \subset u} f_v(u_v),$$

and the Monte Carlo variance decomposes as

$$\sigma^2 = \sum_{u \subseteq \{1, \ldots, s\}} \sigma^2_u, \quad \text{where } \sigma^2_u = \text{Var}[f_u(U)].$$

The $\sigma^2_u$’s can be estimated by MC or RQMC.

Heuristic intuition: Make sure the projections $P_n(u)$ are very uniform for the important subsets $u$ (i.e., with larger $\sigma^2_u$).
Weighted $P_{\gamma, \alpha}$ with projection-dependent weights $\gamma_u$

Denote $u(h) = u(h_1, \ldots, h_s)$ the set of indices $j$ for which $h_j \neq 0$.

$$P_{\gamma, \alpha} = \sum_{0 \neq h \in L_s^*} \gamma_u(h) (\max(1, |h_1|) \cdots \max(1, |h_s|))^{-\alpha}.$$  

For $\alpha/2$ integer $> 0$, with $u_i = (u_{i,1}, \ldots, u_{i,s}) = iv_1 \mod 1$,

$$P_{\gamma, \alpha} = \sum_{\emptyset \neq u \subseteq \{1, \ldots, s\}} \frac{1}{n} \sum_{i=0}^{n-1} \gamma_u \left[ \frac{-( -4\pi^2)^{\alpha/2}}{(\alpha)!} \right] |u| \prod_{j \in u} B_\alpha(u_{i,j}),$$

and the corresponding variation is

$$V^2_\gamma(f) = \sum_{\emptyset \neq u \subseteq \{1, \ldots, s\}} \frac{1}{\gamma_u (4\pi^2)^{\alpha |u|/2}} \int_{[0,1]^{|u|}} \left| \frac{\partial^{\alpha |u|/2}}{\partial u^{\alpha/2}} f_u(u) \right|^2 \, du,$$

for $f : [0,1)^s \to \mathbb{R}$ smooth enough. Then,

$$\text{Var}[\hat{\mu}_{n,rqmc}] = \sum_{u \subseteq \{1, \ldots, s\}} \text{Var}[\hat{\mu}_{n,rqmc}(f_u)] \leq V^2_\gamma(f) P_{\gamma, \alpha}.$$
$\mathcal{P}_{\gamma,\alpha}$ with $\alpha = 2$ and properly chosen weights $\gamma$ is a good practical choice of figure of merit.

Simple choices of weights: order-dependent or product.

**Lattice Builder:** Software to search for good lattices with arbitrary $n$, $s$, weights, etc. See my web page.
ANOVA Variances for estimator of $\mathbb{P}[T > x]$ in Stochastic Activity Network

Stochastic Activity Network

- $x = 64$
- $x = 100$
- CMC, $x = 64$
- CMC, $x = 100$

% of total variance for each cardinality of $u$
Variance for estimator of $\mathbb{P}[T > x]$ for SAN

Variance decreases roughly as $O(n^{-1.2})$. For $\mathbb{E}[T]$, we observe $O(n^{-1.4})$. 
Variance for estimator of $\mathbb{P}[T > x]$ with CMC

Stochastic Activity Network (CMC $x = 64$)

![Graph showing the variance for different estimators: MC, Sobol, Lattice ($\mathcal{P}_2$) + baker, $n^{-2}$, with log-log scale for variance vs. $n$. The graph illustrates the decrease in variance as $n$ increases.]
Histograms

**Single MC draw ($x = 100$)**

**MC estimator ($x = 100$)**

**RQMC estimator ($x = 100$)**
Histograms

Single MC draw (CMC $x = 100$)

MC estimator (CMC $x = 100$)

RQMC estimator (CMC $x = 100$)
Effective dimension

(Caflisch, Morokoff, and Owen 1997).

A function $f$ has effective dimension $d$ in proportion $\rho$ in the superposition sense if

$$\sum_{|u| \leq d} \sigma^2_u \geq \rho \sigma^2.$$

It has effective dimension $d$ in the truncation sense if

$$\sum_{u \subseteq \{1, \ldots, d\}} \sigma^2_u \geq \rho \sigma^2.$$

High-dimensional functions with low effective dimension are frequent. One may change $f$ to make this happen.
Example: Function of a Multinormal vector

Let $\mu = E[f(U)] = E[g(Y)]$ where $Y = (Y_1, \ldots, Y_s) \sim \mathcal{N}(0, \Sigma)$. 
Example: Function of a Multinormal vector

Let $\mu = E[f(U)] = E[g(Y)]$ where $Y = (Y_1, \ldots, Y_s) \sim \mathcal{N}(0, \Sigma)$. 

