Introduction to randomized quasi-Monte Carlo methods in simulation

Part A

Pierre L'Ecuyer

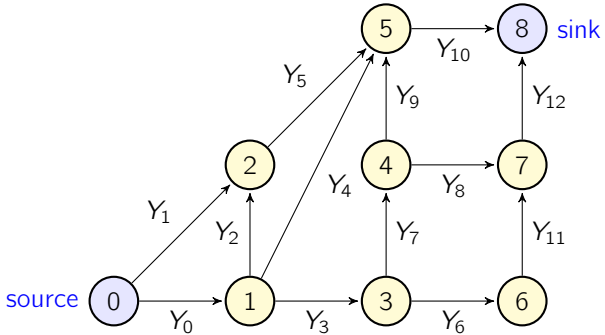
Université de Montréal, Canada

ETICS, Saissac, September 2024

The Monte Carlo Method

Small example: A stochastic activity network

Link k has random length Y_k, with cdf $F_k(y) := \mathbb{P}[Y_k \le y]$.
Goal: estimate $\mathbb{P}[T > x]$ where T = (random) length of longest path from source to sink.



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

Monte Carlo (simulation)

Algorithm: Monte Carlo to estimate $\mathbb{E}[T]$

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

Can also compute confidence interval on $\mathbb{E}[T]$, a histogram for the distribution of T, etc.

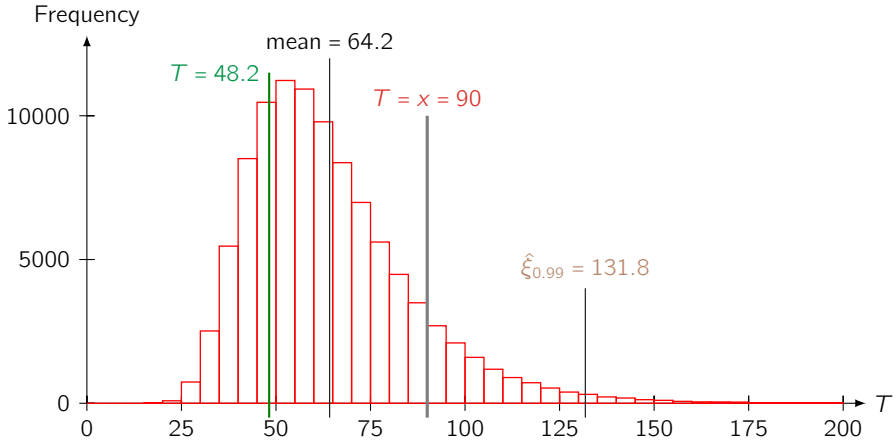
We may pay a penalty if $T > 90$, for example, and want to estimate $\mathbb{P}[T > 90]$.

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 estimates of $\mathbb{E}[T]$ or $\mathbb{P}[T > x]$.

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



Monte Carlo to estimate an expectation

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

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

Monte Carlo (MC) estimator:

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

where $X_i = f(\boldsymbol{U}_i)$ and $\boldsymbol{U}_0, \ldots, \boldsymbol{U}_{n-1}$ i.i.d. uniform over $[0,1)^s$.

We have $\mathbb{E}[\bar{X}_n] = \mu$ and $\mathrm{Var}[\bar{X}_n] = \sigma^2/n = \mathrm{Var}[X]/n$.

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):

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

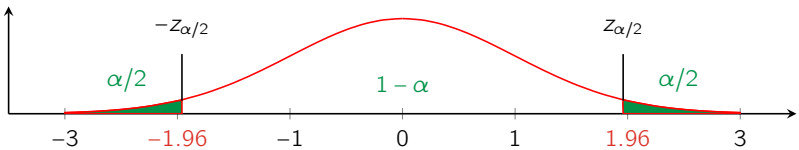
where

$$S_n^2 = \frac{1}{n-1}\sum_{i=0}^{n-1}(X_i - \bar{X}_n)^2.$$

Confidence interval at level α (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 $\alpha = 0.05$.



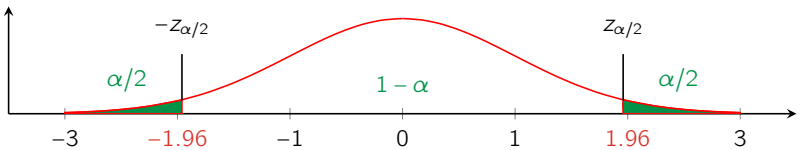
The width of the confidence interval is asymptotically proportional to $\sigma/\sqrt{n}$, so it converges as $O(n^{-1/2})$. Relative error: $\sigma/(\mu\sqrt{n})$.

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

Confidence interval at level α (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.

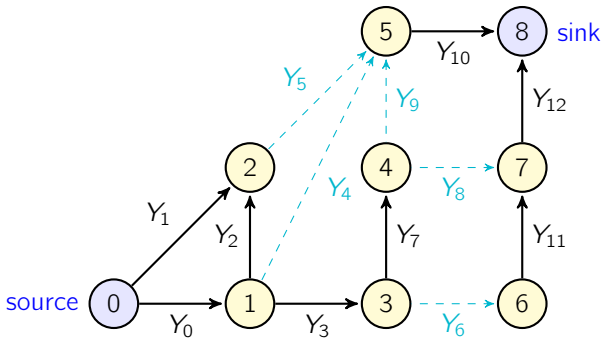
Warning: If the X_i have an asymmetric law, these confidence intervals can have very bad coverage (convergence to normal can be very slow).

Conditional Monte Carlo estimator of $\mathbb{P}[T > x] = \mathbb{E}[\mathbb{I}[T > x]]$.

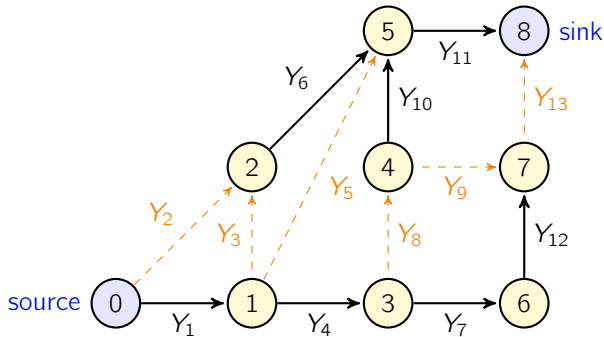
Generate the Y_j's only for the 8 arcs that do not belong to the cut $\mathcal{L} = \{4, 5, 6, 8, 9\}$, and replace the naive estimator $\mathbb{I}[T > x]$ by its conditional expectation given those Y_j's,

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

This reduces the variance and makes the integrand continuous in the U_j's.



Another minimal cut: $\mathcal{L} = \{2, 3, 5, 6, 9, 13\}$.



Works for a general graph: just take $\mathcal{L}$ as a minimal cut between source and sink.

To compute X_e: for each $l \in \mathcal{L}$, say from a_l to b_l, compute the length α_l of the longest path from 1 to a_l, and the length β_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 $\mathbb{P}[Y_l \leq x - \alpha_l - \beta_l] = F_l[x - \alpha_l - \beta_l]$.

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

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

This makes the integrand continuous in the U_j's if the Y_k's are continuous r.v.'s for $k \in \mathcal{L}$.
Also reduces the dimensionality of the integrand!
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

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

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

Algorithm: Option pricing under GBM model

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 $v(s_0, T)$.

Example of contract: Discretely-monitored **Asian call option**:

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

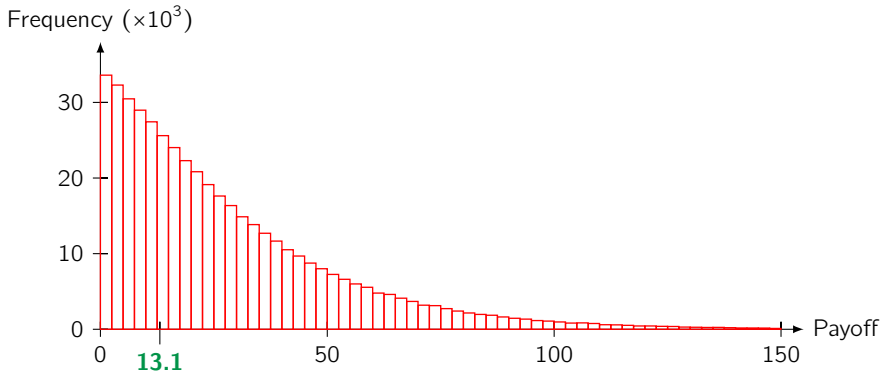
Option price written as an integral over the unit hypercube:

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

$$\begin{aligned} v(s_0, T) &= \int_{[0,1)^s} e^{-rT} \max\left(0, \frac{1}{s}\sum_{i=1}^{s} s_0 \cdot \right. \\ &\qquad \left. \exp\left[(r - \sigma^2/2)t_i + \sigma \sum_{j=1}^{i} \sqrt{t_j - t_{j-1}}\Phi^{-1}(u_j)\right] - K\right) \mathrm{d}u_1 \ldots \mathrm{d}u_s \\ &= \int_{[0,1)^s} f(u_1, \ldots, u_s)\mathrm{d}u_1 \ldots \mathrm{d}u_s. \end{aligned}$$

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:



Variants and extensions

$$\mu = \mathbb{E}[X] = \mathbb{E}[f(\boldsymbol{U})] = \int_{(0,1)^s} f(\boldsymbol{u})\, d\boldsymbol{u}$$

The dimension s can be random and unbounded.

