Number sequences for simulation

Giray Ökten, Florida State University

Research School on Quasi-Monte Carlo Methods and Applications, CIRM.

Number sequences for simulation

Sequences

- Random numbers - an abstraction
- Pseudorandom numbers
- Low-discrepancy sequences (u.d. mod 1, QMC, quasirandom)

Random numbers

Which one of these binary strings look random?

- 00000000000000000000
- 01101010000010011110
- 11011110011101011111

Random numbers

Which one of these binary strings look random?

- 00000000000000000000
- 01101010000010011110
- 11011110011101011111

These sequences are obtained from:

- constant string
- first 20 binary digits of $(2^{1/2} - 1)$
- actual coin tossing

Randomness as Incompressibility

00000000000000000000 $\rightarrow$ "print zero 20 times": can be coded using a little more than $\log_2 20$ bits

11011110011101011111 $\rightarrow$ no pattern: write each digit one by one, requiring 20 bits

Intuition: The second string looks more random because it is more "difficult" to generate.

Randomness as Incompressibility

00000000000000000000 $\rightarrow$ "print zero 20 times": can be coded using a little more than $\log_2 20$ bits

11011110011101011111 $\rightarrow$ no pattern: write each digit one by one, requiring 20 bits

Intuition: The second string looks more random because it is more "difficult" to generate.

The algorithmic complexity $K(x_1, x_2, ..., x_n)$ of a string $x_1, x_2, ..., x_n$ is defined as the size of the smallest input program that outputs the string, when fed to a Turing machine.

Definition (Kolmogorov, Chaitin)

A sequence is random (incompressible) if it has initial segments of high algorithmic complexity.

Randomness as Typicality

Consider infinite sequences of binary digits: 00101000101000...

For this sequence to qualify as a random sequence, we would expect the sequence to satisfy some properties:

- Relative frequencies of 0 and 1 converge to 1/2
- Relative frequencies of 00 and 01 and ...
- ...

Randomness as Typicality

Consider infinite sequences of binary digits: 00101000101000...

For this sequence to qualify as a random sequence, we would expect the sequence to satisfy some properties:

- Relative frequencies of 0 and 1 converge to 1/2
- Relative frequencies of 00 and 01 and ...
- ...

Definition (Ville)

A typical sequence (hence a random sequence) is a sequence that satisfies all properties that occur with probability 1.

Randomness as Typicality

Consider infinite sequences of binary digits: 00101000101000...

For this sequence to qualify as a random sequence, we would expect the sequence to satisfy some properties:

- Relative frequencies of 0 and 1 converge to 1/2
- Relative frequencies of 00 and 01 and ...
- ...

Definition (Ville)

A typical sequence (hence a random sequence) is a sequence that satisfies all properties that occur with probability 1.

Definition

A Martin-Löf random sequence is a sequence that satisfies all Turing computable properties that occur with probability 1.

Random numbers

Theorem (Levin, Schnorr, Chaitin)

These two notions of randomness (typicality and compressibility) are equivalent.

Random numbers

Theorem (Levin, Schnorr, Chaitin)

These two notions of randomness (typicality and compressibility) are equivalent.

Definition

A Martin-Löf random sequence is a sequence that satisfies all Turing computable properties that occur with probability 1.

Random numbers

Theorem (Levin, Schnorr, Chaitin)

These two notions of randomness (typicality and compressibility) are equivalent.

Definition

A Martin-Löf random sequence is a sequence that satisfies all Turing computable properties that occur with probability 1.

Question: Can we see an example of such a sequence?

Random numbers

Theorem (Levin, Schnorr, Chaitin)

These two notions of randomness (typicality and compressibility) are equivalent.

Definition

A Martin-Löf random sequence is a sequence that satisfies all Turing computable properties that occur with probability 1.

Question: Can we see an example of such a sequence?

Answer: No! An element of this sequence is noncomputable: there is no algorithm that generates its digits.

Pseudorandom numbers

- Martin-Löf random sequence: a sequence that passes infinitely many tests

Pseudorandom numbers

- Martin-Löf random sequence: a sequence that passes infinitely many tests
- Pseudorandom sequence: a sequence that passes finitely many (statistical) tests

Pseudorandom numbers

- Martin-Löf random sequence: a sequence that passes infinitely many tests
- Pseudorandom sequence: a sequence that passes finitely many (statistical) tests

Pseudorandom numbers

- Martin-Löf random sequence: a sequence that passes infinitely many tests
- Pseudorandom sequence: a sequence that passes finitely many (statistical) tests

Knuth: *"In practice, we apply about half a dozen different kinds of statistical tests to a sequence, and if it passes them satisfactorily we consider it to be random - it is then presumed innocent until proven guilty."*

Example: Linear Congruential Generators

Introduced by Lehmer in 1949:

$$x_n \equiv a x_{n-1} + c \mod M$$

where a is called the multiplier, c is called the shift or increment, and M is called the modulus of the generator.

Uniform random numbers u_n between 0 and 1 are obtained by

$$u_n = \frac{x_n}{M}$$

Low-discrepancy sequences

What are low-discrepancy sequences?

Low-discrepancy sequences

What are low-discrepancy sequences?

- Martin-Löf random sequence: a sequence that passes infinitely many tests

Low-discrepancy sequences

What are low-discrepancy sequences?

- Martin-Löf random sequence: a sequence that passes infinitely many tests
- Pseudorandom sequence: a sequence that passes finitely many (statistical) tests

Low-discrepancy sequences

What are low-discrepancy sequences?

- Martin-Löf random sequence: a sequence that passes infinitely many tests
- Pseudorandom sequence: a sequence that passes finitely many (statistical) tests
- Low-discrepancy sequence: a sequence that satisfies one property

Pseudorandom sequences

Pseudorandom sequences

There are several methods to generate pseudorandom sequences from the uniform distribution on $(0, 1)$. If numbers from another distribution is needed, then an appropriate transformation method needs to be used - something we will discuss later.

Remark

In practice, a pseudorandom sequence should not output 0 or 1.

Pseudorandom sequences

Linear Congruential Generators

Linear Congruential Generators

Introduced by Lehmer in 1949:

$$x_n \equiv a x_{n-1} + c \mod M$$

where a is called the multiplier, c is called the shift or increment, and M is called the modulus of the generator.

The generator produces a sequence of integers $x_1, x_2, \ldots$ between 0 and $M - 1$, once a seed x_0 is chosen.

Uniform random numbers u_n between 0 and 1 are obtained by

$$u_n = \frac{x_n}{M}$$

Lehmer first considered LCG's where $c = 0$. Such generators are called multiplicative congruential generators (MCG).

Properties of Linear Congruential Generators

What is the maximum number of integers a generator can produce?

- LCG can only produce at most M distinct integers, $\{0, 1, ..., M-1\}$ in some order.
- MCG can only produce at most $M-1$ distinct nonzero integers. We have to omit zero, for if the generator ever outputs zero, it stays at zero!

Definition

The period of a generator is defined as the number of distinct x_n before the generator starts repeating itself.

Period

Example

Consider an LCG with $M = 16, a = 5, c = 4$:

$$x_n \equiv 5x_{n-1} + 4 \mod 16$$

$$\begin{aligned} x_0 &= 0 \to 0, 4, 8, 12, 0 \\ x_0 &= 1 \to 1, 9, 1 \\ x_0 &= 3 \to 3, 3 \end{aligned}$$

Discouraging observations:

- Period may depend on the seed x_0.
- Period may be much shorter than M.

Period

A (bad) idea to obtain full period by the following choice:

$$\begin{aligned} x_n &\equiv x_{n-1} + 1 \mod M \\ x_0 &= 1 \end{aligned}$$

but clearly, what this sequence generates

$$1, 2, 3, ..., M$$

is far from a random looking sequence!

Period

Theorem (Greenberger; Hull, Dobell, 1962)

The period of the LCG $x_n \equiv ax_{n-1} + c \mod M$ *is* M *iff*

1. $gcd(c, M) = 1$ $(c \neq 0)$
2. $a \equiv 1 \mod p$ *for each prime factor* p *of* M
3. $a \equiv 1 \mod 4$ *if 4 divides* M

Example

ANSI C rand() function for Unix, BSD version.

$$M = 2^{31}, a = 1103515245, c = 12345$$

Period

Some popular LCG's that do not have full period.

- $M =$ prime; a is a primitive root modulo M ($a^{M-1} \equiv 1$ and $a^x \not\equiv 1 \mod M$ for any $x < M-1$); $c = 0$; $x_0 \neq 0$. Then the period is $M - 1$.

Example

Maple's random number generator: $M = 10^{12} - 11$, $a = 427419669081$, $c = 0$

Another commonly used generator: $M = 2^{31} - 1$ (Mersenne prime); $a = 7^5$; $c = 0$

Period

- $M = 2^N$; $a \equiv 5 \mod 8$; $c = 0$; x_0 is odd. Then the period is $M/4 = 2^{N-2}$.

Example

Cray Supercomputers. $M = 2^{48}$; $a = 44485709377909$. The seed specifies the lower 32 bits of x_0, with the lowest bit set to prevent the seed taking an even value. The upper 16 bits are set to 0.

A property of LCG's

Theorem (Marsaglia, '68)

The output of any LCG lies on a simple lattice in a k-space with axes representing successive numbers in the output.

A property of LCG's

Theorem (Marsaglia, '68)

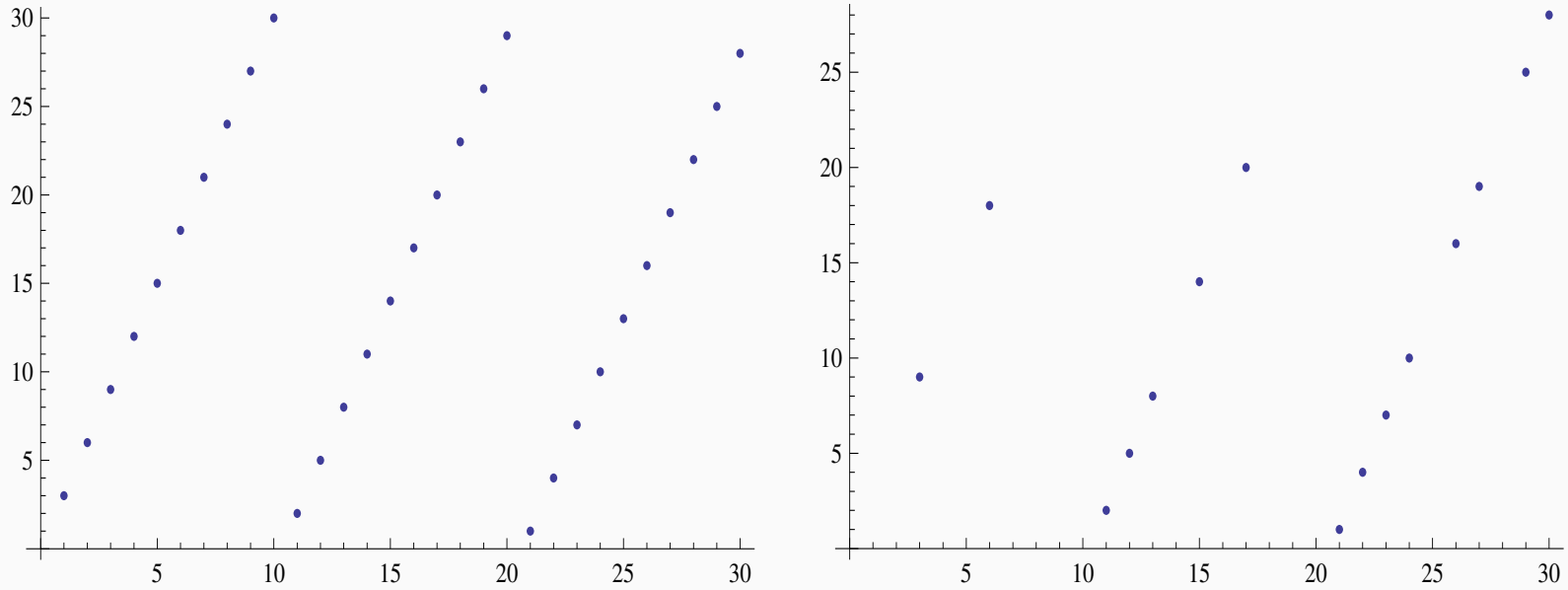
The output of any LCG lies on a simple lattice in a k-space with axes representing successive numbers in the output.