For example, if the payoff of a financial derivative is a function of the values taken by a $c$-dimensional geometric Brownian motion (GMB) at $d$ observations times $0 < t_1 < \cdots < t_d = T$, then we have $s = cd$. 
Example: Function of a Multinormal vector

Let $\mu = E[f(U)] = E[g(Y)]$ where $Y = (Y_1, \ldots, Y_s) \sim N(0, \Sigma)$.

For example, if the payoff of a financial derivative is a function of the values taken by a $c$-dimensional geometric Brownian motion (GMB) at $d$ observations times $0 < t_1 < \cdots < t_d = T$, then we have $s = cd$.

To generate $Y$: Decompose $\Sigma = \mathbf{A}\mathbf{A}^t$, generate $Z = (Z_1, \ldots, Z_s) \sim N(0, I)$ where the (independent) $Z_j$’s are generated by inversion: $Z_j = \Phi^{-1}(U_j)$, and return $Y = AZ$. 
Example: Function of a Multinormal vector

Let $\mu = E[f(U)] = E[g(Y)]$ where $Y = (Y_1, \ldots, Y_s) \sim N(0, \Sigma)$.

For example, if the payoff of a financial derivative is a function of the values taken by a $c$-dimensional geometric Brownian motion (GMB) at $d$ observations times $0 < t_1 < \cdots < t_d = T$, then we have $s = cd$.

To generate $Y$: Decompose $\Sigma = AA^t$, generate $Z = (Z_1, \ldots, Z_s) \sim N(0, I)$ where the (independent) $Z_j$'s are generated by inversion: $Z_j = \Phi^{-1}(U_j)$, and return $Y = AZ$.

Choice of $A$?
Example: Function of a Multinormal vector

Let \( \mu = E[f(U)] = E[g(Y)] \) where \( Y = (Y_1, \ldots, Y_s) \sim N(0, \Sigma) \).

For example, if the payoff of a financial derivative is a function of the values taken by a \( c \)-dimensional geometric Brownian motion (GMB) at \( d \) observations times \( 0 < t_1 < \cdots < t_d = T \), then we have \( s = cd \).

To generate \( Y \): Decompose \( \Sigma = AA^t \), generate \( Z = (Z_1, \ldots, Z_s) \sim N(0, I) \) where the (independent) \( Z_j \)'s are generated by inversion: \( Z_j = \Phi^{-1}(U_j) \), and return \( Y = AZ \).

Choice of \( A \)?

Cholesky factorization: \( A \) is lower triangular.
Principal component decomposition (PCA) (Ackworth et al. 1998):

\[ \mathbf{A} = \mathbf{PD}^{1/2} \]

where \( \mathbf{D} = \text{diag}(\lambda_s, \ldots, \lambda_1) \) (eigenvalues of \( \mathbf{\Sigma} \) in decreasing order) and the columns of \( \mathbf{P} \) are the corresponding unit-length eigenvectors.
Principal component decomposition (PCA) (Ackworth et al. 1998): \( \mathbf{A} = \mathbf{PD}^{1/2} \) where \( \mathbf{D} = \text{diag}(\lambda_s, \ldots, \lambda_1) \) (eigenvalues of \( \mathbf{\Sigma} \) in decreasing order) and the columns of \( \mathbf{P} \) are the corresponding unit-length eigenvectors. With this \( \mathbf{A} \), \( Z_1 \) accounts for the max amount of variance of \( \mathbf{Y} \), then \( Z_2 \) the max amount of variance cond. on \( Z_1 \), etc.
Principal component decomposition (PCA) (Ackworth et al. 1998): $A = PD^{1/2}$ where $D = \text{diag}(\lambda_s, \ldots, \lambda_1)$ (eigenvalues of $\Sigma$ in decreasing order) and the columns of $P$ are the corresponding unit-length eigenvectors. With this $A$, $Z_1$ accounts for the max amount of variance of $Y$, then $Z_2$ the max amount of variance cond. on $Z_1$, etc.