The function f can be very involved.
It can be a function of the trajectory of a Markov chain, for example.

We may want to estimate the whole density of X, or a quantile, etc.

We may want to estimate a global optimizer of f.

Quasi-Monte Carlo

Quasi-Monte Carlo (QMC)

Replace the independent random points $\boldsymbol{U}_i$ by a set of deterministic points $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\}$ that cover $[0,1)^s$ more evenly. Approximate (or estimate)

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

Integration error $E_n = \bar{\mu} - \mu$ (deterministic). This is classical QMC.

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 $\boldsymbol{U}_i$ by a set of deterministic points $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\}$ that cover $[0,1)^s$ more evenly. Approximate (or estimate)

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

Integration error $E_n = \bar{\mu} - \mu$ (deterministic). This is classical QMC.

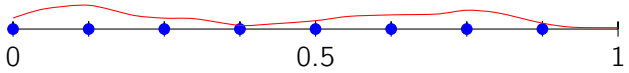
P_n is called a highly-uniform point set or low-discrepancy point set if some measure of discrepancy between the empirical distribution of P_n and the uniform distribution converges to 0 faster than $O(n^{-1/2})$ (the typical rate for independent random points).

Main construction methods: lattice rules and digital nets
(Korobov, Hammersley, Halton, Sobol', Faure, Niederreiter, etc.)

Randomized QMC (RQMC): Randomize P_n so that $\mathbb{E}[E_n] = 0$ and $\mathrm{Var}[E_n] \ll \sigma^2/n$.

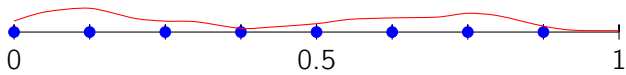
Very simple case: one dimension ($s = 1$)

A simple solution: $P_n = \mathbb{Z}_n/n = \{0, 1/n, \ldots, (n-1)/n\}$ (left Riemann sum), which gives $|E_n| \le K/(2n) = \mathcal{O}(n^{-1})$ if $\sup_{0 \le u \le 1} |f'(u)| \le K$.

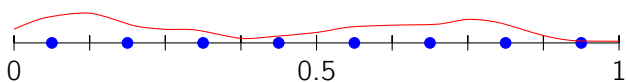


Very simple case: one dimension ($s = 1$)

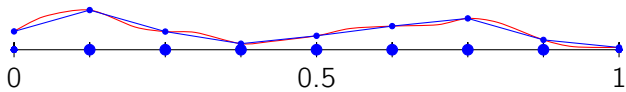
A simple solution: $P_n = \mathbb{Z}_n/n = \{0, 1/n, \ldots, (n-1)/n\}$ (left Riemann sum), which gives $|E_n| \leq K/(2n) = \mathcal{O}(n^{-1})$ if $\sup_{0 \leq u \leq 1} |f'(u)| \leq K$.



Improvement: $P'_n = \{1/(2n), 3/(2n), \ldots, (2n-1)/(2n)\}$ (the midpoint rule), which gives $|E_n| \leq K/(24n^2) = \mathcal{O}(n^{-2})$ if $\sup_{0 \leq u \leq 1} |f''(u)| \leq K$.



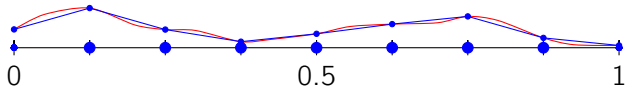
If we allow different weights for the $f(\boldsymbol{u}_i)$, we have the trapezoidal rule, which approximates f by a piecewise-linear interpolation with n pieces:



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

This gives $|E_n| \le K/(12n^2) = \mathcal{O}(n^{-2})$ if $\sup_{0\le u\le 1}|f''(u)| \le K$.

If we allow different weights for the $f(\boldsymbol{u}_i)$, we have the trapezoidal rule, which approximates f by a piecewise-linear interpolation with n pieces:



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

This gives $|E_n| \le K/(12n^2) = \mathcal{O}(n^{-2})$ if $\sup_{0\le u\le 1}|f''(u)| \le K$.

Simpson's rule use a piecewise-quadratic approximation with $n/2$ pieces:

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

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

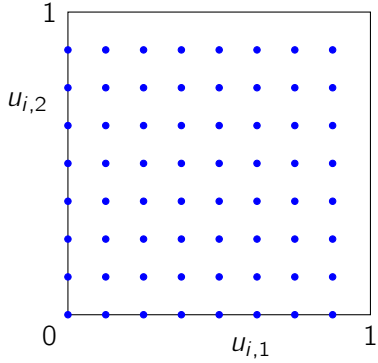
These approximations work well when the derivatives of f (of the appropriate order) are never too large. That is, f must be "smooth".

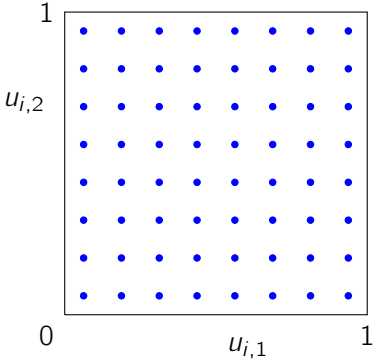
For QMC and RQMC, we will restrict ourselves to equal-weight rules.
For the RQMC points that we will examine, one can prove that equal weights are optimal.

A simplistic solution for when $s > 1$: a rectangular grid (left) with $n = d^s$ points:

$$P_n = \{(i_1/d, \ldots, i_t/d) \text{ tels que } 0 \le i_j < d \quad \forall j\}.$$

We can center it to obtain a s-dimensional midpoint rule (right).





This quickly becomes impractical when s increases.

Error bound for midpoint rule in s dimensions with $n = d^s$ points.

Suppose that

$$\sup_{(u_1,\ldots,u_s)\in[0,1]} \left| \frac{\partial^2 f(u_1,\ldots,u_s)}{\partial u_j^2} \right| \leq K_j < \infty$$

for $j = 1, \ldots, s$ and let $K = K_1 + \cdots + K_s$.
Then the midpoint rule with n points in s dimensions gives

$$|E_n| \leq K d^{-2}/24 = K n^{-2/s}/24 = \mathcal{O}(n^{-2/s}).$$

The s-dimensional trapezoidal rule gives the same rate: $E_n = \mathcal{O}(n^{-2/s})$.

Error bound for midpoint rule in s dimensions with $n = d^s$ points.

Suppose that

$$\sup_{(u_1,\ldots,u_s)\in[0,1]} \left| \frac{\partial^2 f(u_1,\ldots,u_s)}{\partial u_j^2} \right| \le K_j < \infty$$

for $j = 1, \ldots, s$ and let $K = K_1 + \cdots + K_s$.
Then the midpoint rule with n points in s dimensions gives

$$|E_n| \le K d^{-2}/24 = K n^{-2/s}/24 = \mathcal{O}(n^{-2/s}).$$

The s-dimensional trapezoidal rule gives the same rate: $E_n = \mathcal{O}(n^{-2/s})$.

Note that these rates are deterministic, whereas the MC rate is only probabilistic.
In $s = 4$ dimensions, we get the same rate as for MC.
In $s < 4$ dimensions, this rate is better, while for $s > 4$ the MC rate is better.

The degradation comes from the superposition of points in the projections over subsets of coordinates. To see why, note that any integrand f over $[0,1)^s$ can be decomposed as

$$f(u_1,\dots,u_s) = \sum_{j=1}^{s} f_j(u_j) + \sum_{1\le i<j\le s} f_{i,j}(u_i,u_j) + \sum_{1\le i<j<k\le s} f_{i,j,k}(u_i,u_j,u_k) + \cdots$$

This is a sum of s one-dimensional functions, plus other terms. The integration error here will be the total error for the first sum, plus the error for the other terms.

But each one-dimensional projection of P_n contains only the d distinct values $\{0, 1/d, \dots, (d-1)/d\}$, so the first s terms are evaluated at only those d values. Then the error for these terms with the midpoint rule will be $\mathcal{O}(d^{-2}) = \mathcal{O}(n^{-2/s})$ instead of $\mathcal{O}(n^{-2})$.

The degradation comes from the superposition of points in the projections over subsets of coordinates. To see why, note that any integrand f over $[0,1)^s$ can be decomposed as

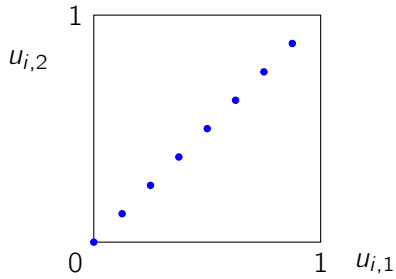
$$f(u_1,\ldots,u_s) = \sum_{j=1}^{s} f_j(u_j) + \sum_{1\le i<j\le s} f_{i,j}(u_i,u_j) + \sum_{1\le i<j<k\le s} f_{i,j,k}(u_i,u_j,u_k) + \cdots$$

This is a sum of s one-dimensional functions, plus other terms. The integration error here will be the total error for the first sum, plus the error for the other terms.

But each one-dimensional projection of P_n contains only the d distinct values $\{0, 1/d, \ldots, (d-1)/d\}$, so the first s terms are evaluated at only those d values. Then the error for these terms with the midpoint rule will be $\mathcal{O}(d^{-2}) = \mathcal{O}(n^{-2/s})$ instead of $\mathcal{O}(n^{-2})$.

If we do not shift the points by $1/(2d)$ for each coordinate (as in the left plot), the error will be $\mathcal{O}(d^{-1}) = \mathcal{O}(n^{-1/s})$ instead of $\mathcal{O}(n^{-1})$.