Example

Consider the simple MCG with $M = 31, a = 3$ and $x_0 = 9$. The period is 30, and 3 is a primitive root modulo 31.The numbers are $\{27, 19, 26, 16, 17, 20, 29, 25, 13, 8, 24, 10, 30, 28, 22, 4, 12, 5, 15, 14, 11, 2, 6, 18, 23, 7, 21, 1, 3, 9, \mathbf{27}\}$.

This output may look like a random sample from the discrete uniform distribution over the integers 1 to 30. However, if we partition the list in an overlapping way, (27,19),(19,26),..., and in a non-overlapping way, (27,19),(26,16)..., and plot the pairs we obtain to following figure:

A property of LCG's



Overlapping (left) and non overlapping (right) partitions

RANDU

A faulty MCG which was widely used in 1960's, on IBM 360.370.

$$\begin{aligned} x_n &\equiv 65539 x_{n-1} \mod 2^{31} \\ &\equiv (65539)^2 x_{n-2} \\ &\equiv (2^{16}+3)^2 x_{n-2} \\ &\equiv 2^{32} x_{n-2} + 6 \cdot 2^{16} x_{n-2} + 9 x_{n-2} \\ &\equiv 6(\underbrace{2^{16} x_{n-2} + 3 x_{n-2}}_{65539 x_{n-2} \equiv x_{n-1}}) - 9 x_{n-2} \\ &\equiv 6 x_{n-1} - 9 x_{n-2} \end{aligned}$$

therefore we get $x_n - 6x_{n-1} + 9x_{n-2} = k2^{31}$ for some integer k. Divide both sides by 2^{31} to get

$$u_n - 6u_{n-1} + 9u_{n-2} = k$$

where $0 < u_i < 1$. Then, $k < 10$ and $k > -6$, and thus the triplets must lie on no more than 15 planes in R^3.

Pseudorandom sequences

Congruential Generators with Dependence on More Terms

Quadratic generators

They have the form

$$x_n \equiv a x_{n-1}^2 + b x_{n-1} + c \mod M$$

An example is a generator by R. Coveyou who suggested

$$\begin{aligned} x_n &\equiv x_{n-1}(x_{n-1}+1) \mod 2^e \\ x_0 &\equiv 2 \mod 4 \end{aligned}$$

Lagged Fibonacci generators

The Fibonacci generator $x_n \equiv x_{n-1} + x_{n-2} \mod M$ is not a good generator; it fails several statistical tests. However, a modification is commonly used today:

$$x_n \equiv x_{n-r} + x_{n-s} \mod M$$

where $n \geq r, 0 < s < r$, and r & s are the lags. The low-order bits of these generators are not very random if lags are small.

Another generalization is to replace addition by another binary operation, such as $-$, $\times$, or exclusive or if $M = 2^e$ (the operation then is equivalent to addition mod 2)

Lagged Fibonacci generators

Example

$r = 607; s = 273; m = 2^{31}$; operation is $+$, i.e.,

$$x_n \equiv x_{n-607} + x_{n-273} \mod 2^{31}$$

has period $(2^{607} - 1)2^{31-1} \approx 10^{191}$, if at least one member of the initial sequence $x_0, ..., x_{606}$ is odd. There are other pairs of lags (r, s) that have period $(2^r - 1)2^{31-1}$.

Multiple Recursive generators

These have the form

$$x_n \equiv a_1 x_{n-1} + \ldots + a_k x_{n-k} \mod p$$

where p is prime. It is possible to find $a_1, \ldots, a_k$ such that the sequence has period length $p^k - 1$. Initial values $x_0, \ldots, x_{k-1}$ can be chosen arbitrarily, but not all zero. A special case, $p = 2$, is commonly used today.

Feedback Shift Register Generators

Introduced by Tausworthe, 1965. Consider the polynomial

$$f(z) = z^k - (a_1 z^{k-1} + ... + a_{k-1} z + a_k)$$

over the field defined over the integers $\{0, 1\}$. Then, the period of

$$x_n \equiv a_1 x_{n-1} + ... + a_k x_{n-k} \mod 2$$

is $2^k - 1$ iff the polynomial is irreducible (provided not all initial values x_i are zero.)

Feedback Shift Register Generators

The sequence of binary digits x_n, $n = 1, \ldots$, is then partitioned into blocks, and each block is transformed to an integer. For example, if blocks of size 4 are formed $[0, 1, 1, 0], [1, 1, 1, 0], \ldots$, then the first block is converted to an integer as $0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 0$.

If the partition into blocks is done with a delay between blocks, then we obtain the so-called generalized feedback shift register generators (GFSR). A good example is tt800 and Mersenne Twister (these generators have a further modification of the basic FSR generators.)

Pseudorandom sequences

Other non-linear congruential generators

Inversive Congruential Generators

Introduced by Eichenauer & Lehn (1986). They have the form

$$x_{n+1} \equiv a x_n^{-1} + c \mod M$$

where x_n^{-1} is the multiplicative inverse of $x_n \mod M$, if it exists, or else it is zero. These generators do not yield regular planes like LCG's, and several tests of randomness reveal their superior properties over LCG's. However, they are computationally demanding.

Other examples:

$$x_{n+1} \equiv g(x_n) \mod M$$

where g is some function. Examples are

1. $g(x) = x^2$; (Blum, Blum, Shub, 1986)
2. $g(x) = ax^{-1} + bx + c$; (Kato, Wu, Yanagihara, 1996)

Combining Generators

Wichmann and Hill (1982, 84) introduced the following generator

$$\begin{aligned} x_n &\equiv 171 x_{n-1} \mod 30269 \\ y_n &\equiv 172 y_{n-1} \mod 30307 \\ z_n &\equiv 170 z_{n-1} \mod 30323 \end{aligned}$$

and

$$u_n \equiv \frac{x_n}{30269} + \frac{y_n}{30307} + \frac{z_n}{30323} \mod 1$$

The period of this generator is 10^{12}.

Combining Generators

L'Ecuyer suggested the following generator

$$\begin{aligned} x_n &\equiv 40014 x_{n-1} \mod 2147483563 \\ y_n &\equiv 40692 y_{n-1} \mod 2147483399 \\ z_n &\equiv x_n - y_n \end{aligned}$$

where if $z_n < 1$ then $z_n = z_n + 2147483562$, and

$$u_n = 4.656613 z_n \times 10^{-10}.$$

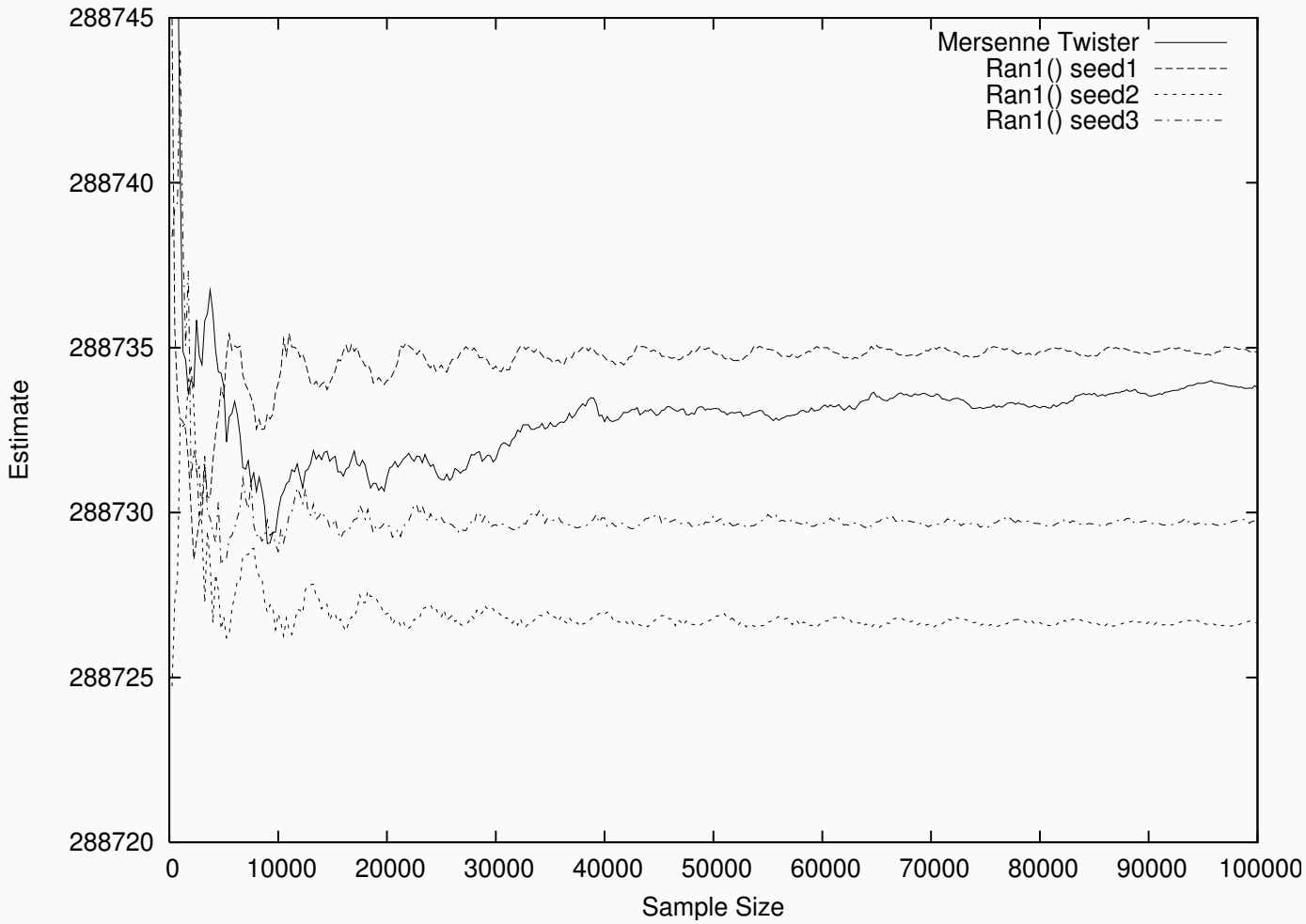
The period of this generator is about 10^{18}.