**Function of a Brownian motion** (or other Lévy process): Payoff depends on $c$-dimensional Brownian motion $\{X(t), t \geq 0\}$ observed at times $0 = t_0 < t_1 < \cdots < t_d = T$. 
Principal component decomposition (PCA) (Ackworth et al. 1998): $A = PD^{1/2}$ where $D = \text{diag}(\lambda_s, \ldots, \lambda_1)$ (eigenvalues of $\Sigma$ in decreasing order) and the columns of $P$ are the corresponding unit-length eigenvectors. With this $A$, $Z_1$ accounts for the max amount of variance of $Y$, then $Z_2$ the max amount of variance cond. on $Z_1$, etc.

Function of a Brownian motion (or other Lévy process):
Payoff depends on $c$-dimensional Brownian motion $\{X(t), t \geq 0\}$ observed at times $0 = t_0 < t_1 < \cdots < t_d = T$.

Sequential (or random walk) method: generate $X(t_1)$, then $X(t_2) - X(t_1)$, then $X(t_3) - X(t_2)$, etc.
Principal component decomposition (PCA) (Ackworth et al. 1998): $A = P D^{1/2}$ where $D = \text{diag}(\lambda_s, \ldots, \lambda_1)$ (eigenvalues of $\Sigma$ in decreasing order) and the columns of $P$ are the corresponding unit-length eigenvectors. With this $A$, $Z_1$ accounts for the max amount of variance of $Y$, then $Z_2$ the max amount of variance cond. on $Z_1$, etc.

**Function of a Brownian motion** (or other Lévy process): Payoff depends on $c$-dimensional Brownian motion $\{X(t), t \geq 0\}$ observed at times $0 = t_0 < t_1 < \cdots < t_d = T$.

**Sequential (or random walk) method:** generate $X(t_1)$, then $X(t_2) - X(t_1)$, then $X(t_3) - X(t_2)$, etc.

**Bridge sampling** (Moskowitz and Caflisch 1996). Suppose $d = 2^m$. generate $X(t_d)$, then $X(t_{d/2})$ conditional on $(X(0), X(t_d))$,
Principal component decomposition (PCA) (Ackworth et al. 1998): \[ \mathbf{A} = \mathbf{PD}^{1/2} \] where \( \mathbf{D} = \text{diag}(\lambda_s, \ldots, \lambda_1) \) (eigenvalues of \( \mathbf{\Sigma} \) in decreasing order) and the columns of \( \mathbf{P} \) are the corresponding unit-length eigenvectors. With this \( \mathbf{A} \), \( Z_1 \) accounts for the max amount of variance of \( \mathbf{Y} \), then \( Z_2 \) the max amount of variance cond. on \( Z_1 \), etc.

**Function of a Brownian motion** (or other Lévy process):
Payoff depends on \( c \)-dimensional Brownian motion \( \{ \mathbf{X}(t), t \geq 0 \} \) observed at times \( 0 = t_0 < t_1 < \cdots < t_d = T \).

Sequential (or random walk) method: generate \( \mathbf{X}(t_1) \), then \( \mathbf{X}(t_2) - \mathbf{X}(t_1) \), then \( \mathbf{X}(t_3) - \mathbf{X}(t_2) \), etc.

Bridge sampling (Moskowitz and Caflisch 1996). Suppose \( d = 2^m \).
generate \( \mathbf{X}(t_d) \), then \( \mathbf{X}(t_{d/2}) \) conditional on \( (\mathbf{X}(0), \mathbf{X}(t_d)) \), then \( \mathbf{X}(t_{d/4}) \) conditional on \( (\mathbf{X}(0), \mathbf{X}(t_{d/2})) \), and so on.
The first few \( N(0, 1) \) r.v.’s already sketch the path trajectory.
Principal component decomposition (PCA) (Ackworth et al. 1998): \( \mathbf{A} = \mathbf{PD}^{1/2} \) where \( \mathbf{D} = \text{diag}(\lambda_s, \ldots, \lambda_1) \) (eigenvalues of \( \mathbf{\Sigma} \) in decreasing order) and the columns of \( \mathbf{P} \) are the corresponding unit-length eigenvectors. With this \( \mathbf{A} \), \( Z_1 \) accounts for the max amount of variance of \( \mathbf{Y} \), then \( Z_2 \) the max amount of variance cond. on \( Z_1 \), etc.

**Function of a Brownian motion** (or other Lévy process):
Payoff depends on \( c \)-dimensional Brownian motion \( \{X(t), \ t \geq 0\} \) observed at times \( 0 = t_0 < t_1 < \cdots < t_d = T \).