To avoid this superposition of points, we would like to construct P_n so that each one-dimensional projection is $\{0, 1/n, \ldots, (n-1)/n\}$.



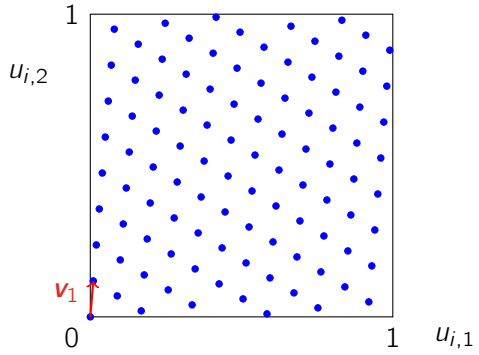
We also want these n values to be enumerated in a different order for the different coordinates. Otherwise all the points are on a diagonal line.

We must take a different permutation of $\{0, 1/n, \ldots, (n-1)/n\}$ for each coordinate, and choose these permutations so that P_n is highly uniform over $[0,1)^s$. **How can we do that?**

We may also center the points by adding $1/(2n)$ to each coordinate, as in the midpoint rule.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/n$

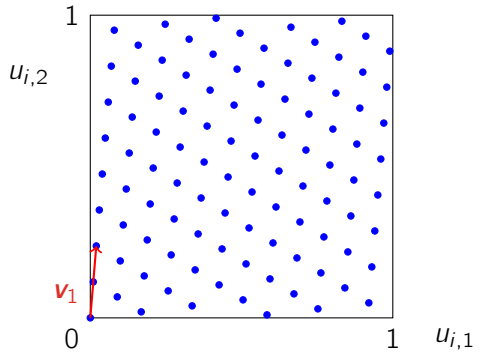
$$
\begin{aligned}
P_n &= \{\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1) : i = 0, \ldots, n-1\} \\
&= \{(0,0),\ (1/101, 12/101),\ (2/101,\ 43/101), \ldots\}.
\end{aligned}
$$



Here, each one-dimensional projection is $\{0,\ 1/n,\ \ldots,\ (n-1)/n\}$.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/n$

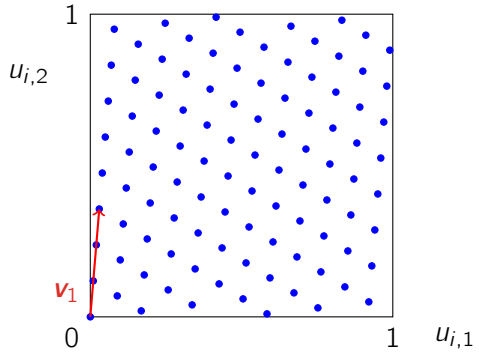
$$
\begin{aligned}
P_n &= \{\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1) : i = 0, \ldots, n-1\} \\
&= \{(0,0),\ (1/101, 12/101),\ (2/101,\ 43/101), \ldots\}.
\end{aligned}
$$



Here, each one-dimensional projection is $\{0,\ 1/n,\ \ldots,\ (n-1)/n\}$.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/n$

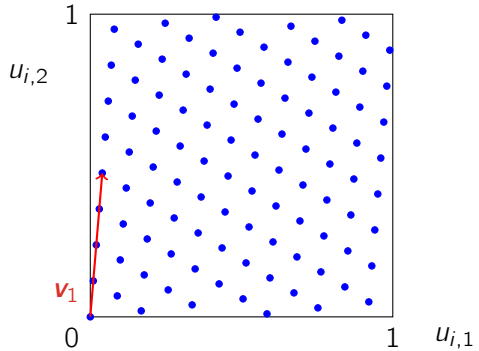
$$
\begin{aligned}
P_n &= \{\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1) : i = 0, \ldots, n-1\} \\
&= \{(0,0),\ (1/101, 12/101),\ (2/101,\ 43/101), \ldots\}.
\end{aligned}
$$



Here, each one-dimensional projection is $\{0,\ 1/n,\ \ldots,\ (n-1)/n\}$.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/n$

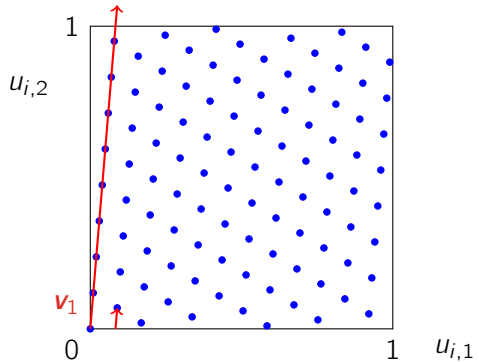
$$
\begin{aligned}
P_n &= \{\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1) : i = 0, \ldots, n-1\} \\
&= \{(0,0),\ (1/101, 12/101),\ (2/101,\ 43/101), \ldots\}.
\end{aligned}
$$



Here, each one-dimensional projection is $\{0,\ 1/n,\ \ldots,\ (n-1)/n\}$.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/n$

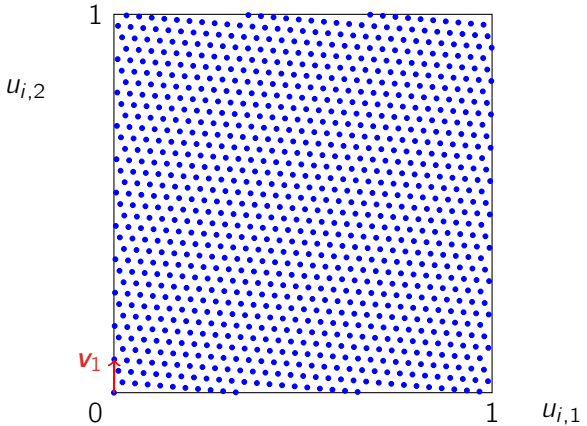
$$
\begin{aligned}
P_n &= \{\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1) : i = 0, \ldots, n-1\} \\
&= \{(0,0),\ (1/101, 12/101),\ (2/101,\ 43/101), \ldots\}.
\end{aligned}
$$



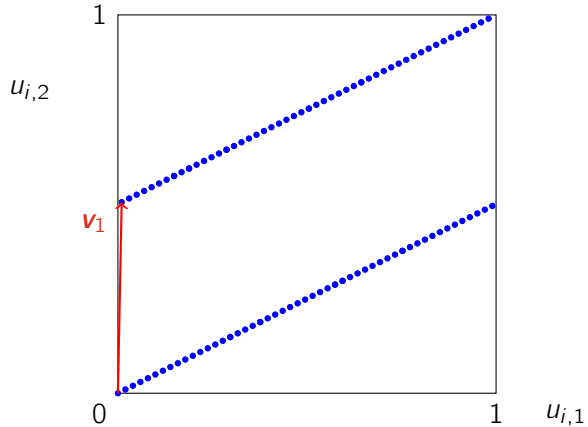
Here, each one-dimensional projection is $\{0,\ 1/n,\ \ldots,\ (n-1)/n\}$.

Another example: $s = 2$, $n = 1021$, $\boldsymbol{v}_1 = (1, 90)/n$

$$
\begin{aligned}
P_n &= \{\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1 : i = 0, \ldots, n-1\} \\
&= \{(i/1021,\ (90i/1021) \bmod 1) : i = 0, \ldots, 1020\}.
\end{aligned}
$$



A bad lattice: $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 51)/n$



Good uniformity in one dimension, but not in two!

Lattice rules (Korobov, Sloan, etc.)

Integration lattice:

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

where $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s \in \mathbb{R}^s$ are linearly independent over $\mathbb{R}$ and where L_s contains $\mathbb{Z}^s$.

Lattice rule: Take $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\} = L_s \cap [0,1)^s$.

Lattice rules (Korobov, Sloan, etc.)

Integration lattice:

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

where $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s \in \mathbb{R}^s$ are linearly independent over $\mathbb{R}$ and where L_s contains $\mathbb{Z}^s$.

Lattice rule: Take $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\} = L_s \cap [0,1)^s$.

Lattice rule of rank 1: $\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1$ for $i = 0, \ldots, n-1$,
where $n\boldsymbol{v}_1 = \boldsymbol{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n-1\}^s$. These are the most popular.

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

Lattice rules (Korobov, Sloan, etc.)

Integration lattice:

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

where $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s \in \mathbb{R}^s$ are linearly independent over $\mathbb{R}$ and where L_s contains $\mathbb{Z}^s$.

Lattice rule: Take $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\} = L_s \cap [0,1)^s$.

Lattice rule of rank 1: $\boldsymbol{u}_i = i\boldsymbol{v}_1 \bmod 1$ for $i = 0, \ldots, n-1$,
where $n\boldsymbol{v}_1 = \boldsymbol{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n-1\}^s$. These are the most popular.

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

For any $\mathfrak{u} \subset \{1, \ldots, s\}$, the projection $L_s(\mathfrak{u})$ of L_s is also a lattice.

If $\text{pgcd}(n, a_j) = 1$, then coordinate j takes each value in $\mathbb{Z}_n/n = \{0, 1/n, 2/n, \ldots, (n-1)/n\}$ exactly once, in a certain order.

Remarks and insight

Ces remarques seront précisées par des théorèmes plus loin.

1. Avec de tels points, si f est une somme de fonctions à une variable (pas d'autres termes), l'erreur va converger à la même vitesse que pour une fonction à une dimension. Mais s'il y a d'autres termes, la convergence sera moins bonne en général.