See Glasserman's book (pages 52, 53) for a C code for another combined generator suggested by L'Ecuyer.

A recent faulty generator

A combined random number generator, ran1(), that was given in Numerical Recipes in C, First Edition (Press et. al. 1988), used two LCG's to construct a uniform number, and then a third LCG to shuffle the order numbers are generated. However, after this generator was applied to pricing collateralized mortgage obligations (CMO), it was discovered to be faulty. The generator has a cycle of $360 \times 10,800$, and since CMO is a 360-dimensional problem, there are only 10,800 different paths. The following graph shows how estimates obtained from Ran1 converges to different values for different seeds. The estimates given by Mersenne Twister can be taken as the "true" price of the CMO.

A recent faulty generator



Ran1() with different seeds and Mersenne Twister. 360 months

Suggestions

Use at least two different types of well tested random number generators. Having the Mersenne Twister and one of the combined generators in your library is probably a good idea. Subtle correlations in a generator can always cause problems in a particular problem.

Low-discrepancy sequences

Low-discrepancy sequences

Definition

A sequence $x_1, x_2, \ldots$ in $I = [0, 1)$ is said to be uniformly distributed modulo 1 (u.d. mod 1) if

$$\lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} 1_{[a,b)}(x_n) = b - a$$

for any $0 \leq a < b \leq 1$.

This definition is easily generalized to higher dimensions:

$$I = [0, 1) \to I^s = [0, 1)^s$$

$$[a, b) \to \prod_{i=1}^{s} [a_i, b_i)$$

Low-discrepancy sequences

Why are these sequences important in applications?

Theorem

A sequence $x_1, x_2, \ldots$ in I^s is u.d. mod 1 iff

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(x_n) = \int_{I^s} f(x)dx$$

for all Riemann integrable functions f on I^s.

How to measure the quality of a sequence?

Given a sequence $x_1, \ldots, x_N$ in $[0,1)^s$, its extreme discrepancy is

$$D_N(x_n) = \sup_{[a,b)} \left| \frac{1}{N} \sum_{n=1}^{N} 1_{[a,b)}(x_n) - \lambda_s([a,b)) \right|$$

If the supremum is taken over all intervals of the form $[0,a)$ then we obtain star-discrepancy

$$D_N^*(x_n) = \sup_{[0,a)} \left| \frac{1}{N} \sum_{n=1}^{N} 1_{[0,a)}(x_n) - \lambda_s([0,a)) \right|$$

How to measure the quality of a sequence?

Given a sequence $x_1, \ldots, x_N$ in $[0,1)^s$, its extreme discrepancy is

$$D_N(x_n) = \sup_{[a,b)} \left| \frac{1}{N} \sum_{n=1}^{N} 1_{[a,b)}(x_n) - \lambda_s([a,b)) \right|$$

If the supremum is taken over all intervals of the form $[0,a)$ then we obtain star-discrepancy

$$D_N^*(x_n) = \sup_{[0,a)} \left| \frac{1}{N} \sum_{n=1}^{N} 1_{[0,a)}(x_n) - \lambda_s([0,a)) \right|$$

Theorem

$D_N(x_n) \to 0$ iff $\{x_n\}$ is a u.d. mod 1 sequence.

Remark

Sequences with best known bounds for star-discrepancy satisfy $O((\log N)^s / N)$. (The term "low-discrepancy" sequence is used for these sequences.)

Low-discrepancy sequences

Halton, permuted Halton, Faure, and (t, s) sequences

van der Corput and Halton sequences

The van der Corput sequence $\phi_2(n), n = 1, 2, ...$ in base 2 is defined as follows:

n	base 2	inversion	van der Corput seq
0	0	0.0	0
1	1	0.1	1/2
2	10	0.01	1/4
3	11	0.11	3/4
...			

van der Corput and Halton sequences

The van der Corput sequence $\phi_2(n), n = 1, 2, ...$ in base 2 is defined as follows:

n	base 2	inversion	van der Corput seq
0	0	0.0	0
1	1	0.1	1/2
2	10	0.01	1/4
3	11	0.11	3/4
...			

Formally, for any base b, if

$$n = (a_k \cdots \ a_1 a_0)_b = a_0 + a_1 b + ... + a_k b^k$$

then

$$\phi_b(n) = (.a_0 a_1 \cdots a_k)_b = \frac{a_0}{b} + \frac{a_1}{b^2} + ... + \frac{a_k}{b^{k+1}}$$

Halton sequence

The Halton sequence is a generalization of the van der Corput sequence to higher dimensions. The s-dimensional Halton sequence in the bases $b_1, ..., b_s$ is defined as

$$(\phi_{b_1}(n), ..., \phi_{b_s}(n))_{n=1}^{\infty}$$

Theorem

The Halton sequence is u.d. mod 1 if its bases $b_1, ..., b_s$ are relatively prime.

Halton sequence

Example

The 2-dimensional Halton sequence in bases 2 and 3.

van der Corput in base 2

$$\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \cdots$$

van der Corput in base 3

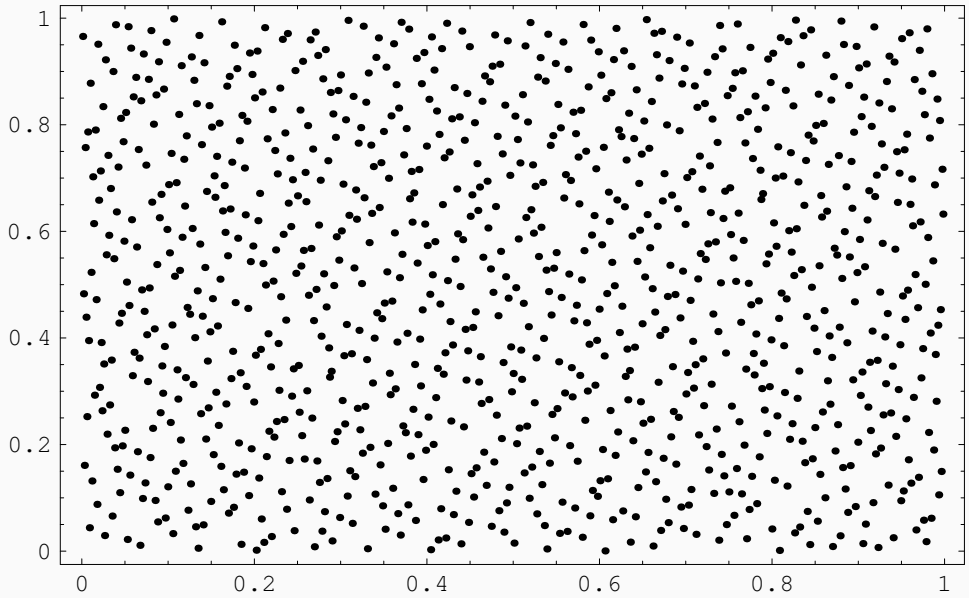
$$\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \cdots$$

Halton sequence in bases 2 and 3:

$$\left(\frac{1}{2}, \frac{1}{3}\right), \left(\frac{1}{4}, \frac{2}{3}\right), \left(\frac{3}{4}, \frac{1}{9}\right), \left(\frac{1}{8}, \frac{4}{9}\right) \cdots$$

Halton sequence

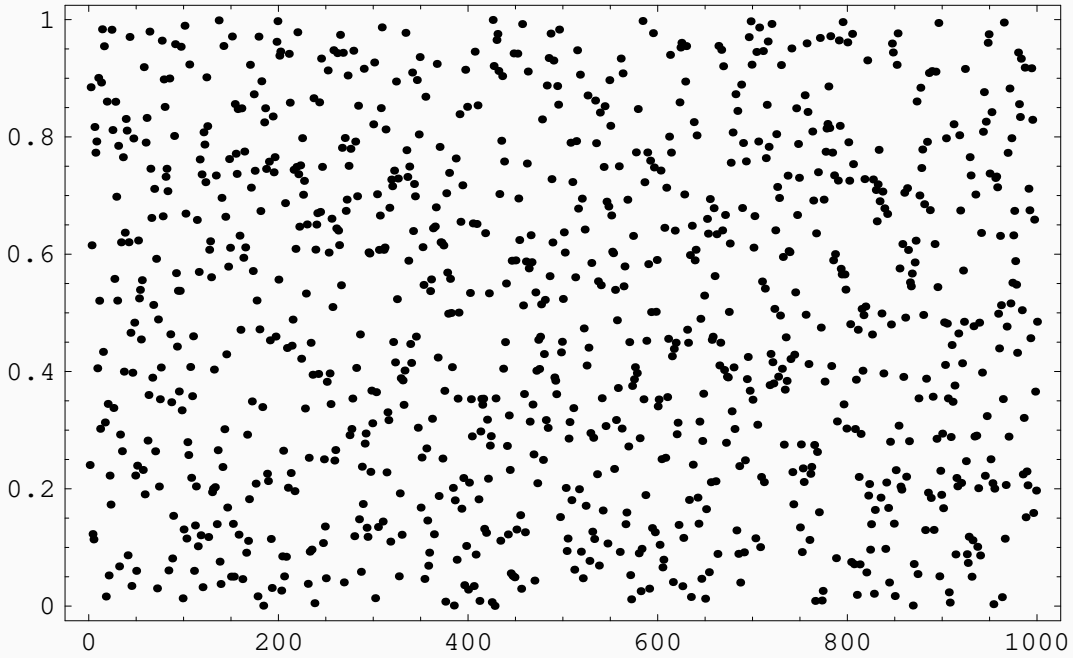
A plot of 2-dimensional Halton vectors when the bases are 2 and 3.



The reason u.d. mod 1 sequences can be used to approximate integrals of functions, or global extrema values of functions is because they fill the space in a "uniform" way.

Pseudorandom numbers

A plot of 2-dimensional pseudorandom vectors.