Sequential (or random walk) method: generate \( X(t_1) \), then \( X(t_2) - X(t_1) \), then \( X(t_3) - X(t_2) \), etc.

Bridge sampling (Moskowitz and Caflisch 1996). Suppose \( d = 2^m \).
generate \( X(t_d) \), then \( X(t_d/2) \) conditional on \( (X(0), X(t_d)) \), then \( X(t_d/4) \) conditional on \( (X(0), X(t_d/2)) \), and so on.

The first few \( N(0, 1) \) r.v.’s already sketch the path trajectory.
Each of these methods corresponds to some matrix \( \mathbf{A} \).
Choice has a large impact on the ANOVA decomposition of \( f \).
Example: Pricing an Asian basket option

We have $c$ assets, $d$ observation times. Want to estimate $\mathbb{E}[f(U)]$, where

$$f(U) = e^{-rT} \max \left[ 0, \frac{1}{cd} \sum_{i=1}^{c} \sum_{j=1}^{d} S_i(t_j) - K \right]$$

is the net discounted payoff and $S_i(t_j)$ is the price of asset $i$ at time $t_j$. 

Suppose $(S_1(t), \ldots, S_c(t))$ obeys a geometric Brownian motion. Then, $f(U) = g(Y)$ where $Y = (Y_1, \ldots, Y_s) \sim N(0, \Sigma)$ and $s = cd$.

Even with Cholesky decompositions of $\Sigma$, the two-dimensional projections often account for more than 99% of the variance: low effective dimension in the superposition sense. With PCA or bridge sampling, we get low effective dimension in the truncation sense. In realistic examples, the first two coordinates $Z_1$ and $Z_2$ often account for more than 99.99% of the variance!
**Example: Pricing an Asian basket option**

We have $c$ assets, $d$ observation times. Want to estimate $\mathbb{E}[f(U)]$, where

$$f(U) = e^{-rT} \max \left[ 0, \frac{1}{cd} \sum_{i=1}^{c} \sum_{j=1}^{d} S_i(t_j) - K \right]$$

is the net discounted payoff and $S_i(t_j)$ is the price of asset $i$ at time $t_j$.

Suppose $(S_1(t), \ldots, S_c(t))$ obeys a geometric Brownian motion. Then, $f(U) = g(Y)$ where $Y = (Y_1, \ldots, Y_s) \sim N(0, \Sigma)$ and $s = cd$. 
Example: Pricing an Asian basket option

We have $c$ assets, $d$ observation times. Want to estimate $\mathbb{E}[f(U)]$, where

$$f(U) = e^{-rT} \max \left[ 0, \frac{1}{cd} \sum_{i=1}^{c} \sum_{j=1}^{d} S_i(t_j) - K \right]$$

is the net discounted payoff and $S_i(t_j)$ is the price of asset $i$ at time $t_j$.

Suppose $(S_1(t), \ldots, S_c(t))$ obeys a geometric Brownian motion. Then, $f(U) = g(Y)$ where $Y = (Y_1, \ldots, Y_s) \sim N(0, \Sigma)$ and $s = cd$.

Even with Cholesky decompositions of $\Sigma$, the two-dimensional projections often account for more than 99% of the variance: low effective dimension in the superposition sense.

With PCA or bridge sampling, we get low effective dimension in the truncation sense. In realistic examples, the first two coordinates $Z_1$ and $Z_2$ often account for more than 99.99% of the variance!
Numerical experiment with $c = 10$ and $d = 25$

This gives a 250-dimensional integration problem.

Let $\rho_{i,j} = 0.4$ for all $i \neq j$, $T = 1$, $\sigma_i = 0.1 + 0.4(i - 1)/9$ for all $i$, $r = 0.04$, $S(0) = 100$, and $K = 100$. (Imai and Tan 2002).
Numerical experiment with $c = 10$ and $d = 25$

This gives a 250-dimensional integration problem.

Let $\rho_{i,j} = 0.4$ for all $i \neq j$, $T = 1$, $\sigma_i = 0.1 + 0.4(i - 1)/9$ for all $i$, $r = 0.04$, $S(0) = 100$, and $K = 100$. (Imai and Tan 2002).