2. Plus la dimension s augmente, plus c'est difficile de bien couvrir l'hypercube unitaire uniformément par des points. Ainsi, QMC est généralement moins efficace en plus grande dimension. Les taux de convergence de l'erreur vont réfléter cela.

3. Par contre, si f est bien approximée par une somme de fonctions de petite dimension, par exemples de 1 à 3 dimensions, et si les points sont bien construits pour les projections correspondantes, alors on peut gagner beaucoup.

4. Pour que QMC/RQMC fasse beaucoup mieux que MC, il faut en général que f et ses dérivées soient lisses et ne varient pas trop. QMC n'aime pas les fonctions discontinues ou qui oscillent beaucoup.

Digital net in base b (Sobol', Niederreiter, ...)

This is a different type of construction for $n = b^k$ points in s dimensions.

Select a prime number b (the base, most commonly $b = 2$), two integers $w \geq k \geq 0$, and s generating matrices $\boldsymbol{C}_1, \cdots, \boldsymbol{C}_s$ of dimensions $w \times k$, with entries in $\mathbb{Z}_b$, and whose first k rows are linearly independent. For $i = 0, \ldots, b^k - 1$ and $j = 1, \ldots, s$, let:

$$
\begin{aligned}
i &= a_{i,0} + a_{i,1} b + \cdots + a_{i,k-1} b^{k-1} && (k \text{ digits en base b}) \\
\begin{pmatrix} u_{i,j,1} \\ \vdots \\ u_{i,j,w} \end{pmatrix} &= \boldsymbol{C}_j \begin{pmatrix} a_{i,0} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \bmod b, && (w \text{ digits en base b}) \\
u_{i,j} &= \sum_{\ell=1}^{w} u_{i,j,\ell} b^{-\ell}, \qquad \boldsymbol{u}_i = (u_{i,1}, \ldots, u_{i,s}) \in [0,1)^s.
\end{aligned}
$$

Digital net in base b (Sobol', Niederreiter, ...)

This is a different type of construction for $n = b^k$ points in s dimensions.

Select a prime number b (the base, most commonly $b = 2$), two integers $w \geq k \geq 0$, and s generating matrices $\boldsymbol{C}_1, \cdots, \boldsymbol{C}_s$ of dimensions $w \times k$, with entries in $\mathbb{Z}_b$, and whose first k rows are linearly independent. For $i = 0, \ldots, b^k - 1$ and $j = 1, \ldots, s$, let:

$$
\begin{aligned}
i &= a_{i,0} + a_{i,1} b + \cdots + a_{i,k-1} b^{k-1} && (k \text{ digits en base b}) \\
\begin{pmatrix} u_{i,j,1} \\ \vdots \\ u_{i,j,w} \end{pmatrix} &= \boldsymbol{C}_j \begin{pmatrix} a_{i,0} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \bmod b, && (w \text{ digits en base b}) \\
u_{i,j} &= \sum_{\ell=1}^{w} u_{i,j,\ell} b^{-\ell}, && \boldsymbol{u}_i = (u_{i,1}, \ldots, u_{i,s}) \in [0,1)^s.
\end{aligned}
$$

Digital sequence: each $\boldsymbol{C}_j$ has an unlimited number of columns.
Gives an infinite sequence of points. Can stop at $n = b^k$ points for any k.

Uniformity

For each one-dimensional projection, we have exactly one point in each interval $[i/n, (i+1)/n)$, for $i = 0, \ldots, n-1$.

Uniformity

For each one-dimensional projection, we have exactly one point in each interval $[i/n, (i+1)/n)$, for $i = 0, \ldots, n-1$.

The higher-dimensional uniformity will depend on how we construct the $\boldsymbol{C}_j$'s with respect to each other. In s dimensions, we can partition $[0,1)^s$ in n rectangular boxes of the same size and request that each box contains exactly one point $\boldsymbol{u}_i$.

Uniformity

For each one-dimensional projection, we have exactly one point in each interval $[i/n, (i+1)/n)$, for $i = 0, \ldots, n-1$.

The higher-dimensional uniformity will depend on how we construct the $\boldsymbol{C}_j$'s with respect to each other. In s dimensions, we can partition $[0,1)^s$ in n rectangular boxes of the same size and request that each box contains exactly one point $\boldsymbol{u}_i$.

In two dimensions, if $k = k_1 + k_2$ and if we partition the first axis in b^{k_1} equal intervals and the second axis in b^{k_2} equal intervals, we obtain a partition into $n = b^k$ rectangles of the same size. The first k_1 digits $u_{i,1}$ and the first k_2 digits of $u_{i,2}$ determine in which box the point $\boldsymbol{u}_i$ will fall. There will be one point in each box iff the vector formed by those $k_1 + k_2$ digits takes each of its b^k possible values when i goes from 0 to $n-1$. This happens iff the matrix formed by the first k_1 rows of $\boldsymbol{C}_1$ and the first k_2 rows of $\boldsymbol{C}_2$ is invertible.

Uniformity

For each one-dimensional projection, we have exactly one point in each interval $[i/n, (i+1)/n)$, for $i = 0, \ldots, n-1$.

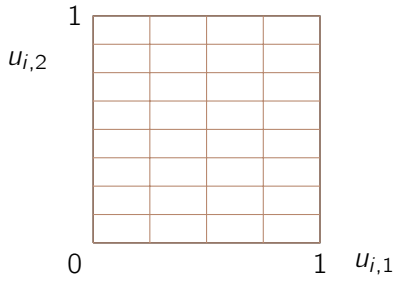
The higher-dimensional uniformity will depend on how we construct the $\boldsymbol{C}_j$'s with respect to each other. In s dimensions, we can partition $[0,1)^s$ in n rectangular boxes of the same size and request that each box contains exactly one point $\boldsymbol{u}_i$.

In two dimensions, if $k = k_1 + k_2$ and if we partition the first axis in b^{k_1} equal intervals and the second axis in b^{k_2} equal intervals, we obtain a partition into $n = b^k$ rectangles of the same size. The first k_1 digits $u_{i,1}$ and the first k_2 digits of $u_{i,2}$ determine in which box the point $\boldsymbol{u}_i$ will fall. There will be one point in each box iff the vector formed by those $k_1 + k_2$ digits takes each of its b^k possible values when i goes from 0 to $n-1$. This happens iff the matrix formed by the first k_1 rows of $\boldsymbol{C}_1$ and the first k_2 rows of $\boldsymbol{C}_2$ is invertible.

One easy way to obtain this property: Select a $k \times k$ invertible matrix $\boldsymbol{C}_1$, then put the rows of $\boldsymbol{C}_1$ in reverse order to define $\boldsymbol{C}_2$. Then the matrix that contains the first k_1 rows of $\boldsymbol{C}_1$ and the first $k_2 = k - k_1$ rows of $\boldsymbol{C}_2$ is invertible, because it has the same rows as $\boldsymbol{C}_1$.

Two-dimensional example with $b = 2$, $k_1 = 2$, $k_2 = 3$, $k = 5$.

Let $\boldsymbol{c}_{j,\ell}$ denote row ℓ of matrix $\boldsymbol{C}_j$.



$$\begin{pmatrix} u_{i,1,1} \\ u_{i,1,2} \\ u_{i,2,1} \\ u_{i,2,2} \\ u_{i,2,3} \end{pmatrix} = \begin{pmatrix} c_{1,1} \\ c_{1,2} \\ c_{2,1} \\ c_{2,2} \\ c_{2,3} \end{pmatrix} \begin{pmatrix} a_{i,0} \\ a_{i,1} \\ a_{i,2} \\ a_{i,3} \\ a_{i,4} \end{pmatrix} \bmod 2,$$

The box number of point $\boldsymbol{u}_i$ is determined by the first two bits of $u_{i,1}$ (the first two rows of $\boldsymbol{C}_1$) and the first three bits of $u_{i,2}$ (the first three rows of $\boldsymbol{C}_2$).

We have one point per box iff the linear mapping above is one-to-one,
iff the five rows $\boldsymbol{c}_{1,1}, \cdots, \boldsymbol{c}_{2,3}$ are linearly independent.