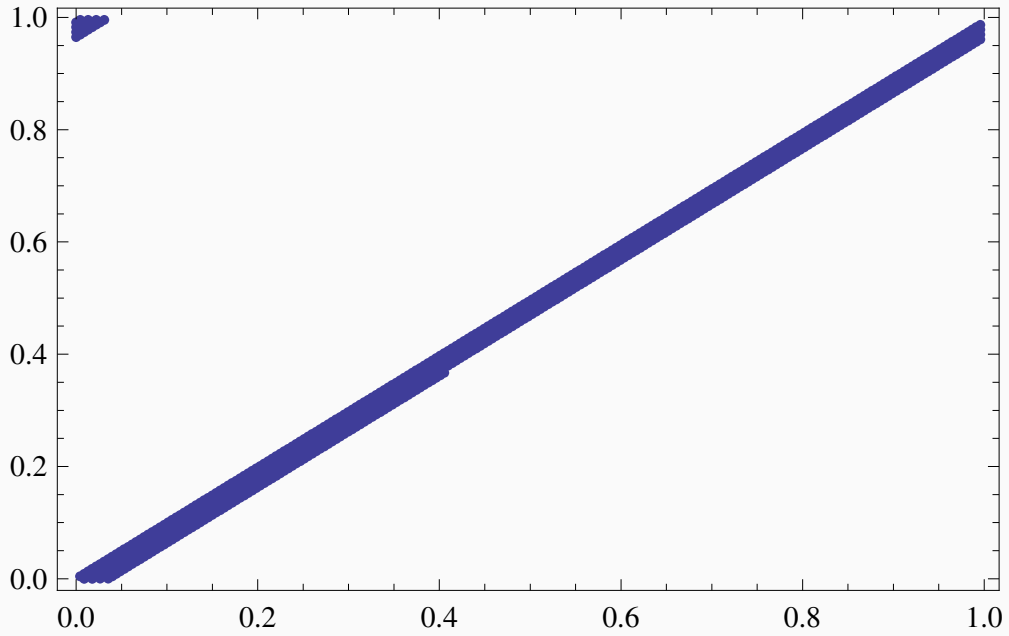


Halton sequence

Another picture: First 1000 vectors of the Halton sequence in bases 227 and 229 (49th and 50th primes)

Halton sequence

Another picture: First 1000 vectors of the Halton sequence in bases 227 and 229 (49th and 50th primes)



This phenomenon is usually called "high correlation between higher bases"; first observed by Braaten & Weller, and Faure in late 1970s.

A remedy: permuted van der Corput and Halton sequences

van der Corput:

$$\phi_b(n) = \frac{a_0}{b} + \frac{a_1}{b^2} + \ldots + \frac{a_k}{b^{k+1}}$$

Permuted van der Corput:

$$\phi_b(n) = \frac{\sigma_b(a_0)}{b} + \frac{\sigma_b(a_1)}{b^2} + \ldots + \frac{\sigma_b(a_k)}{b^{k+1}}$$

where σ_b is a permutation on the digit set $\{0, \ldots, b-1\}$.

A further generalization:

$$\phi_b(n) = \frac{\sigma_{b,1}(a_0)}{b} + \frac{\sigma_{b,2}(a_1)}{b^2} + \ldots + \frac{\sigma_{b,k+1}(a_k)}{b^{k+1}}$$

where different permutations are used for each digit.

Permuted Halton sequences are obtained from these in the usual way.

Where do we find good permutations?

- Several permutations are suggested in the literature: Braaten & Weller, Atanassov, Faure, Chi & Mascagni & Warnock, Faure & Lemieux, Kocis & Whitten, Tuffin, Vandewoestyne & Cools, Warnock, Ökten & Shah.

Where do we find good permutations?

- Several permutations are suggested in the literature: Braaten & Weller, Atanassov, Faure, Chi & Mascagni & Warnock, Faure & Lemieux, Kocis & Whitten, Tuffin, Vandewoestyne & Cools, Warnock, Ökten & Shah.
- How are the permutations obtained?

Where do we find good permutations?

- Several permutations are suggested in the literature: Braaten & Weller, Atanassov, Faure, Chi & Mascagni & Warnock, Faure & Lemieux, Kocis & Whitten, Tuffin, Vandewoestyne & Cools, Warnock, Ökten & Shah.
- How are the permutations obtained?
  - Heuristics

Where do we find good permutations?

- Several permutations are suggested in the literature: Braaten & Weller, Atanassov, Faure, Chi & Mascagni & Warnock, Faure & Lemieux, Kocis & Whitten, Tuffin, Vandewoestyne & Cools, Warnock, Ökten & Shah.
- How are the permutations obtained?
  - Heuristics
  - Searching for the optimal permutations that minimize the discrepancy of the one-dimensional or two-dimensional projections

(t, m, s) nets, (t, s) sequences, Faure sequence

Definition

Let $s \geq 1, b \geq 2$ be integers. An elementary interval in base b is a subinterval of $[0,1)^s$ of the form

$$\prod_{j=1}^{s} \left[\frac{a_j}{b^{d_j}}, \frac{a_j + 1}{b^{d_j}} \right)$$

where integers a_j, d_j satisfy $d_j \geq 0$ and $0 \leq a_j < b^{d_j}$.

Example

Elementary intervals in base $b = 3$ and $s = 1$ are:

$$\begin{aligned} d_1 &= 0 \Rightarrow [0,1) \\ d_1 &= 1 \Rightarrow [0,1/3), [1/3,2/3), [2/3,1) \\ d_1 &= 2 \Rightarrow [0,1/9), [1/9,2/9), [2/9,3/9), \ldots \\ &\ldots \end{aligned}$$

Example

Elementary intervals in base $b = 2$ and $s = 2$ are:

- $d_1 = 0, d_2 = 0 \Rightarrow [0, 1) \times [0, 1)$
- $d_1 = 1, d_2 = 1 \Rightarrow [0, 1/2) \times [0, 1/2)$ and $[1/2, 1) \times [1/2, 1)$ and $[0, 1/2) \times [1/2, 1)$ and $[1/2, 1) \times [0, 1/2)$
- $d_1 = 1, d_2 = 2 \Rightarrow$
  $[0, 1/2) \times [0, 1/4)$ and $[0, 1/2) \times [1/4, 3/4)$ $\cdots$
  and $[1/2, 1) \times [0, 1/4)$ and $[1/2, 1) \times [1/4, 3/4)$ $\cdots$

(t, m, s) nets

Definition

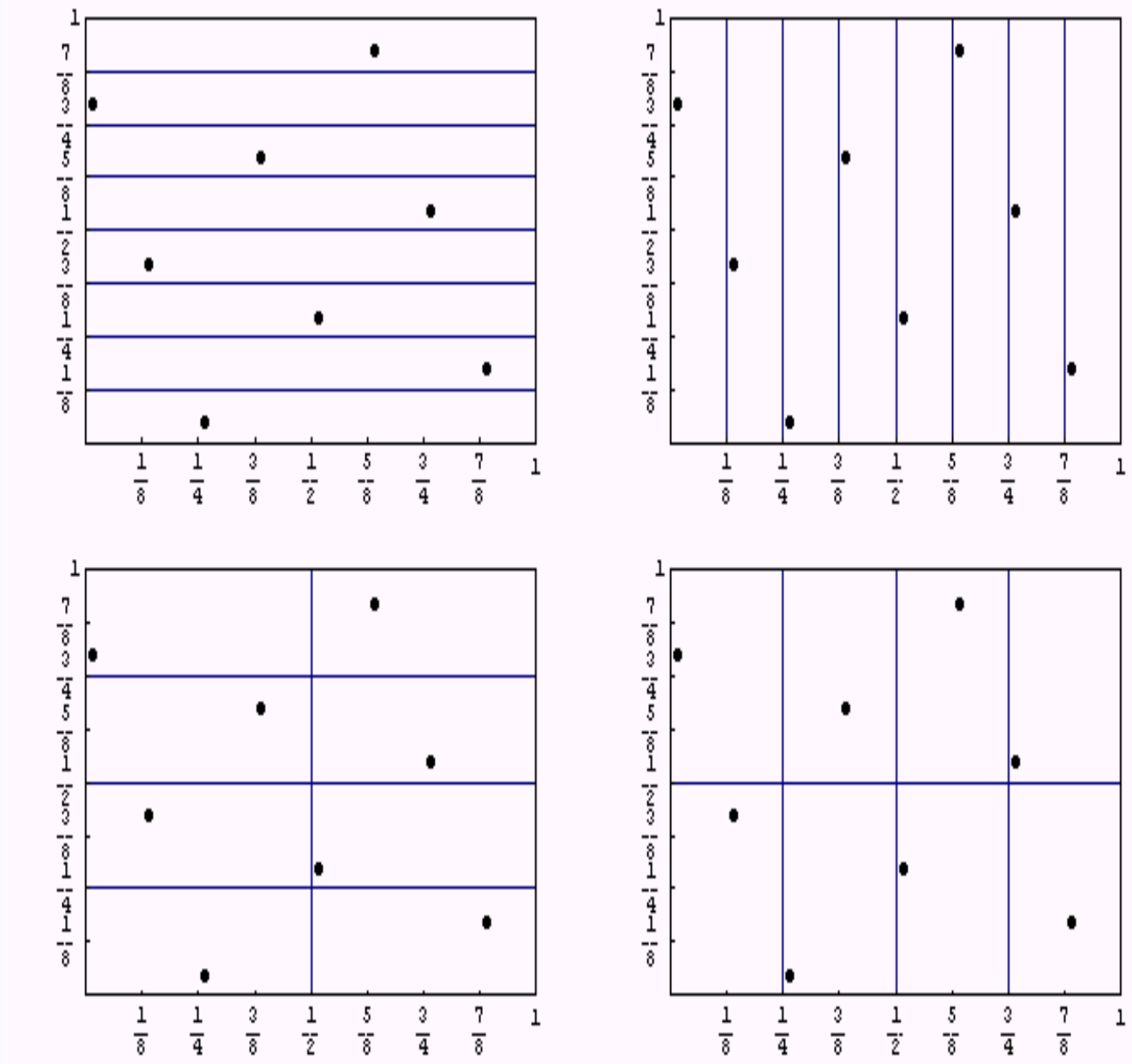
For $0 \leq t \leq m$, a finite sequence of b^m points in $[0, 1)^s$ is a (t, m, s)-net in base b if every elementary interval in base b of volume b^{t-m} contains exactly b^t points of the sequence.

Example

Let $t = 0, m = 3, s = 2$, and $b = 2$. A $(0, 3, 2)$-net in base 2 is

- a collection of $b^m = 2^3 = 8$ points (vectors) in dimension $s = 2$
- such that every elementary interval of area $b^{-m} = 2^{-3} = 1/8$ contains exactly one point

(0,3,2)-net in base 2 with elementary intervals of area 1/8



(t, s) sequences

Definition

An infinite sequence of points $q_1, q_2, \ldots$ is called a (t, s)-sequence in base b if the finite sequence $q_{kb^m+1}, \ldots, q_{(k+1)b^m}$ is a (t, m, s)-net in base b for all $k \geq 0$ and $m \geq t$.

Example

The Faure sequence is an example of a *digital* (t, s) *sequence,* with $t = 0$. The word "digital" refers to a special construction of (t, s)-sequences. The base of the Faure sequence is the smallest prime number greater than or equal to s.