Variance reduction factors for Cholesky (left) and PCA (right) (experiment from 2003):

<table>
<thead>
<tr>
<th>Korobov Lattice Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 16381$</td>
</tr>
<tr>
<td>$a = 5693$</td>
</tr>
<tr>
<td>$n = 65521$</td>
</tr>
<tr>
<td>$a = 944$</td>
</tr>
<tr>
<td>$n = 262139$</td>
</tr>
<tr>
<td>$a = 21876$</td>
</tr>
<tr>
<td>Lattice+shift</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>878</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>1504</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>2643</td>
</tr>
<tr>
<td>Lattice+shift+baker</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>4553</td>
</tr>
<tr>
<td>46</td>
</tr>
<tr>
<td>3657</td>
</tr>
<tr>
<td>43</td>
</tr>
<tr>
<td>7553</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sobol’ Nets</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 2^{14}$</td>
</tr>
<tr>
<td>$n = 2^{16}$</td>
</tr>
<tr>
<td>$n = 2^{18}$</td>
</tr>
<tr>
<td>Sobol+Shift</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>1299</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>3184</td>
</tr>
<tr>
<td>32</td>
</tr>
<tr>
<td>6046</td>
</tr>
<tr>
<td>Sobol+LMS+Shift</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>4232</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>9219</td>
</tr>
<tr>
<td>35</td>
</tr>
<tr>
<td>16557</td>
</tr>
</tbody>
</table>

Note: The payoff function is not smooth and also unbounded!
ANOVA Variances for ordinary Asian Option

Asian Option with $S(0) = 100$, $K = 100$, $r = 0.05$, $\sigma = 0.5$

- $s = 3$, seq.
- $s = 3$, BB
- $s = 3$, PCA
- $s = 6$, seq.
- $s = 6$, BB
- $s = 6$, PCA
- $s = 12$, seq.
- $s = 12$, BB
- $s = 12$, PCA

% of total variance for each cardinality of $u$
Asian Option ($s = 6$) with $S(0) = 100$, $K = 100$, $r = 0.05$, $\sigma = 0.5$
Asian Option (PCA) $s = 12$, $S(0) = 100$, $K = 100$, $r = 0.05$, $\sigma = 0.5$
Asian Option on a Single Asset, with control variate

Let $c = 1$, $S(0) = 100$, $r = \ln(1.09)$, $\sigma_i = 0.2$, $T = 120/365$, 
$t_j = D_1/365 + (T - D_1/365)(j - 1)/(d - 1)$ for $j = 1, \ldots, d$,
Asian Option on a Single Asset, with control variate

Let $c = 1$, $S(0) = 100$, $r = \ln(1.09)$, $\sigma_i = 0.2$, $T = 120/365$, $t_j = D_1/365 + (T - D_1/365)(j - 1)/(d - 1)$ for $j = 1, \ldots, d$,

We estimated the optimal CV coefficient by pilot runs for MC and for each combination of sampling scheme, RQMC method, and $n$. 
Asian Option on a Single Asset, with control variate

Let $c = 1$, $S(0) = 100$, $r = \ln(1.09)$, $\sigma_j = 0.2$, $T = 120/365$, $t_j = D_1/365 + (T - D_1/365)(j - 1)/(d - 1)$ for $j = 1, \ldots, d$,

We estimated the optimal CV coefficient by pilot runs for MC and for each combination of sampling scheme, RQMC method, and $n$.

<table>
<thead>
<tr>
<th>$d$</th>
<th>$D_1$</th>
<th>$K$</th>
<th>$\mu$</th>
<th>$\sigma^2$</th>
<th>VRF of CV</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>111</td>
<td>90</td>
<td>13.008</td>
<td>105</td>
<td>$1.53 \times 10^6$</td>
</tr>
<tr>
<td>10</td>
<td>111</td>
<td>100</td>
<td>5.863</td>
<td>61</td>
<td>$1.07 \times 10^6$</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>90</td>
<td>11.367</td>
<td>46</td>
<td>5400</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>100</td>
<td>3.617</td>
<td>23</td>
<td>3950</td>
</tr>
<tr>
<td>120</td>
<td>1</td>
<td>90</td>
<td>11.207</td>
<td>41</td>
<td>5050</td>
</tr>
<tr>
<td>120</td>
<td>1</td>
<td>100</td>
<td>3.367</td>
<td>20</td>
<td>4100</td>
</tr>
</tbody>
</table>
VRFs (per run) for RQMC vs MC, with \( n \approx 2^{16} \).

*Sequential* sampling (left), *bridge* sampling (middle), and *PCA* (right).