Example: Hammersley point set: $b = 2$, $s = 2$, $\boldsymbol{C}_1$ = reflected identity, $\boldsymbol{C}_2$ = identity:

$$\boldsymbol{C}_1 = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 1 & 0 \\ \vdots & \iddots & 0 & 0 \\ 1 & \cdots & 0 & 0 \end{pmatrix}, \qquad \boldsymbol{C}_2 = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Example: Hammersley point set: $b = 2$, $s = 2$, $\boldsymbol{C}_1$ = reflected identity, $\boldsymbol{C}_2$ = identity:

$$\boldsymbol{C}_1 = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 1 & 0 \\ \vdots & \iddots & 0 & 0 \\ 1 & \cdots & 0 & 0 \end{pmatrix}, \qquad \boldsymbol{C}_2 = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

For $k = 8$, this gives the $n = 2^8 = 256$ points (en binary):

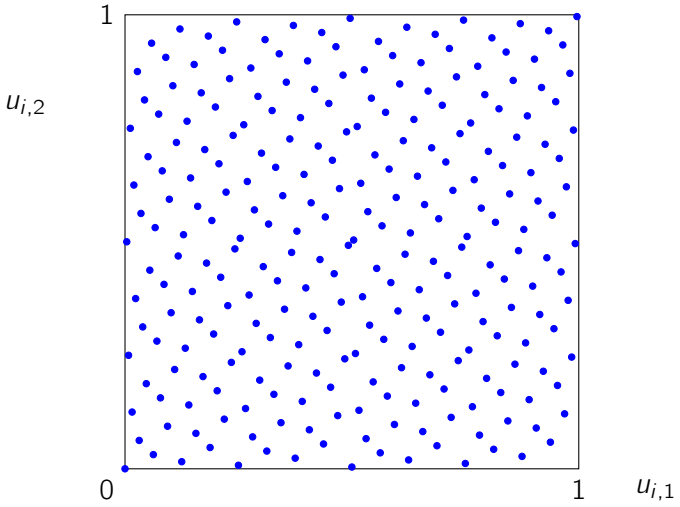
i	$u_{1,i}$	$u_{2,i}$
0	.00000000	.0
1	.00000001	.1
2	.00000010	.01
3	.00000011	.11
4	.00000100	.001
5	.00000101	.101
⋮	⋮	⋮
254	.11111110	.01111111
255	.11111111	.11111111

Left column: $0, 1/n, 2/n, \ldots$

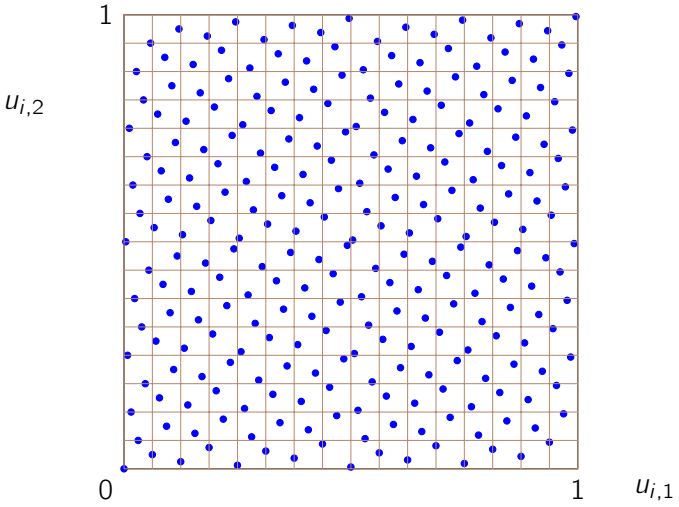
Right column: van der Corput sequence in base 2.

Works for any $k > 0$.

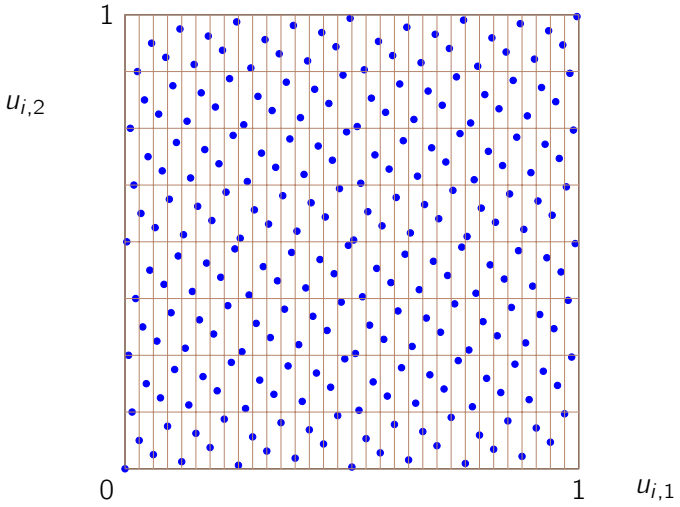
Hammersley point set in base 2, $n = 2^8 = 256$, $s = 2$.



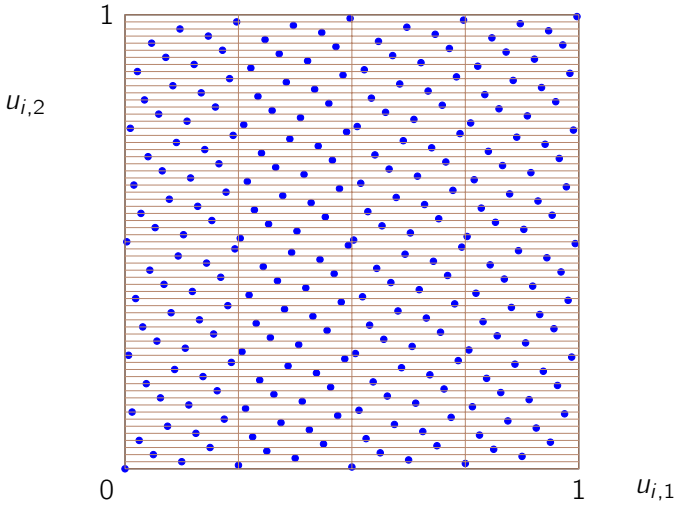
Hammersley point set in base 2, $n = 2^8 = 256$, $s = 2$.



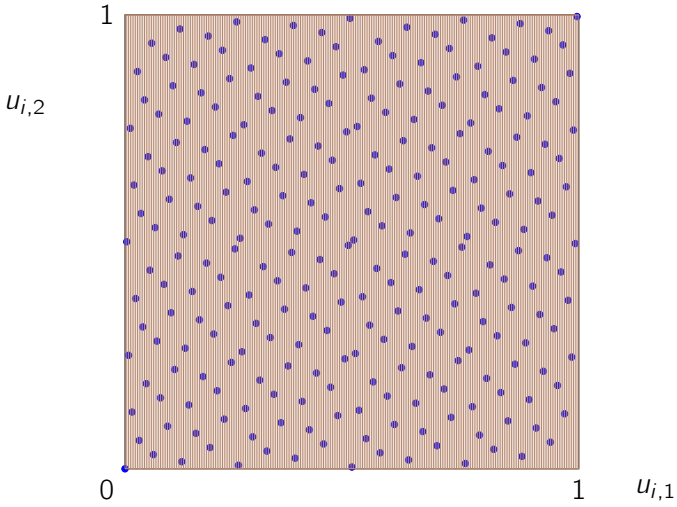
Hammersley point set in base 2, $n = 2^8 = 256$, $s = 2$.



Hammersley point set in base 2, $n = 2^8 = 256$, $s = 2$.



Hammersley point set in base 2, $n = 2^8 = 256$, $s = 2$.



Hammersley point set in base 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.

Hammersley point set in base 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.

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

Can also have $s = \infty$ and/or $n = \infty$ (infinite sequence of points).

Equidistribution in s dimensions

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 $\boldsymbol{C}_1$, the first q_2 rows of $\boldsymbol{C}_2$, ..., the first q_s rows of $\boldsymbol{C}_s$, 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 $q_1+\cdots+q_s = k-t$.
t-value of a net: smallest t for which it is a (t,k,s)-net.
It is possible to have $t=0$ only if $b \geq s-1$.
Example: Hammersley points form a $(0,k,2)$-net in base 2.

Equidistribution in s dimensions

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 $\boldsymbol{C}_1$, the first q_2 rows of $\boldsymbol{C}_2$, ..., the first q_s rows of $\boldsymbol{C}_s$, 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 $q_1+\cdots+q_s = k-t$.
t-value of a net: smallest t for which it is a (t,k,s)-net.
It is possible to have $t=0$ only if $b \geq s-1$.
Example: Hammersley points form a $(0,k,2)$-net in base 2.

An infinite sequence $\{\boldsymbol{u}_0,\boldsymbol{u}_1,\ldots,\}$ in $[0,1)^s$ is a (t,s)-sequence in base b if for all $k>0$ and $\nu \geq 0$, $Q(k,\nu) = \{\boldsymbol{u}_i : i = \nu b^k,\ldots,(\nu+1)b^k-1\}$, is a (t,k,s)-net in base b.
This is possible for $t=0$ only if $b \geq s$.

Faure nets and sequences in base b

Faure (1982) proposed the matrices

$$\boldsymbol{C}_j = \boldsymbol{P}^j \bmod b = \boldsymbol{P}\boldsymbol{C}_{j-1} \bmod b$$

with $\boldsymbol{C}_0 = \boldsymbol{I}$ and $\boldsymbol{P} = (p_{l,c})$ upper triangular where

$$p_{l,c} = \binom{c}{l} = \frac{c!}{l!(c-l)!}$$

for $l \le c$.

Faure nets and sequences in base b

Faure (1982) proposed the matrices

$$\boldsymbol{C}_j = \boldsymbol{P}^j \bmod b = \boldsymbol{P}\boldsymbol{C}_{j-1} \bmod b$$

with $\boldsymbol{C}_0 = \boldsymbol{I}$ and $\boldsymbol{P} = (p_{l,c})$ upper triangular where

$$p_{l,c} = \binom{c}{l} = \frac{c!}{l!(c-l)!}$$

for $l \le c$.

Faure proved that if b is prime and $b \ge s$, this gives a $(0, s)$-sequence in base b.

Thus, for all $k > 0$ and $\nu \ge 0$, $Q(k, \nu) = \{\boldsymbol{u}_i : i = \nu b^k, \ldots, (\nu+1)b^k - 1\}$
(which contains $n = b^k$ points) is a $(0, k, s)$-net in base b.

In this set, each coordinate j visits all values in $\{0, 1/n, \ldots, (n-1)/n\}$ once and only once.

Sobol' nets and sequences

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

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

Sobol' nets and sequences

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

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

Column c of $\boldsymbol{C}_j$ 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

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

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

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 $t \le d_0 + \cdots + d_{s-1} + 1 - s$.