Constructing the Faure sequence

Let $\boldsymbol{q}$ be an s-dimensional digital (t, s)-sequence in base b

$$\boldsymbol{q} : \begin{bmatrix} q_n^{(1)} \\ \vdots \\ q_n^{(s)} \end{bmatrix}, n = 1, 2, \ldots.$$

To describe how $q_n^{(i)}$, $i = 1, \ldots, s$, is computed, we will first write it in its base b expansion:

$$q_n^{(i)} = (0.c_1 c_2 \ldots c_k)_b.$$

Now assume the base b expansion of n is:

$$n = (a_k \ldots a_2 a_1)_b.$$

Constructing the Faure sequence

The digits c_j are obtained from the digits a_j by the following matrix vector product, done in modular arithmetic mod b:

$$\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} = \left[C^{(i)} \right] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{bmatrix}$$

where $C^{(i)}$, $i = 1, 2, \ldots, s$, are square matrices of dimension k, and called the *generator matrices* of the sequence.

Constructing the Faure sequence

The generator matrix for the Faure sequence is

$$\mathbf{C}^{(k)} = \mathbf{P}^{k-1}, \ k = 1, ..., s$$

where $\mathbf{P}$ is the $l \times l$ Pascal matrix mod p: this is an upper triangular matrix where the ijth entry is the binomial term $\binom{j-1}{i-1}$ mod p. For example, the 3×3 Pascal matrix is $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$.

Remark

Powers of the Pascal matrix can be computed without matrix multiplication as follows. Define $P(a)$ as the upper triangular matrix with the ijth entry equal to $a^{j-i}\binom{j-1}{i-1}$. Then we have the following useful identity: $\mathbf{P}^k = P(k)$ for any integer k (adopting the convention $0^0 = 1$ so that $P^0 = P(0) = I$.)

Low-discrepancy sequences

Numerical integration, Koksma-Hlawka, uniform point sets

Numerical integration

Consider the problem of estimating

$$\theta = \int_{I^s} f(x)dx$$

In both Monte Carlo and quasi-Monte Carlo, we estimate the integral by averages of the form:

$$\theta_N = \frac{1}{N}\sum_{n=1}^{N} f(x_n)$$

- In Monte Carlo, x_n are pseudorandom numbers, and $\theta_N \to \theta$ as $N \to \infty$ a.s. The error $\theta_N - \theta$ is a normal random variable with zero mean and variance $\left(\int_{I^s} f^2(x)dx - \theta^2\right)/N$

Numerical integration

Consider the problem of estimating

$$\theta = \int_{I^s} f(x)dx$$

In both Monte Carlo and quasi-Monte Carlo, we estimate the integral by averages of the form:

$$\theta_N = \frac{1}{N}\sum_{n=1}^{N} f(x_n)$$

- In Monte Carlo, x_n are pseudorandom numbers, and $\theta_N \to \theta$ as $N \to \infty$ a.s. The error $\theta_N - \theta$ is a normal random variable with zero mean and variance $\left(\int_{I^s} f^2(x)dx - \theta^2\right)/N$
- In QMC, x_n come from an $s-$dimensional u.d. mod 1 sequence, and $\theta_N \to \theta$ as $N \to \infty$. Now we discuss the error of this approximation, $\theta_N - \theta$.

Koksma-Hlawka inequality

Theorem

If f has bounded variation $V(f)$ in the sense of Hardy and Krause over $[0,1]^s$, then, for any $x_1, ..., x_N \in [0,1)^s$, we have

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(x_n) - \int_{I^s} f(x) dx \right| \leq V(f) D_N^*(x_n)$$

This inequality shows that QMC error bound (worst case) is $O((\log N)^s / N)$, and smaller error is possible if we can make $V(f)$ or $D_N^*(x_n)$ smaller.

Variation in the sense of Hardy & Krause

- $f(x_1, ..., x_s) \rightarrow$ real valued function on $[0, 1]^s$

Variation in the sense of Hardy & Krause

- $f(x_1, ..., x_s) \rightarrow$ real valued function on $[0, 1]^s$
- $P \rightarrow$ a partition of $[0, 1]^s$, i.e., a set of s finite sequences

$$P : \nu_0^{(j)}, \nu_1^{(j)}, ..., \nu_{m(j)}^{(j)} \text{ for } j = 1, ..., s$$

with $0 = \nu_0^{(j)} \leq \nu_1^{(j)} \leq ... \leq \nu_{m(j)}^{(j)} = 1$

Variation in the sense of Hardy & Krause

- $f(x_1, ..., x_s) \rightarrow$ real valued function on $[0,1]^s$
- $P \rightarrow$ a partition of $[0,1]^s$, i.e., a set of s finite sequences

$$P : \nu_0^{(j)}, \nu_1^{(j)}, ..., \nu_{m(j)}^{(j)} \text{ for } j = 1, ..., s$$

with $0 = \nu_0^{(j)} \leq \nu_1^{(j)} \leq ... \leq \nu_{m(j)}^{(j)} = 1$

- Δ_j (jth difference operator) $\rightarrow$

$$\Delta_j f(\cdot ... \cdot, v_i^{(j)}, \cdot ... \cdot) = f(\cdot ... \cdot, v_{i+1}^{(j)}, \cdot ... \cdot) - f(\cdot ... \cdot, v_i^{(j)}, \cdot ... \cdot)$$

and let $\Delta_{j_1 ... j_p} = \Delta_{j_1} ... \Delta_{j_p}$

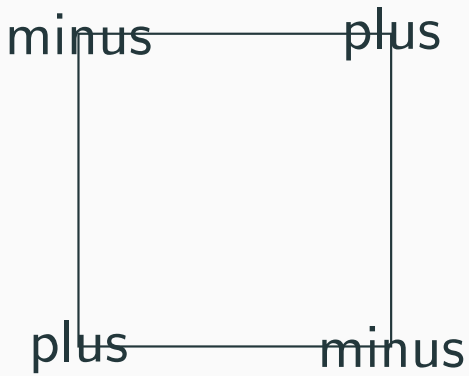
Variation in the sense of Hardy & Krause

Example

$s = 2$ and $P : \nu_0^{(1)}, \nu_1^{(1)}$, and $\nu_0^{(2)}, \nu_1^{(2)}$

$$\Delta_{12} f(\nu_0^{(1)}, \nu_0^{(2)}) = \Delta_1(\Delta_2 f(\nu_0^{(1)}, \nu_0^{(2)})) = \Delta_1(f(\nu_0^{(1)}, \nu_1^{(2)}) - f(\nu_0^{(1)}, \nu_0^{(2)}))$$

$$= f(\nu_1^{(1)}, \nu_1^{(2)}) - f(\nu_0^{(1)}, \nu_1^{(2)}) - f(\nu_1^{(1)}, \nu_0^{(2)}) + f(\nu_0^{(1)}, \nu_0^{(2)})$$



Variation in the sense of Hardy & Krause

The variation of f in the sense of **Vitali** is

$$V^s(f) = \sup_P \sum_{i_1=0}^{m(1)-1} \dots \sum_{i_s=0}^{m(s)-1} \left| \Delta_{1,\dots,s} f(v_{i_1}^{(1)}, \dots, v_{i_s}^{(s)}) \right|$$

The variation of f in the sense of **Hardy and Krause** is

$$V(f) = \sum_{k=1}^{s} \sum_{1 \le i_1 < i_2 < \dots < i_k \le s} V^k(f; i_1, \dots, i_k)$$

where $V^k(f; i_1, \dots, i_k)$ is the variation in the sense of Vitali of the restriction of f to the k-dimensional face

Drawbacks of Koksma-Hlawka inequality

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(x_n) - \int_{I^s} f(x) dx \right| \leq V(f) D_N^*(x_n)$$

- For a given f, the inequality can be loose
- In general, $V(f)$ and $D_N^*(x_n)$ are not known exactly, and need to be numerically estimated
- Variation in the sense of H&K is difficult to work with

Example

A simple function like

$$f(x, y) = \begin{cases} 1, x \geq y \\ 0, x < y \end{cases}$$

is not of finite bounded variation!

Uniform point sets: a generalization of (t, m, s)-nets

Let

- $(X, \mathcal{B}, \mu) \rightarrow$ a general probability space
- $\mathcal{M} \rightarrow$ a collection of subsets of X, i.e., a subset of $\mathcal{B}$
- $\mathcal{P} = \{x_1, \ldots, x_N\} \rightarrow$ a point set of elements of X
- $A(M; \mathcal{P}) \rightarrow$ counts how many elements of $\mathcal{P}$ belong to M

then, we say $\mathcal{P}$ is $(\mathcal{M}, \mu)$-uniform if for all $M \epsilon \mathcal{M}$,

$$\frac{A(M; \mathcal{P})}{N} = \mu(M)$$

Example

A (t, m, s)-net in base b is a $(\mathcal{M}, \mu)$-uniform point set where μ is the Lebesgue measure on $(0, 1)^s$, and $\mathcal{M}$ is collection of all elementary intervals in base b of volume b^{t-m}.

Error bounds for uniform point sets

Consider a special case where $\mathcal{M} = \{M_1, ..., M_k\}$ is a partition of X. Then

Theorem

For any $(\mathcal{M}, \mu)$-uniform point set $\mathcal{P} = \{x_1, \ldots, x_N\}$ and any bounded μ integrable function f on X we have

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(x_n) - \int_X f d\mu \right| \leq \sum_{j=1}^{k} \mu_j(M_j)(\sup_{t \in M_j} f(t) - \inf_{t \in M_j} f(t))$$

Randomized quasi-Monte Carlo methods

Randomized quasi-Monte Carlo

- Want a method that will enable the use of statistical methods to analyze error in the context of u.d. mod 1 sequences

Randomized quasi-Monte Carlo

- Want a method that will enable the use of statistical methods to analyze error in the context of u.d. mod 1 sequences
- Let β be a low-discrepancy sequence. *Randomize* β in order to obtain *independent* copies $\beta_1, ..., \beta_m$

Randomized quasi-Monte Carlo

- Want a method that will enable the use of statistical methods to analyze error in the context of u.d. mod 1 sequences
- Let β be a low-discrepancy sequence. *Randomize* β in order to obtain *independent* copies $\beta_1, ..., \beta_m$
- If the problem is to estimate $\theta = \int_{I^s} f(x)dx$, then each β_i gives an estimate, θ_i. If the randomization is done properly, we will estimate θ using the sample mean

$$\hat{\theta} = \frac{1}{m}\sum_{i=1}^{m}\theta_i$$

and estimate error by sample variance,

$$Var(\hat{\theta}) = \frac{\sum_{i=1}^{m}(\theta_i - \theta)^2}{m-1}$$

Randomized quasi-Monte Carlo

In randomized quasi-Monte Carlo, we have

- $\beta_u \rightarrow$ a family of s-dimensional sequences indexed by the random parameter u
- $Q(\beta_u) = \frac{1}{N}\sum_{n=1}^{N} f(\beta_u^{(n)}) \rightarrow$ corresponding random quadrature rule
- The final estimate of the integral is

$$\frac{Q(\beta_{u_1}) + \ldots + Q(\beta_{u_m})}{m}$$

Randomized quasi-Monte Carlo

In randomized quasi-Monte Carlo, we have

- $\beta_u \rightarrow$ a family of s-dimensional sequences indexed by the random parameter u
- $Q(\beta_u) = \frac{1}{N}\sum_{n=1}^{N} f(\beta_u^{(n)}) \rightarrow$ corresponding random quadrature rule
- The final estimate of the integral is

$$\frac{Q(\beta_{u_1}) + ... + Q(\beta_{u_m})}{m}$$

Most RQMC methods satisfy:

- $E[Q(\beta_u)] = \theta$
- $Var(Q(\beta_u)) = O\left(\frac{(\log N)^{2s}}{N^2}\right)$, or better

Randomized quasi-Monte Carlo

Some of the RQMC methods are:

1. Random shifting
2. Scrambled nets and sequences
3. Digit scrambling, and linear scrambling of nets and sequences
4. Random-start Halton sequences

We discuss these methods next.