<table>
<thead>
<tr>
<th>( d )</th>
<th>( D_1 )</th>
<th>( K )</th>
<th>( P_n )</th>
<th>without CV</th>
<th>with CV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>( \text{SEQ} )</td>
<td>( \text{BBS} )</td>
<td>( \text{PCA} )</td>
</tr>
<tr>
<td>10</td>
<td>111</td>
<td>90</td>
<td>Kor+S</td>
<td>5943</td>
<td>6014</td>
</tr>
<tr>
<td>10</td>
<td>111</td>
<td>90</td>
<td>Kor+S+B</td>
<td>88927</td>
<td>256355</td>
</tr>
<tr>
<td>10</td>
<td>111</td>
<td>90</td>
<td>Sob+DS</td>
<td>9572</td>
<td>12549</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>90</td>
<td>Kor+S</td>
<td>442</td>
<td>1720</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>90</td>
<td>Kor+S+B</td>
<td>1394</td>
<td>26883</td>
</tr>
<tr>
<td>10</td>
<td>12</td>
<td>90</td>
<td>Sob+DS</td>
<td>2205</td>
<td>9053</td>
</tr>
<tr>
<td>120</td>
<td>1</td>
<td>90</td>
<td>Kor+S</td>
<td>192</td>
<td>2025</td>
</tr>
<tr>
<td>120</td>
<td>1</td>
<td>90</td>
<td>Kor+S+B</td>
<td>394</td>
<td>15575</td>
</tr>
<tr>
<td>120</td>
<td>1</td>
<td>90</td>
<td>Sob+DS</td>
<td>325</td>
<td>7079</td>
</tr>
</tbody>
</table>

For \( d = 10 \), Sobol’ with PCA combined with CV reduces the variance approximately by a factor of \( 6.8 \times 10^9 \), without increasing the CPU time.

For \( d = 120 \), PCA is slower than SEQ by a factor of 2 or 3, but worth it.
Array-RQMC for Markov Chains

Setting: A Markov chain with state space $\mathcal{X} \subseteq \mathbb{R}^\ell$, evolves as

$$X_0 = x_0, \quad X_j = \varphi_j(X_{j-1}, U_j), \quad j \geq 1,$$

where the $U_j$ are i.i.d. uniform r.v.’s over $(0, 1)^d$. Want to estimate

$$\mu = \mathbb{E}[Y] \quad \text{where} \quad Y = \sum_{j=1}^{\tau} g_j(X_j).$$

Ordinary MC: $n$ i.i.d. realizations of $Y$. Requires $\tau s$ uniforms.

Simulate an “array” (or population) of $n$ chains in “parallel.”

Goal: Want small discrepancy between empirical distribution of states $S_{n,j} = \{X_{0,j}, \ldots, X_{n-1,j}\}$ and theoretical distribution of $X_j$, at each step $j$. At each step, use RQMC point set to advance all the chains by one step.
**Some RQMC insight:** To simplify, suppose $X_j \sim U(0, 1)^\ell$. We estimate

$$
\mu_j = \mathbb{E}[g_j(X_j)] = \mathbb{E}[g_j(\varphi_j(X_{j-1}, U))] = \int_{[0,1]^\ell+d} g_j(\varphi_j(x,u)) dx \, du
$$

by

$$
\hat{\mu}_{\text{arqmc}, j, n} = \frac{1}{n} \sum_{i=0}^{n-1} g_j(X_{i,j}) = \frac{1}{n} \sum_{i=0}^{n-1} g_j(\varphi_j(X_{i,j-1}, U_{i,j})).
$$

This is (roughly) RQMC with the point set $Q_n = \{(X_{i,j-1}, U_{i,j}), \ 0 \leq i < n\}$. We want $Q_n$ to have low discrepancy (LD) over $[0, 1)^{\ell+d}$. 

**Some RQMC insight:** To simplify, suppose $X_j \sim U(0, 1)^{\ell}$.

We estimate

$$
\mu_j = \mathbb{E}[g_j(X_j)] = \mathbb{E}[g_j(\varphi_j(X_{j-1}, U))] = \int_{[0,1]^{\ell+d}} g_j(\varphi_j(x,u))dxdu
$$

by

$$
\hat{\mu}_{\text{arqmc},j,n} = \frac{1}{n} \sum_{i=0}^{n-1} g_j(X_{i,j}) = \frac{1}{n} \sum_{i=0}^{n-1} g_j(\varphi_j(X_{i,j-1}, U_{i,j})).
$$

This is (roughly) RQMC with the point set $Q_n = \{(X_{i,j-1}, U_{i,j}), 0 \leq i < n\}$.