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 $\boldsymbol{C}_0 = \boldsymbol{I}$), 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, ℓ)-equidistribution for $\ell = 1$ and $\ell = 2$ (only the first two bits).

Joe and Kuo (2008) tabulated direction numbers giving good t-values for the two-dimensional projections, for given s and k.

Other constructions

Niederreiter-Xing point sets and sequences: better t-values than Sobol'

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.):

$$|\hat{\mu}_{n,\text{rqmc}} - \mu| \;\le\; V(f) \cdot D(P_n)$$

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 α:

$D(P_n) = \mathcal{O}(n^{-\alpha+\epsilon})$ for arbitrary small ϵ.

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

$D(P_n) = \mathcal{O}(n^{-1}(\ln n)^s) = \mathcal{O}(n^{-1+\epsilon})$ for arbitrary small ϵ.

More recent constructions offer $\mathcal{O}(n^{-\alpha+\epsilon})$ rates for smooth functions.

Worst-case error bounds

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

$$|\hat{\mu}_{n,\text{rqmc}} - \mu| \;\le\; V(f) \cdot D(P_n)$$

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 α:

$D(P_n) = \mathcal{O}(n^{-\alpha+\epsilon})$ for arbitrary small ϵ.

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

$D(P_n) = \mathcal{O}(n^{-1}(\ln n)^s) = \mathcal{O}(n^{-1+\epsilon})$ for arbitrary small ϵ.

More recent constructions offer $\mathcal{O}(n^{-\alpha+\epsilon})$ rates for smooth functions.

However, these bounds are conservative and too hard to compute in practice.

Randomized quasi-Monte Carlo (RQMC)

$$\hat{\mu}_{n,\text{rqmc}} = \frac{1}{n}\sum_{i=0}^{n-1} f(\boldsymbol{U}_i),$$

with $P_n = \{\boldsymbol{U}_0, \ldots, \boldsymbol{U}_{n-1}\} \subset (0,1)^s$ an RQMC point set:

(i) each point $\boldsymbol{U}_i$ has the uniform distribution over $(0,1)^s$;

(ii) P_n as a whole is a low-discrepancy point set.

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

$$\text{Var}[\hat{\mu}_{n,\text{rqmc}}] = \frac{\text{Var}[f(\boldsymbol{U}_i)]}{n} + \frac{2}{n^2}\sum_{i<j}\text{Cov}[f(\boldsymbol{U}_i), f(\boldsymbol{U}_j)].$$

We want the last sum to be as negative as possible.

Randomized quasi-Monte Carlo (RQMC)

$$\hat{\mu}_{n,\text{rqmc}} = \frac{1}{n}\sum_{i=0}^{n-1} f(\boldsymbol{U}_i),$$

with $P_n = \{\boldsymbol{U}_0, \ldots, \boldsymbol{U}_{n-1}\} \subset (0,1)^s$ an RQMC point set:

(i) each point $\boldsymbol{U}_i$ has the uniform distribution over $(0,1)^s$;

(ii) P_n as a whole is a low-discrepancy point set.

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

$$\text{Var}[\hat{\mu}_{n,\text{rqmc}}] = \frac{\text{Var}[f(\boldsymbol{U}_i)]}{n} + \frac{2}{n^2}\sum_{i<j}\text{Cov}[f(\boldsymbol{U}_i), f(\boldsymbol{U}_j)].$$

We want the last sum to be as negative as possible.
Weak attempts to do this:
antithetic variates ($n = 2$), Latin hypercube sampling (LHS), stratification, ...

Variance estimation:

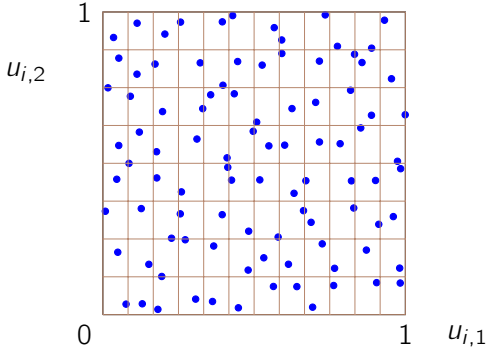
Can compute m independent realizations $X_1, \ldots, X_m$ of $\hat{\mu}_{n,\text{rqmc}}$, then estimate μ and $\mathrm{Var}[\hat{\mu}_{n,\text{rqmc}}]$ by their sample mean $\bar{X}_m$ and sample variance S_m^2.
Could be used to compute a confidence interval.

Beware: $\bar{X}_m$ does not obey a central-limit theorem in general when m is fixed and $n \to \infty$.

Stratification of the unit hypercube

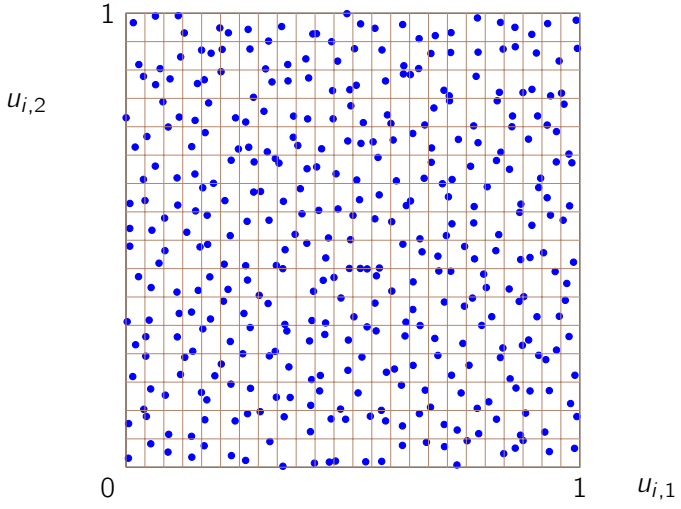
Partition axis j in $k_j \geq 1$ equal parts, for $j = 1, \ldots, s$.
Draw $n = k_1 \cdots k_s$ random points, one per box, independently.

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:

$$X_{\mathrm{s},n} = \frac{1}{n}\sum_{j=0}^{n-1} f(\boldsymbol{U}_j).$$

The crude MC variance with n points can be decomposed as

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

where μ_j is the mean over box j. The more the μ_j differ, the more the variance is reduced.
This is a simple form of stratification with proportional allocation.
Gives an unbiased estimator, and variance can be estimated by replicating $m \geq 2$ times.

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

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

Stratified estimator:

$$X_{s,n} = \frac{1}{n}\sum_{j=0}^{n-1} f(\boldsymbol{U}_j).$$

The crude MC variance with n points can be decomposed as

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

where μ_j is the mean over box j. The more the μ_j differ, the more the variance is reduced. This is a simple form of stratification with proportional allocation.
Gives an unbiased estimator, and variance can be estimated by replicating $m \geq 2$ times.

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

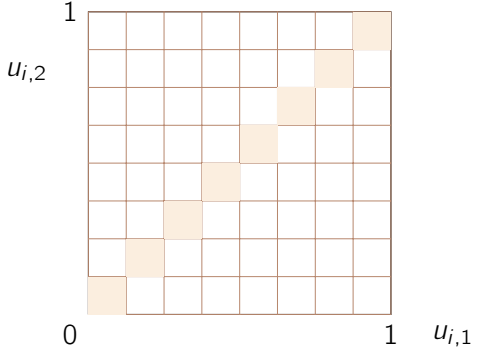
$$\mathrm{Var}[X_{s,n}] = \mathcal{O}(n^{-1-2/s}).$$

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.

Latin Hypercube Sampling (LHS)

Also a form of stratification with proportional allocation. We stratify in one dimension (only) for each coordinate, independently. The number of points is equal to the number of one-dimensional intervals, so s can be very large and there is no curse of dimensionality.

We partition the unit hypercube in k^s subcubes, and we generate k points in total in a way that for each coordinate j, there is exactly one point in each of the k sub-intervals of $(0, 1)$.

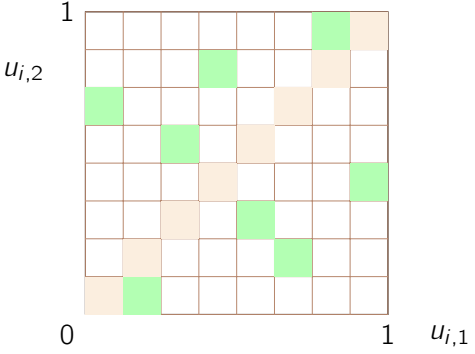


Permute the columns randomly, for each coordinate.
Then generate one point randomly in each selected box.

Latin Hypercube Sampling (LHS)

Also a form of stratification with proportional allocation. We stratify in one dimension (only) for each coordinate, independently. The number of points is equal to the number of one-dimensional intervals, so s can be very large and there is no curse of dimensionality.

We partition the unit hypercube in k^s subcubes, and we generate k points in total in a way that for each coordinate j, there is exactly one point in each of the k sub-intervals of $(0, 1)$.

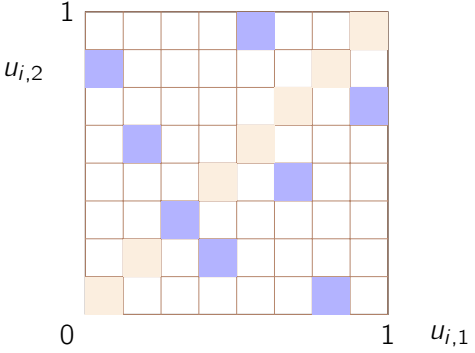


Permute the columns randomly, for each coordinate.
Then generate one point randomly in each selected box.