Randomized quasi-Monte Carlo methods

Random shifting

Random shifting

Consider the first six Halton vectors in dimension 2, using bases 2 and 3:

$$\begin{bmatrix} 1/2 \\ 1/3 \end{bmatrix}, \begin{bmatrix} 1/4 \\ 2/3 \end{bmatrix}, \begin{bmatrix} 3/4 \\ 1/9 \end{bmatrix}, \begin{bmatrix} 1/8 \\ 4/9 \end{bmatrix}, \begin{bmatrix} 5/8 \\ 7/9 \end{bmatrix}, \begin{bmatrix} 3/8 \\ 2/9 \end{bmatrix}$$

Random shifting

Consider the first six Halton vectors in dimension 2, using bases 2 and 3:

$$\begin{bmatrix} 1/2 \\ 1/3 \end{bmatrix}, \begin{bmatrix} 1/4 \\ 2/3 \end{bmatrix}, \begin{bmatrix} 3/4 \\ 1/9 \end{bmatrix}, \begin{bmatrix} 1/8 \\ 4/9 \end{bmatrix}, \begin{bmatrix} 5/8 \\ 7/9 \end{bmatrix}, \begin{bmatrix} 3/8 \\ 2/9 \end{bmatrix}$$

- Generate two random numbers u_1, u_2 from $U(0,1)$, say, $u_1 = 0.74$ and $u_2 = 0.62$.

Random shifting

Consider the first six Halton vectors in dimension 2, using bases 2 and 3:

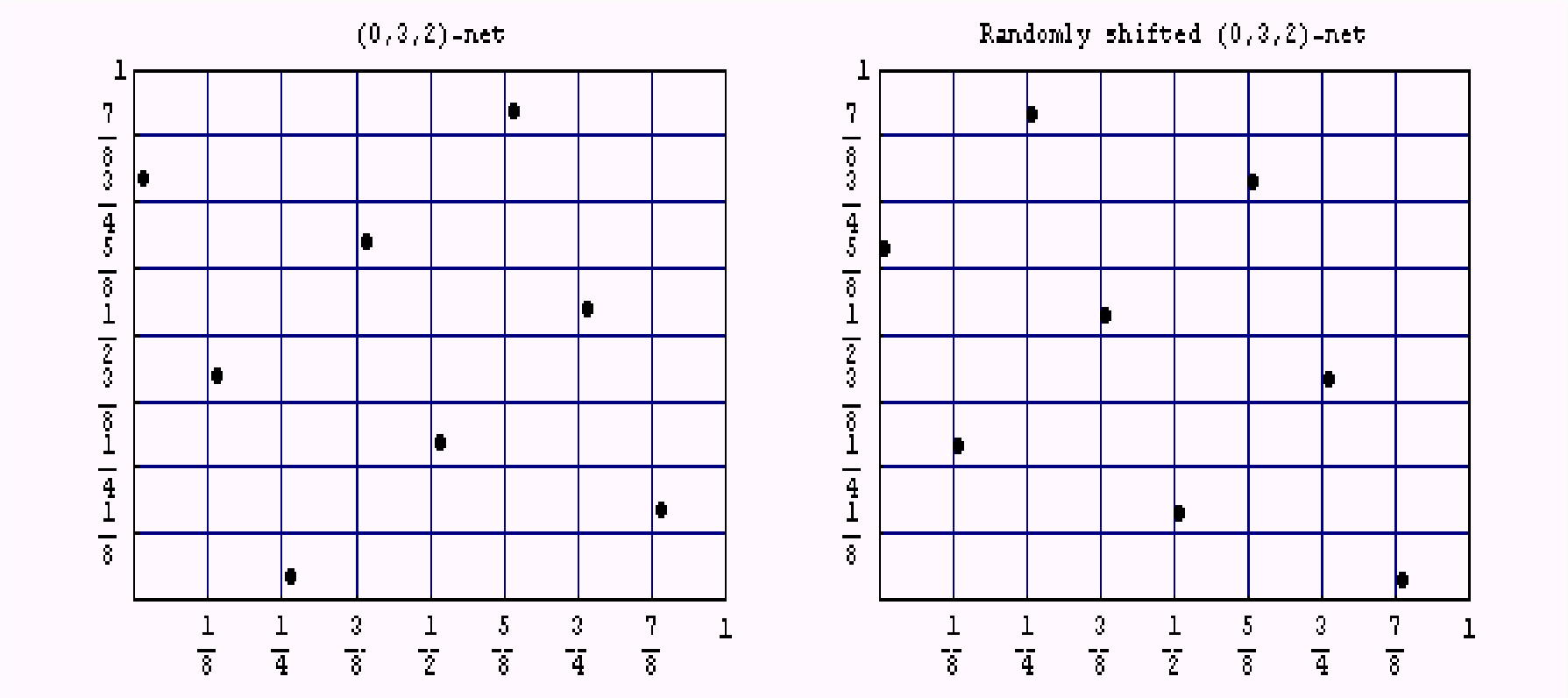
$$\begin{bmatrix} 1/2 \\ 1/3 \end{bmatrix}, \begin{bmatrix} 1/4 \\ 2/3 \end{bmatrix}, \begin{bmatrix} 3/4 \\ 1/9 \end{bmatrix}, \begin{bmatrix} 1/8 \\ 4/9 \end{bmatrix}, \begin{bmatrix} 5/8 \\ 7/9 \end{bmatrix}, \begin{bmatrix} 3/8 \\ 2/9 \end{bmatrix}$$

- Generate two random numbers u_1, u_2 from $U(0,1)$, say, $u_1 = 0.74$ and $u_2 = 0.62$.
- Shift all the above vectors by the vector $u = \begin{bmatrix} 0.74 \\ 0.62 \end{bmatrix}$, by componentwise addition and then by taking the fractional part of the sum. For example, the first vector becomes

$$\begin{bmatrix} \{1/2 + 0.74\} \\ \{1/3 + 0.62\} \end{bmatrix} = \begin{bmatrix} 0.24 \\ 0.95 \end{bmatrix}$$

Random shifting

Our $(0, 3, 2)-$net and its shifted version. The shift is done by the random vector $u = \begin{bmatrix} 0.74 \\ 0.62 \end{bmatrix}$.



Randomized quasi-Monte Carlo methods

Scrambled nets and sequences

Scrambled nets and sequences

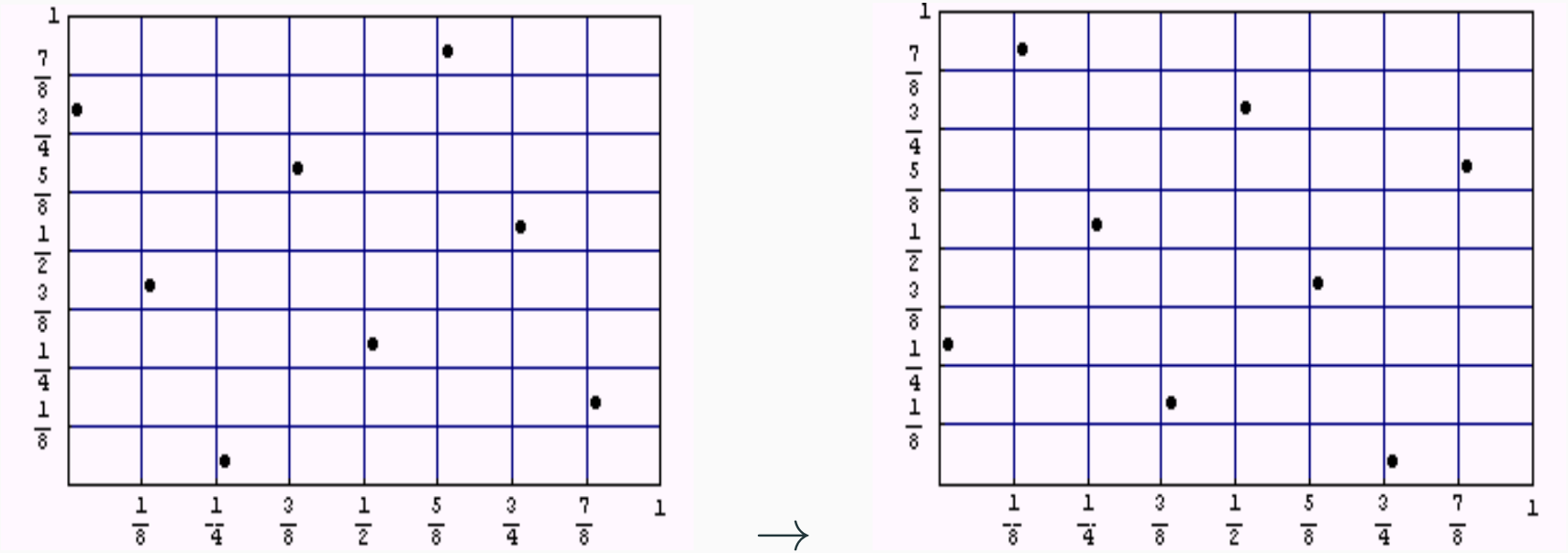
Let's apply the method to the $(0, 3, 2)$-net we discussed earlier.

The first step is to "scramble" the first coordinates: Pick a random permutation of the digits $\{0, 1\}$, say $0 \to 1$ and $1 \to 0$, and apply it to the first digit of the x-coordinates of each vector in the net.

	x-coord.	y-coord.			x-coord.new 1st digits
$(\frac{1}{64}, \frac{51}{64})$	000001	110011		$(\frac{33}{64}, \frac{51}{64})$	**1**00001
$(\frac{33}{64}, \frac{39}{64})$	100001	010011		$(\frac{1}{64}, \frac{39}{64})$	**0**00001
$(\frac{17}{64}, \frac{3}{64})$	010001	000011		$(\frac{49}{64}, \frac{3}{64})$	**1**10001
$(\frac{49}{64}, \frac{35}{64})$	110001	100011	$\longrightarrow$	$(\frac{17}{64}, \frac{35}{64})$	**0**10001
$(\frac{9}{64}, \frac{27}{64})$	001001	011011		$(\frac{41}{64}, \frac{27}{64})$	**1**01001
$(\frac{41}{64}, \frac{59}{64})$	101001	111011		$(\frac{9}{64}, \frac{59}{64})$	**0**01001
$(\frac{25}{64}, \frac{43}{64})$	011001	101011		$(\frac{57}{64}, \frac{43}{64})$	**1**11001
$(\frac{57}{64}, \frac{11}{64})$	111001	001011		$(\frac{25}{64}, \frac{11}{64})$	**0**11001

Scrambled nets and sequences

Geometrically, the effect of this scrambling is to swap the two half-rectangles $[0, 1/2] \times [0, 1]$ and $[1/2, 1] \times [0, 1]$ in the original net.



Scrambling the first digits

Scrambled nets and sequences

Let's now scramble the second digits: Generate two random permutations at random. Say the first permutation is $0 \to 1$ and $1 \to 0$, and the second permutation is the identity permutation.

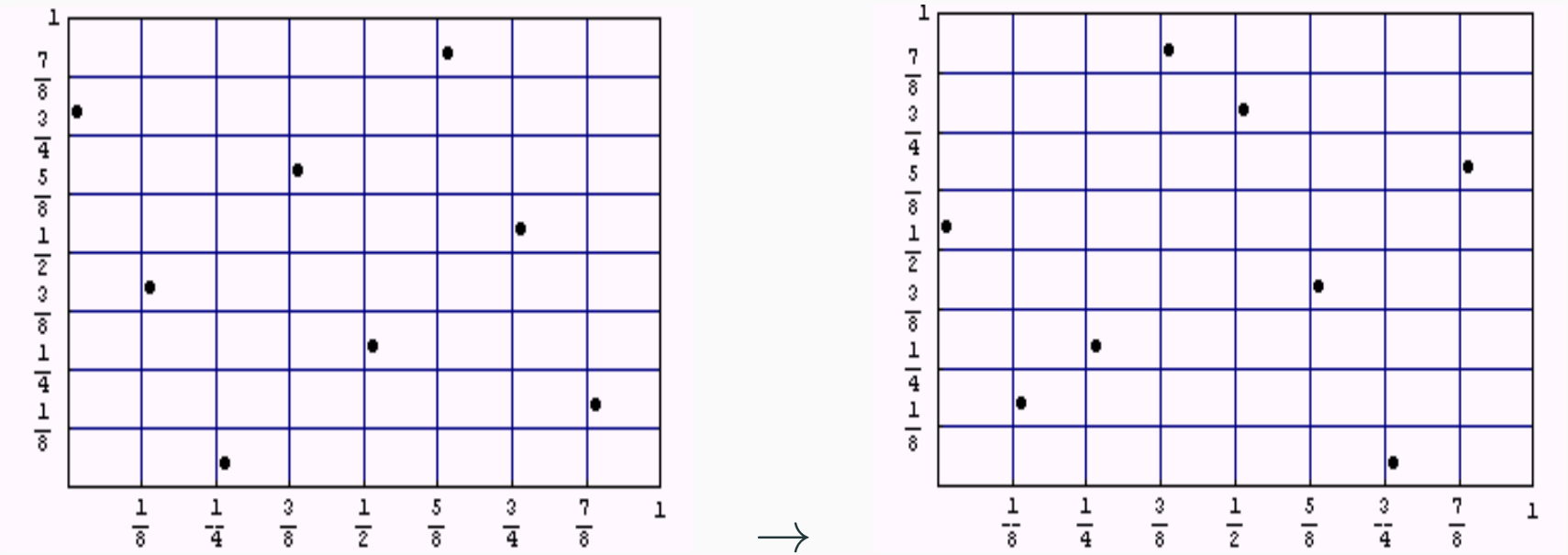
Use first permutation to scramble numbers whose first digits are 0

	x-coord.			x-coord. new 2nd digits
$(\frac{33}{64}, \frac{51}{64})$	100001		$(\frac{33}{64}, \frac{51}{64})$	100001
$(\frac{1}{64}, \frac{39}{64})$	000001		$(\frac{17}{64}, \frac{39}{64})$	0**1**0001
$(\frac{49}{64}, \frac{3}{64})$	110001		$(\frac{49}{64}, \frac{3}{64})$	110001
$(\frac{17}{64}, \frac{35}{64})$	010001	$\longrightarrow$	$(\frac{1}{64}, \frac{35}{64})$	0**0**0001
$(\frac{41}{64}, \frac{27}{64})$	101001		$(\frac{41}{64}, \frac{27}{64})$	101001
$(\frac{9}{64}, \frac{59}{64})$	001001		$(\frac{25}{64}, \frac{59}{64})$	0**1**1001
$(\frac{57}{64}, \frac{43}{64})$	111001		$(\frac{57}{64}, \frac{43}{64})$	111001
$(\frac{25}{64}, \frac{11}{64})$	011001		$(\frac{9}{64}, \frac{11}{64})$	0**0**1001

Scrambled nets and sequences

Geometrically, the effect of this permutation is to swap the rectangles $[0, 1/4] \times [0, 1]$ and $[1/4, 1/2] \times [0, 1]$.

The second permutation is then applied to those numbers whose first digits are 1. Since the second permutation is the identity permutation, this does not change anything. This is what happens to our $(0, 3, 2)$-net after the scrambling of the first 2 digits:



Scrambled nets and sequences

If we continue this process until all digits are scrambled, and scramble the y-coordinates of points in a similar manner, then we obtain a scrambled version of the $(0, 3, 2)$-net.

Let β_u denote a scrambled (t, s)-sequence in base b.

Theorem (Owen)

The scrambled sequence β_u is a (t, s)-sequence in base b with probability 1. Furthermore the resulting quadrature rules satisfy $E\left[Q(\beta_u)\right] = \theta$ and $Var(Q(\beta_u)) = O(\frac{(\log N)^{2s}}{N^2})$.

Remark

Additional properties of scrambled nets and sequences are established most notably by Owen, Hickernell, Loh. For example, for "smooth" functions Owen shows $Var(Q(\beta_u))$ can be as small as $O\left(N^{-3}(\log N)^{s-1}\right)$.

Efficient scrambling methods

- The scrambling method we discussed in the previous section is computationally expensive. Several more efficient scrambling methods were introduced by Matousek.

Efficient scrambling methods

- The scrambling method we discussed in the previous section is computationally expensive. Several more efficient scrambling methods were introduced by Matousek.
- These methods limit the amount of randomness in the original scrambling method in return for efficiency. They have the same mean square L^2 discrepancy as the original scrambling approach of Owen. However, the variance could be worse.

Efficient scrambling methods

- The scrambling method we discussed in the previous section is computationally expensive. Several more efficient scrambling methods were introduced by Matousek.
- These methods limit the amount of randomness in the original scrambling method in return for efficiency. They have the same mean square L^2 discrepancy as the original scrambling approach of Owen. However, the variance could be worse.
- We will discuss two such methods: Random digit scrambling and linear scrambling

Random digit scrambling

Generate one independent random permutation for each digit of the x, y coordinates. Back to our $(0, 3, 2)$-net:

	x-coord.	y-coord.
$(\frac{1}{64}, \frac{51}{64})$	000001	110011
$(\frac{33}{64}, \frac{39}{64})$	100001	010011
$(\frac{17}{64}, \frac{3}{64})$	010001	000011
$(\frac{49}{64}, \frac{35}{64})$	110001	100011
$(\frac{9}{64}, \frac{27}{64})$	001001	011011
$(\frac{41}{64}, \frac{59}{64})$	101001	111011
$(\frac{25}{64}, \frac{43}{64})$	011001	101011
$(\frac{57}{64}, \frac{11}{64})$	111001	001011

We need 12 random independently generated permutations. Say we obtain the following: φ,id,id,φ,id,φ,id,id,id,φ,φ,id, where φ is the permutation that sends 0 to 1 and 1 to 0, and id is the identity.

Random digit scrambling

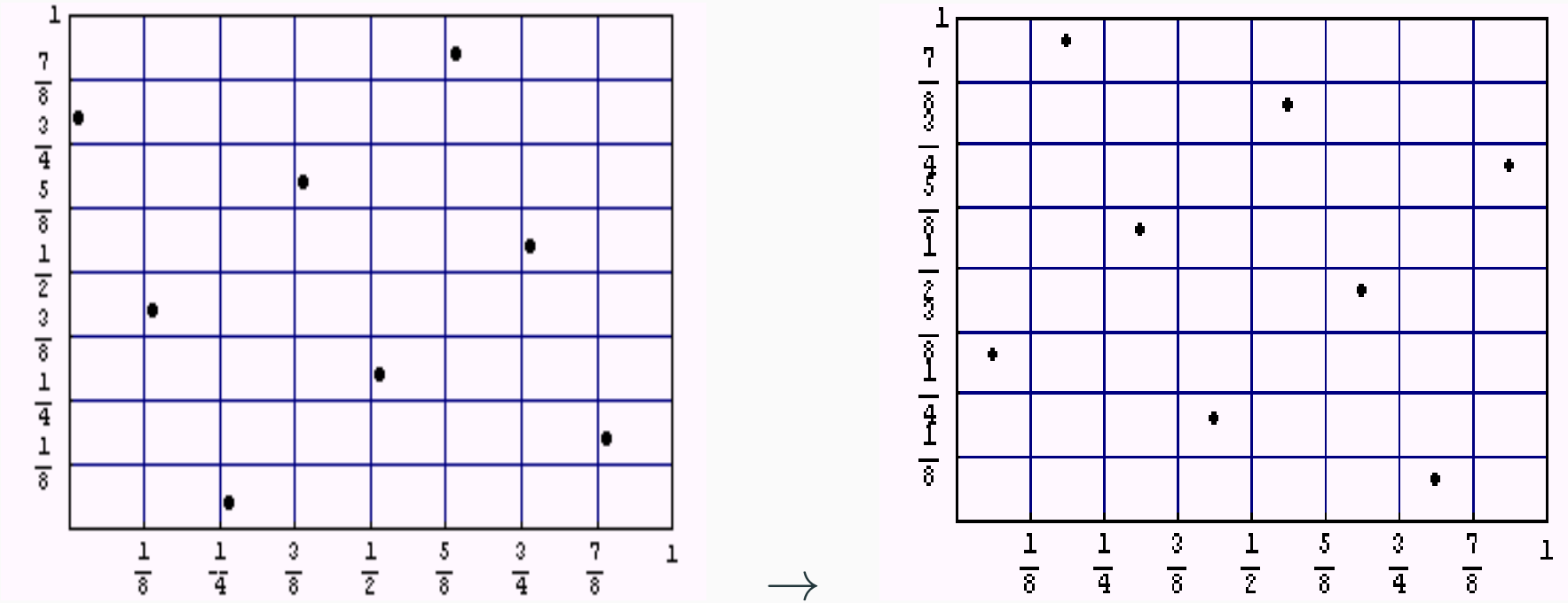
Permutations: φ,id,id,φ,id,φ,id,id,id,φ,φ,id

	x-coord.	y-coord.			x-coord.	y-coord.
$(\frac{1}{64}, \frac{51}{64})$	000001	110011		$(\frac{36}{64}, \frac{53}{64})$	100100	110101
$(\frac{33}{64}, \frac{39}{64})$	100001	010011		$(\frac{4}{64}, \frac{21}{64})$	000100	010101
$(\frac{17}{64}, \frac{3}{64})$	010001	000011		$(\frac{52}{64}, \frac{5}{64})$	110100	000101
$(\frac{49}{64}, \frac{35}{64})$	110001	100011	$\longrightarrow$	$(\frac{20}{64}, \frac{37}{64})$	010100	100101
$(\frac{9}{64}, \frac{27}{64})$	001001	011011		$(\frac{44}{64}, \frac{29}{64})$	101100	011101
$(\frac{41}{64}, \frac{59}{64})$	101001	111011		$(\frac{12}{64}, \frac{61}{64})$	001100	111101
$(\frac{25}{64}, \frac{43}{64})$	011001	101011		$(\frac{60}{64}, \frac{45}{64})$	111100	101101
$(\frac{57}{64}, \frac{11}{64})$	111001	001011		$(\frac{28}{64}, \frac{13}{64})$	011100	001101

Digit scrambling of the $(0, 3, 2)$-net

Random digit scrambling

Here is the $(0, 3, 2)$-net and its digit scrambled version. Notice that the net structure is preserved by the digit scrambling method.



Linear scrambling

- Generate s lower triangular square matrices $L^{(k)}$, $k = 1, \ldots, s$, such that

Linear scrambling

- Generate s lower triangular square matrices $L^{(k)}$, $k = 1, ..., s$, such that
  - non-diagonal entries are random integers between 0 and $p - 1$

Linear scrambling

- Generate s lower triangular square matrices $L^{(k)}$, $k = 1, \ldots, s$, such that
  - non-diagonal entries are random integers between 0 and $p - 1$
  - diagonal entries are random integers between 1 and $p - 1$

Linear scrambling

- Generate s lower triangular square matrices $L^{(k)}$, $k = 1, ..., s$, such that
  - non-diagonal entries are random integers between 0 and $p - 1$
  - diagonal entries are random integers between 1 and $p - 1$
- Generate s vectors $g^{(k)}$ with random entries between 0 and $p - 1$

Linear scrambling

- Generate s lower triangular square matrices $L^{(k)}$, $k = 1, ..., s$, such that
  - non-diagonal entries are random integers between 0 and $p - 1$
  - diagonal entries are random integers between 1 and $p - 1$
- Generate s vectors $g^{(k)}$ with random entries between 0 and $p - 1$
- Recall how a digital (t, s)-sequence $q_1, q_2, ...,$ is generated:

$$\phi(q_n^{(k)}) = \mathbf{C}^{(k)} d_n, \;\; k = 1, ..., s$$

where d_n has the digits of n and $\phi(q_n^{(k)})$ has the digits of $q_n^{(k)}$

Linear scrambling

- Generate s lower triangular square matrices $L^{(k)}$, $k = 1, ..., s$, such that
  - non-diagonal entries are random integers between 0 and $p - 1$
  - diagonal entries are random integers between 1 and $p - 1$
- Generate s vectors $g^{(k)}$ with random entries between 0 and $p - 1$
- Recall how a digital (t, s)-sequence $q_1, q_2, ...,$ is generated:

$$\phi(q_n^{(k)}) = \mathbf{C}^{(k)} d_n, \;\; k = 1, ..., s$$

  where d_n has the digits of n and $\phi(q_n^{(k)})$ has the digits of $q_n^{(k)}$
- A linear scrambled version of the sequence replaces

$$\mathbf{C}^{(k)} \rightarrow \mathbf{L}^{(k)} \mathbf{C}^{(k)}, \;\; k = 1, ..., s$$

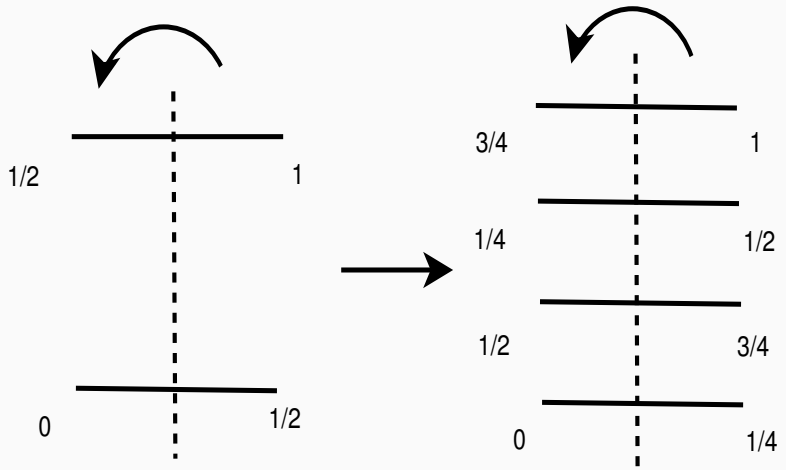
  and $\phi(q_n^{(k)}) = \mathbf{C}^{(k)} d_n + g^{(k)}, \;\; k = 1, ..., s$

Randomized quasi-Monte Carlo methods

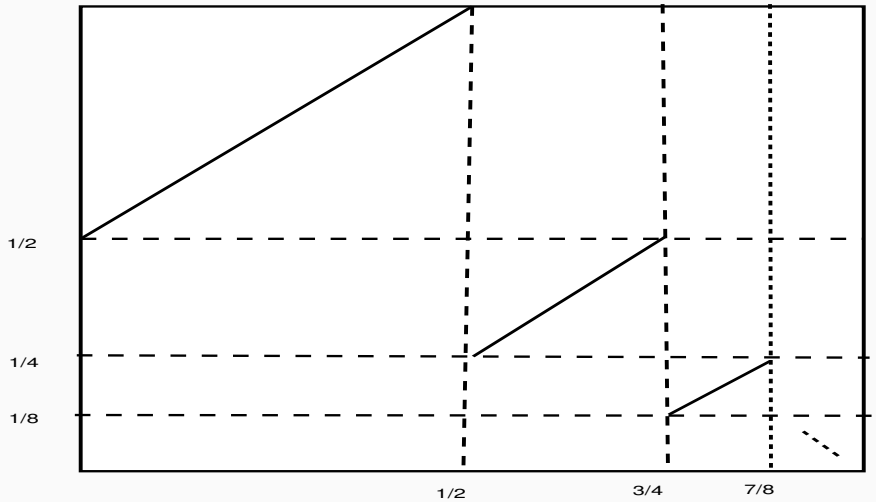
Random-start Halton sequences

Random-start Halton sequences: von Neumann-Kakutani transformation

An ergodic and measure-preserving transf. constructed inductively



Splitting & stacking process



$T : [0, 1) \to [0, 1)$

Remark

The orbit of 0 under T is the van der Corput sequence in base 2.

von Neumann-Kakutani transformation

- If $\mathbf{b} = (b_1, ..., b_s)$ is a vector of positive integers that are pairwise relatively prime and $\mathbf{x} = (x_1, ..., x_s)$ is a vector in $[0, 1)^s$, then the s-dimensional von Neuman-Kakutani transformation is given by

$$\mathbf{T_b}(\mathbf{x}) = (T_{b_1}(x_1), ..., T_{b_s}(x_s))$$

  The orbit of $(0, ..., 0)$ under $\mathbf{T_b}$ gives the Halton sequence in bases $b_1, ..., b_s$.

von Neumann-Kakutani transformation

- If $\mathbf{b} = (b_1, ..., b_s)$ is a vector of positive integers that are pairwise relatively prime and $\mathbf{x} = (x_1, ..., x_s)$ is a vector in $[0, 1)^s$, then the s-dimensional von Neuman-Kakutani transformation is given by

$$\mathbf{T_b}(\mathbf{x}) = (T_{b_1}(x_1), ..., T_{b_s}(x_s))$$

  The orbit of $(0, ..., 0)$ under $\mathbf{T_b}$ gives the Halton sequence in bases $b_1, ..., b_s$.
- The orbit of any vector $\mathbf{x}$ in $[0, 1)^s$ under $\mathbf{T_b}$, $\{\mathbf{T_b^n}(\mathbf{x})\}_{n=1}^{\infty}$, is a u.d. mod 1 sequence (a.s.) from Birkhoff's ergodic theorem.

von Neumann-Kakutani transformation

- If $\mathbf{b} = (b_1, ..., b_s)$ is a vector of positive integers that are pairwise relatively prime and $\mathbf{x} = (x_1, ..., x_s)$ is a vector in $[0, 1)^s$, then the s-dimensional von Neuman-Kakutani transformation is given by

$$\mathbf{T_b}(\mathbf{x}) = (T_{b_1}(x_1), ..., T_{b_s}(x_s))$$

  The orbit of $(0, ..., 0)$ under $\mathbf{T_b}$ gives the Halton sequence in bases $b_1, ..., b_s$.
- The orbit of any vector $\mathbf{x}$ in $[0, 1)^s$ under $\mathbf{T_b}$, $\{\mathbf{T_b^n}(\mathbf{x})\}_{n=1}^{\infty}$, is a u.d. mod 1 sequence (a.s.) from Birkhoff's ergodic theorem.
- The transformation can be generalized so that the orbit gives the permuted Halton sequence for any permutations.

Random-start permuted Halton sequences

Randomly select "starting values", $u \in U(0,1)^s$: let β_u be the orbit of u under $\mathbf{T_b}$. These are called random-start permuted Halton sequences.

Define random quadrature rule $Q(\beta_u)$ as

$$Q(\beta_u) = \frac{1}{N} \sum_{i=0}^{N-1} f(\mathbf{T}_\mathbf{b}^i(u))$$

Theorem

The random-start permuted Halton sequences are u.d. mod 1 a.s.

- $E[Q(\beta_u)] = \theta$
- $Var(Q(\beta_u)) = O(\frac{(\log N)^{2s}}{N^2})$ *if the following conjecture is true*

Random-start permuted Halton sequences

Conjecture

$D_N^*(\mathbf{T}_{\mathbf{b}}^n(\mathbf{x}))$ is $O(N^{-1}(\log N)^s)$ for arbitrary permutations used in the splitting and stacking process, arbitrary $\mathbf{x}$, and any relatively prime bases $b_1, ..., b_s$.

Remark

The conjecture is true if

- *the permutations used in splitting and stacking process are the identity (Lapeyre & Pages, '92)*
- $\mathbf{x} = 0$ *(Atanassov, '04)*