We want $Q_n$ to have low discrepancy (LD) over $[0, 1)^{\ell+d}$.

We do not choose the $X_{i,j-1}$'s in $Q_n$: they come from the simulation.

We select a LD point set

$$
\tilde{Q}_n = \{(w_0, U_{0,j}), \ldots, (w_{n-1}, U_{n-1,j})\},
$$

where the $w_i \in [0, 1)^{\ell}$ are fixed and each $U_{i,j} \sim U(0, 1)^d$.

Permute the states $X_{i,j-1}$ so that $X_{\pi_j(i),j-1}$ is “close” to $w_i$ for each $i$ (LD between the two sets), and compute $X_{i,j} = \varphi_j(X_{\pi_j(i),j-1}, U_{i,j})$ for each $i$.

Example: If $\ell = 1$, can take $w_i = (i + 0.5)/n$ and just sort the states. For $\ell > 1$, there are various ways to define the matching (multivariate sort).
Array-RQMC algorithm

\( X_{i,0} \leftarrow x_0 \) (or \( X_{i,0} \leftarrow x_{i,0} \)) for \( i = 0, \ldots, n - 1 \);

for \( j = 1, 2, \ldots, \tau \) do

Compute the permutation \( \pi_j \) of the states (for matching);
Randomize afresh \( \{U_{0,j}, \ldots, U_{n-1,j}\} \) in \( \tilde{Q}_n \);
\( X_{i,j} = \varphi_j(X_{\pi_j(i),j-1}, U_{i,j}) \), for \( i = 0, \ldots, n - 1 \);
\( \hat{\mu}_{\text{arqmc},j,n} = \bar{Y}_{n,j} = \frac{1}{n} \sum_{i=0}^{n-1} g(X_{i,j}); \)
Estimate \( \mu \) by the average \( \bar{Y}_n = \hat{\mu}_{\text{arqmc},n} = \sum_{j=1}^{\tau} \hat{\mu}_{\text{arqmc},j,n}. \)
Array-RQMC algorithm

\[ X_{i,0} \leftarrow x_0 \] (or \( X_{i,0} \leftarrow x_{i,0} \)) for \( i = 0, \ldots, n - 1 \);

for \( j = 1, 2, \ldots, \tau \) do

- Compute the permutation \( \pi_j \) of the states (for matching);
- Randomize afresh \( \{U_{0,j}, \ldots, U_{n-1,j}\} \) in \( \tilde{Q}_n \);

\[ X_{i,j} = \varphi_j(X_{\pi_j(i),j-1}, U_{i,j}), \text{ for } i = 0, \ldots, n - 1; \]

\[ \hat{\mu}_{arqmc,j,n} = \bar{Y}_{n,j} = \frac{1}{n} \sum_{i=0}^{n-1} g(X_{i,j}); \]

Estimate \( \mu \) by the average \( \bar{Y}_n = \hat{\mu}_{arqmc,n} = \sum_{j=1}^{\tau} \hat{\mu}_{arqmc,j,n} \).

**Proposition:** (i) The average \( \bar{Y}_n \) is an **unbiased** estimator of \( \mu \).
(ii) The **empirical variance** of \( m \) independent realizations gives an unbiased estimator of \( \text{Var}[\bar{Y}_n] \).
Some generalizations

L., Lécot, and Tuffin [2008]: $\tau$ can be a random stopping time w.r.t. the filtration $\mathcal{F}\{(j, X_j), j \geq 0\}$.


L. and Sanvido [2010]: Combination with coupling from the past for exact sampling.

Dion and L. [2010]: Combination with approximate dynamic programming and for optimal stopping problems.

Gerber and Chopin [2015]: Sequential QMC.
Convergence results and applications

L., Lécot, and Tuffin [2006, 2008]: Special cases: convergence at MC rate, one-dimensional, stratification, etc. $\mathcal{O}(n^{-3/2})$ variance.

Lécot and Tuffin [2004]: Deterministic, one-dimension, discrete state.

El Haddad, Lécot, L. [2008, 2010]: Deterministic, multidimensional. $\mathcal{O}(n^{-1/(\ell+1)})$ worst-case error under some conditions.

Fakhererredine, El Haddad, Lécot [2012, 2013, 2014]: LHS, stratification, Sudoku sampling, ...