Latin Hypercube Sampling (LHS)

Also a form of stratification with proportional allocation. We stratify in one dimension (only) for each coordinate, independently. The number of points is equal to the number of one-dimensional intervals, so s can be very large and there is no curse of dimensionality.

We partition the unit hypercube in k^s subcubes, and we generate k points in total in a way that for each coordinate j, there is exactly one point in each of the k sub-intervals of $(0, 1)$.

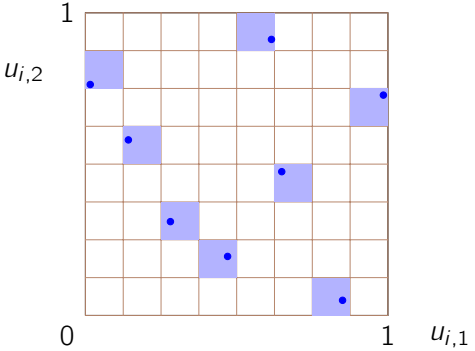


Permute the columns randomly, for each coordinate.
Then generate one point randomly in each selected box.

Latin Hypercube Sampling (LHS)

Also a form of stratification with proportional allocation. We stratify in one dimension (only) for each coordinate, independently. The number of points is equal to the number of one-dimensional intervals, so s can be very large and there is no curse of dimensionality.

We partition the unit hypercube in k^s subcubes, and we generate k points in total in a way that for each coordinate j, there is exactly one point in each of the k sub-intervals of $(0, 1)$.



Permute the columns randomly, for each coordinate.
Then generate one point randomly in each selected box.

Algorithm to generate the k LHS points

$$\boldsymbol{U}_1 = (U_{1,1}, \ldots, U_{1,j}, \ldots, U_{1,s}), \quad \ldots, \quad \boldsymbol{U}_k = (U_{k,1}, \ldots, U_{k,j}, \ldots, U_{k,s}) :$$

in s dimensions:

for coordinate $j = 1, \ldots, s$ **do**
  generate a random permutation $(\pi_{1,j}, \ldots, \pi_{k,j})$ of the integers $\{1, \ldots, k\}$
  for $i = 1, \ldots, k$ **do**
    generate $U_{i,j}$ uniformly in $((\pi_{i,j} - 1)/k,\ \pi_{i,j}/k)$

Then we compute the average $X_{\text{lh}} = [f(\boldsymbol{U}_1) + \cdots + f(\boldsymbol{U}_k)]/k$.

Properties of LHS

For each i, we have $\boldsymbol{U}_i \sim U[0,1]^s$, so the LHS estimator X_{lh} is unbiased.
For each j, $\{U_{1,j}, \ldots, U_{k,j}\}$ is a stratified sample of the $U(0,1)$ distribution.

Properties of LHS

For each i, we have $\boldsymbol{U}_i \sim U[0,1]^s$, so the LHS estimator X_{lh} is unbiased.
For each j, $\{U_{1,j}, \ldots, U_{k,j}\}$ is a stratified sample of the $U(0,1)$ distribution.

We can replicate the procedure m times independently to estimate the mean and the variance as in RQMC.

LHS does not always reduce the variance compared with MC, but under certain conditions, we know that it does. For example:

Theorem. If $f(U_1, \ldots, U_s)$ is monotone in U_j for all j, then $\mathrm{Var}[X_{\text{lh}}] \leq \mathrm{Var}[\bar{X}_k]$.

Properties of LHS

For each i, we have $\boldsymbol{U}_i \sim U[0,1]^s$, so the LHS estimator X_{lh} is unbiased.
For each j, $\{U_{1,j}, \ldots, U_{k,j}\}$ is a stratified sample of the $U(0,1)$ distribution.

We can replicate the procedure m times independently to estimate the mean and the variance as in RQMC.

LHS does not always reduce the variance compared with MC, but under certain conditions, we know that it does. For example:

Theorem. If $f(U_1, \ldots, U_s)$ is monotone in U_j for all j, then $\mathrm{Var}[X_{\text{lh}}] \leq \mathrm{Var}[\bar{X}_k]$.

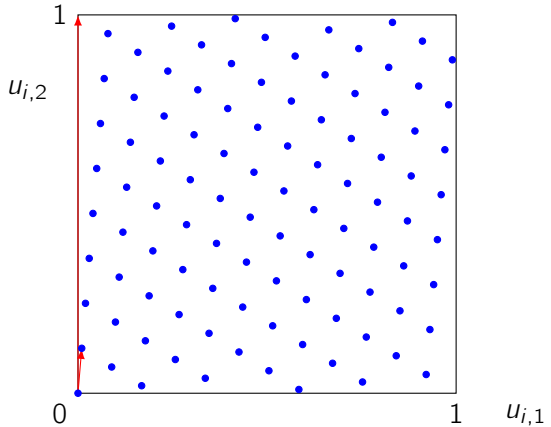
There is also a central-limit theorem for LHS. Unless f is a sum of one-dimensional functions, the variance remains in $\mathcal{O}(k^{-1})$ as for standard MC, so LHS does not improve the convergence rate of the variance or standard deviation.

The reason: LHS only ensures uniformity of one-dimensional projections. RQMC, on the other hand, can also provide better uniformity for higher-dimensional projections.

Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the two RQMC conditions.

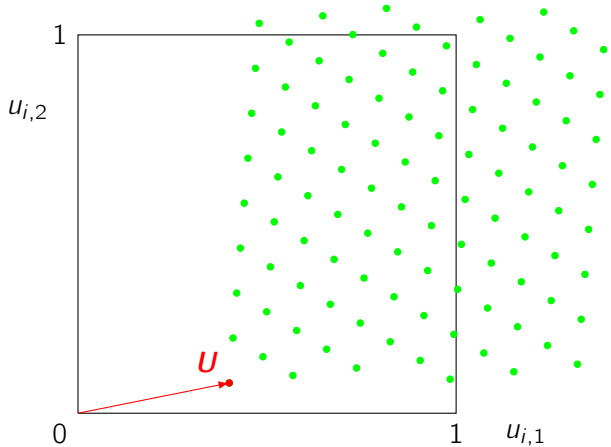
Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/101$



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the two RQMC conditions.

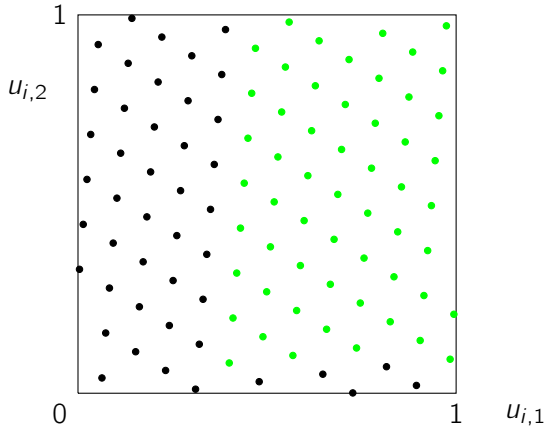
Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/101$



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the two RQMC conditions.

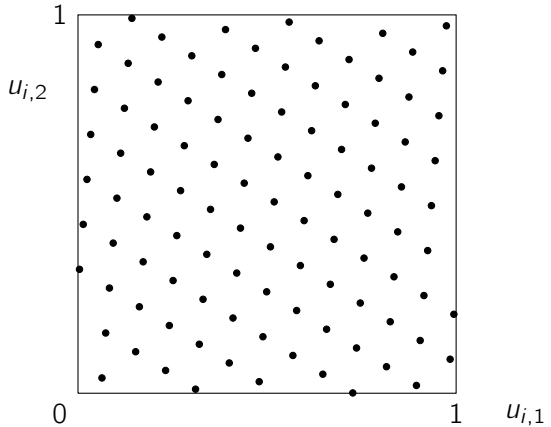
Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/101$



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the two RQMC conditions.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{v}_1 = (1, 12)/101$



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 $\boldsymbol{U} \sim U(0,1)^s$ and XOR it bitwise with each $\boldsymbol{u}_i$.

Example for $b = 2$ and $s = 2$:

$$\begin{aligned} \boldsymbol{u}_i &= (0.01100100\ldots,\ 0.10011000\ldots)_2 \\ \boldsymbol{U} &= (0.00100000\ldots,\ 0.01010001\ldots)_2 \\ \boldsymbol{u}_i \oplus \boldsymbol{U} &= (0.01000100\ldots,\ 0.11001001\ldots)_2. \end{aligned}$$

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

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 $\boldsymbol{U} \sim U(0,1)^s$ and XOR it bitwise with each $\boldsymbol{u}_i$.

Example for $b = 2$ and $s = 2$:

$$\begin{aligned} \boldsymbol{u}_i &= (0.01100100\ldots,\ 0.10011000\ldots)_2 \\ \boldsymbol{U} &= (0.00100000\ldots,\ 0.01010001\ldots)_2 \\ \boldsymbol{u}_i \oplus \boldsymbol{U} &= (0.01000100\ldots,\ 0.11001001\ldots)_2. \end{aligned}$$

Each randomized point has $U(0,1)^s$ distribution.
Preservation of the equidistribution ($k_1 = 3$, $k_2 = 5$):

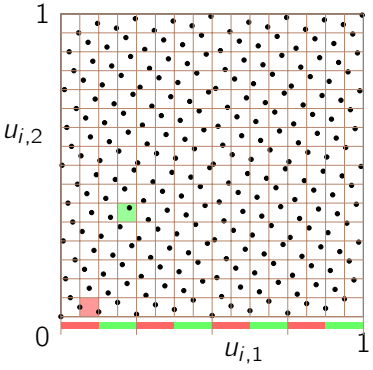
$$\begin{aligned} \boldsymbol{u}_i &= (0.***,\ 0.*****) \\ \boldsymbol{U} &= (0.001,\ 0.01010)_2 \\ \boldsymbol{u}_i \oplus \boldsymbol{U} &= (0.***,\ 0.*****) \end{aligned}$$

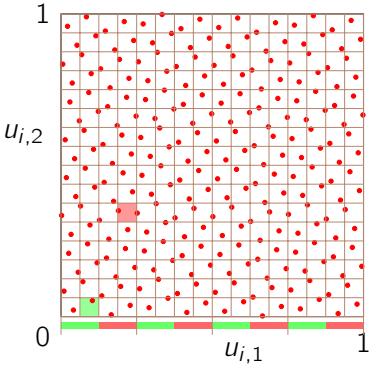
The red bits are changed for all the points. It permutes the rectangular boxes.

Example with

$$\begin{aligned} \boldsymbol{U} &= (0.1270111220,\ 0.3185275653)_{10} \\ &= (0.\ 0010\ 0000100000111100,\ 0.\ 0101\ 0001100010110000)_2. \end{aligned}$$

We flip the bits 3, 9, 15, 16, 17, 18 of $u_{i,1}$
and the bits 2, 4, 8, 9, 13, 15, 16 of $u_{i,2}$.





Random digital shift in base b

We have $u_{i,j} = \sum_{\ell=1}^{w} u_{i,j,\ell} b^{-\ell}$.
Let $\boldsymbol{U} = (U_1, \ldots, U_s) \sim U[0,1)^s$ where $U_j = \sum_{\ell=1}^{w} U_{j,\ell}\, b^{-\ell}$.
We replace each $u_{i,j}$ by $\tilde{U}_{i,j} = \sum_{\ell=1}^{w} [(u_{i,j,\ell} + U_{j,\ell}) \bmod b] b^{-\ell}$.

Proposition. $\tilde{P}_n$ is $(q_1, \ldots, q_s)$-equidistributed in base b iff P_n is.
For $w = \infty$, each point $\tilde{\boldsymbol{U}}_i$ has the uniform distribution over $(0,1)^s$.

The two following permutation methods preserve equidistribution and they both provably reduces the variance to $O(n^{-3}(\log n)^s)$ when f is sufficiently smooth.

Linear matrix scrambling (Matoušek 1998, Hickernell et Hong, Tezuka, Owen):
Apply a linear transformation to each $\boldsymbol{C}_j$. There are several variants.

Most common is left matrix srambling (LMS): For Sobol' upper-triangular matrices $\boldsymbol{C}_j$, generate random lower-triangular invertible matrices $\boldsymbol{L}_j$, to obtain the new generating matrices $\tilde{\boldsymbol{C}}_j = \boldsymbol{L}_j \boldsymbol{C}_j$. These $\tilde{\boldsymbol{C}}_j$ inherit the equidistribution properties of the $\boldsymbol{C}_j$.

$$\boldsymbol{L}_j = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ \ell_{2,1} & 1 & 0 & \dots & 0 \\ & \vdots & & \ddots & 0 \\ \ell_{k,1} & \ell_{k,2} & & & 1 \\ \ell_{k+1,1} & \ell_{k+1,2} & & & \ell_{k+1,k} \\ & \vdots & & & \vdots \\ \ell_{w,1} & \ell_{w,2} & & & \ell_{w,k} \end{pmatrix} \quad \text{and} \quad \boldsymbol{C}_j = \begin{pmatrix} 1 & v_{1,2} & \dots & v_{1,k} \\ 0 & 1 & \dots & v_{2,k} \\ \vdots & 0 & \ddots & \vdots \\ & \vdots & & 1 \end{pmatrix}.$$

This only changes the $\boldsymbol{C}_j$'s.
After that, we must apply a random digital shift in base b to get RQMC (unbiasedness).

Nested uniform scrambling (NUS) (Owen 1995).

Deeper scrambling. More costly than LMS, mostly because we must store all the points explicitly.

It was recently proved that LMS + random digital shift, with $w = \infty$, gives the same variance as NUS.

Numerical Illustrations

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 mod 1.
Sobol: left matrix scramble + random digital shift.

Variance estimated from $m = 100$ indep. randomizations.
VRF = (MC variance) / $(n\mathrm{Var}[X_{s,n}])$

method	n	$\bar{X}_m$	nS_m^2	VRF
stratif.	2^{10}	17.100	232.8	4
lattice	2^{10}	17.092	20.8	45
Sobol	2^{10}	17.094	1.66	563
stratif.	2^{16}	17.046	135.3	7
lattice	2^{16}	17.096	4.38	213
Sobol	2^{16}	17.096	0.037	25,330
stratif.	2^{20}	17.085	117.6	8
lattice	2^{20}	17.096	0.112	8,318
Sobol	2^{20}	17.096	0.0026	360,000

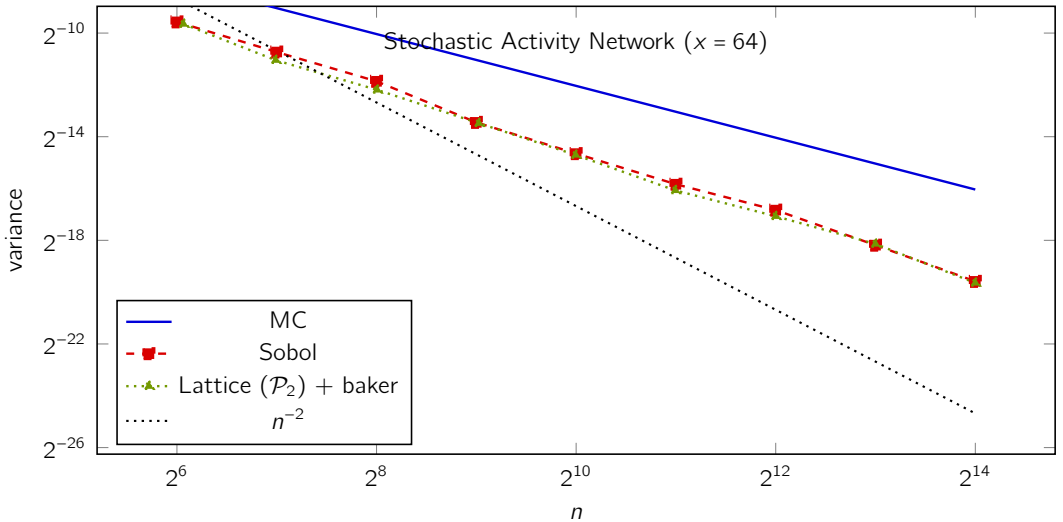
$s = d = 12$. $\mu \approx 13.122$. MC variance: **516.3**.

Lattice: Korobov + random shift.
Sobol: left matrix scramble + random digital shift.

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

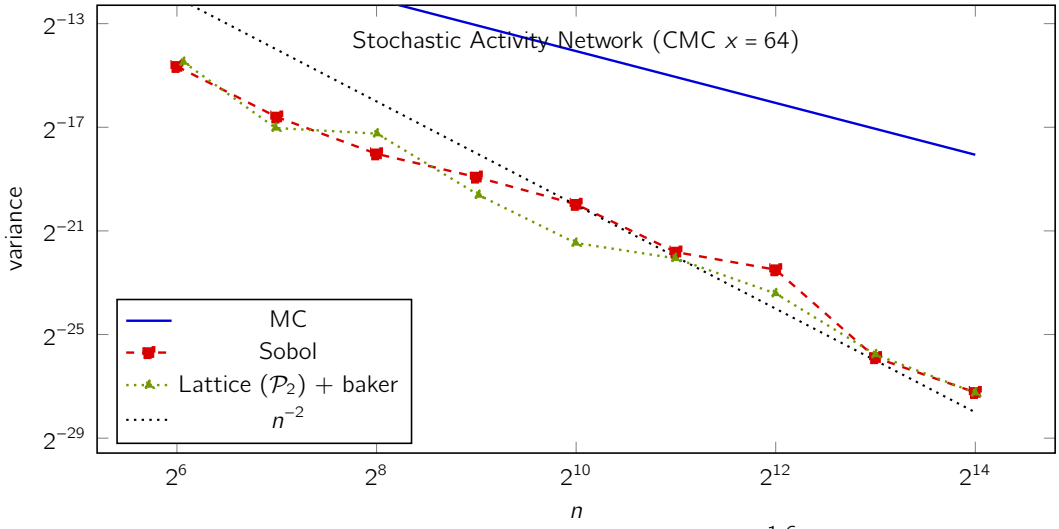
method	n	$\bar{X}_m$	nS_m^2	VRF
lattice	2^{10}	13.114	39.3	13
Sobol	2^{10}	13.123	5.9	88
lattice	2^{16}	13.122	6.61	78
Sobol	2^{16}	13.122	1.63	317
lattice	2^{20}	13.122	8.59	60
Sobol	2^{20}	13.122	0.89	579

Variance for estimator of $\mathbb{P}[T > x]$ for SAN



RQMC variance decreases roughly as $\mathcal{O}(n^{-1.2})$. For $\mathbb{E}[T]$, we observe $\mathcal{O}(n^{-1.4})$ instead.

Variance for estimator of $\mathbb{P}[T > x]$ with CMC



RQMC variance decreases roughly as $\mathcal{O}(n^{-1.6})$.