L., Lécot, Munger, and Tuffin [2016]: Survey, comparing sorts, and further examples, some with $\mathcal{O}(n^{-3})$ empirical variance.

Wächter and Keller [2008]: Applications in computer graphics.

Gerber and Chopin [2015]: Sequential QMC (particle filters), Owen nested scrambling and Hilbert sort. $o(n^{-1})$ variance.
A (4,4) mapping

States of the chains

Sobol’ net in 2 dimensions after random digital shift
A (4,4) mapping

States of the chains

Sobol’ net in 2 dimensions after random digital shift
A (4,4) mapping

States of the chains

Sobol’ net in 2 dimensions after random digital shift
A (4,4) mapping

States of the chains

Sobol’ net in 2 dimensions after random digital shift
Hilbert curve sort
Map the state to $[0, 1]$, then sort.

States of the chains
Hilbert curve sort

Map the state to $[0, 1]$, then sort.
Hilbert curve sort
Map the state to $[0, 1]$, then sort.

States of the chains
Hilbert curve sort
Map the state to [0, 1], then sort.

States of the chains

![Graph showing Hilbert curve sort with states mapped to [0, 1] and then sorted.](image-url)
Example: Asian Call Option

\[ S(0) = 100, \ K = 100, \ r = 0.05, \ \sigma = 0.15, \ t_j = j/52, \ j = 0, \ldots, \tau = 13. \]

RQMC: Sobol’ points with linear scrambling + random digital shift. Similar results for randomly-shifted lattice + baker’s transform.

\[ \log_2 \text{Var}[\hat{\mu}_{\text{RQMC},n}] \]

\[ \text{log}_2 n \]

-10
-20
-30
-40
8 10 12 14 16 18 20

- \hat{\mu}_{\text{crude MC}}
- RQMC sequential
- array-RQMC, split sort
- \text{n}^{-1}
- \text{n}^{-2}
Example: Asian Call Option

<table>
<thead>
<tr>
<th>Sort</th>
<th>RQMC points</th>
<th>$\log_2 \frac{\text{Var}[\bar{Y}_{n,j}]}{\log_2 n}$</th>
<th>VRF</th>
<th>CPU (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch sort (n₁ = n₂)</td>
<td>SS</td>
<td>-1.38</td>
<td>2.0 × 10²</td>
<td>744</td>
</tr>
<tr>
<td></td>
<td>Sobol</td>
<td>-2.03</td>
<td>4.2 × 10⁶</td>
<td>532</td>
</tr>
<tr>
<td></td>
<td>Sobol+NUS</td>
<td>-2.03</td>
<td>2.8 × 10⁶</td>
<td>1035</td>
</tr>
<tr>
<td></td>
<td>Korobov+baker</td>
<td>-2.04</td>
<td>4.4 × 10⁶</td>
<td>482</td>
</tr>
<tr>
<td>Hilbert sort (logistic map)</td>
<td>SS</td>
<td>-1.55</td>
<td>2.4 × 10³</td>
<td>840</td>
</tr>
<tr>
<td></td>
<td>Sobol</td>
<td>-2.03</td>
<td>2.6 × 10⁶</td>
<td>534</td>
</tr>
<tr>
<td></td>
<td>Sobol+NUS</td>
<td>-2.02</td>
<td>2.8 × 10⁶</td>
<td>724</td>
</tr>
<tr>
<td></td>
<td>Korobov+baker</td>
<td>-2.01</td>
<td>3.3 × 10⁶</td>
<td>567</td>
</tr>
</tbody>
</table>

VRF for $n = 2^{20}$. CPU time for $m = 100$ replications.
Conclusion, discussion, etc.

- RQMC can improve the accuracy of estimators considerably in some applications.
- Cleverly modifying the function $f$ can often bring huge statistical efficiency improvements in simulations with RQMC.
- There are often many possibilities for how to change $f$ to make it smoother, periodic, and reduce its effective dimension.
- Point set constructions should be based on discrepancies that take that into account. Can take a weighted average (or worst-case) of uniformity measures over a selected set of projections.
- Nonlinear functions of expectations: RQMC also reduces the bias.
- RQMC for density estimation.
- RQMC for optimization.
- Array-RQMC for Markov chains. Sequential RQMC. Other QMC methods for Markov chains.
- Still a lot to learn and do ...
Some basic references on QMC and RQMC:


Some references on Array-RQMC:


