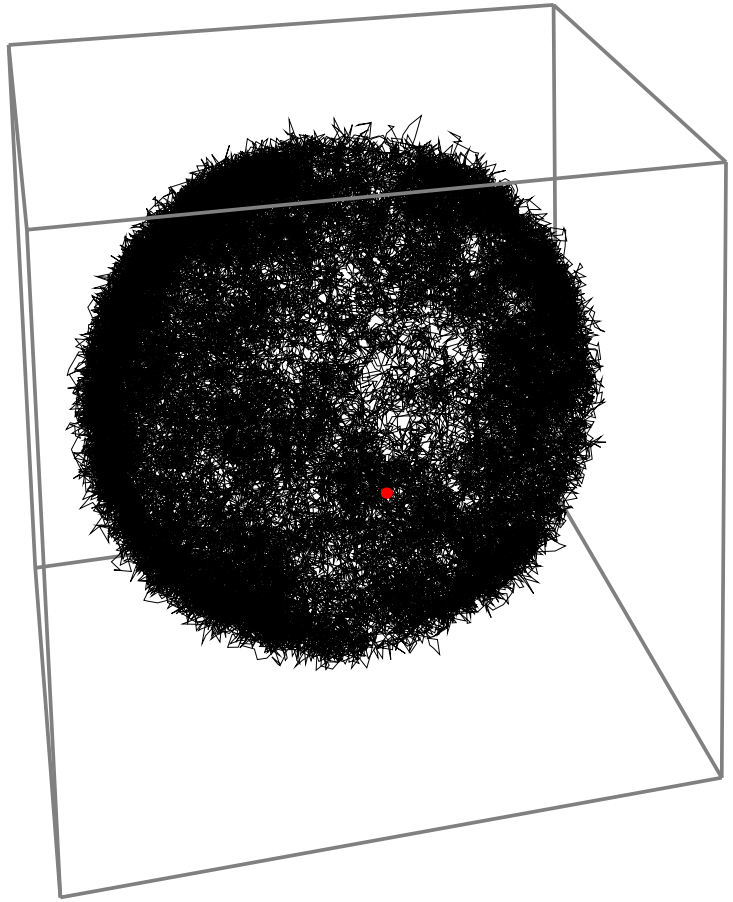
Certificate in Quantitative Finance

Peter Jäckel*

MONTE CARLO METHODS



*OTC Analytics

Contents

I. Basics

A *Monte Carlo* method is any method that involves a *numerical* approximation by means of *sampling a domain*.

Against popular belief, the term *Monte Carlo method* does not, in any way, imply that any form of randomness is involved.

The most common calculation evaluated by Monte Carlo methods is

$$\int_{\mathcal{D}} g(x)\,\mathrm{d}x \tag{1}$$

over some domain $\mathcal{D}$ for some integrand $g(x)$. Almost always, the integrand $g(x)$ is comprised of two parts: the value function $f(x)$ and the *measure* $\psi(x)$ by

$$g(x) = f(x) \cdot \psi(x)\ . \tag{2}$$

II. The connection to statistics

- A *random experiment* is a process or action subject to non-deterministic uncertainty.
- We call the outcome of a random experiment a *draw* or a *variate* from a distribution.
- A *distribution density* or *probability density function* is a generalised function that assigns *likelihood* or *probability density* to all possible results of a random experiment.
- For our purposes, a *generalised function* can be an ordinary function, or a linear combination of an ordinary function and any finite number of *Dirac densities* $\delta(x - x_0)$.

- The *Dirac density* is the equivalent of assigning a finite probability to a single number on a continuous interval. This means, the Dirac density $\delta(x - x_0)$ is zero everywhere where it is defined, and strictly speaking undefined at x_0. However, its integral is given by the *Heaviside function* $h(x)$, i.e.

$$h(x - x_0) = \int_{x'=-\infty}^{x} \delta(x' - x_0)\,\mathrm{d}x' \ ,$$

  which is zero for $x < x_0$ and one for $x > x_0$.

- We call the set of all attainable outcomes X of a random experiment the *domain $\mathcal{D}(X)$ of the random experiment*. Whenever $\mathcal{D}(X)$ is an ordered set, i.e. when we can decide whether any one element is less than any of the other elements of $\mathcal{D}(X)$, we define the *cumulative probability function* as

$$\Psi(x) \ = \int_{x'=\inf(\mathcal{D})}^{x} \psi(x')\,\mathrm{d}x' \ = \ \mathsf{Pr}\left[X < x\right] \ . \tag{3}$$

- All distribution densities are, by definition, normalised, i.e.

$$\int_{\mathcal{D}} \psi(x)\,\mathrm{d}x = 1\ . \tag{4}$$

- The *expected value* of a quantity subject to uncertainty is the probability weighted average. Our notation for the expected value of a quantity f with respect to a probability density ψ is

$$\mathsf{E}_\psi[f] \quad : \quad \text{expected value of } f \text{ with respect to } \psi. \tag{5}$$

- An alternative notation frequently encountered for $\mathsf{E}_\psi[f]$ is

$$\langle f \rangle\ , \tag{6}$$

- The *variance* of a quantity subject to uncertainty is defined as

$$\mathsf{E}_\psi\left[f^2\right] - \left(\mathsf{E}_\psi[f]\right)^2 \quad = \quad \left\langle f^2 \right\rangle - \langle f \rangle^2\ . \tag{7}$$

- Very often, there is no ambiguity about the relevant distribution. In this case we may just write $\mathsf{E}[f]$. Alternatively, there may be a parametric description of the distribution of f. This means that f is actually a function of some other uncertain variable x. Given the distribution density of x, say $\psi(x)$, we would denote the expectation of f as $\mathsf{E}_{\psi(x)}[f(x)]$. This is just to say

$$\mathsf{E}_{\psi(x)}[f(x)] \;=\; \int_{-\infty}^{\infty} f(x)\psi(x)\,\mathrm{d}x\;. \tag{8}$$

- The connection between Monte Carlo methods and the statistical quantity known as the *expectation* is simply the fact that the target of a Monte Carlo calculation is to compute

$$\int f(x)\psi(x)\,\mathrm{d}x\;. \tag{9}$$

which, in a statistical sense, is, of course, equal to $\mathsf{E}_{\psi(x)}[f(x)]$.

III. The Feynman-Kac theorem

R. Feynman and M. Kac [Fey48, Kac51]:

Given the set of stochastic processes

$$\mathrm{d}X_i = b_i \mathrm{d}t + \sum_{j=1}^{n} a_{ij} \mathrm{d}W_j \quad \text{for} \quad i = 1..n \ , \tag{10}$$

with formal solution

$$X_i(T) = X_i(t) + \int_t^T b_i \mathrm{d}t + \int_t^T \sum_{j=1}^{n} a_{ij} \mathrm{d}W_j \ , \tag{11}$$

any function $V(t, \boldsymbol{X})$ with boundary conditions

$$V(T, \boldsymbol{X}) = f(\boldsymbol{X}) \tag{12}$$

that satisfies the partial differential equation

$$\frac{\partial V}{\partial t} + \sum_{i=1}^{n} b_i \frac{\partial V}{\partial X_i} + \frac{1}{2} \sum_{i,j=1}^{n} c_{ij} \frac{\partial^2 V}{\partial X_i \partial X_j} + g = rV \tag{13}$$

with

$$c_{ij} := \sum_{k=1}^{n} a_{ik} a_{jk} \tag{14}$$

can be represented as the expectation

$$V(t, \boldsymbol{X}) = \mathsf{E}\left[f(\boldsymbol{X}_T) \mathrm{e}^{-\int_t^T r \, \mathrm{d}u} + \int_t^T g \, \mathrm{e}^{-\int_t^s r \, \mathrm{d}u} \mathrm{d}s \right] . \tag{15}$$

Hereby, all of the coefficients a_{ij}, b_i, r, and g can be functions both of time t and the state vector $\boldsymbol{X}(t)$.

As with most mathematical theorems, there is a whole host of additional conditions for good behaviour of all the coefficients and functions involved and the reader is referred to, e.g., Karatzas and Shreve [KS91] (page 366).

IV. The basic Monte Carlo algorithm

The easiest form of *Monte Carlo integration* of integrals like equation (9) can be summarised as follows.

- Establish a procedure of drawing variates $\boldsymbol{x}$ from the target distribution density $\psi(\boldsymbol{x})$.

- Set up a running sum variable

```
double runingSum=0;
```

  and a counter variable

```
unsigned long i=0;
```

- Draw a variate vector $\boldsymbol{x}_i$ and evaluate $f_i := f(\boldsymbol{x}_i)$.
- Add the computed function value to `runningSum`, i.e.

```
runningSum += f;
```

- Increment i, i.e.

```
++i;
```

- Set the running average to

```
const double runningAverage = runningSum/i;
```

  This gives us the *Monte Carlo estimator*

$$\hat{v}_N := \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}_i) \quad . \tag{16}$$

- Keep iterating until either the required number of iterations has been carried out, or a specific error estimate has decreased below a predetermined threshold.

A practitioner's list of key points to remember:-

- Do not trust any single Monte Carlo simulation result unless you feel you have good reasons to believe that it is sufficiently converged.
- Do not trust the fact that an interval of one standard error around your result is still within an acceptable range for your purposes - one in three simulations should be outside one standard error interval!
- Do not repeat a simulation using a different "seed" - you have no idea how much overlap you may have!
- If you wish to repeat a simulation, do so using a different number generation algorithm.
- The most intuitive control over convergence is given by a convergence diagram - I recommend plotting the running average as a function of the number of iterations N with plot points for N being powers of 2.

V. The standard error

A statistical error estimate for a quantity subject to probabilistic uncertainty is the *standard deviation*

$$\sigma_a \;=\; \left\langle a^2 \right\rangle - \left\langle a \right\rangle^2 \;. \tag{17}$$

Note: the standard deviation is strictly only applicable when the distribution of a itself is normal. It can still be used as a vague measure of uncertainty, **if and only if**

- the distribution of a is unimodal, i.e. its probability density function has a single local maximum which must be its global maximum;

- the probability density function of a is concave only in a singly connected domain near its maximum and non-concave everywhere else.

In general, we don't actually know the variance σ_a^2 of the random variate a whose expectation we are trying to estimate. However, by virtue of the combination of the central limit and the continuous mapping theorem, we can use the variance of the simulation instead as an estimate for σ_a^2:

$$\hat{\sigma}_a^{(N)} = \sqrt{\left(\frac{1}{N}\sum_{i=1}^{N} a_i^2\right) - \left(\frac{1}{N}\sum_{i=1}^{N} a_i\right)^2} \tag{18}$$

This leads us to the definition of the *standard error*:

$$\epsilon_a^{(N)} = \frac{\hat{\sigma}_a^{(N)}}{\sqrt{N}} \tag{19}$$

Important:

- Every single Monte Carlo simulation result for a normally distributed variate a has a probability of approximately **one third** of being more than one standard error away from the unknown true expectation.

- In practice, almost nothing we are ever interested in is ultimately normally distributed.

- The less similarities the distribution of the target quantity has with the normal (Gaussian) distribution, the less meaning has the standard error figure.

Examples where the target distribution is distinctly non-normal:

– Correlation estimator

$$\hat{\rho}_{ab} = \frac{\left(\frac{1}{N}\sum_{i=1}^{N} a_i b_i\right) - \left(\frac{1}{N}\sum_{i=1}^{N} a_i\right)\left(\frac{1}{N}\sum_{i=1}^{N} b_i\right)}{\sqrt{\left[\left(\frac{1}{N}\sum_{i=1}^{N} a_i^2\right) - \left(\frac{1}{N}\sum_{i=1}^{N} a_i\right)^2\right]\left[\left(\frac{1}{N}\sum_{i=1}^{N} b_i^2\right) - \left(\frac{1}{N}\sum_{i=1}^{N} b_i\right)^2\right]}} \tag{20}$$

– Option payoff

Despite all these shortcomings, without further knowledge of the task at hand, the *standard error* is still the best generic estimator of convergence available, irrespective of whether the simulation algorithm was based on statistical methods or not.

VI. Uniform variates

Standard uniform variates are supposed to be equally distributed on $[0,1]$, $[0,1)$, or $(0,1)$ (depending on the algorithm used). For applications in finance, *without exception*, we need to ensure that the number generator is confined to $(0,1)$. There are two main variations of number generation algorithms:-

- Pseudo-random number generators. Can be used as *number pipelines*.

  These generators have no built-in mechanism for sampling efficiency. They typically lead to *clusters* and *gaps* which may or may not be desirable.

  $\Rightarrow$ emulate randomness.

- Low-discrepancy number generators (sometimes also, *misleadingly*, referred to as quasi-random number generators).

  These algorithms are designed to take advantage of our knowledge of the dimensionality and ordering of importance of the given simulation problem at hand.

  $\Rightarrow$ holistic sampling scheme aiming for as little randomness as possible.

Pseudo-random numbers

- Uniform coverage is ensured by virtue of *serial decorrelation*.
- The most common simple algorithms are *congruential generators*

$$m_{n+1} = (a\,m_n + c) \mod M\ . \tag{21}$$

for the integer m_n. Real-valued variates on $(0,1)$ are then produced from m_n by setting

$$x_n = (m_n + 1)/(M + 1)\ . \tag{22}$$

- There are many number generators: see the reviews in [PTVF92, SB02, L'E01]. To name but one reliable one: the Mersenne Twister [MN97].
- Number generators are highly sensitive to tampering - don't do it! Any changes to an existing number generator without a deep understanding of the involved number theory is practically guaranteed to destroy the desired features!

Low-discrepancy numbers

- Uniform coverage is ensured by *selective placement* within one dimension and *incommensurate sampling* across dimensions.

- The mathematics can be very involved, though the resulting run-time algorithm is usually extremely efficient.

- Monte Carlo simulations using low-discrepancy numbers are, in theory, expected to converge as

$$\sim c(d)\frac{(\ln N)^d}{N}$$

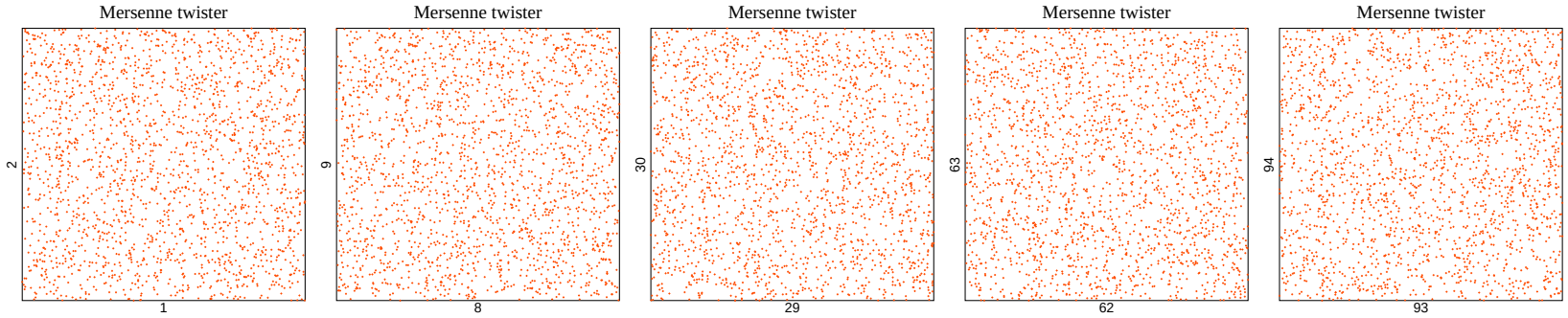
whereas the convergence using random numbers is, expected to be

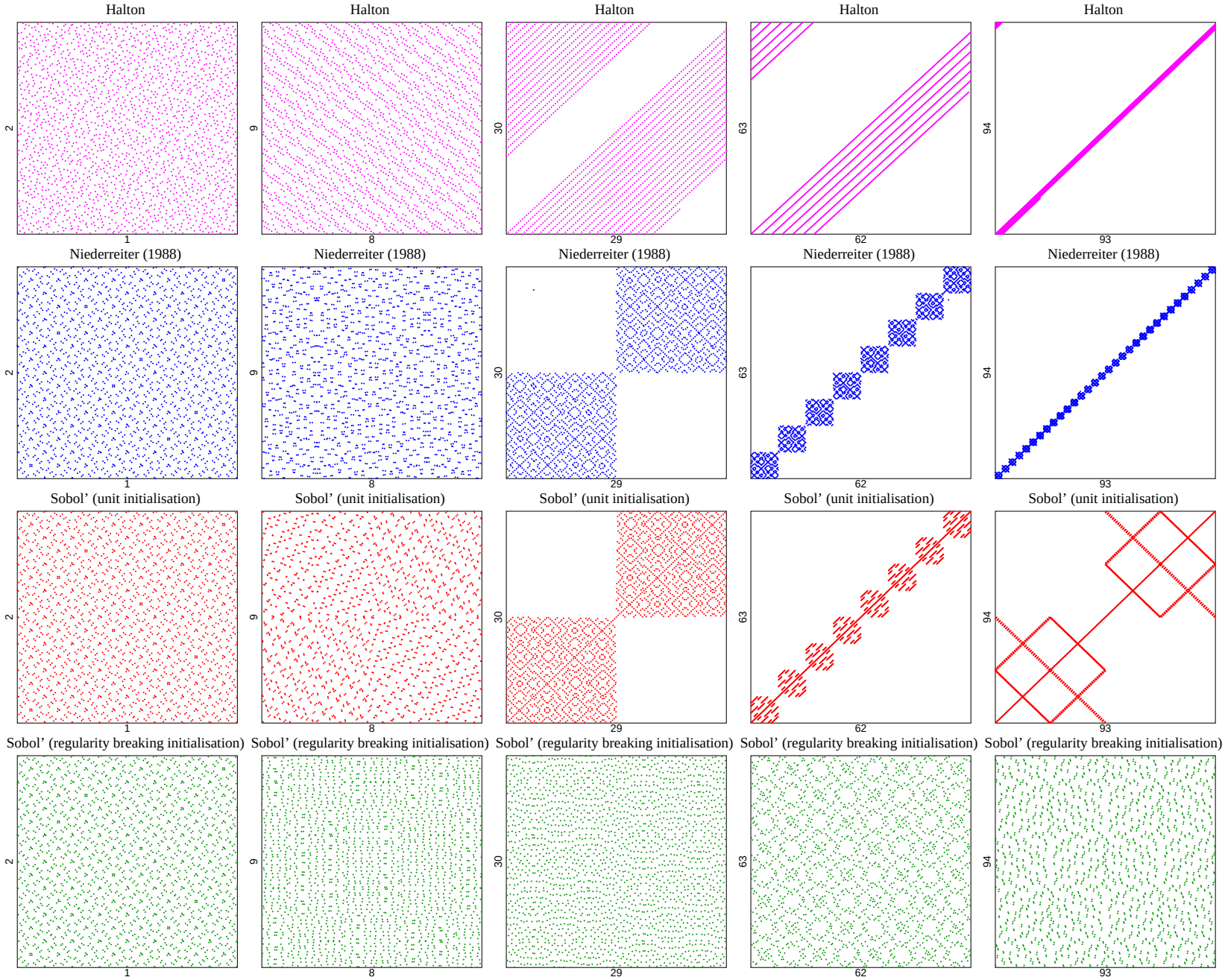
$$\sim \frac{1}{\sqrt{N}}\,.$$

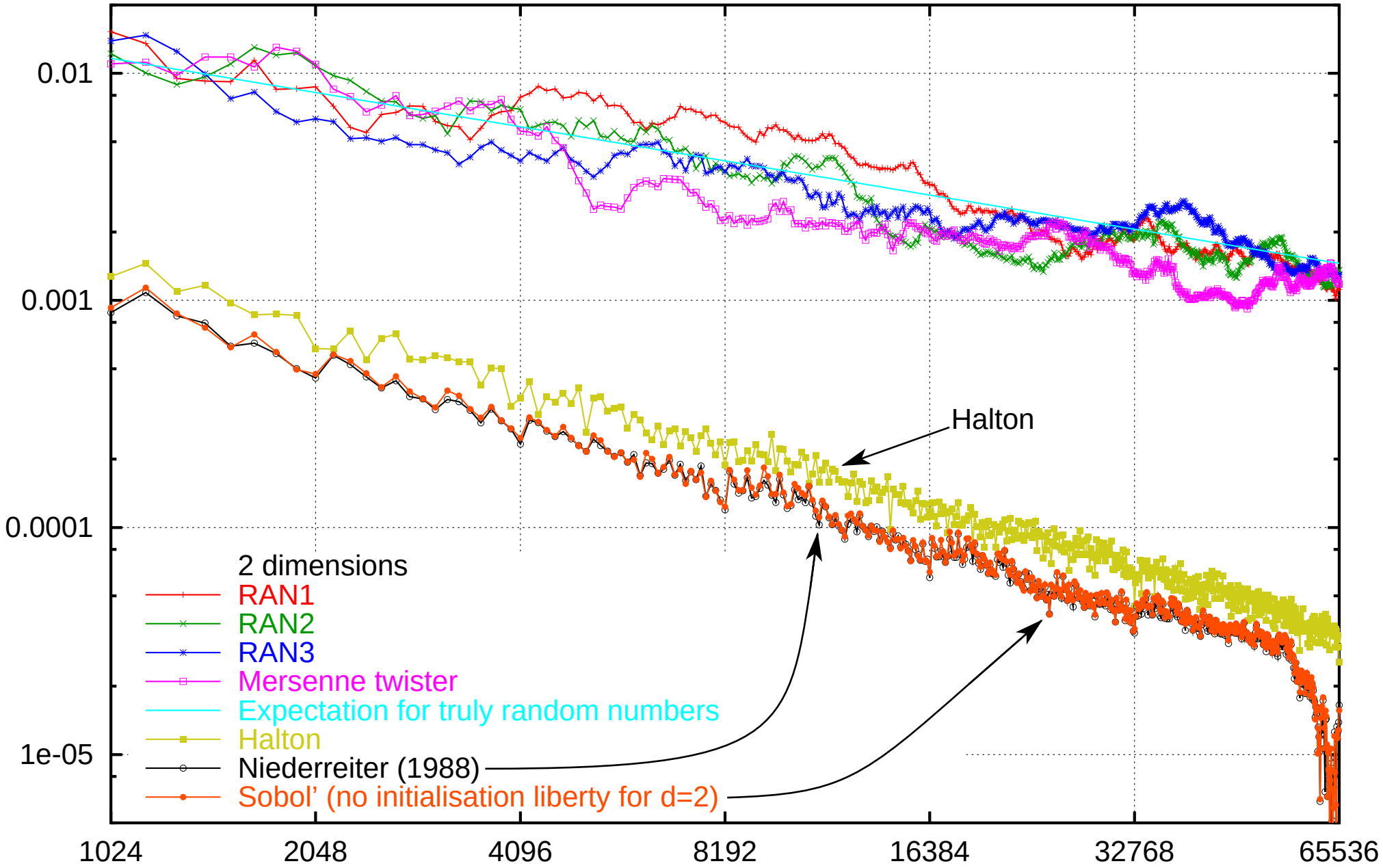
- The problem with low-discrepancy numbers is that $c(d)$ can explode geometrically with d.

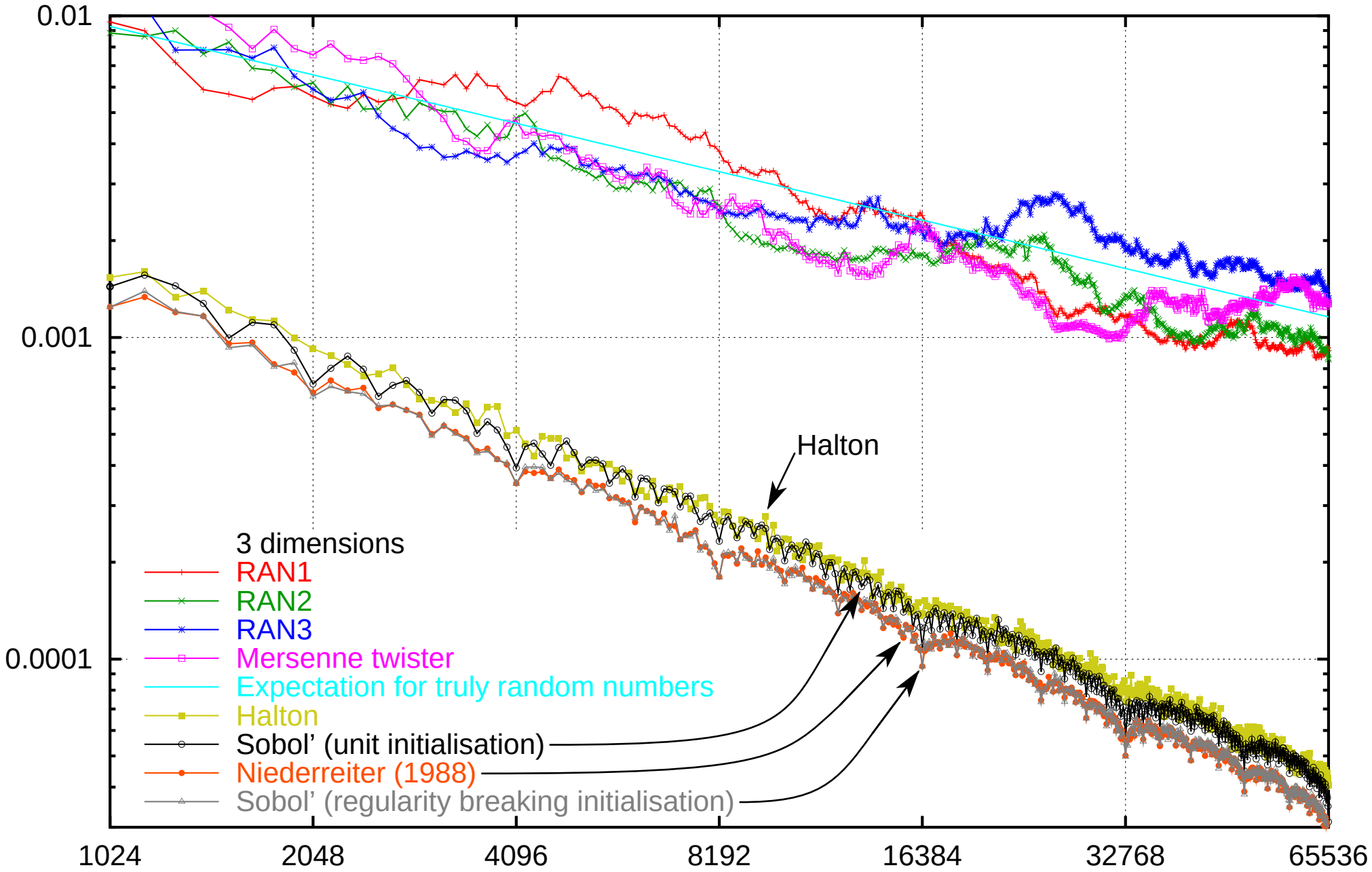
- However, it has been shown that there are low-discrepancy algorithms that do not suffer the dimensionality breakdown, namely Sobol' numbers with appropriate initialisation [Sob76], and Niederreiter-Xing numbers [NX95, NX96].

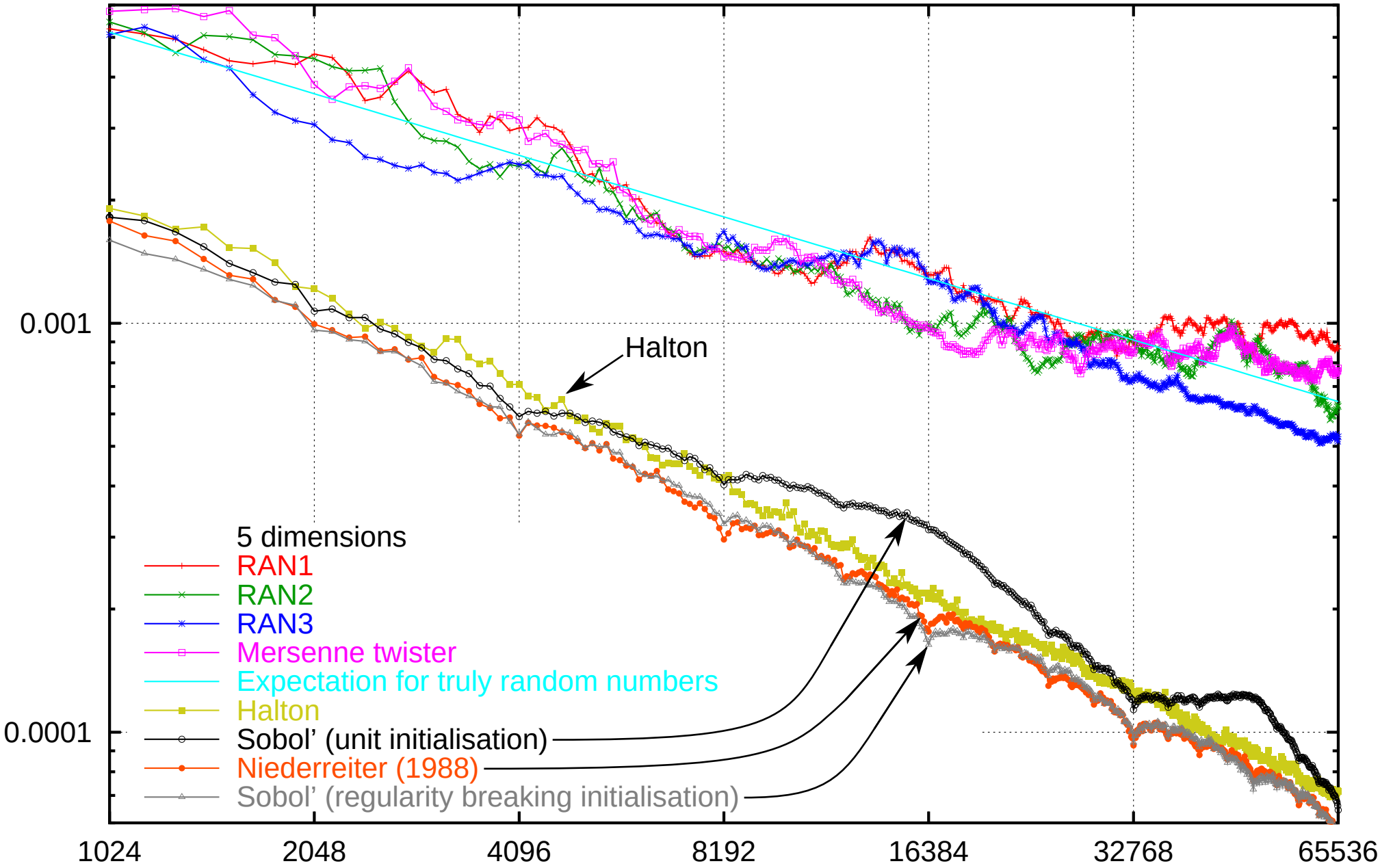
- Two methods are commonly employed to test low-discrepancy numbers: numerical evaluation of a measure for non-uniformity (i.e. discrepancy), and pairwise projections:-

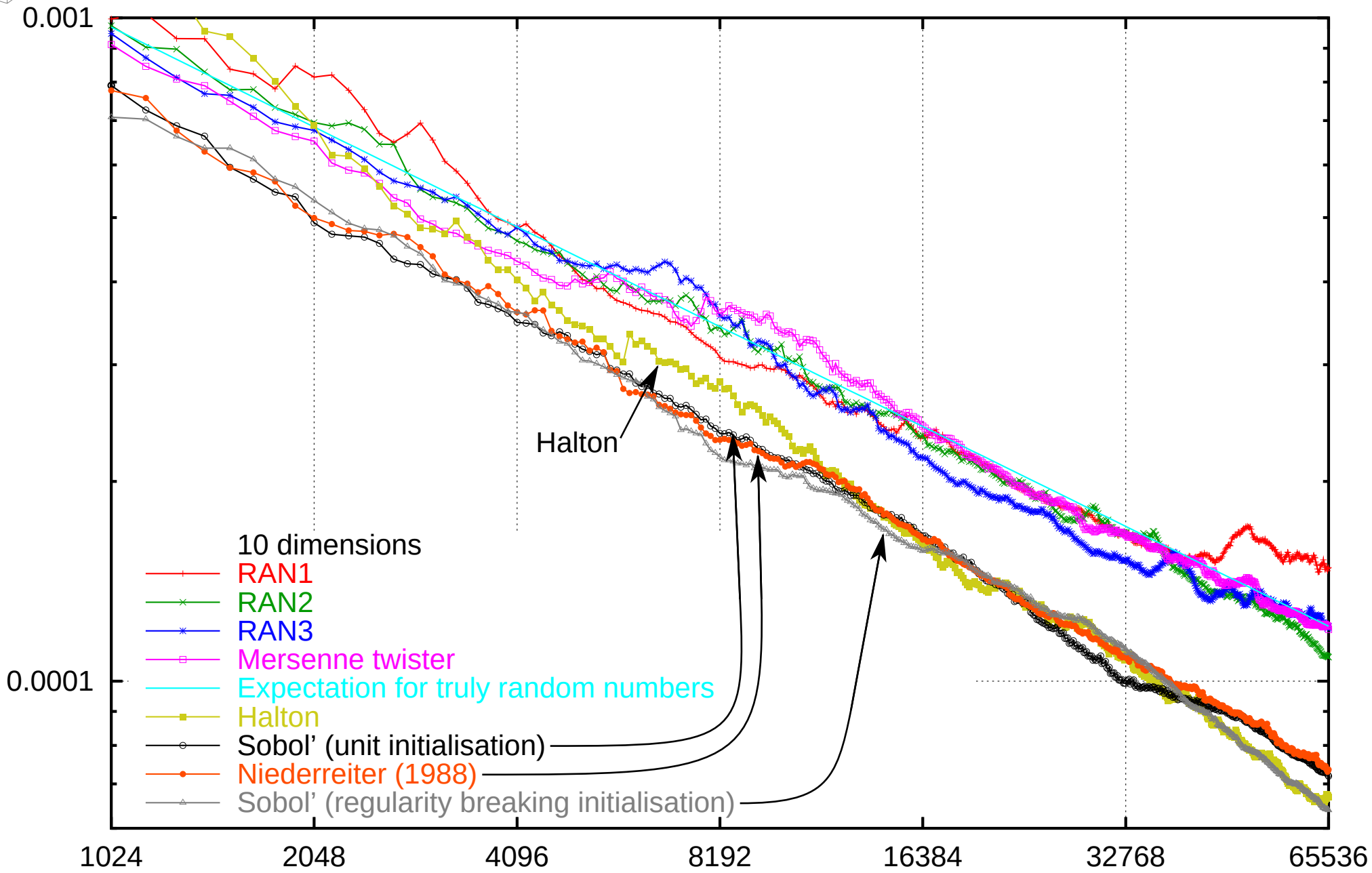


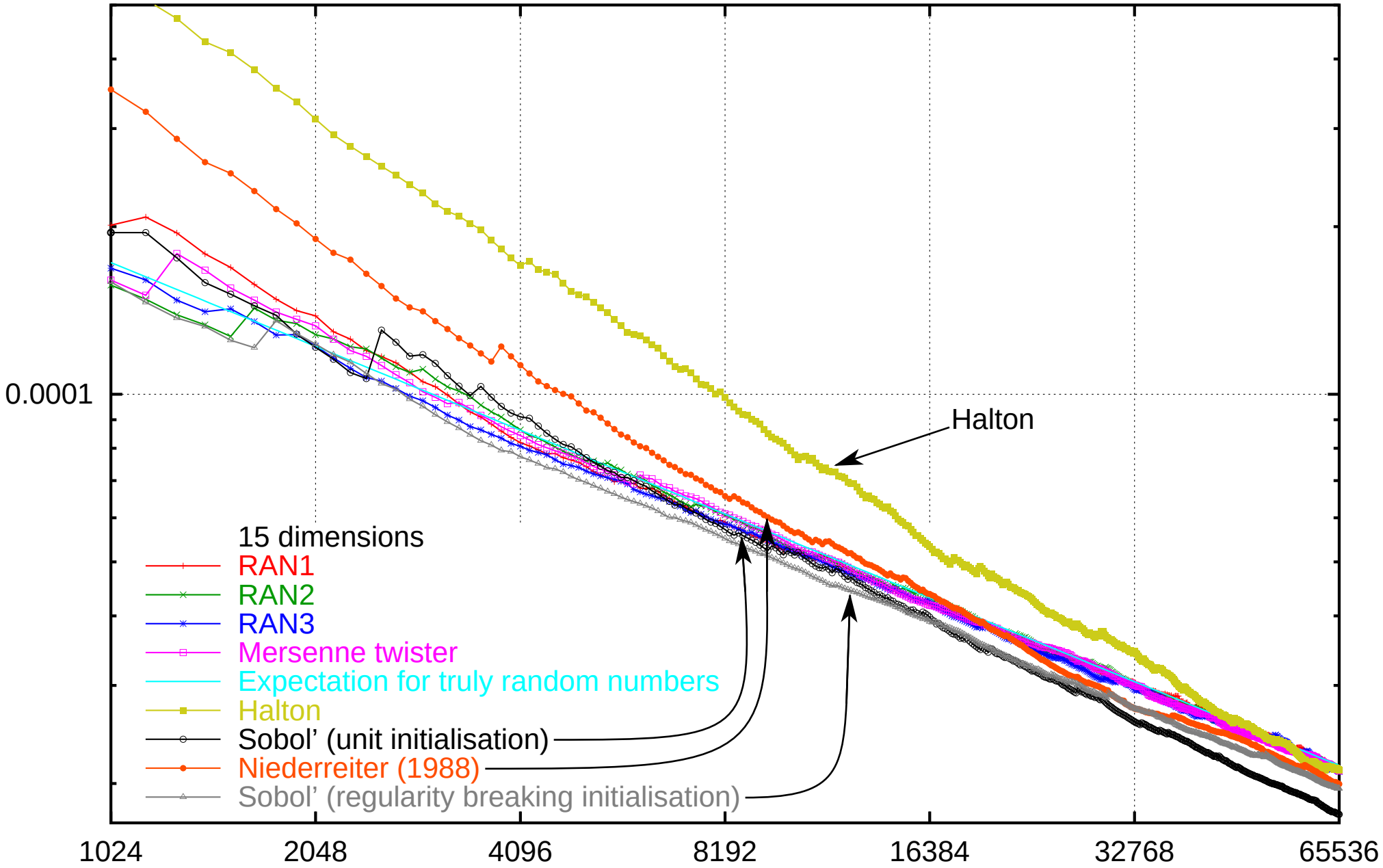

Halton
Halton
Halton
Halton
Halton
Niederreiter (1988)
Niederreiter (1988)
Niederreiter (1988)
Niederreiter (1988)
Niederreiter (1988)
Sobol' (unit initialisation)
Sobol' (unit initialisation)
Sobol' (unit initialisation)
Sobol' (unit initialisation)
Sobol' (unit initialisation)
Sobol' (regularity breaking initialisation)
Sobol' (regularity breaking initialisation)
Sobol' (regularity breaking initialisation)
Sobol' (regularity breaking initialisation)
Sobol' (regularity breaking initialisation)
2
9
30
63
94
1
8
29
62
93

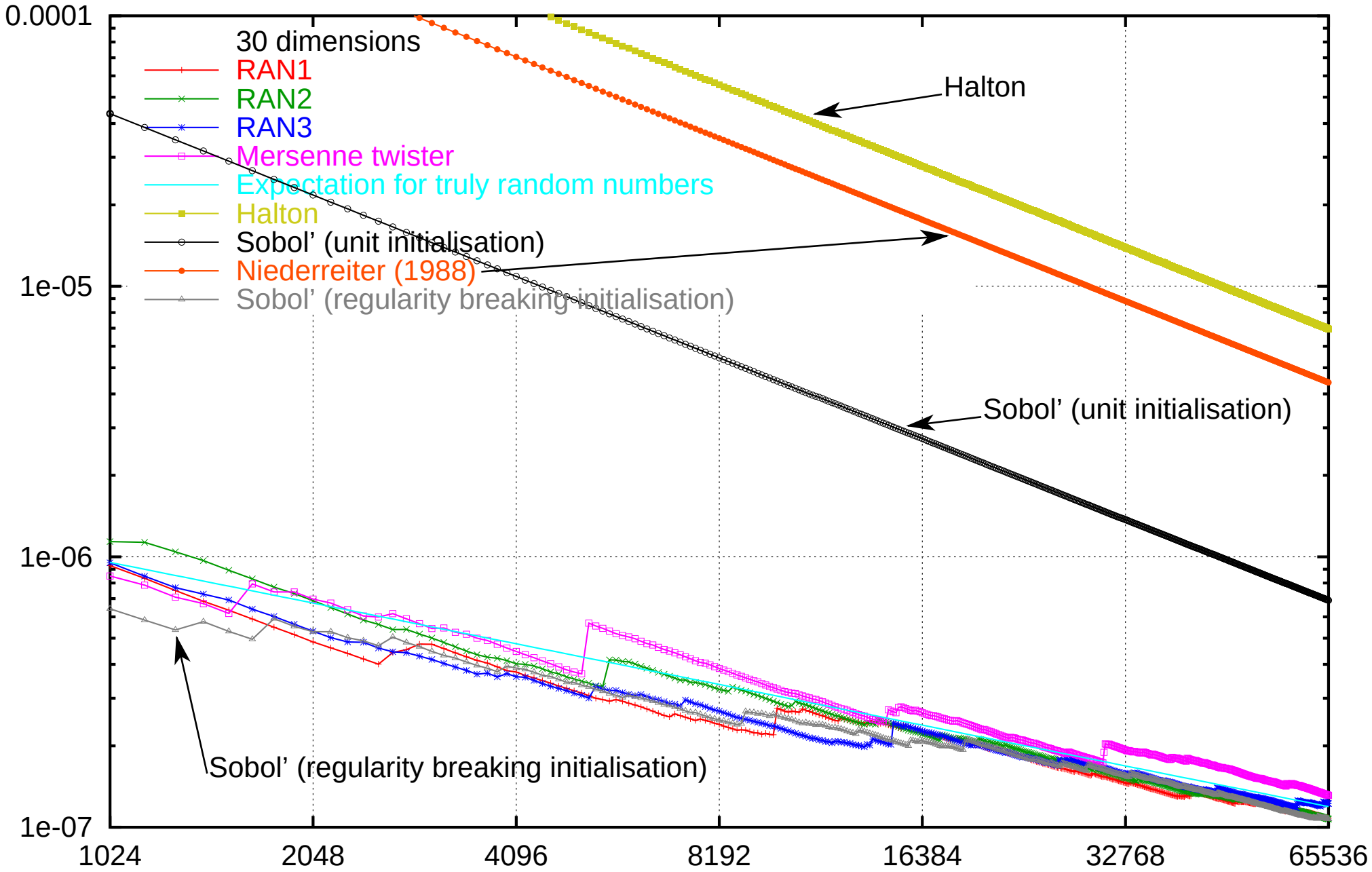

0.01
0.001
0.0001
1e-05
1024
2048
4096
8192
16384
32768
65536
Halton
2 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Niederreiter (1988)
Sobol' (no initialisation liberty for d=2)

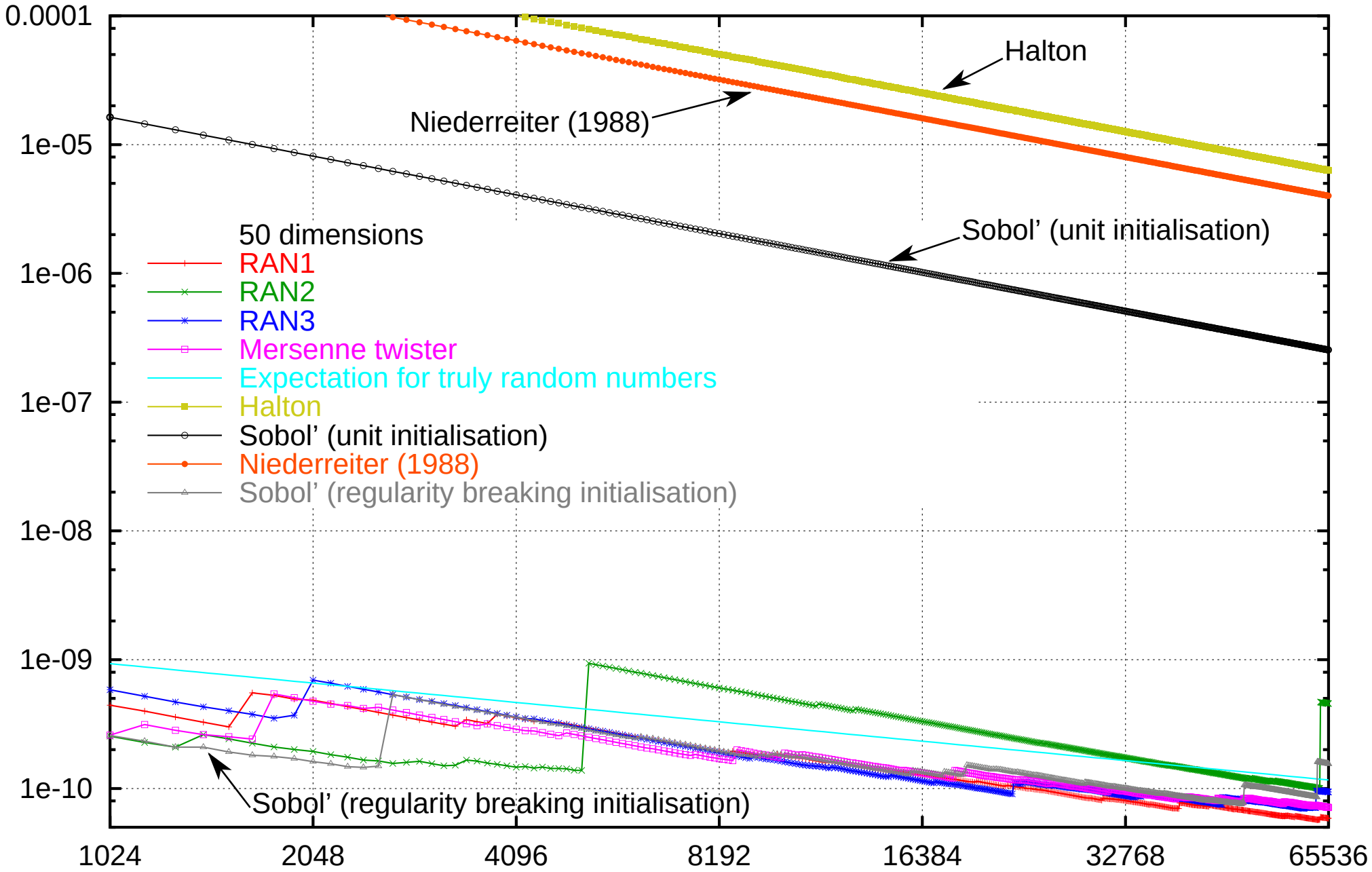

0.01
0.001
0.0001
1024
2048
4096
8192
16384
32768
65536
Halton
3 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)

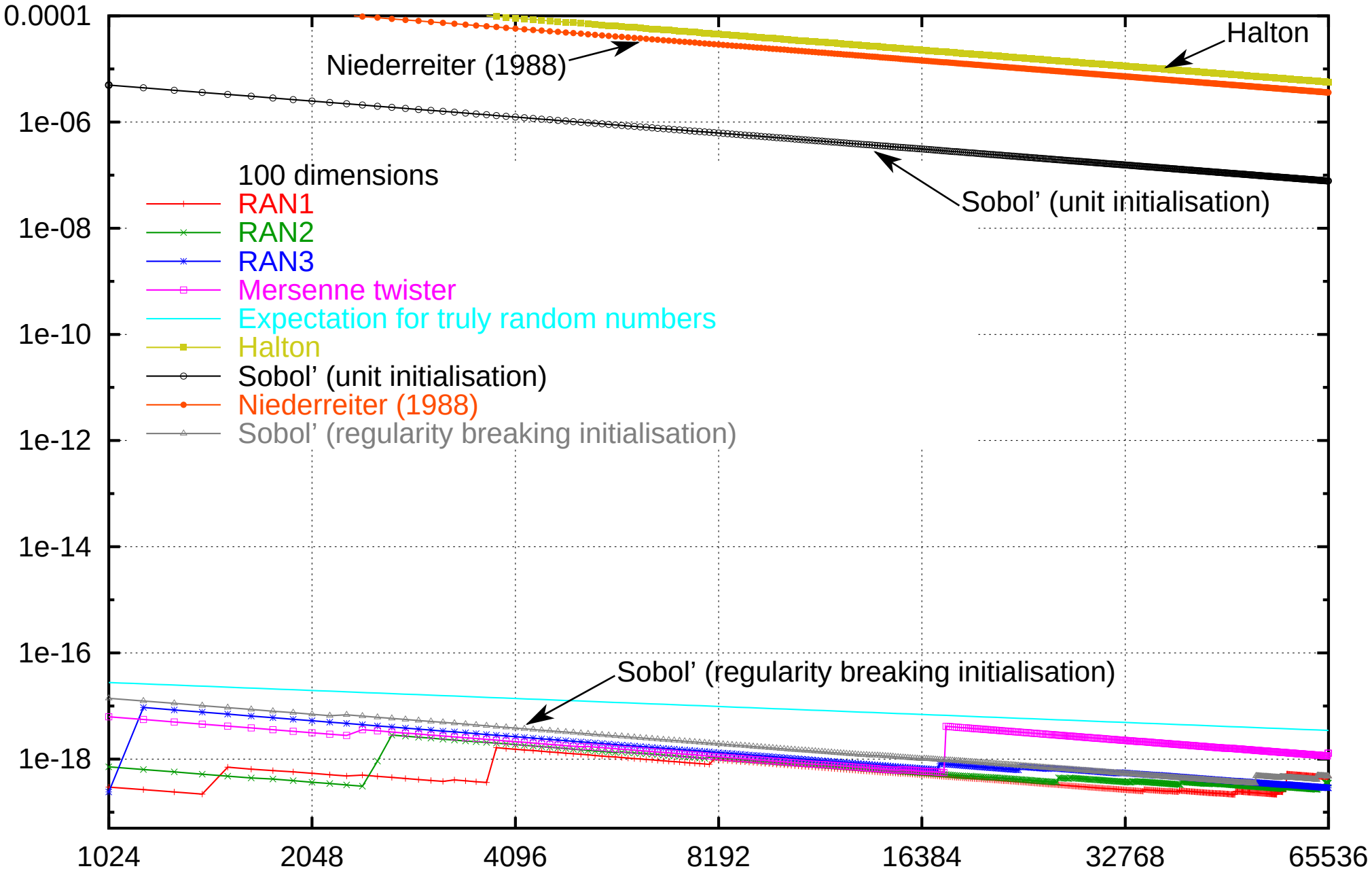

0.001
0.0001
1024
2048
4096
8192
16384
32768
65536
Halton
5 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)


0.001
0.0001
1024
2048
4096
8192
16384
32768
65536
Halton
10 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)


0.0001
1024
2048
4096
8192
16384
32768
65536
Halton
15 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)


0.0001
1e-05
1e-06
1e-07
1024
2048
4096
8192
16384
32768
65536
30 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)
Halton
Sobol' (unit initialisation)
Sobol' (regularity breaking initialisation)


0.0001
1e-05
1e-06
1e-07
1e-08
1e-09
1e-10
1024
2048
4096
8192
16384
32768
65536
Halton
Niederreiter (1988)
Sobol' (unit initialisation)
Sobol' (regularity breaking initialisation)
50 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)


0.0001
1e-06
1e-08
1e-10
1e-12
1e-14
1e-16
1e-18
1024
2048
4096
8192
16384
32768
65536
Halton
Niederreiter (1988)
Sobol' (unit initialisation)
Sobol' (regularity breaking initialisation)
100 dimensions
RAN1
RAN2
RAN3
Mersenne twister
Expectation for truly random numbers
Halton
Sobol' (unit initialisation)
Niederreiter (1988)
Sobol' (regularity breaking initialisation)

VII. Non-uniform variates

Inverting the cumulative distribution function

Given a uniform variate $u \in (0,1)$, the distribution density $\psi(x)$, its integral $\Psi(x)$ (i.e. the cumulative distribution function), and the inverse of the cumulative distribution function $\Psi^{-1}(u)$, the variate $x := \Psi^{-1}(u)$ is distributed $\sim \psi(x)$.

Whenever the cumulative distribution function can be accurately inverted at moderate computational cost, this is the preferred method of choice.

When an efficient and accurate approximation for the inverse of $\Psi(x)$ is not available but $\Psi(x)$ is, an efficient inverse lookup table can be set up as follows:-

1. Choose the number of interpolation points to be used.

2. Populate a vector of values x_i such that the whole domain of x is reasonably equally covered.

3. Compute the associated cumulative probability values $\Psi(x_i)$ and store as elements u_i in a second vector.

4. Build an interpolation object of at least second order interpolation accuracy using the vector $\boldsymbol{u}$ as abscissa, and the vector $\boldsymbol{x}$ as ordinate. Simple polynomials of third or higher order are usually already sufficient but the more accurate the better [Kva00]. Ensure that your interpolation object selects the interpolation bracket given any lookup variate u using a binary nesting algorithm.

5. The resulting interpolation object now represents a numerical instantiation of the inverse cumulative function $\Psi^{-1}(u)$

The use of such an interpolation object for the transformation of uniform $(0,1)$ variates to a target distribution $\psi(x)$ is highly efficient and completely generic.

Ensure that $u \in (0,1)$, i.e. that $u \neq 0$ and $u \neq 1$ since at least one of the two end points would for most distributions of interest result in `NaN` (also known as `#Num`).

Using a sampler density

Assume that the target density, $\psi(x)$, is much harder to match directly than a very similar density, $\widetilde{\psi}(x)$, and that $\widetilde{\psi}(x)$ is non-zero wherever $\psi(x)$ is non-zero.

Since

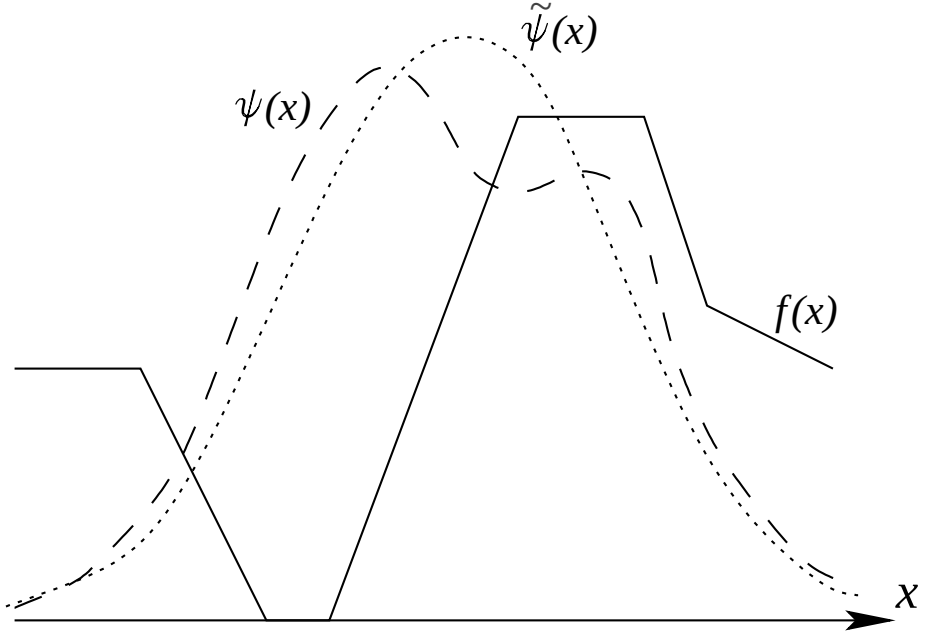
$$\int f(x)\cdot\psi(x)\,\mathrm{d}x \;=\; \int f(x)\frac{\psi(x)}{\widetilde{\psi}(x)}\cdot\widetilde{\psi}(x)\,\mathrm{d}x \tag{23}$$

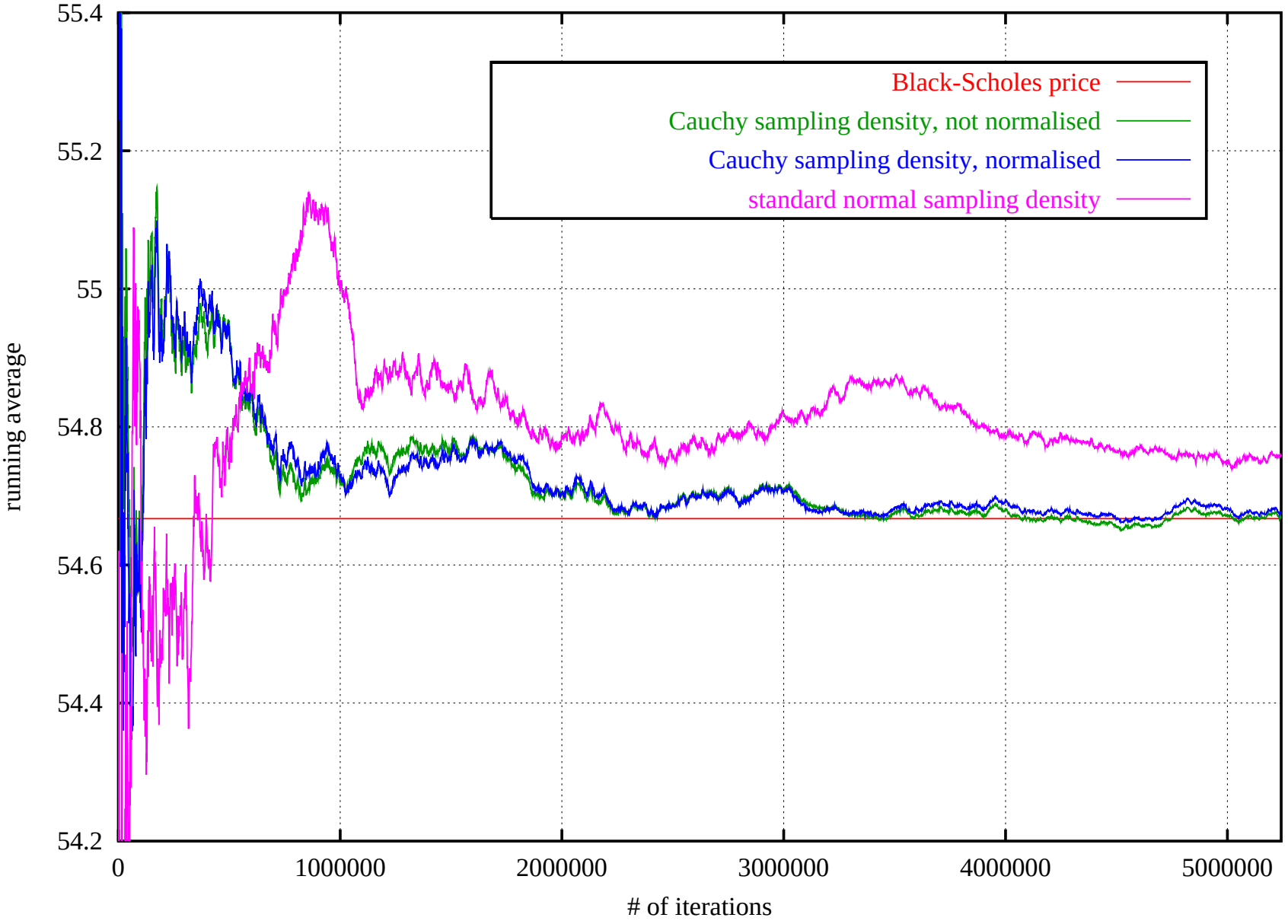
we can, instead of (16), use the *sampler density Monte Carlo estimator*:

$$\hat{v}_N \;:=\; \frac{1}{N}\sum_{i=1}^{N} f(x_i)\left(\frac{\psi(x_i)}{\widetilde{\psi}(x_i)}\right) \quad . \tag{24}$$

When $f(x)$ is nearly constant, it may be preferable to use the *normalised sampler density Monte Carlo estimator*:

$$\hat{v}_N := \frac{\sum_{i=1}^{N} f(x_i)\left(\frac{\psi(x_i)}{\widetilde{\psi}(x_i)}\right)}{\sum_{i=1}^{N}\left(\frac{\psi(x_i)}{\widetilde{\psi}(x_i)}\right)} \quad . \tag{25}$$





Pricing an out-of-the-money call option using a Cauchy sampling density $\widetilde{\psi}(x) = \frac{1}{\pi} \cdot \frac{1}{1+x^2}$ with $\widetilde{\Psi}^{-1}(u) = tan\left(\pi \cdot (u - 1/2)\right)$.

Importance sampling

is but a suitable choice of a sampler density such that the Monte Carlo simulation to compute

$$\mathsf{E}_{\widetilde{\psi}(x)}\left[f(x)\left(\frac{\psi(x_i)}{\widetilde{\psi}(x_i)}\right)\right]$$

converges more rapidly than

$$\mathsf{E}_{\psi(x)}[f(x)]\ ,$$

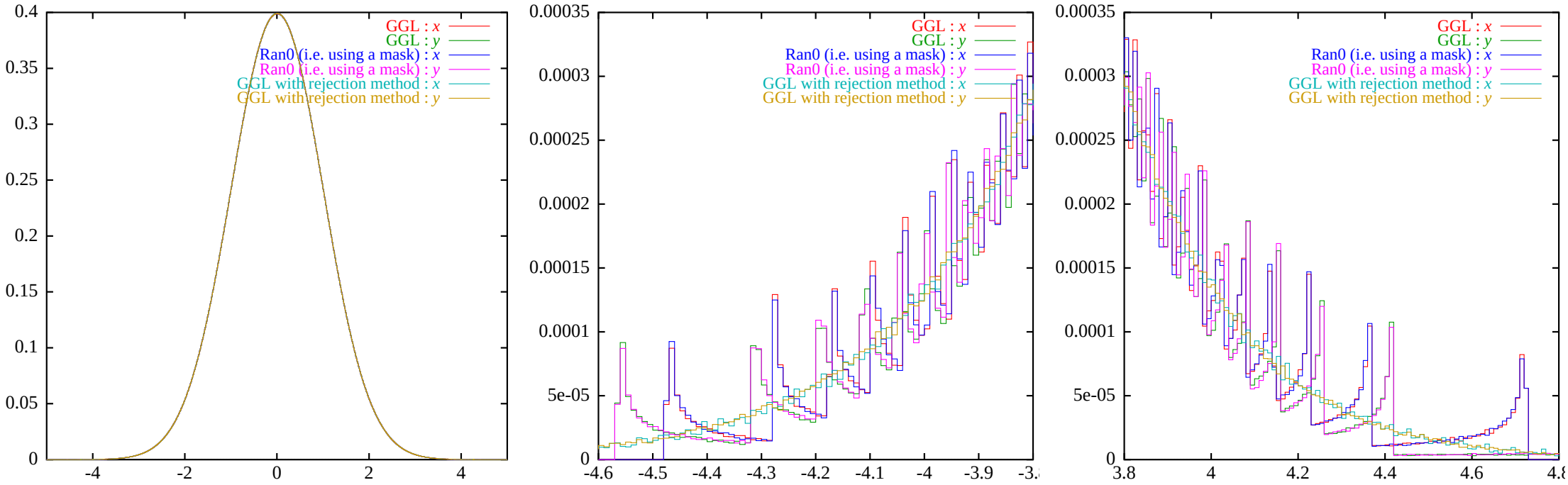
even though their convergence levels are identical.

Normal variates

- The preferred method should always be Peter Acklam's direct inverse cumulative normal [Ack00].
- *Never* use Excel's `Application.NormSInv` for any simulation.
- Box-Muller is deprecated for three reasons:
  1. It requires a pairwise pipelined generator.
  2. It is less efficient than the direct inverse cumulative normal.
  3. The Neave effect and similar entrainments.

The Neave effect

On a global scale, all combinations look pretty much the same



The dramatic Neave [Nea73] effect above only arises for the direct trigonometric version of the Box-Muller algorithm [BM58].

However, there are indications that other undesirable effects [Tez95] can arise due to the interaction of the nonlinear coupling of the transformation scheme and the number generation algorithm itself. In the physical and mathematical sciences of nonlinear dynamics, this phenomenon is known as *entrainment*. It is practically impossible to predict for what number generators it can arise. It is much safer to stay away from transformation methods that introduce unintended coupling.

VIII. Efficiency ratio and yield

Assume that we are using a variate generation algorithm based on the sampler density $\chi(x)$, and that the target density is $\psi(x)$, with the domain where ψ is non-zero being a subset of the domain where χ is non-zero. This means that for any x, we have

$$\psi(x) > 0 \quad \Rightarrow \quad \chi(x) > 0 \tag{26}$$

$$\chi(x) \quad \leq \quad \psi(x) \tag{27}$$

The standard error of a simulation based on χ whose objective is to compute $\mathrm{E}_{\psi(x)}[f(x)]$ is governed by the variance

$$V_\chi[f] \;=\; \int \left(f(x) \frac{\psi(x)}{\chi(x)} \right)^2 \chi(x)\,\mathrm{d}x \;\; - \;\; \left(\int f(x) \frac{\psi(x)}{\chi(x)} \chi(x)\,\mathrm{d}x \right)^2 . \tag{28}$$

We define the *efficiency ratio* of the Monte Carlo simulation based on the

sampler density $\chi(x)$ as

$$\eta_\chi[f] \;:=\; \frac{V_\psi[f]}{V_\chi[f]}\;. \tag{29}$$

Note that the efficiency ratio is a functional of the function f whose expectation we intend to compute.

Next, we define the *yield* of the Monte Carlo simulation based on the sampler density $\chi(x)$ as

$$y_\chi[f] \;:=\; \frac{\int f^2(x)\psi(x)\,\mathrm{d}x}{\int f^2(x)\frac{\psi(x)}{\chi(x)}\psi(x)\,\mathrm{d}x}\;. \tag{30}$$

Without any knowledge of f, we can use the *unbiased yield*

$$y_\chi \;:=\; \left(\int \frac{\psi(x)}{\chi(x)}\psi(x)\,\mathrm{d}x\right)^{-1} \tag{31}$$

to assess the computational burden of a sampler density algorithm.

Note that, whilst the functional yield $y_\chi[f]$ can be greater than 100% (importance sampling methods), the unbiased yield y_χ cannot.

IX. Codependence

Given the joint distribution density $\psi(x, y)$ of the two variables x and y, the *marginal* distribution density function of x is defined as

$$\psi_x(x) = \int \psi(x, y)\, \mathrm{d}y \ , \tag{32}$$

and analogously,

$$\psi_y(y) = \int \psi(x, y)\, \mathrm{d}x \ . \tag{33}$$

The marginal distribution density of any one of the two variables is nothing other than the probability density disregarding the value of the second variable.

Two variates x and y are considered *independent* if their joint distribution density function separates into the product of their individual distribution density functions, i.e.

$$\psi(x, y) = \psi_x(x)\psi_y(y) \ . \tag{34}$$

There are several ways to measure codependence.

Linear correlation

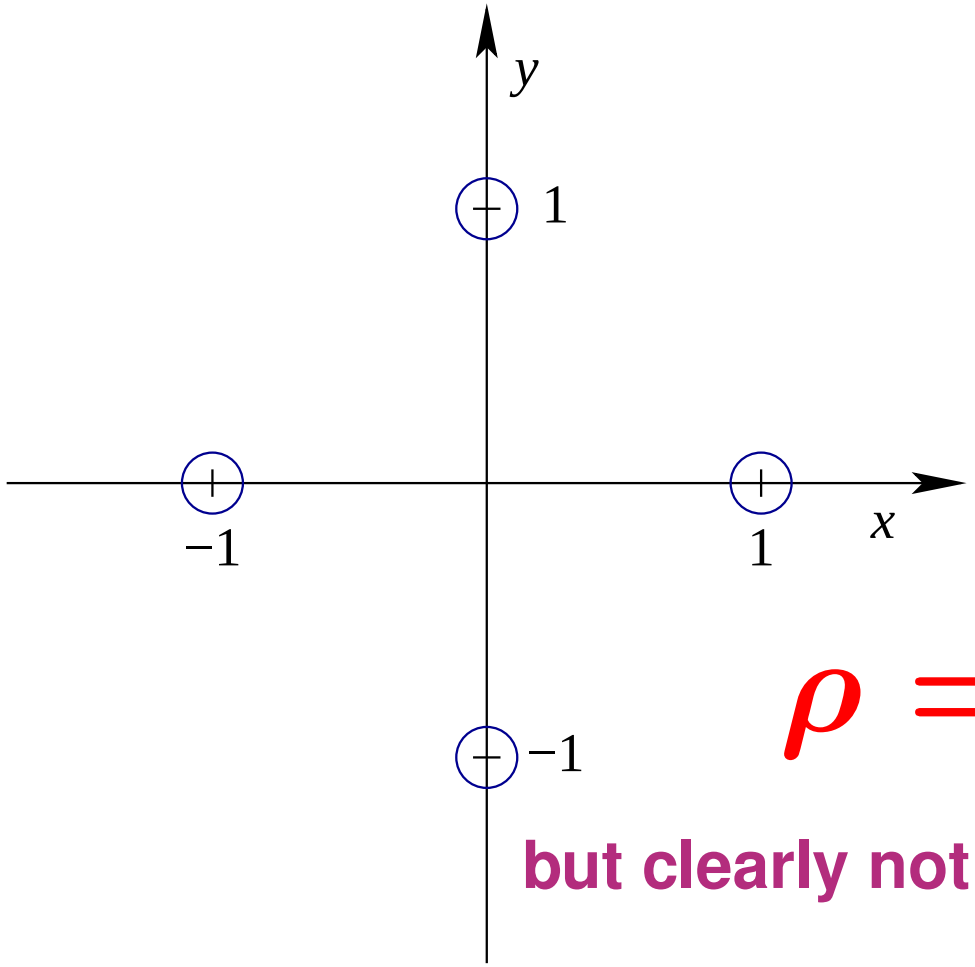
The *linear correlation* $\rho_{xy} := \mathsf{Corr}[x, y]$ of two variates x and y is defined as

$$\rho_{xy} = \frac{\mathsf{Cov}[x, y]}{\sqrt{\mathsf{V}[x]\,\mathsf{V}[y]}} = \frac{\int xy\psi(x, y)\mathrm{d}x\mathrm{d}y - \int x\psi_x(x)\mathrm{d}x \int y\psi_y(y)\mathrm{d}y}{\sqrt{\int x^2\psi_x(x)\mathrm{d}x - \left[\int x\psi_x(x)\mathrm{d}x\right]^2}\sqrt{\int y^2\psi_y(y)\mathrm{d}y - \left[\int y\psi_y(y)\mathrm{d}y\right]^2}}\,. \tag{35}$$

Linear correlation is a good measure for the co-dependence of normal variates.

For strongly non-normal distributions, it can be highly misleading.

Consider the discrete distribution where the variate pair (x, y) can take on the possible combinations $\{(0, 1), (0, -1), (1, 0), (-1, 0)\}$ with equal probability:



$$\rho = 0,$$

but clearly not independent!

Correlation is a term that makes sense only for some types of codependence.

Spearman's rho

Spearman's rho is defined as the linear correlation coefficient of the probability-transformed variates, i.e. of the variates transformed by their own cumulative marginal distribution functions:

$$\rho_{\mathrm{S}} := \frac{\iint \Psi_x(x)\Psi_y(y)\psi(x,y)\mathrm{d}x\mathrm{d}y - \int \Psi_x(x)\psi_x(x)\mathrm{d}x \int \Psi_y(y)\psi_y(y)\mathrm{d}y}{\sqrt{\int \Psi_x(x)^2\psi_x(x)\mathrm{d}x - \left[\int \Psi_x(x)\psi(x)\mathrm{d}x\right]^2}\sqrt{\int \Psi_y(y)^2\psi_y(y)\mathrm{d}y - \left[\int \Psi_y(y)\psi(y)\mathrm{d}y\right]^2}} \tag{36}$$

Spearman's rho can be expressed as

$$\rho_{\mathrm{S}} := 12\iint \Psi_x(x)\Psi_y(y)\psi(x,y)\,\mathrm{d}x\mathrm{d}y - 3\ . \tag{37}$$

Spearman's rho is independent with respect to variable transformations, whether linear or not.

Kendall's tau

For continuous distributions, *Kendall's tau* is given by

$$\tau_{\mathsf{K}} = 4\iint \Psi(x,y)\psi(x,y)\,\mathrm{d}x\mathrm{d}y - 1\ . \tag{38}$$

Since Kendall's tau is defined on the joint cumulative probability, it is also invariant with respect to transformations.

Kendall's tau and Spearman's rho belong to the category of *rank correlations*.

Rank correlations have the nice property that for any two marginal distribution densities $\psi_x(x)$ and $\psi_y(y)$, there always exists a joint distribution density $\psi(x,y)$ for every possible value in $[-1,1]$ of the rank correlation.

This is not necessarily guaranteed for linear correlation!

Gaussian variates

Given the covariance matrix C, and *any* pseudo-square root A of C such that $C = A \cdot A^\top$, a vector $\boldsymbol{z}$ of independent Gaussian variates can be transformed into a vector of correlated Gaussian variates $\boldsymbol{x}$ by setting

$$\boldsymbol{x} = A \cdot \boldsymbol{z} \ . \tag{39}$$

The covariances of the entries of the vector x are then

$$\begin{aligned}
\langle \boldsymbol{x} \cdot \boldsymbol{x}^\top \rangle &= \langle A \cdot \boldsymbol{z} \cdot \boldsymbol{z}^\top \cdot A^\top \rangle \\
&= A \cdot \langle \boldsymbol{z} \cdot \boldsymbol{z}^\top \rangle \cdot A^\top \\
&= A \cdot \mathbf{1} \cdot A^\top \\
&= A \cdot A^\top \\
&= C
\end{aligned}$$

as desired.

For specific covariance matrices, individual ways to compute a pseudo-square root are available. For instance, the covariance matrix of the path realisations of a Wiener process can be split into the *Brownian bridge*.

The two most popular methods to split any *generic* covariance matrix C into $A \cdot A^\top$ are:-

- Cholesky decomposition.

  Advantages: the effort in computing $\boldsymbol{x} = A \cdot \boldsymbol{z}$ for $\boldsymbol{z} \in \mathbb{R}^n$ is only $n \cdot \frac{n+1}{2}$.

  Disadvantages:

  – Telescopic dependence of the last variates on the first makes this method susceptible to roundoff error accumulation.
  – The standard Cholesky method as found in textbooks on numerical analysis fails for rank-deficient or ill conditioned covariance matrices. However, with some extra care, this problem can be remedied since the Cholesky algorithm can be readily amended to handle rank deficiencies.

- Spectral decomposition. Take the eigendecomposition

$$C = S \cdot \Lambda \cdot S^\top \tag{40}$$

with Λ being a diagonal matrix of non-negative entries and set

$$A := S \cdot \sqrt{\Lambda}\,. \tag{41}$$

Disadvantages: the effort in computing $\boldsymbol{x} = A \cdot \boldsymbol{z}$ for $\boldsymbol{z} \in \mathbb{R}^n$ is always n^2.

Advantages:

– Rank deficiencies are handled gracefully, especially if a stable eigensystem calculation algorithm is used.
– The resulting matrix A can be designed to associate the modes responsible for the majority of the variance in C with any particular entry of $\boldsymbol{z}$ at will.

Copulæ

A $copula$ of two variables x and y is a cumulative probability function defined directly as a function of the marginal cumulative probabilities of x and y.

A copula of n variables is a function $C : [0,1]^n \to [0,1]$.

$$\Psi(x,y) = C\left(\Psi_x(x), \Psi_y(y)\right) \ . \tag{42}$$

For strictly increasing cumulative marginals $\Psi_x(x)$ and $\Psi_y(y)$, we can also write

$$C(u,v) = \Psi\left(\Psi_x^{-1}(u), \Psi_y^{-1}(v)\right) \ . \tag{43}$$

The copula of independent variables, not surprisingly, is given by

$$C_{\text{independent}}(u,v) = u \cdot v. \tag{44}$$

By virtue of the definition on the cumulative marginal distribution functions, the copula of a set of variables (x,y) is invariant with respect to a set of strictly increasing transformations $(f(x), g(y))$.

The Gaussian Copula

Assume that we are given a symmetric, positive-semidefinite matrix R of codependence coefficients that are to be used to generate vectors of codependent uniform $(0,1)$ variates:

- Find a suitable pseudo-square root A of R such that $R = A \cdot A^\top$.
- Draw a vector $\boldsymbol{z} \in \mathbb{R}^n$ of uncorrelated standard normal variates.
- Compute $\tilde{\boldsymbol{z}} := A \cdot \boldsymbol{z}$.
- Map $\tilde{\boldsymbol{z}}$ back to a vector of uniform variates $\boldsymbol{v} \in [0,1]^n$ by setting $v_i = \mathsf{N}(\tilde{z}_i)$.

It can be shown [LMS01, Kau01] that Kendall's tau of two variables connected by a Gaussian copula with codependence coefficient ρ is given by

$$\tau_{\mathsf{K}} = \frac{2}{\pi} \arcsin \rho \; . \tag{45}$$

Example: *Two uniform variates under the Gaussian copula*

The linear correlation η between the two dependent uniform variates under the Gaussian copula can be calculated:

$$\eta(\rho) = 12 \iint \mathsf{N}(z_1)\mathsf{N}(\rho z_1 + \sqrt{1-\rho^2} z_2)\varphi(z_1)\varphi(z_2)\,\mathrm{d}z_1\mathrm{d}z_2 - 3\ . \tag{46}$$

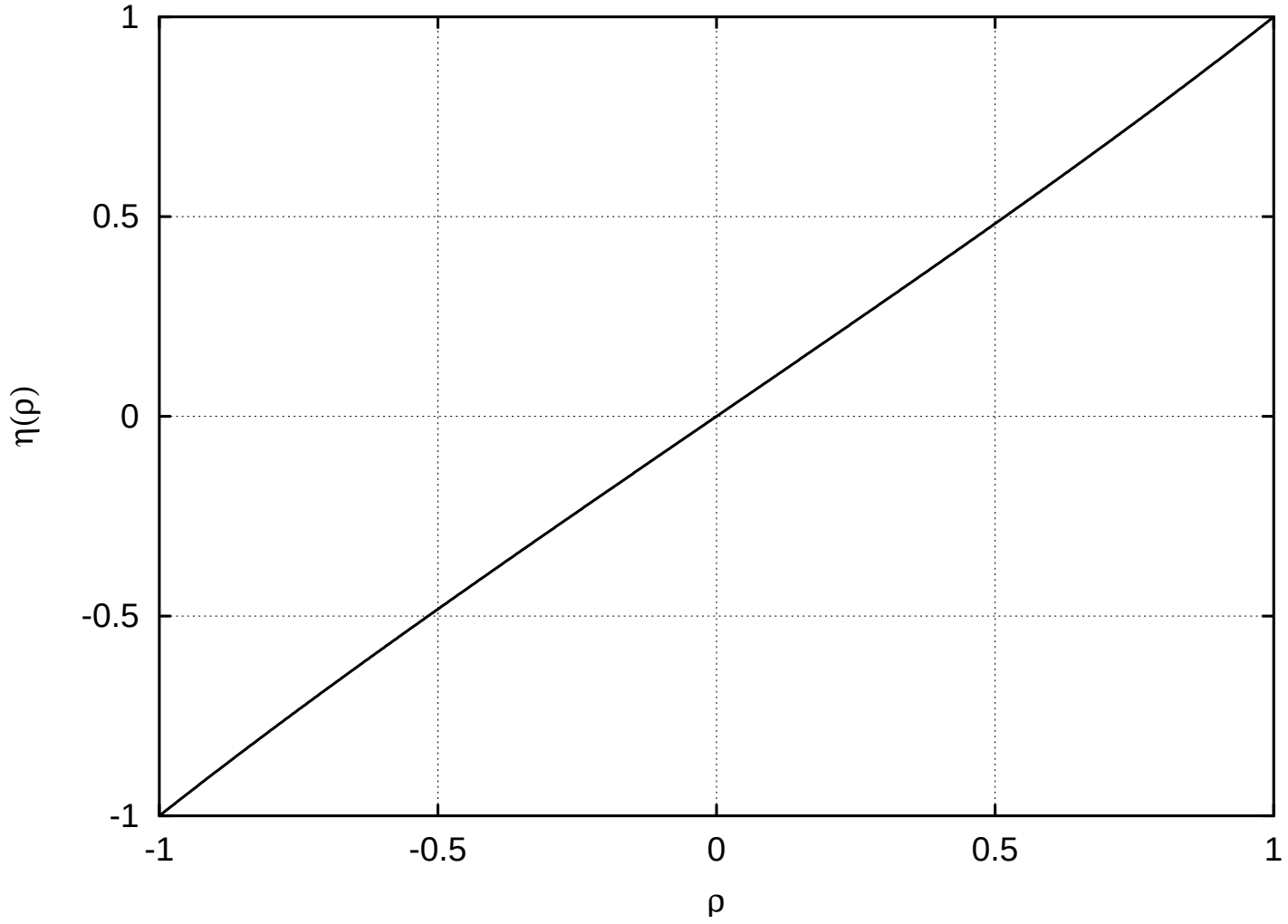
Using equation (26.3.19) in [AS84], namely

$$\mathsf{N}(0,0,\rho) \ = \ \rho_{\mathsf{S}} \ = \ \frac{1}{4} + \frac{1}{2\pi}\arcsin\rho\ ,$$

we have the exact relationship

$$\eta(\rho) \ = \ \rho_{\mathsf{S}} \ = \ 2\cdot\frac{3}{\pi}\arcsin\frac{\rho}{2}\ . \tag{47}$$

Near the origin, this means: $\eta(\rho) \ = \ \rho_{\mathsf{S}} \approx \frac{3}{\pi}\rho$ for $|\rho| \ll 1$.



Don’t be misled by the apparently straight line: there is a little bit of curvature in there, although a straight line would certainly be a good approximation for it.

Example: *Two exponential variates under the Gaussian copula*

$$\tau \sim \lambda \mathrm{e}^{-\lambda\tau} \ . \tag{48}$$

Random draws for τ can be generated from a uniform variate u by setting

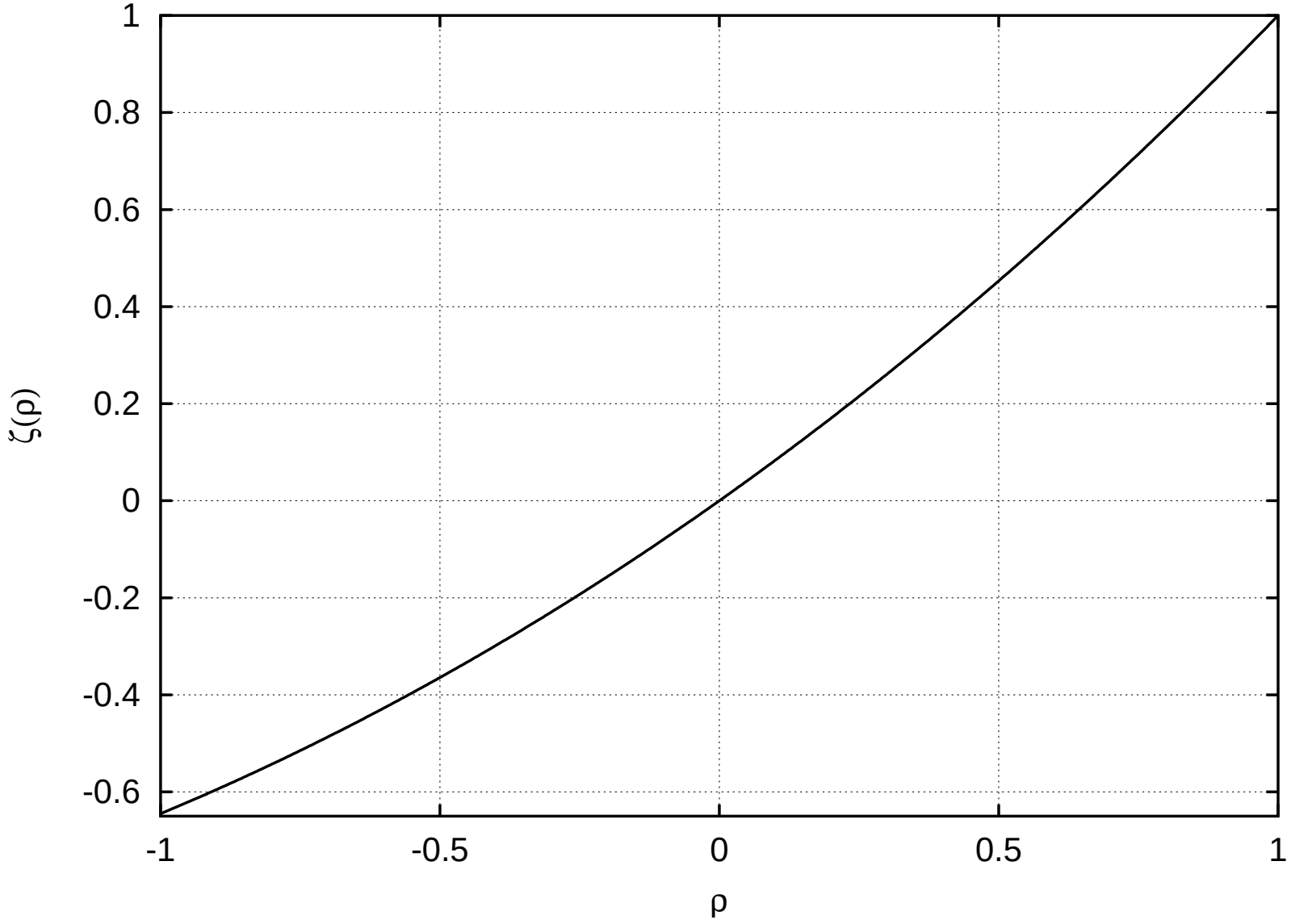
$$\tau = -\frac{\ln(1-u)}{\lambda} \ . \tag{49}$$

The linear correlation of two events τ_a and τ_b using a Gaussian copula with codependence coefficient ρ is given by

$$\zeta(\rho) = \iint \ln(1-\mathsf{N}(z_a))\ln(1-\mathsf{N}(\rho z_a + \sqrt{1-\rho^2}z_b))\varphi(z_a)\varphi(z_b)\,\mathrm{d}z_a\mathrm{d}z_b - 1 \ . \tag{50}$$

We can evaluate analytically what interval $\zeta(\rho)$ is confined to:

$$\zeta \in [1-\frac{\pi^2}{6}, 1] \quad \text{for} \quad \rho \in [-1,1] \ , \quad \text{where} \quad 1-\frac{\pi^2}{6} \approx -0.6449341.$$


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
ζ(ρ)
-1
-0.5
0
0.5
1
ρ

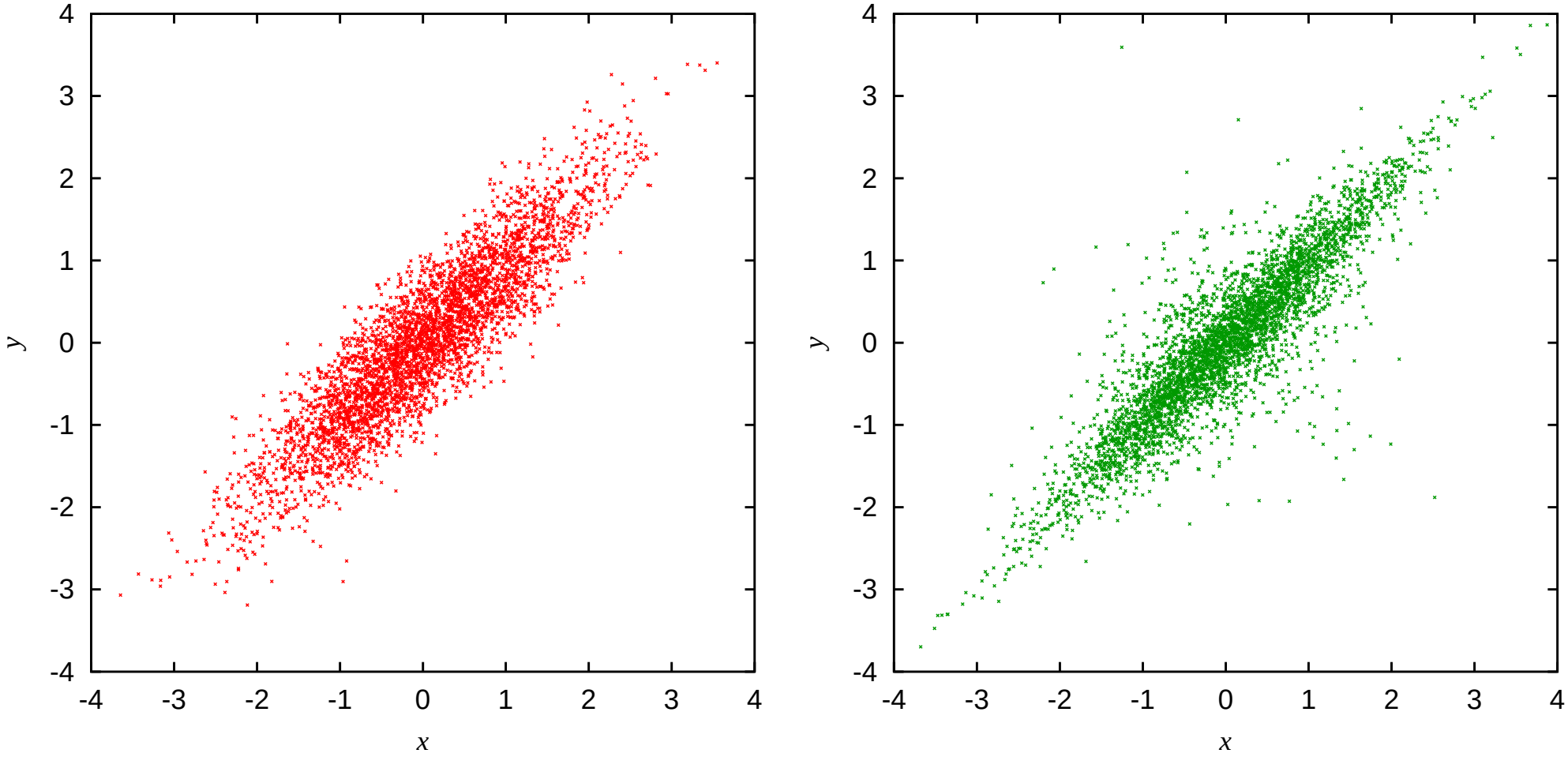
Generic elliptic copulæ

- Select a standard correlation coefficient matrix R and find a pseudo-square root A of R such that $R = A \cdot A^\top$.
- Draw a vector $\boldsymbol{z} \in \mathbb{R}^n$ of uncorrelated standard normal variates.
- Compute $\tilde{\boldsymbol{z}} := A \cdot \boldsymbol{z}$.
- Draw an independent, non-negative, variate q, from some distribution χ.
- Set $\boldsymbol{x} := q \cdot \tilde{\boldsymbol{z}}$.
- Map $\boldsymbol{x}$ back to a vector of uniform variates $\boldsymbol{v} \in [0,1]^n$ using the cumulative probability function of $q \cdot z$ with $q \sim \chi(q)$ and $z \sim \varphi(z)$ that has to be computed according to the choice of χ.

Kendall's tau for any pair of variates connected by a generic elliptic copula is also given by equation (45).

The t-copula

is an elliptic copula with the specific choice that $q := \sqrt{\frac{\nu}{s}}$ with $s \sim \chi^2_\nu(s)$, i.e. s is a chi-square variate with ν degrees of freedom. The distribution of $q \cdot z$ is that of a Student-t variate with ν degrees of freedom.



Both figures show data that are marginally normally distributed. The figure on the left depicts ordinary correlated normal variates with $\rho = 0.9$, and the figure on the right was created from data under a t_2-copula, also with $\rho = 0.9$.

There are many other copulæ. In particular, there is the family of *Archimedean copulæ*. All members of this class of copulæ have in common that they are generated by a strictly decreasing convex function $\phi(u)$ which maps the interval $(0,1]$ onto $[0,\infty)$ such that $\lim_{\epsilon\to 0}\phi(\epsilon)=\infty$ and $\phi(1)=0$. An Archimedean copula is generated from a given function $\phi(u)$ by

$$C(u,v)=\phi^{-1}\left(\phi(u)+\phi(v)\right)\ . \tag{51}$$

Two uniform variates u and v under any Archimedean copula can be produced by the following algorithm:

- Draw two independent uniform variates s and q.
- Solve

$$q=t-\frac{\phi(t)}{\phi'(t)} \tag{52}$$

  for t.

- Set

$$u := \phi^{-1}\left(s\phi(t)\right) \tag{53}$$

$$v := \phi^{-1}\left((1-s)\phi(t)\right) \ . \tag{54}$$

For further details see [ELM01].

The main drawback of these copulæ is that it is difficult to extend them to higher dimensions whilst allowing for one separate codependence coefficient for every possible pair.

X. Wiener path construction

In many applications, we need to construct a simulated discretised path of a standard Wiener process over a set $\{t_i\}, i = 1..n$, points in time.

A discretised Wiener path is given by the associated values $w_i := W(t_i)$.

We can view the values $w_i := W(t_i)$ of the Wiener process at those points in time as a vector of Gaussian variates.

The elements of their covariance matrix C, are given by

$$c_{ij} = \mathrm{Cov}[W(t_i), W(t_j)] = \min(t_i, t_j) \ . \tag{55}$$

The special structure of the covariance matrix gives us immediately three choices for the construction of paths.

Incremental

$$w_{i+1} = w_i + \sqrt{\Delta t_{i+1}} \cdot z_i, \quad \text{with} \quad z_i \sim \mathcal{N}(0,1)\ . \tag{56}$$

The construction matrix of the incremental method is given by the Cholesky decomposition of the covariance matrix:

$$A_{\text{incremental}} = \begin{pmatrix} \sqrt{\Delta t_1} & 0 & 0 & 0 & \cdots & 0 \\ \sqrt{\Delta t_1} & \sqrt{\Delta t_2} & 0 & 0 & \cdots & 0 \\ \sqrt{\Delta t_1} & \sqrt{\Delta t_2} & \sqrt{\Delta t_3} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \cdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sqrt{\Delta t_1} & \sqrt{\Delta t_2} & \sqrt{\Delta t_3} & \cdots & \cdots & \sqrt{\Delta t_n} \end{pmatrix} \tag{57}$$

The advantage of this method is that it is fast. For each path over n points in time, we need a total of n multiplications, and $n-1$ additions.

Spectral

The most efficient and reliable way to construct Wiener paths from a spectral decomposition of the pathwise covariance is to use the generic spectral decomposition approach.

Whilst certain analytical approximations are possible in the general case, and whilst an exact decomposition is known when all time steps are exactly equal, all in all it is hardly ever worth considering anything other than an efficient numerical eigensystem decomposition as described, for instance, in [PTVF92].

The Brownian bridge

Similar to the spectral path construction method, the *Brownian bridge* is a way to construct a discretised Wiener process path by using the first Gaussian variates in a vector draw $\boldsymbol{z}$ to shape the overall features of the path, and then add more and more of the fine structure.

The very first variate z_1 is used to determine the realisation of the Wiener path at the final time t_n of our n-point discretisation of the path by setting $W_{t_n} = \sqrt{t_n} z_1$.

The next variate is then used to determine the value of the Wiener process as it was realised at an intermediate timestep t_j conditional on the realisation at t_n (and at $t_0 = 0$ which is, of course, zero).

The procedure is then repeated to gradually fill in all of the realisations of the Wiener process at all intermediate points, in an ever refining algorithm.

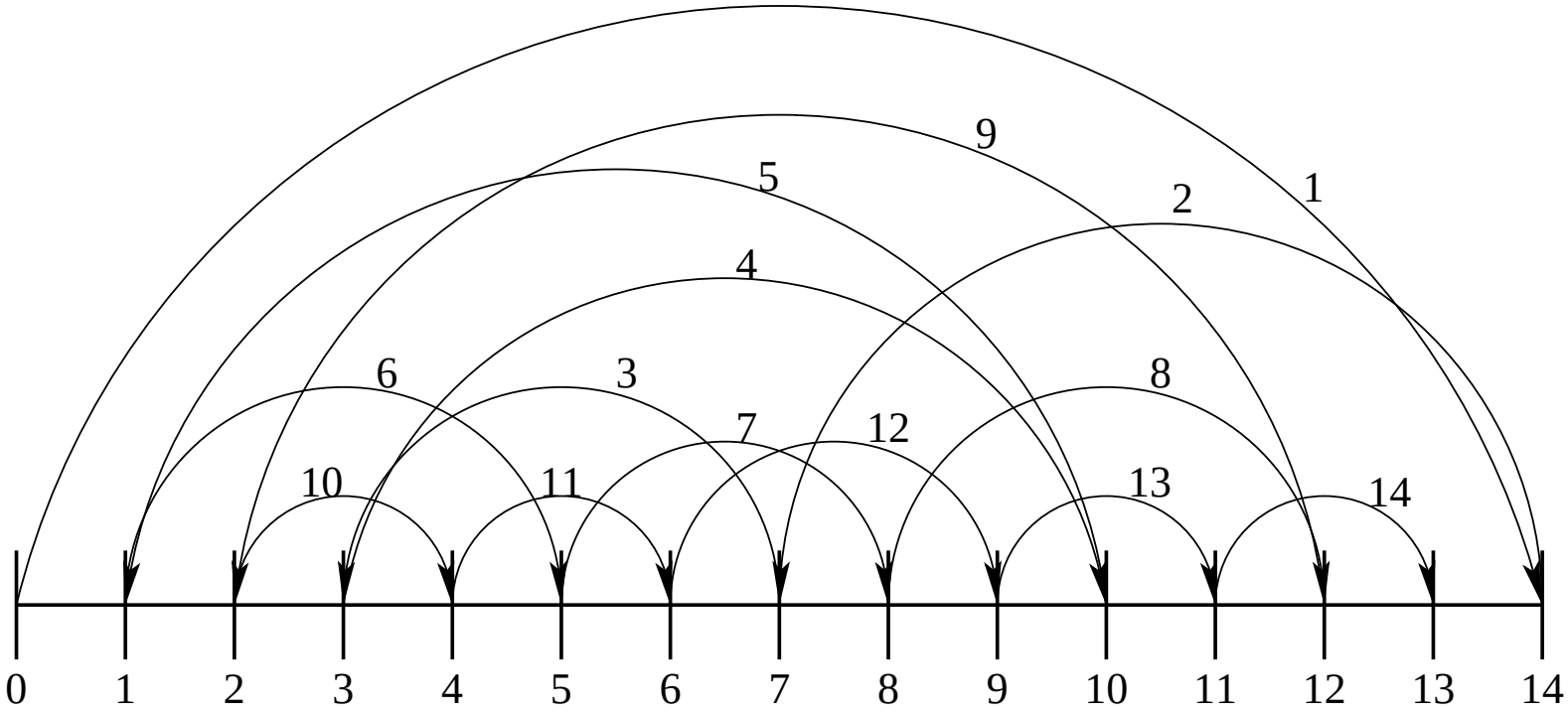
In each step of the refinement procedure to determine W_{t_j} given that we have already established W_{t_i} and W_{t_k} with $t_i < t_j < t_k$, we make use of the fact

that the conditional distribution of W_{t_j} is Gaussian with mean

$$\mathsf{E}\left[W_{t_j}\right] = \left(\frac{t_k - t_j}{t_k - t_i}\right) W_{t_i} + \left(\frac{t_j - t_i}{t_k - t_i}\right) W_{t_k} \tag{58}$$

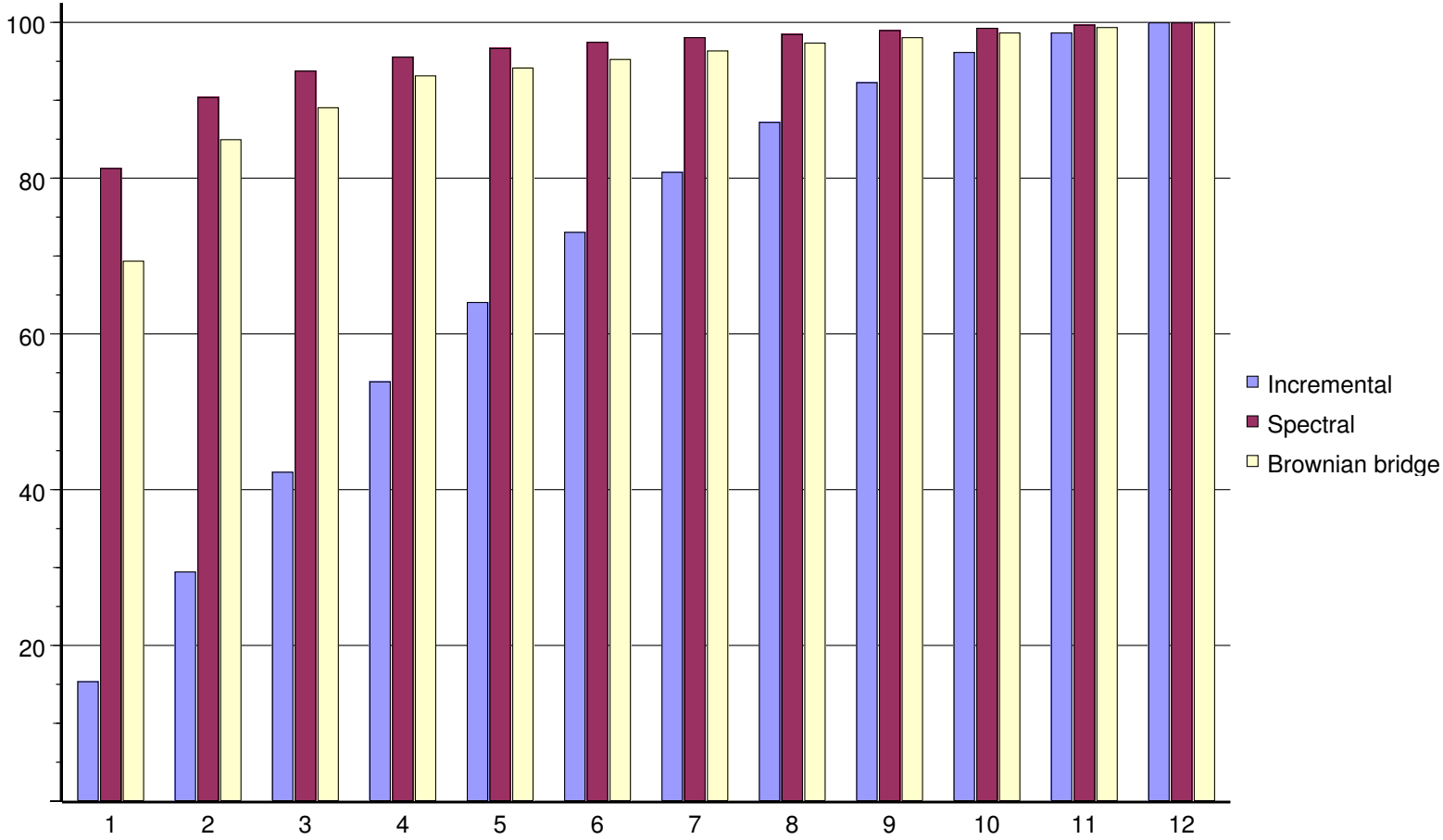
and variance

$$\mathsf{V}\left[W_{t_j}\right] = \frac{(t_j - t_i)\,(t_k - t_j)}{(t_k - t_i)}\,. \tag{59}$$



Comparison of path construction methods

The *variability explained* is the variance that remains if only the first m column vectors in a complete path construction matrix A are used. It is given by the sum of all of the squares of the elements of the first m column vectors of A.

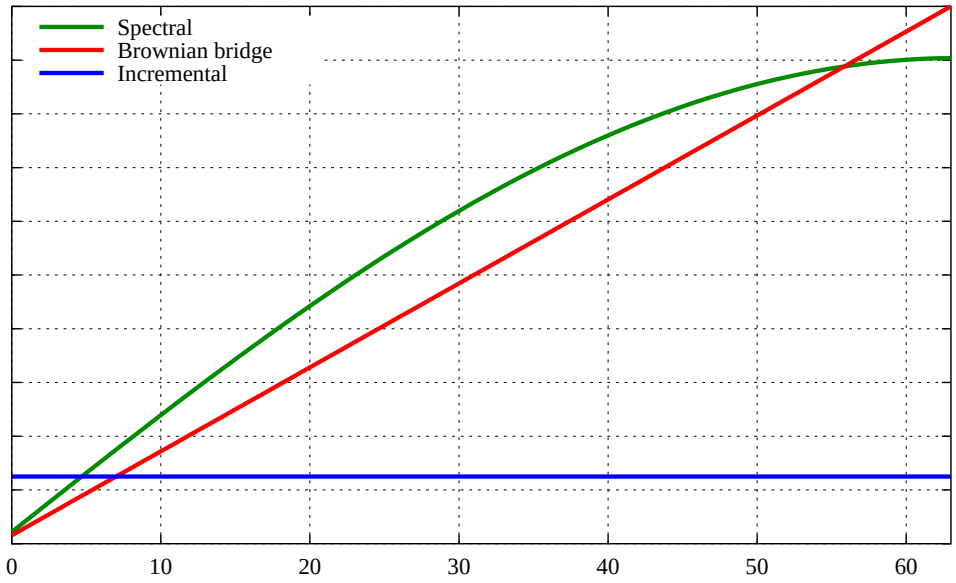


The cumulative variability explained (in %) by an increasing number of dimensions for three path construction methods over 12 equal time steps.

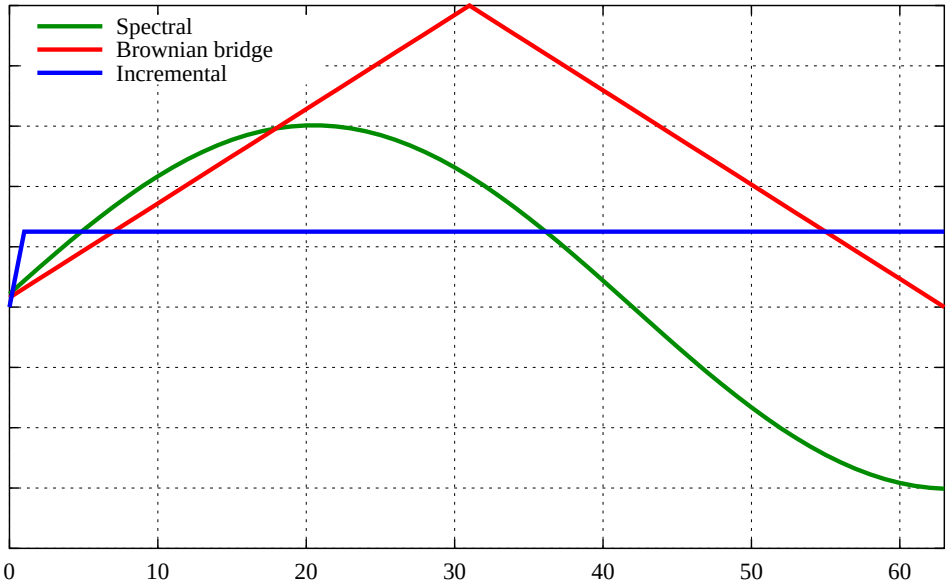
The Brownian bridge construction captures variability similarly to the spectral method whilst being almost as efficient as incremental path construction.

The construction modes of the Brownian bridge are akin to a piecewise linear approximation to the construction modes of the spectral method.

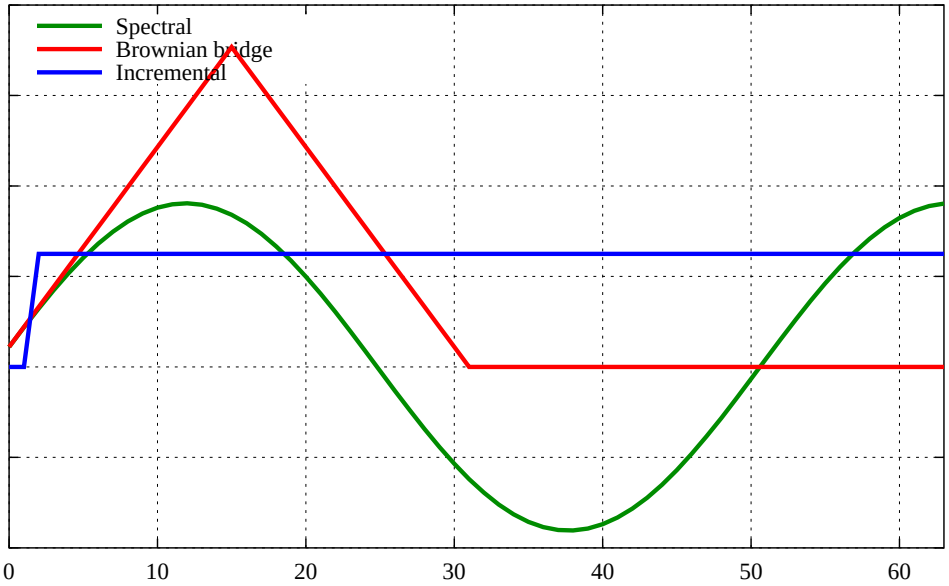
First column vectors of the construction matrices



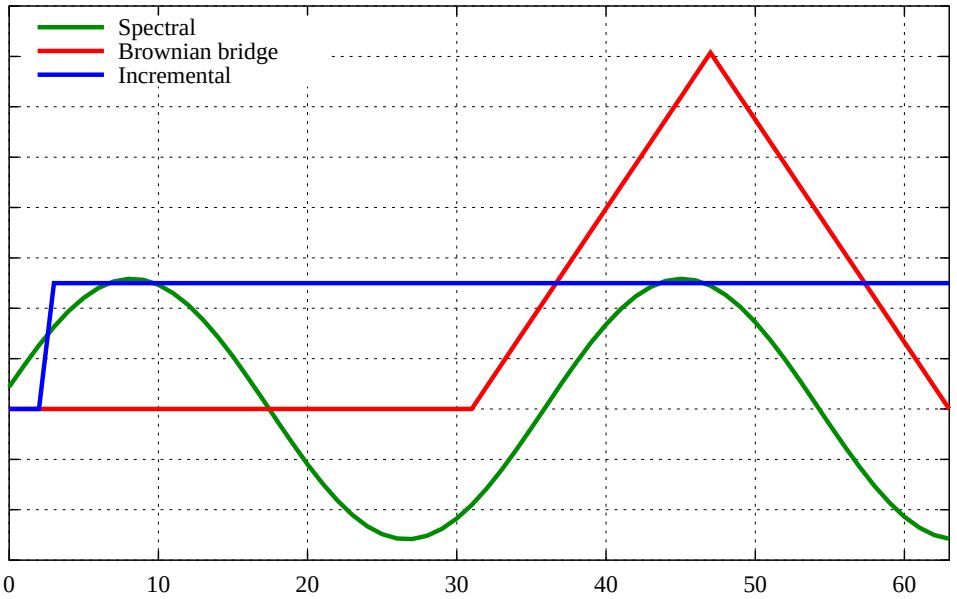
Second column vectors of the construction matrices



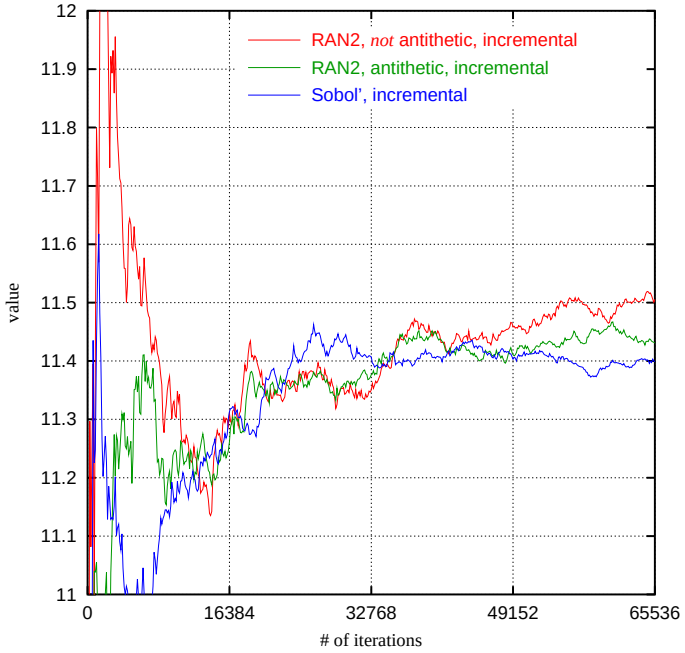
Third column vectors of the construction matrices

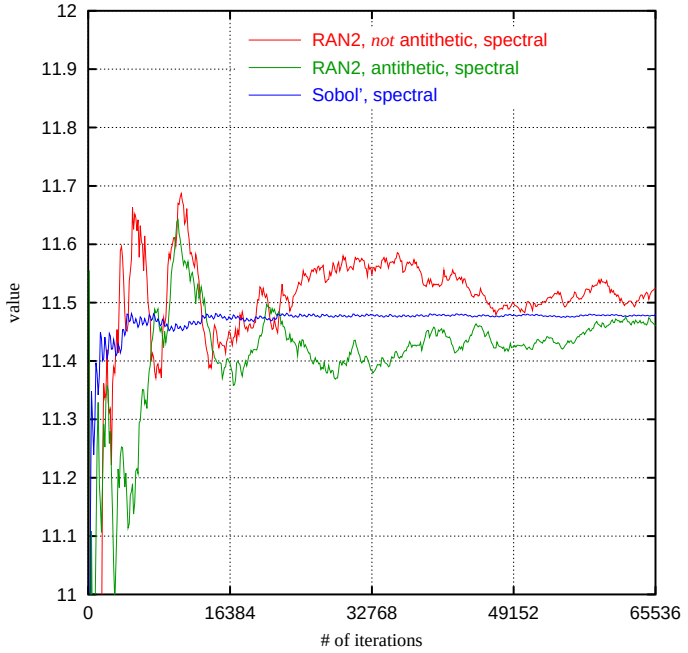


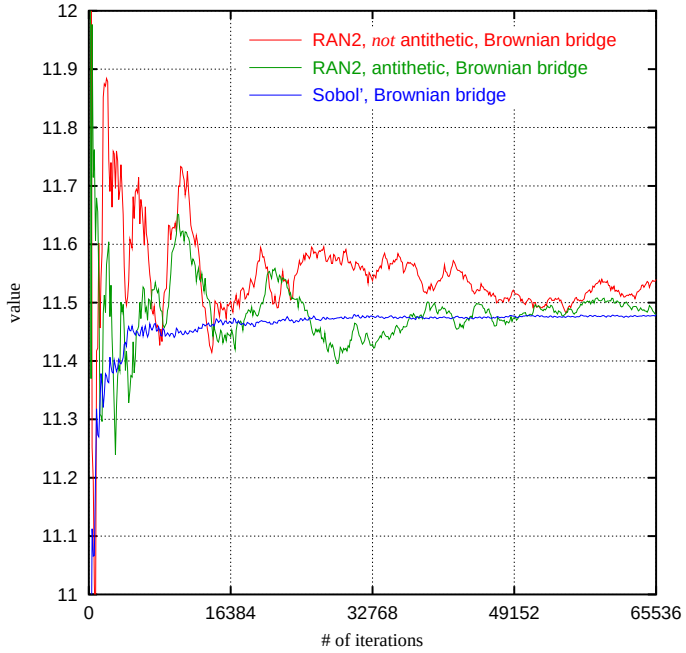
Fourth column vectors of the construction matrices



This explains the convergence behaviour we see for the different path construction methods in comparative Monte Carlo simulations.

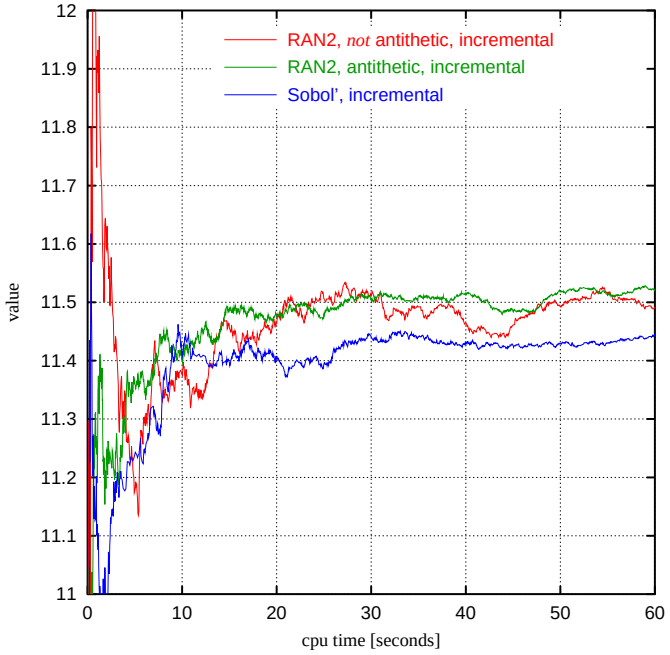


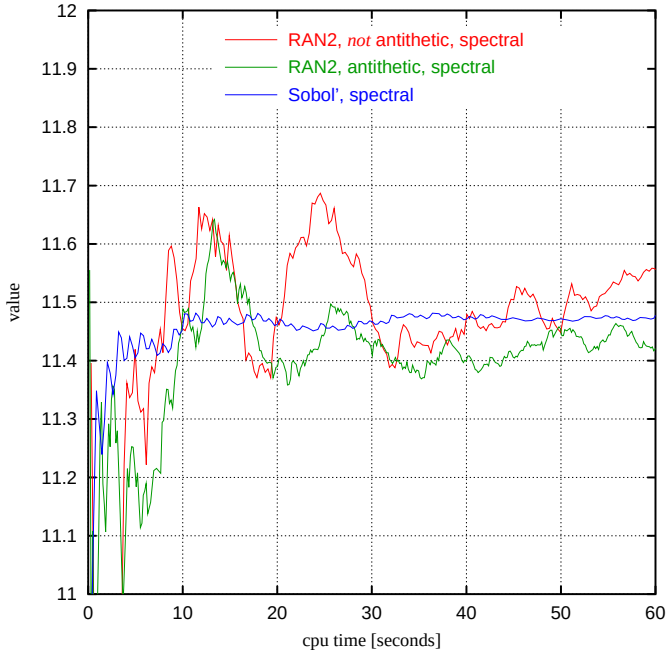


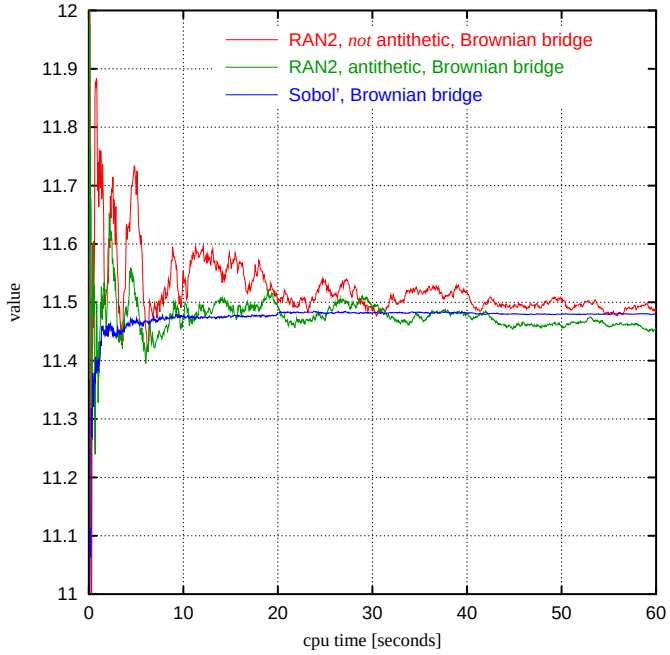


The example was a standard Asian option with one year to maturity and 252 monitoring points.

If we take into account the time spent in the path construction, the Brownian bridge immediately becomes the method of choice, irrespective of the specifics of the calculation.







XI. Poisson path construction

A Poisson process of intensity λ is given by the increments $\mathrm{d}N$ that over any infinitesimal timer interval $\mathrm{d}t$ are:-

$$\mathrm{d}N = \begin{cases} 1 & \text{with probability} \quad \lambda \mathrm{d}t \\ 0 & \text{with probability} \quad 1 - \lambda \mathrm{d}t \end{cases} . \tag{60}$$

A Poisson path is fully described by the set of event times. There are (at least) three ways to construct such a Poisson path:-

1. Draw Bernoulli variates over a time discretisation of interval size Δt. *Very bad*.

2. Draw independent exponential variates $\tau \sim \lambda \mathrm{e}^{-\lambda\tau}$ that represent the time between events. *Good and easy*.

3. Construct a *Poisson bridge* [Fox96]. *Excellent but somewhat more involved*.

A sketch of the Poisson bridge from 0 to T:

- Draw the number of events n to occur for this path in $(0, T)$ from the Poisson distribution
$$\mathrm{Pr}\left[n = k\right] = \mathrm{e}^{-\lambda T}\frac{(\lambda T)^k}{k!}\ . \tag{61}$$
- *Conditional on there being* n *events on* $(0, T)$, they are uniform on $(0, T)$.
- Set j to be the largest integer less than or equal to $n/2$.
- Draw the u_j from a beta distribution $\beta(j, n + 1 - j)$.
- Set the event time t_j to be $u_j \cdot T$.
- Now we need to fill $j - 1$ events into the interval $(0, t_j)$ and $n - j - 1$ events into the interval (t_j, T), which is to be done using the same algorithm after rescaling.

The generalization of a Poisson process leads to a *Gamma* process which can be constructed with a *Gamma bridge* [ALT03].

XII. Numerical integration of stochastic differential equations

Anyone ever considering numerical integration of SDEs should buy [KP99].

Let us assume that we have a stochastic differential equation of the form

$$\mathrm{d}X = a\,\mathrm{d}t + b\,\mathrm{d}W \ . \tag{62}$$

Note that both a and b can be functions of the process variable X and time. In the multi-dimensional case of m state variables X_i driven by d independent Wiener processes, we have

$$\mathrm{d}X_i = a_i(t, \boldsymbol{X})\,\mathrm{d}t + \sum_{j=1}^{d} b_{ij}(t, \boldsymbol{X})\,\mathrm{d}W_j \ . \tag{63}$$

The Euler scheme

Denote the numerical approximation to the solution of (62) for a scheme over equal steps of size Δt at time $n \cdot \Delta t$ as $Y(t_n)$. The explicit Euler scheme is then given by

$$Y(t_{n+1}) = Y(t_n) + a(t_n, Y(t_n))\,\Delta t + b(t_n, Y(t_n))\,\Delta W\ . \tag{64}$$

Not surprisingly, the implicit Euler scheme is given by

$$Y(t_{n+1}) = Y(t_n) + a(t_n, Y(t_{n+1}))\,\Delta t + b(t_n, Y(t_{n+1}))\,\Delta W\ . \tag{65}$$

The Milstein scheme

The Milstein scheme involves the addition of the next order terms of the Itô-Taylor expansion of equation (62). This gives

$$Y(t_{n+1}) = Y(t_n) + a(t_n, Y(t_n))\,\Delta t + b(t_n, Y(t_n))\,\Delta W + \tfrac{1}{2} b b' \left[\Delta W^2 - \Delta t\right] . \tag{66}$$

with

$$b' = \frac{\partial b(t, X)}{\partial X} . \tag{67}$$

The Milstein scheme is definitely manageable in the one-dimensional case.

However, its general multi-dimensional extension is not as straightforward as one may expect since it requires not only the drawing of standard normal variates for the simulation of the standard Wiener process increments ΔW for each dimension, but also additional ones to account for the Itô integrals involving the mixing terms $\sum_{j=1}^{d} b_{ij}(t, \boldsymbol{X})\,\mathrm{d}W_j$.

Predictor-Corrector

First take an explicit Euler step to arrive at the *predictor*

$$\bar{Y}_i(t_{n+1}) \quad = \quad Y_i(t_n) + a_i(t_n, \boldsymbol{Y}(t_n))\,\Delta t + \sum_{j=1}^{d} b_{ij}(t_n, \boldsymbol{Y}(t_n))\,\Delta W_j \ . \tag{68}$$

Next, select two weighting coefficients α and η in the interval $[0,1]$, usually near $1/2$, and calculate the *corrector*

$$\begin{aligned} Y_i(t_{n+1}) \quad = \quad & Y_i(t_n) + \left\{\alpha\,\bar{a}_i(t_{n+1}, \bar{\boldsymbol{Y}}(t_{n+1}); \eta) + (1-\alpha)\bar{a}_i(t_n, \boldsymbol{Y}(t_n); \eta)\right\}\Delta t \\ & + \sum_{j=1}^{m} \left\{\eta\, b_{ij}(t_{n+1}, \bar{\boldsymbol{Y}}(t_{n+1})) + (1-\eta) b_{ij}(t_n, \boldsymbol{Y}(t_n))\right\}\sqrt{\Delta t}\, z_j \end{aligned} \tag{69}$$

with

$$\bar{a}_i(t, \boldsymbol{Y}; \eta) := a_i(t, \boldsymbol{Y}) - \eta \sum_{j=1}^{m}\sum_{k=1}^{d} b_{kj}(t, \boldsymbol{Y})\partial_{Y_k} b_{ij}(t, \boldsymbol{Y}) \ . \tag{70}$$

The predictor-corrector scheme is very easy to implement, in particular for the special case that the coefficients b_{ij} don't depend on the state variables.

Practitioner's hint: if at all possible, transform your equations to shift all dependence on the state variables from the volatility term over to the drift term!

The biggest problems arise when none of the above schemes are guaranteed to remain in the domain of the process for the stochastic differential equation at hand.

In that case, we may need to select a boundary condition for the numerical scheme.

Alternatively, we can try to find a numerical approximation to Doss's approach of *pathwise solutions* (see [Dos77] or [KS91] (pages 295–296)).

Numerical integration by approximation of pathwise solutions

If there is a strong solution to

$$\mathrm{d}X = a(X)\mathrm{d}t + b(X)\mathrm{d}W \tag{71}$$

then it can be written as a function of the standard Wiener process W (with $W(0) = 0$) and a second process Y, i.e.

$$X = u(W, Y) \quad \text{with boundary condition} \quad u(0, Y) = Y \tag{72}$$

whereby the dynamics of the process Y are governed by an *ordinary differential equation*

$$\mathrm{d}Y = f(W, Y)\mathrm{d}t \tag{73}$$

for some function $f(W, Y)$ to be determined [Dos77]. Itô's lemma for X gives us

$$\mathrm{d}X = \frac{\partial u(W,Y)}{\partial W}\mathrm{d}W + \frac{\partial u(W,Y)}{\partial Y}\mathrm{d}Y + \frac{1}{2}\frac{\partial^2 u(W,Y)}{\partial W^2}\mathrm{d}t \tag{74}$$

which means that u must satisfy

$$\frac{\partial u(W,Y)}{\partial W} = b(u) \qquad \text{with} \qquad u(0,Y) = Y \ . \tag{75}$$

Differentiating with respect to Y gives us

$$\frac{\partial^2 u(W,Y)}{\partial W \partial Y} = b'(u)\frac{\partial u(W,Y)}{\partial Y} \ . \tag{76}$$

This is equivalent to

$$\frac{\partial \frac{\partial u(W,Y)}{\partial Y}}{\partial W} = b'(u)\frac{\partial u(W,Y)}{\partial Y} \tag{77}$$

which has the formal solution

$$\frac{\partial u(W,Y)}{\partial Y} = \mathrm{e}^{\int_0^W b'(u(z,Y))\mathrm{d}z} \tag{78}$$

Matching the terms proportional to $\mathrm{d}t$ in equations (71) and (74) gives

$$\frac{\partial u(W,Y)}{\partial Y} f(W,Y) + \frac{1}{2} b(u) b'(u) = a(u) \ . \tag{79}$$

Therefore, we have from (75),

$$f(W,Y) \ = \ \left[a\left(u(W,Y)\right) - \frac{1}{2} \cdot b\left(u(W,Y)\right) \cdot b'\left(u(W,Y)\right) \right] \cdot \mathrm{e}^{-\int_0^W b'(u(z,Y))\,\mathrm{d}z} \tag{80}$$

aid the numerical scheme which is now solely governed by

$$\mathrm{d}Y \ = \ f(W,Y)\,\mathrm{d}t \tag{81}$$

and the state variable X at any point in time is given by

$$X(t) \ = \ u\left(W(t), Y(t)\right) \ . \tag{82}$$

Naturally, we must have $X(0) \ = \ u\left(W(0), Y(0)\right) \ = \ u\left(0, Y(0)\right) \ = \ Y(0)$.

Example:

$$\mathrm{d}X = \kappa(\theta - X)\mathrm{d}t + \sigma X \mathrm{d}W \tag{83}$$

Assuming

$$\kappa \geq 0\,, \quad \theta \geq 0\,, \quad \sigma \geq 0\,, \quad \text{and} \quad X(0) \geq 0\,,$$

we must have

$$X(t) \geq 0 \qquad \text{for all} \qquad t > 0\,.$$

Now for equation (75):

$$\frac{\partial u(W,Y)}{\partial W} = \sigma u \quad \text{and thus} \quad u(W,Y) = Y\mathrm{e}^{\sigma W}\,. \tag{84}$$

This means

$$\mathrm{d}Y \;=\; \left[\kappa\theta\mathrm{e}^{-\sigma W} - \left(\kappa + \tfrac{1}{2}\sigma^2\right) Y\right] \mathrm{d}t \tag{85}$$

We cannot solve this equation directly.

Also, a directly applied explicit Euler scheme would permit Y to cross over to the negative half of the real axis and thus $X = u(W,Y) = Y\mathrm{e}^{\sigma W}$ would leave the domain of (83).

An explicit Euler scheme applied to equation (85) would mean that, within the scheme, we interpret the $W(t)$ as a piecewise constant function.

We can do better than that!

Recall that, for the given time discretisation, we explicitly construct the Wiener process values $W(t_i)$ and thus, for the purpose of numerical integration of equation (83), they are known along any one given path.

If we now approximate $W(t)$ as a piecewise linear function in between the known values at t_n and t_{n+1}, i.e.

$$W(t) \;\simeq\; \alpha_n + \beta_n t \quad \text{for} \quad t \in [t_n, t_{n+1}] \tag{86}$$

with

$$\alpha_n \;=\; W(t_n) - \beta_n t_n \qquad \text{and} \qquad \beta_n \;=\; \frac{W(t_n) - W(t_{n+1})}{t_n - t_{n+1}}\,,$$

then we have the approximating ordinary differential equation

$$\mathrm{d}\hat{Y} \;=\; \left[\kappa\theta \mathrm{e}^{-\sigma(\alpha_n+\beta_n t)} - \left(\kappa + \tfrac{1}{2}\sigma^2\right)\hat{Y}\right]\mathrm{d}t\,. \tag{87}$$

Using the abbreviations

$$\delta_n := \kappa + \tfrac{1}{2}\sigma^2 - \sigma\beta_n \ , \qquad \Delta t_n := t_{n+1} - t_n \ , \qquad \text{and} \qquad W_{n+1} := W(t_{n+1})$$

we can write the solution to equation (87) as

$$\hat{Y}_{n+1} = \hat{Y}_n \mathrm{e}^{-\left(\kappa+\frac{1}{2}\sigma^2\right)\Delta t_n} + \kappa\theta \cdot \mathrm{e}^{-\sigma W_{n+1}} \cdot \left(\frac{1-\mathrm{e}^{-\delta_n \Delta t_n}}{\delta_n}\right) \ . \tag{88}$$

This scheme is unconditionally stable!

Note: The ordinary differential equation of Doss's pathwise solution can also be obtained by looking for the transformation $Y = Y(W, X)$ such that the stochastic differential equation for Y resulting from Itô's lemma degenerates into an ordinary differential equation.

For the example: apply Itô's lemma to $Y = X\mathrm{e}^{-\sigma W}$:

$$\begin{aligned}
\mathrm{d}Y &= \frac{\partial Y}{\partial X}\mathrm{d}X + \frac{\partial Y}{\partial W}\mathrm{d}W \\
&\quad + \frac{1}{2}\frac{\partial^2 Y}{\partial X^2}\sigma^2 X^2\mathrm{d}t + \frac{1}{2}\frac{\partial^2 Y}{\partial W^2}\mathrm{d}t + \frac{\partial^2 Y}{\partial W \partial X}\sigma X\mathrm{d}t \\
&= \mathrm{e}^{-\sigma W}\cdot\left[\kappa(\theta - X)\mathrm{d}t + \sigma X\,\mathrm{d}W\right] - \sigma Y\,\mathrm{d}W \\
&\quad + \frac{1}{2}\sigma^2 Y\,\mathrm{d}t - \sigma\mathrm{e}^{-\sigma W}\sigma X\mathrm{d}t \\
&= \left[\kappa\theta\mathrm{e}^{-\sigma W} - \left(\kappa + \tfrac{1}{2}\sigma^2\right)Y\right]\mathrm{d}t
\end{aligned}$$

Alternative stable schemes for the numerical solution of equation (85):-

Implicit.

$$\begin{aligned} Y_{n+1} - Y_n &= \left[\kappa\theta \mathrm{e}^{-\sigma W_{n+1}} - \left(\kappa + \tfrac{1}{2}\sigma^2\right) Y_{n+1}\right] \Delta t_n \\ Y_{n+1} &= \frac{Y_n + \kappa\theta \mathrm{e}^{-\sigma W_{n+1}} \Delta t_n}{1 + \left(\kappa + \tfrac{1}{2}\sigma^2\right) \Delta t_n} \end{aligned}$$

Mixed.

$$\begin{aligned} Y_{n+1} - Y_n &= \left[\kappa\theta \mathrm{e}^{-\sigma \frac{1}{2}(W_n + W_{n+1})} - \left(\kappa + \tfrac{1}{2}\sigma^2\right) Y_{n+1}\right] \Delta t_n \\ Y_{n+1} &= \frac{Y_n + \kappa\theta \mathrm{e}^{-\sigma \frac{1}{2}(W_n + W_{n+1})} \Delta t_n}{1 + \left(\kappa + \tfrac{1}{2}\sigma^2\right) \Delta t_n} \end{aligned}$$

XIII. Variance reduction techniques

Reducing the variance of the Monte Carlo estimator (which is a variate itself), improves the Monte Carlo convergence behaviour.

This is why Monte Carlo convergence enhancements are referred to as *variance reduction techniques*.

- A good method to construct paths weighting the dimensions of relevance by their importance is one of the most important variance reduction methods.

- Importance sampling, naturally arising from a fortunate choice of sampler density, is another powerful variance reduction method.

Antithetic sampling for Gaussian variates

For each drawn Gaussian variate vector $\boldsymbol{z}$, reevaluate the functional $v(\boldsymbol{z})$ with $-\boldsymbol{z}$, i.e. compute $v(-\boldsymbol{z})$.

Since $\boldsymbol{z}$ and $-\boldsymbol{z}$ have equal probability, this method automatically matches the first moment.

Only count the pairwise average $\bar{v}_i = \frac{1}{2}(v(\boldsymbol{z}_i) + v(-\boldsymbol{z}_i))$ as an individual sample, because the pairwise averages $\bar{v}_i$ are independent!

This method improves convergence whenever $v(\boldsymbol{z})$ is monotonic in $\boldsymbol{z}$. When v is symmetric in $\boldsymbol{z}$, it decreases performance.

Antithetic sampling is good for:

- one sided payoff functionals

Antithetic sampling is bad for:

- double sided payoff functionals (double knock-outs, range accruals)
- payoff functionals with pronounced discretely symmetric features

Note: Sobol' numbers, whenever the number of iterations is a Mersenne number, i.e. $2^n - 1$ for some integer n, unlike pseudo-random numbers, have the antithetic feature (approximately) built into them.

$\Rightarrow$ **Do not use this method with low-discrepancy numbers!**

Matching the second moment

Moment matching used to be a very popular method before efficient and reliable low-discrepancy numbers became available. This method does usually give more accurate results for calculations that use pseudo-random numbers.

Matching the second moment is not guaranteed to improve convergence.

Assume a Monte Carlo simulation is to be carried out using a total of N variate vectors $\boldsymbol{v}$ of dimensionality d of a known joint distribution density $\psi(\boldsymbol{v})$.

Then, we can calculate the moments actually realised by the drawn variate vector set $V := \{v_{ij}\}$ with $i = 1..N$ and $j = 1..d$. The first moment for dimension j is given by

$$\langle \boldsymbol{v} \rangle_j = \frac{1}{N} \sum_{i=1}^{N} v_{ij} \ , \quad j = 1..d \ . \tag{89}$$

Using (89), we can construct a set of first-moment-corrected variates $\tilde{V}$ by subtraction of the average in each dimension, i.e.

$$\tilde{v}_{ij} = v_{ij} - \langle \boldsymbol{v} \rangle_j \ . \tag{90}$$

The realised covariance of the mean-corrected variate set can be concisely represented as

$$\tilde{C} = \tilde{V}^\top \tilde{V} \tag{91}$$

if we view $\tilde{V}$ as a matrix whose rows comprise the individual d-dimensional mean-corrected vector draws.

We can construct a new matrix $\hat{V}$ whose entries will meet the desired covariance C of the target distribution density ψ exactly.

Define the elements of the desired covariance matrix C as

$$c_{jk} = \int v_j v_k \psi(\boldsymbol{v}) \mathrm{d}v_j \mathrm{d}v_k \ . \tag{92}$$

Also, define the pseudo-square roots of both C and $\tilde{C}$ by

$$\tilde{C} = \tilde{A} \cdot \tilde{A}^\top \quad \text{and} \quad C = A \cdot A^\top \,. \tag{93}$$

The correction matrix K that transforms $\tilde{V}$ to $\hat{V}$ can be computed by solving the linear system

$$\tilde{A}^\top \cdot K = A^\top \,, \quad \text{i.e.} \quad K = \tilde{A}^{\top\,-1} \cdot A^\top \,. \tag{94}$$

The moment-corrected variate matrix is thus

$$\hat{V} = K \cdot \tilde{V} \,. \tag{95}$$

Since $\tilde{C}$ may (very rightfully) be rank-deficient, you must always use a failsafe method for the solution of the linear system (94) such as the Moore-Penrose pseudo-inverse.

Notes:

- When using this method to correct the first and the second moment of a set of drawn variates it should be applied to the variates *after* having transformed them from the uniform $(0, 1)$ distribution to whatever distribution is actually used, e.g. a joint normal distribution.

- For anything but comparatively low-dimensional methods, the calculation of the covariance matrix and the subsequent correction of each drawn vector variate can very easily dominate the overall computational effort *by several orders of magnitude*!

- Sobol' numbers (approximately) match the second moment (as well as the first moment), when the number of iterations is a Mersenne number.

- **Do not use this method with low-discrepancy numbers!**

Control variates

Many Monte Carlo calculations are carried out for problems that we can almost solve analytically.

The idea behind *control variates* is as follows:

Let's assume that we wish to calculate the expectation $\mathsf{E}[v]$ of a function $v(\boldsymbol{u})$ for some underlying vector draw $\boldsymbol{u}$, and that there is a related function $g(\boldsymbol{u})$ whose expectation $g^* := \mathsf{E}[g]$ we know exactly. Then, we have

$$\mathsf{E}\left[\frac{1}{n}\sum_{i=1}^{n} v(\boldsymbol{u}_i)\right] = \mathsf{E}\left[\frac{1}{n}\sum_{i=1}^{n} v(\boldsymbol{u}_i) + \beta\left(g^* - \frac{1}{n}\sum_{i=1}^{n} g(\boldsymbol{u}_i)\right)\right] \tag{96}$$

for any given $\beta \in \mathbb{R}$ and thus we can replace the ordinary Monte Carlo estimator

$$\hat{v} = \frac{1}{n}\sum_{i=1}^{n} v(\boldsymbol{u}_i) \tag{97}$$

by

$$\hat{v}_{\mathsf{CV}} = \frac{1}{n}\sum_{i=1}^{n}\left[v(\boldsymbol{u}_i) + \beta\left(g^* - g(\boldsymbol{u}_i)\right)\right] \ . \tag{98}$$

The optimal choice of β is

$$\beta^* = \frac{\mathsf{Cov}[v,g]}{\mathsf{V}[g]} \tag{99}$$

which minimises the variance of of $\hat{v}_{\mathsf{CV}}$.

Note that the function $g(\boldsymbol{u}_i)$ does not have to be the payoff of an analytically known option. It could also be the profit from a self-financing dynamic hedging strategy, i.e. a strategy that starts with zero investment capital. For risk-neutral measures, the expected profit from any such strategy is zero which means that the control variate is simply the payoff from the dynamic hedging strategy along any one path.

An intuitive understanding of the control variate method is to consider the case when v and g are positively correlated. For any draw $v(\boldsymbol{u}_i)$ that overestimates the result, $g(\boldsymbol{u}_i)$ is likely to overestimate g^*. As a result, the term multiplied by β in equation (98) is likely to correct the result by subtracting the aberration.

The precise value of β^* is, of course, not known but can be estimated from the same simulation that is used to calculate $\hat{v}_{\mathrm{CV}}$. As in all the situations when the parameters determining the result are calculated from the same simulation, this can introduce a bias that is difficult to estimate. In the limit of very large numbers of iterations, this bias vanishes, but the whole point of variance reduction techniques is to require *fewer* simulations and thus shorter run time. A remedy to the problem of bias due to a correlated estimate of the control parameter β is to use an initial simulation, possibly with fewer iterates than the main run, to estimate β^* in isolation.

Key point: **Choose a good control variate specific to each problem. A (very) bad control variate can make matters worse!**

XIV. Sensitivity calculations

A Monte Carlo estimator, invariably, is not just a function of the payoff-specific parameters, but also of the initial values of the state variables and the model parameters

$$\hat{v} = \hat{v}(\boldsymbol{x}(0), \boldsymbol{\lambda}) \; . \tag{100}$$

The partial derivatives with respect to the elements of $\boldsymbol{x}(0)$ and $\boldsymbol{\lambda}$ are known as *Greeks* and are one of the biggest problems of Monte Carlo simulations.

Finite differencing with path recycling

We can always re-do the entire valuation with varied inputs reflecting the potential change in the underlying asset, and use an explicit finite-differencing approach to compute the Greek we are after.

Forward differencing for Delta:

$$\text{Delta} = \frac{\partial v}{\partial S_0} \approx \frac{v(S_0 + \Delta S_0) - v(S_0)}{\Delta S_0} \; . \tag{101}$$

Centre differencing for Delta:

$$\text{Delta} = \frac{\partial v}{\partial S_0} \approx \frac{v(S_0 + \Delta S_0) - v(S_0 - \Delta S_0)}{2\Delta S_0} \,. \tag{102}$$

Using the centre-differencing approach in equation (102) has the added advantage that we can then directly approximate Gamma as

$$\text{Gamma} = \frac{\partial^2 v}{\partial S_0^2} \approx \frac{v(S_0 + \Delta S_0) - 2v(S_0) + v(S_0 - \Delta S_0)}{\Delta S_0^2} \,. \tag{103}$$

Note: $\frac{\Delta S_0}{S_0}$ should scale like the fourth root of your machine precision!

For Gamma of a plain vanilla option, this means we are trying to compute the value of an approximation to the Dirac-spike as a payoff!

Monte Carlo Greeks with conventional finite differencing can be prohibitively noisy!

Importance sampling can help!

Pathwise differentiation

All occurrences of the true price v in equation (102) are numerically evaluated as a Monte Carlo approximation to it. Thus,

$$\widehat{\text{Delta}} = \frac{\partial \hat{v}}{\partial S_0} = \frac{\partial}{\partial S_0}\left[\frac{1}{m}\sum_{j=1}^{m}\pi\Big(\boldsymbol{S}(\boldsymbol{z}^j; S_0)\Big)\right] \quad . \tag{104}$$

An infinitesimal change of the initial spot level S_0 can only give rise to infinitesimal changes of the spot level at any of the monitoring dates.

For Lipschitz-continuous payoff functions, the order of differentiation and expectation can be interchanged. For such payoff functions, it is perfectly consistent to assign a Delta of zero to all paths that terminate out of the money, and a Delta equal to $\sum_i \frac{\partial\pi\left(\boldsymbol{S}(\boldsymbol{z}^j;S_0)\right)}{\partial S_i}\cdot\frac{\partial S_i}{\partial S_0}$.

This method is called *pathwise differentiation* and can easily be transferred to other Greeks such as Vega. However, for the calculation of Gamma, we still have to implement a finite differencing scheme for two individually calculated

Deltas for an up and a down shift, and both of these can individually be computed using pathwise differentiation.

Alas, this method does not apply (easily) to discontinuous payoff functionals, of which there are so many.

The likelihood ratio method

The option pricing problem by Monte Carlo is a numerical approximation to an integration:

$$v = \int \pi(S)\ \psi(S)\,\mathrm{d}S \tag{105}$$

Numerically, we construct evolutions of the underlying assets represented by S given a risk-neutral distribution $\psi(S)$.

We hereby typically construct the paths by the aid of a set of standard normal variates which corresponds to

$$v = \int \pi(S(z;\alpha))\ \varphi(z)\,\mathrm{d}z\ , \tag{106}$$

and all dependence on further pricing parameters (herein represented by α) such as the spot level at inception, volatility, time to maturity, etc., is absorbed into the path construction $S(z;\alpha)$.

Any derivative with respect to any of the parameters will thus suffer from any discontinuities of π in S:

$$\frac{\partial v}{\partial \alpha} = \int \frac{\partial}{\partial \alpha} \pi(S(z;\alpha))\ \varphi(z)\,\mathrm{d}z\ . \tag{107}$$

The key insight behind the *likelihood ratio* method is to shift the dependence on any of the parameters over into the density function. In other words, a transformation of the density is required to look at the pricing problem in the form of equation (105). This way, the Greek evaluation problem becomes

$$\frac{\partial v}{\partial \alpha} = \int \pi(S)\ \frac{\partial}{\partial \alpha}\psi(S;\alpha)\,\mathrm{d}S = \int \pi(S)\ \frac{\frac{\partial \psi(S;\alpha)}{\partial \alpha}}{\psi(S;\alpha)}\,\psi(S;\alpha)\,\mathrm{d}S\ . \tag{108}$$

The calculation of the desired Greek now looks exactly like the original pricing

problem, only with a new payoff function

$$\chi(S;\alpha) := \pi(S) \cdot \omega(S;\alpha) \tag{109}$$

with

$$\omega(S;\alpha) := \frac{\frac{\partial\psi(S;\alpha)}{\partial\alpha}}{\psi(S;\alpha)} \; . \tag{110}$$

The term $\omega(S;\alpha)$ may be interpreted as a *likelihood ratio* since it is the quotient of two density functions, whence the name of the method. Using this definition, the Greek calculation becomes

$$\frac{\partial v}{\partial\alpha} = \int \chi(S;\alpha)\,\psi(S;\alpha)\,\mathrm{d}S \; . \tag{111}$$

The beauty of this idea is that for the probability density functions that we typically use such as the one corresponding to geometric Brownian motion, the function $\chi(S;\alpha)$ is $\in \mathcal{C}^\infty$ in the parameter α and thus doesn't cause the trouble that we have when we approach the Greek calculation problem in the form of equation (106).

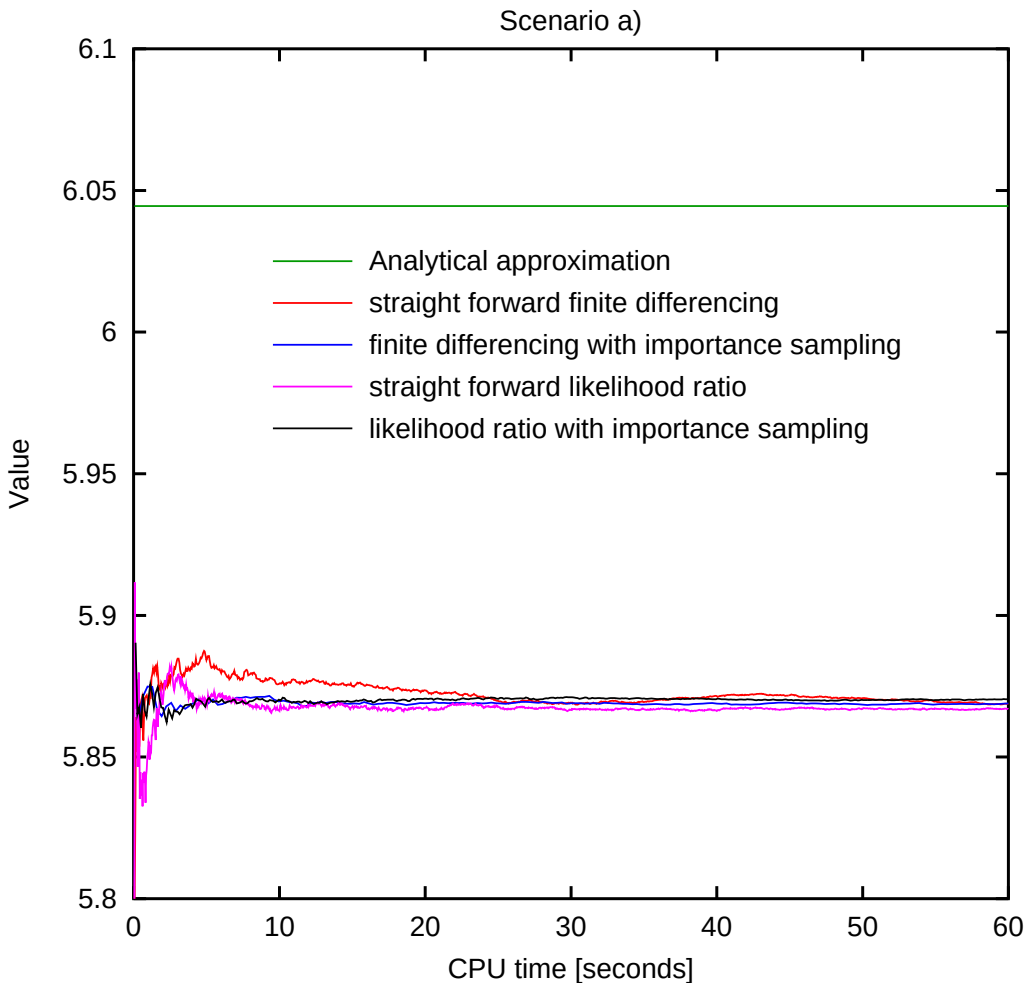
The application is now straightforward. Alongside the calculation of the option price, for each constructed path, apart from calculating the payoff $\pi(S)$, also calculate the likelihood ratio $\omega(S;\alpha)$. The approximation for Delta, for instance, thus becomes

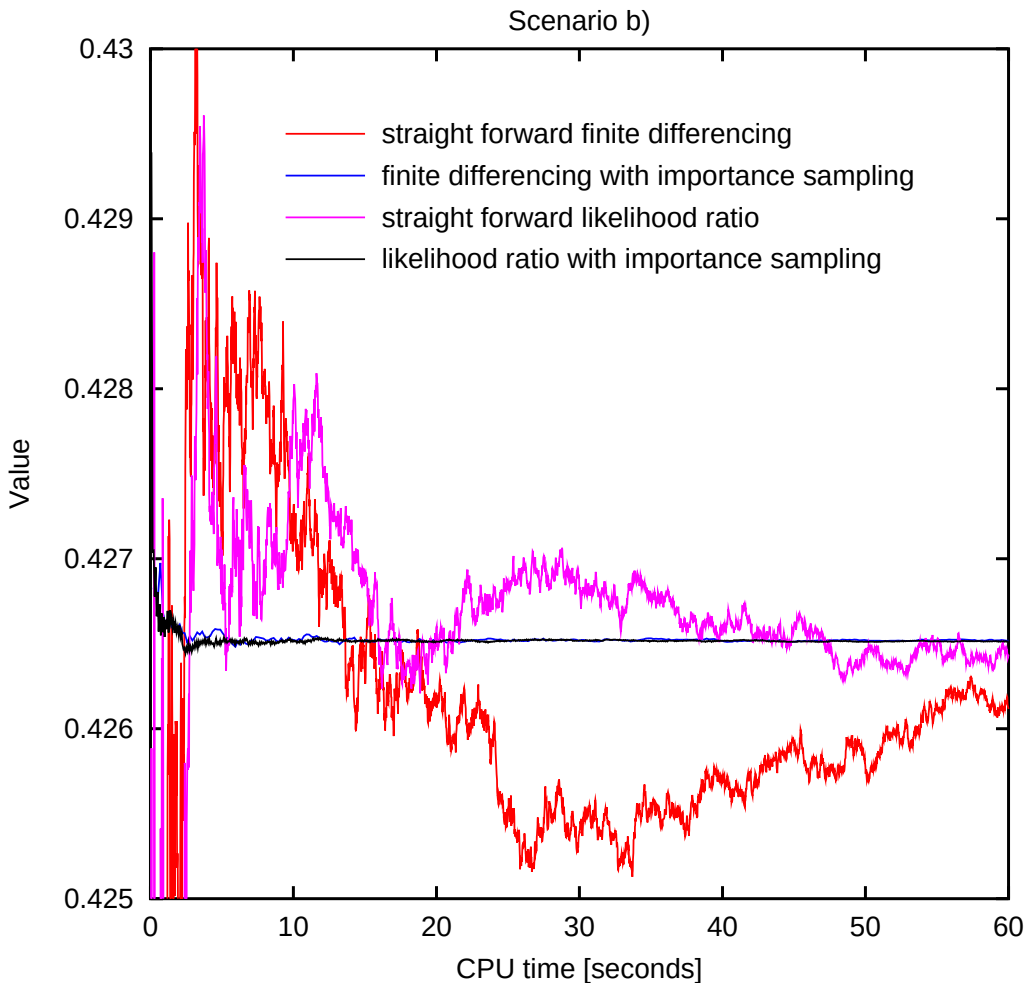
$$\widehat{\text{Delta}} = \frac{1}{m}\sum_{j=1}^{m} \pi\left(\boldsymbol{S}^j; S_0\right)\right) \, \omega\left(\boldsymbol{S}^j; S_0\right)\right) \ . \tag{112}$$

For geometric Brownian motion:

$$\omega_{\widehat{\text{Delta}}} = \frac{z_1}{S_0\sigma\sqrt{\Delta t_1}} \tag{113}$$

$$\omega_{\widehat{\text{Gamma}}} = \frac{z_1^2 - z_1\sigma\sqrt{\Delta t_1} - 1}{S_0^2\sigma^2\Delta t_1} \tag{114}$$

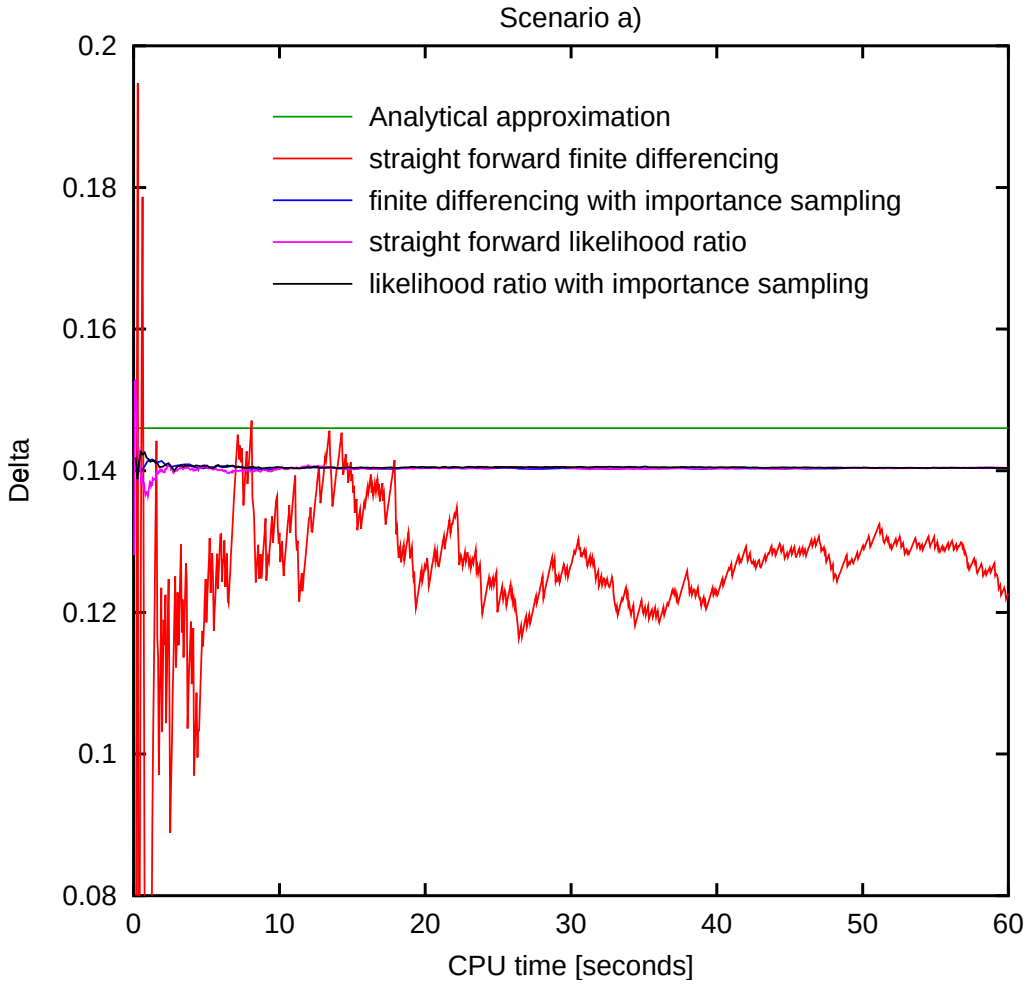


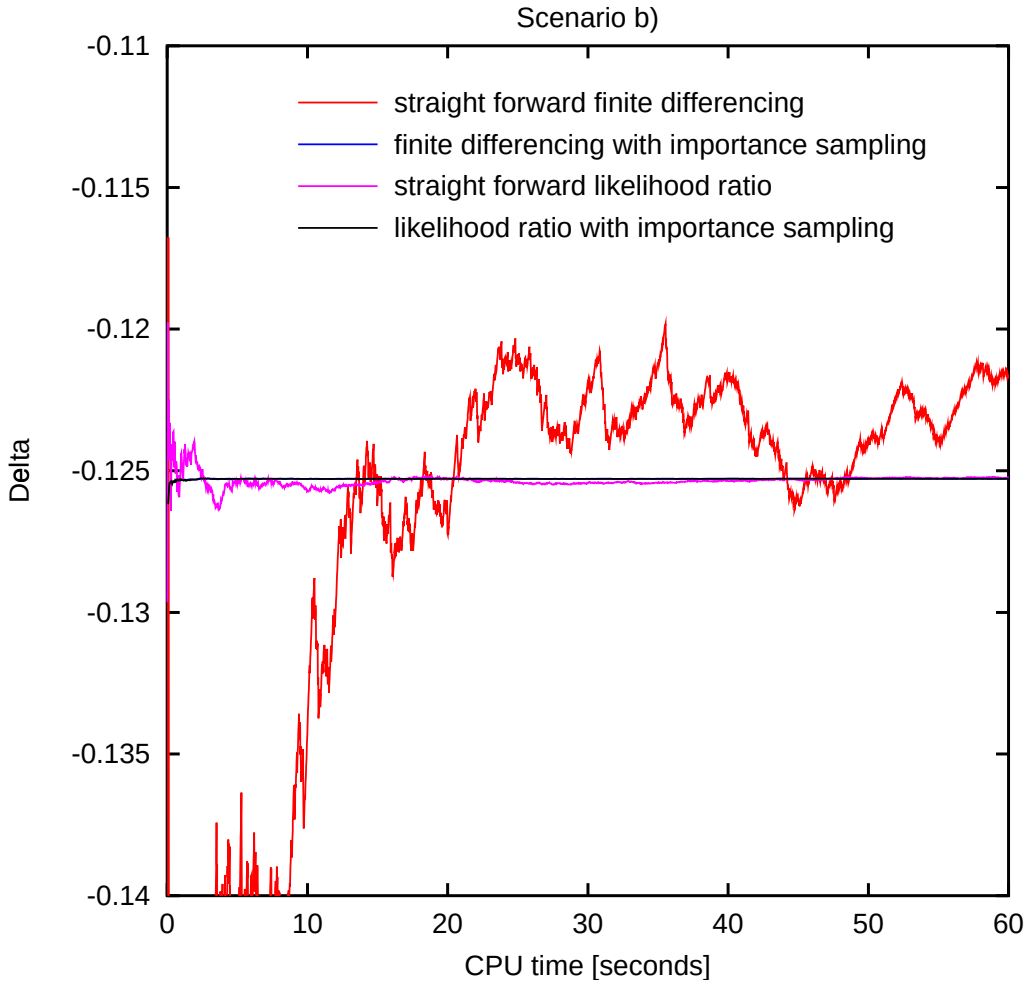


The value of two Up-Out-Call options.

Scenario a): $T = 1, S = 100, K = 100, H = 150, \sigma = 30\%, t_i = \frac{i}{12}$.

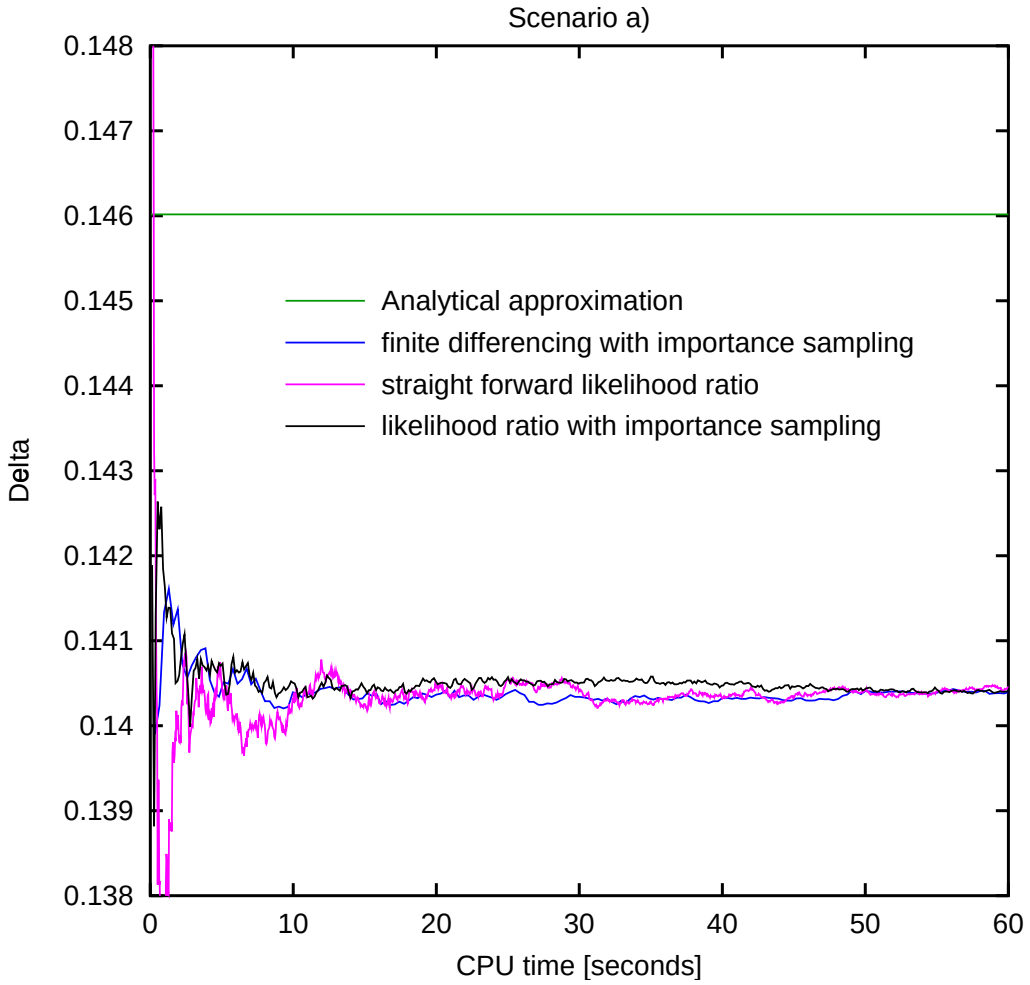
Scenario b): $T = 0.52, S = 160, K = 100, H = 150, \sigma = 30\%, t_1 = \frac{5}{250}, t_i = \frac{i}{12} + \frac{5}{250}$ for $i = 2, 3, 4, 5, 6, 7$.

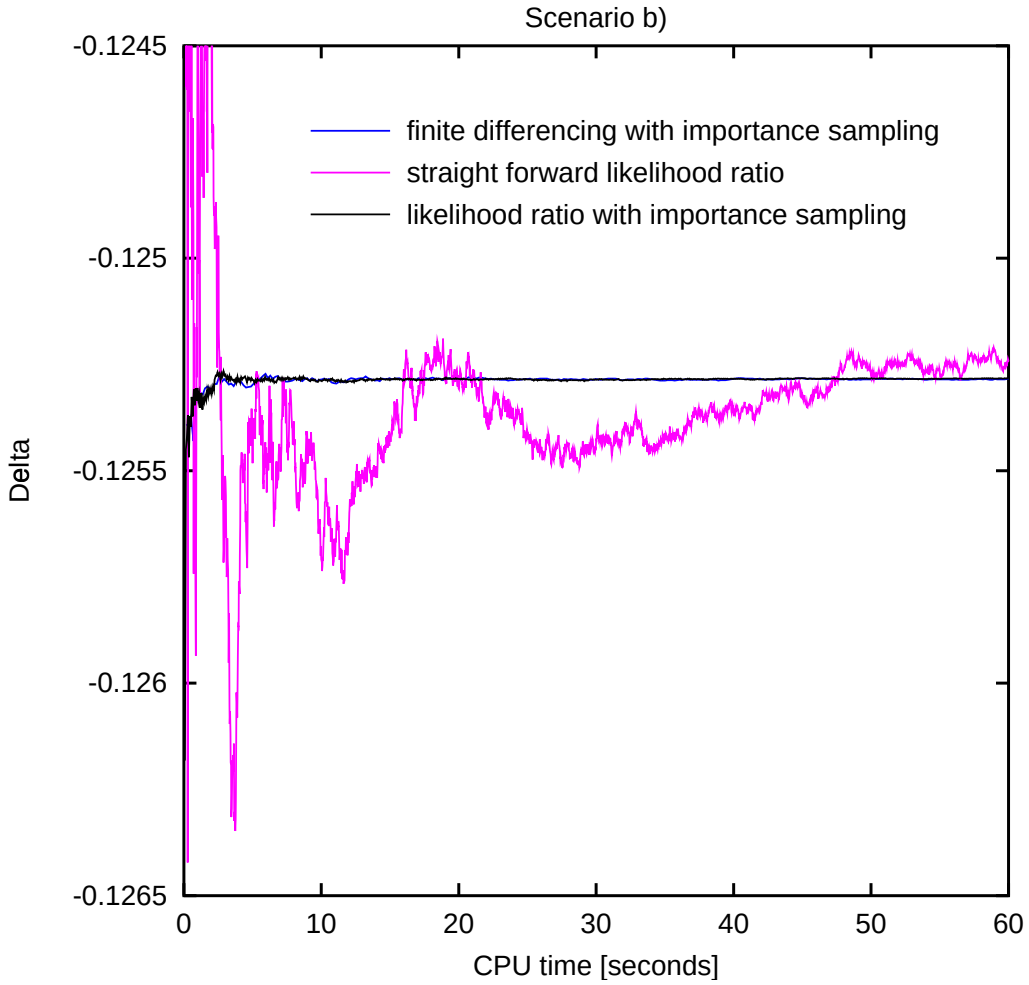




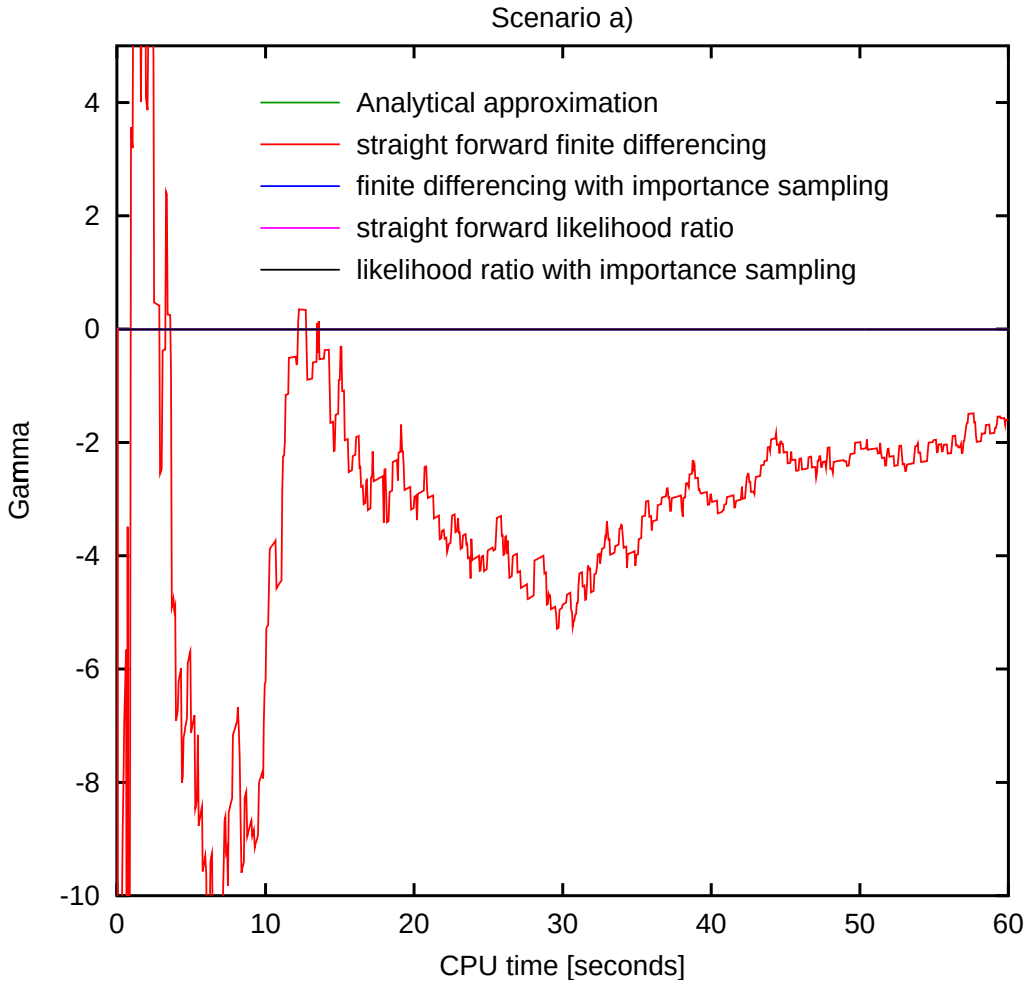
Delta of two Up-Out-Call option.

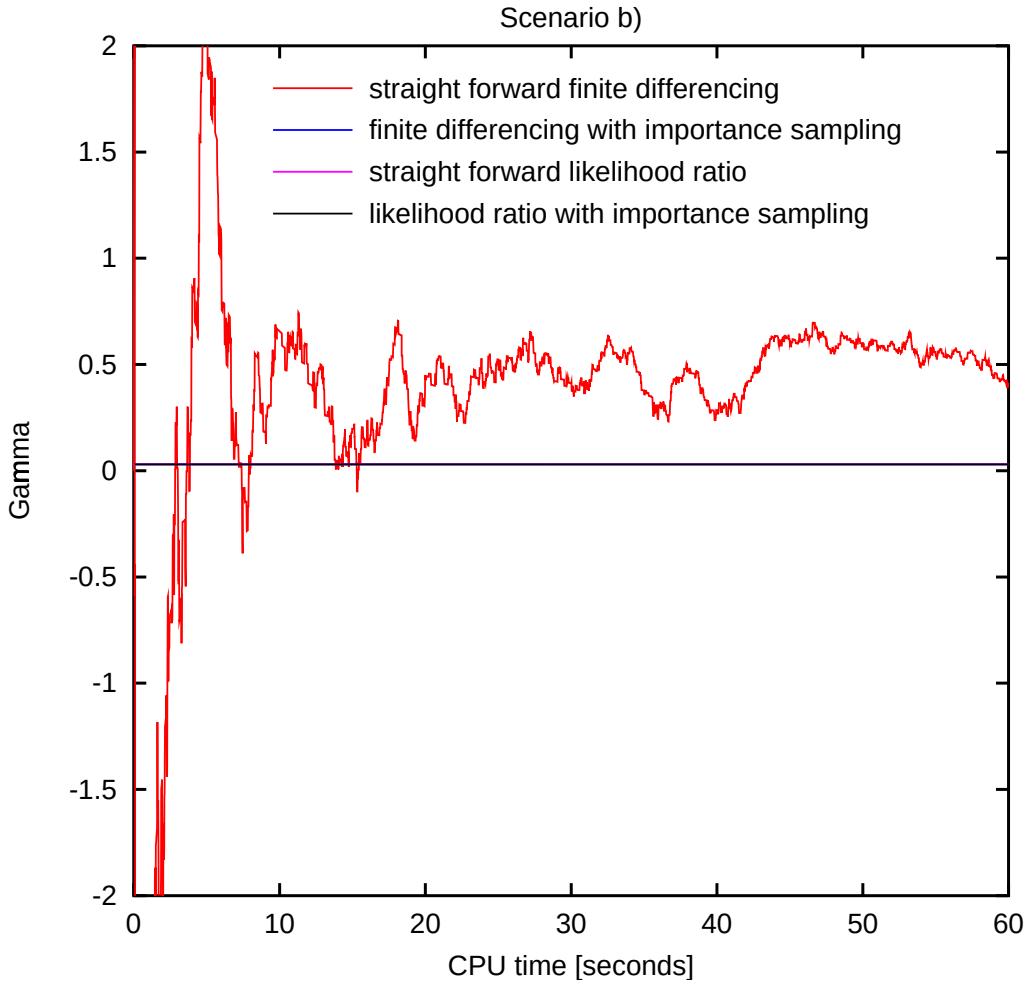
Hardware: AMD K6-III processor running at 400MHz.



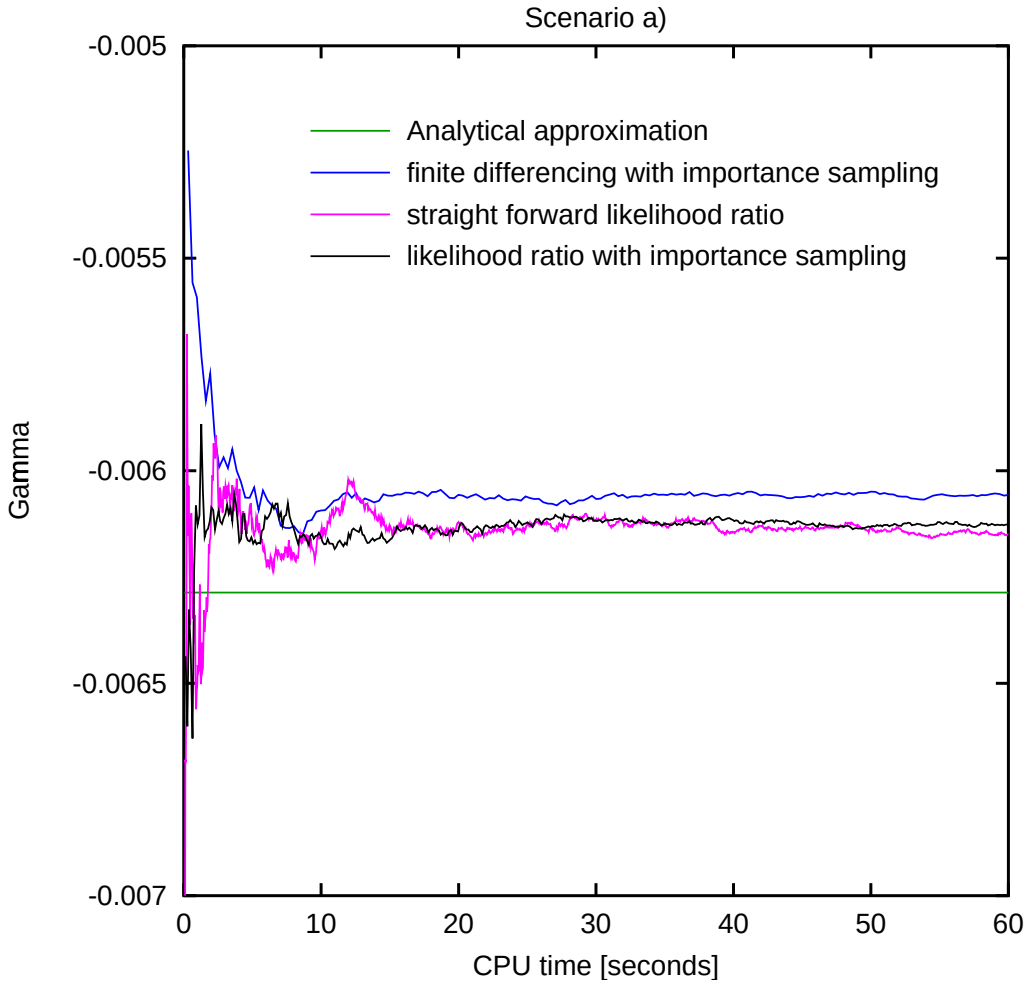


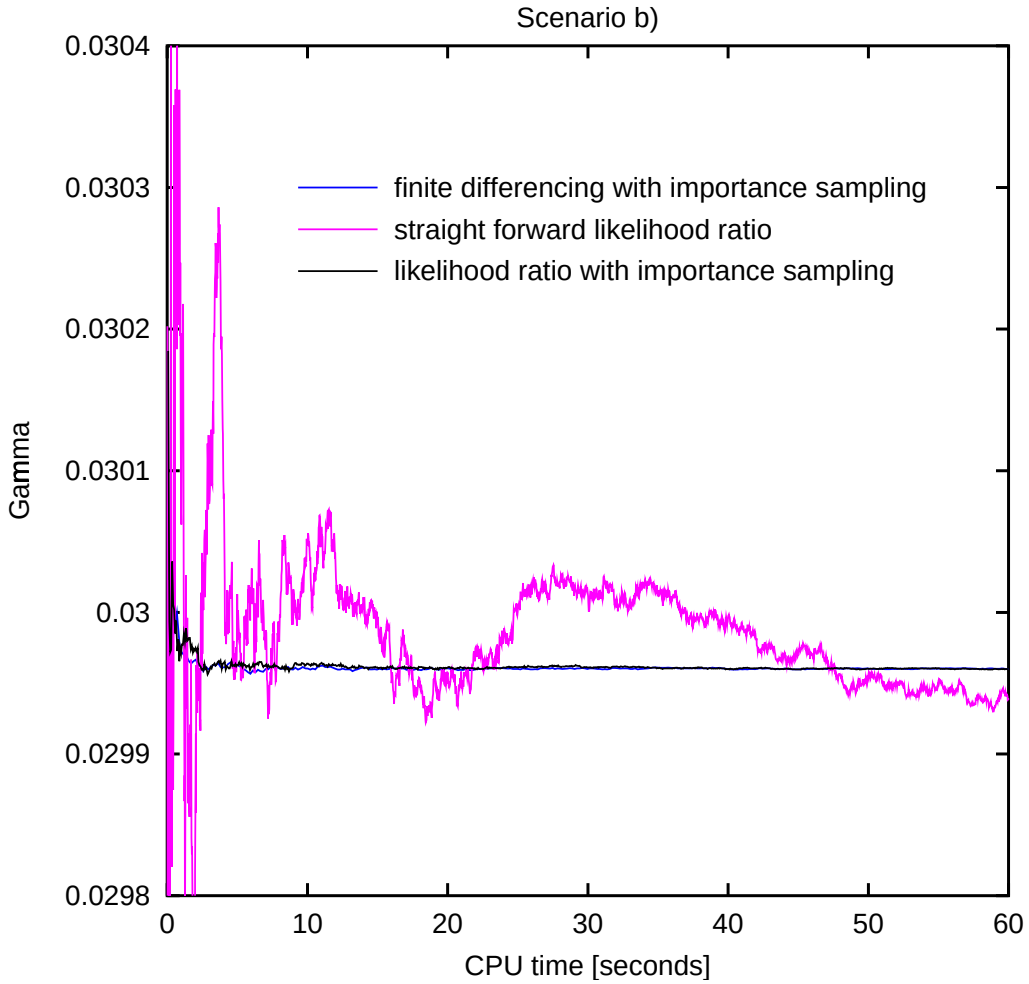
Enlargement for Delta.





Gamma of two Up-Out-Call options.





Enlargements for Gamma of the two Up-Out-Call options.

XV. Weighted Monte Carlo

These methods are also known as under the names *equivalent entropy projection methods* and *perturbation of measures* and are due to Avellaneda and Gamba [AG02].

A standard Monte Carlo estimator over N paths is given by

$$\langle v \rangle_N \quad = \quad \sum_{i=1}^{N} \frac{1}{N} v_i \ . \tag{115}$$

where v_i is the net present value of all cashflows (numéraire dominated payoffs).

The concept of *Weighted Monte Carlo* is to generalise this to

$$\widetilde{\langle v \rangle}_N \quad = \quad \sum_{i=1}^{N} p_i v_i \quad = \quad \boldsymbol{p}^\top \cdot \boldsymbol{v} \ . \tag{116}$$

Define the value of the M calibration instruments as g_j^* for $j = 1..M$, and the vector of these values as $\boldsymbol{g}^*$.

Define the aggregated net present value of all cashflows occuring in path # i for the calibration instrument # j as $g_{ij} = (G)_{ij}$.

In order to have stable hedge ratios, we demand a perfect match of the market values of the calibration instruments:

$$\boldsymbol{g}^{*\top} \;=\; \boldsymbol{p}^{\top} \cdot G \tag{117}$$

Another constraint is

$$\sum_{i=1}^{N} p_i \;=\; 1\,. \tag{118}$$

Since $M \ll N$, this alone does not define all of the p_i uniquely.

One can remedy the ambiguity by the aid of a strictly convex objective function of $\boldsymbol{p}$ in combination with (117) and (118) using Lagrange multipliers.

Avellaneda and Gamba suggest

$$\sum_{i=1}^{N}\Psi(p_i)-\sum_{j=1}^{M}\lambda_j\cdot\left[\sum_{i=1}^{N}p_i g_{ij}-g_j^*\right]-\mu\cdot\left[\sum_{i=1}^{N}p_i-1\right] \tag{119}$$

with $\Psi(x)$ a convex function. Specifically, we will consider the following two choices:-

1. The Euclidean distance from equiprobability given by the quadratic entropy function
$$\Psi^{(\mathrm{QE})}(p)=\left(p-1/N\right)^2\ . \tag{120}$$

2. The Shannon entropy [Sha48] function (which is identical to the Kullback-Leibler relative entropy function against a uniform distribution [KL51])
$$\Psi^{(\mathrm{KL})}(p)\ =\ p\ln p\ . \tag{121}$$

Differentiation of (119),

$$\nabla_{\boldsymbol{p}} \cdot \left[\sum_{i=1}^{N} \Psi(p_i) - \left(\boldsymbol{p}^\top \cdot G - \boldsymbol{g}^{*\top} \right) \cdot \boldsymbol{\lambda} - \mu \cdot \left(\sum_{i=1}^{N} p_i - 1 \right) \right] \;=\; 0 \qquad (122)$$

yields that the minimum must satisfy

$$p_i \;=\; \phi\left(\textstyle\sum_{j=1}^{M} g_{ij} \lambda_j + \mu \right) \qquad \text{with} \qquad \phi(x) \;:=\; {\Psi'}^{-1}(x) \qquad (123)$$

which, for the specific choices of (120) and (121), means

$$p_i^{(\mathrm{QE})} \;=\; \frac{1}{N} + \frac{1}{2} \left(\sum_{j=1}^{M} g_{ij} \lambda_j^{(\mathrm{QE})} + \mu \right) \qquad (124)$$

and

$$p_i^{(\mathrm{KL})} \;=\; \mathrm{e}^{\sum_{j=1}^{M} g_{ij} \lambda_j^{(\mathrm{KL})} + \mu - 1} \,. \qquad (125)$$

Combining this with the condition $\sum_{i=1}^{N} p_i = 1$, we obtain

$$p_i^{(\mathrm{QE})} = \frac{1}{N} + \frac{1}{2}\sum_{j=1}^{M}\left(g_{ij} - \langle g_j\rangle_N\right)\cdot\lambda_j^{(\mathrm{QE})} \tag{126}$$

and

$$p_i^{(\mathrm{KL})} = \frac{\mathrm{e}^{\sum_{j=1}^{M} g_{ij}\lambda_j^{(\mathrm{KL})}}}{Z(\boldsymbol{\lambda}^{(\mathrm{KL})})} \tag{127}$$

with

$$\langle g_j\rangle_N := \frac{1}{N}\sum_{i=1}^{N} g_{ij} \qquad \text{and} \qquad Z(\boldsymbol{\lambda}^{(\mathrm{KL})}) := \sum_{i=1}^{N}\mathrm{e}^{\sum_{j=1}^{M} g_{ij}\lambda_j^{(\mathrm{KL})}} \tag{128}$$

The key is now to find the values for $\boldsymbol{\lambda}$ such that (117) and (118), i.e.

$$g_j^* = \widetilde{\langle g_j\rangle}_N = \sum_{i=1}^{N} p_i g_{ij} \qquad \text{and} \qquad \sum_{i=1}^{N} p_i = 1\,,$$

are satisfied.

For the quadratic penalty function (120), this means we have to solve the linear system

$$\underbrace{\left(\langle \boldsymbol{g}\cdot\boldsymbol{g}^\top\rangle_N - \langle\boldsymbol{g}\rangle_N\cdot\langle\boldsymbol{g}\rangle_N^\top\right)}_{\text{sample covariance}}\cdot\boldsymbol{\lambda}^{(\mathrm{QE})} \;=\; \frac{2}{N}\left(\boldsymbol{g}^* - \langle\boldsymbol{g}\rangle_N\right)\;. \tag{129}$$

Substituting this back into (126) and (116) yields

$$p_i^{(\mathrm{QE})} \;=\; \frac{1}{N}\cdot\left(1+\sum_{j=1}^M\left(g_{ij}-\langle g_j\rangle_N\right)\cdot\gamma_j\right) \quad\text{with}\quad \boldsymbol{\gamma} \;:=\; \langle\boldsymbol{g},\boldsymbol{g}^\top\rangle_N^{-1}\cdot\left(\boldsymbol{g}^*-\langle\boldsymbol{g}\rangle_N\right) \tag{130}$$

and

$$\widetilde{\langle v\rangle}_N^{(\mathrm{QE})} \;=\; \langle v\rangle_N + \langle v,\boldsymbol{g}^\top\rangle_N\cdot\langle\boldsymbol{g},\boldsymbol{g}^\top\rangle_N^{-1}\cdot\left(\boldsymbol{g}^*-\langle\boldsymbol{g}\rangle_N\right) \tag{131}$$

where $\langle\cdot,\cdot\rangle_N$ stands for the sample covariance.

The choice of a quadratic entropy function centered at the equiprobability level is equivalent to the use of the calibration instruments as conventional control variates.

For the Kullback-Leibler entropy function (121), Avellaneda and Gamba noticed that

$$\sum_{i=1}^{N} p_i^{(\mathrm{KL})} g_{ij} \;=\; \frac{\partial \ln Z(\boldsymbol{\lambda}^{(\mathrm{KL})})}{\partial \lambda_j^{(\mathrm{KL})}} \;. \tag{132}$$

Thus, instead of solving (117) directly, i.e.

$$\boldsymbol{g}^* \;=\; \boldsymbol{p}^{(\mathrm{KL})\,\top} \cdot G \;=\; \nabla_{\boldsymbol{\lambda}^{(\mathrm{KL})}} \cdot \ln Z(\boldsymbol{\lambda}^{(\mathrm{KL})}) \;, \tag{133}$$

one may minimise

$$\ln Z(\boldsymbol{\lambda}^{(\mathrm{KL})}) - \sum_{j=1}^{M} g_j^* \lambda_j^{(\mathrm{KL})} \;. \tag{134}$$

This can be done, for instance, using the Broydn-Fletcher-Goldfarb-Shanno algorithm [PTVF92]. A second order expansion in $\boldsymbol{\lambda}$ for $\ln Z(\boldsymbol{\lambda}^{(\mathrm{KL})})$ gives

$$\boldsymbol{\lambda}^{(\mathrm{KL})} \;\simeq\; N/2 \cdot \boldsymbol{\lambda}^{(\mathrm{QE})} \tag{135}$$

which can be used as the initial guess for the iteration.

The connection between the Monte Carlo estimator of the exotic derivative $\widetilde{\langle v \rangle}_N$ and the calibration instruments $\boldsymbol{g}^*$ is given by the fact that the discrete distribution encoded in the probability vector $\boldsymbol{p}$ is a function of the parameter vector $\boldsymbol{\lambda}$, i.e.

$$\boldsymbol{p} \;=\; \boldsymbol{p}(\boldsymbol{\lambda}) \qquad \text{with} \qquad \boldsymbol{p} \in \mathbb{R}^N, \boldsymbol{\lambda} \in \mathbb{R}^M \;, \tag{136}$$

and that the vector $\boldsymbol{\lambda}$ is chosen to match the calibration constraints (117), i.e.

$$\boldsymbol{\lambda} \;=\; \boldsymbol{\lambda}(\boldsymbol{g}^*) \qquad \text{with} \qquad \boldsymbol{g}^* \in \mathbb{R}^M \;. \tag{137}$$

By virtue of parametric adjustments of the sample distribution $\boldsymbol{p}(\boldsymbol{\lambda})$, we can adjust for changes in the given calibration instruments.

Model calibration, in general, is given by the process of parametric adjustments of a model's probability measures, i.e. implied market variable distributions, to match the prices of calibration instruments.

This means, we can estimate the effect of changes in the values of calibration instruments on $\widetilde{\langle v \rangle}_N$ by calculating the changes in $\widetilde{\langle v \rangle}_N = \boldsymbol{p}^\top \cdot \boldsymbol{v}$ in response to parametric changes of the sample distribution:

$$\nabla_{\boldsymbol{g}^*} \cdot \widetilde{\langle v \rangle}_N \;=\; J \cdot \nabla_{\boldsymbol{\lambda}} \cdot \boldsymbol{p}^\top \cdot \boldsymbol{v} \quad \text{with} \quad J \;:=\; \left(\frac{(\partial \boldsymbol{\lambda})}{(\partial \boldsymbol{g}^*)} \right) \quad \text{such that} \quad J_{kl} = \partial_{g_k^*} \lambda_l \tag{138}$$

From (123), we have

$$\partial_{\lambda_j} p_k \;=\; \phi_k' \cdot \left(g_{kj} + \partial_{\lambda_j} \mu \right) \tag{139}$$

where we have set

$$\phi_k' := \phi' \left(\textstyle\sum_{j=1}^M g_{kj} \lambda_j + \mu \right) \;.$$

Differentiating the probability normalisation condition (118) with respect to λ_j gives us

$$0 \;=\; \sum_{k=1}^N \partial_{\lambda_j} p_k \;=\; \sum_{k=1}^N \phi_k' \cdot \left(g_{kj} + \partial_{\lambda_j} \mu \right) \;, \tag{140}$$

i.e.

$$\partial_{\lambda_j}\mu \;=\; -\frac{\sum_{k=1}^{N}\phi'_k g_{kj}}{\sum_{k=1}^{N}\phi'_k} \;=\; -\widehat{\langle g_j\rangle}_N \;. \tag{141}$$

where we have defined

$$s := \sum_{k=1}^{N}\phi'_k \;,\quad w_i := \phi'_i/s \;,\quad \text{and}\quad \widehat{\langle v\rangle}_N := \sum_{i=1}^{N} w_i v_i \;. \tag{142}$$

This enables us to write concisely

$$\partial_{\lambda_j}\widetilde{\langle v\rangle}_N \;=\; s\cdot\sum_{k=1}^{N}\Big(w_k g_{kj} v_k - w_k\widehat{\langle g_j\rangle}_N v_k\Big) \tag{143}$$

and thus

$$\nabla_{\boldsymbol{\lambda}}\cdot\widetilde{\langle v\rangle}_N \;=\; s\cdot\widehat{\langle \boldsymbol{g}, v\rangle}_N \tag{144}$$

with

$$\widehat{\langle a,b\rangle}_N := \widehat{\langle ab\rangle}_N - \widehat{\langle a\rangle}_N\widehat{\langle b\rangle}_N \;. \tag{145}$$

Since this holds for any derivative contract priced in this measure, it also holds for the calibration instruments:

$$\underbrace{\nabla_{\boldsymbol{g}^*} \cdot \widetilde{\langle \boldsymbol{g} \rangle}_N^{\top}}_{\text{identity matrix}} = J \cdot \nabla_{\boldsymbol{\lambda}} \cdot \widetilde{\langle \boldsymbol{g} \rangle}_N^{\top} = J \cdot s \cdot \widehat{\langle \boldsymbol{g}, \boldsymbol{g}^{\top} \rangle}_N \tag{146}$$

whence

$$J = \frac{1}{s} \widehat{\langle \boldsymbol{g}, \boldsymbol{g}^{\top} \rangle}_N^{-1} . \tag{147}$$

We finally arrive at the formula for the hedge positions in the calibration instruments:

$$\nabla_{\boldsymbol{g}^*} \cdot \widetilde{\langle v \rangle}_N = \widehat{\langle \boldsymbol{g}, \boldsymbol{g}^{\top} \rangle}_N^{-1} \cdot \widehat{\langle \boldsymbol{g}, v \rangle}_N \tag{148}$$

The vector of price-sensitivities is equal to the vector of regression coefficients (under the calibration-adjusted measure $\widehat{\langle \cdot \rangle}$) of the payoff of the target contract on the linear space generated by the payoff-vectors of the calibration instruments [AG02].

In the case of the quadratic entropy function, we have:

$$\begin{aligned}
\Psi^{(\mathrm{QE})}(p) &= \left(p-\frac{1}{N}\right)^2 \\
\Psi'^{(\mathrm{QE})}(p) &= 2\left(p-\frac{1}{N}\right) \\
\phi^{(\mathrm{QE})}(y) &= \frac{y}{2}+\frac{1}{N} \\
\phi'^{(\mathrm{QE})}(y) &= \frac{1}{2} \\
s^{(\mathrm{QE})} &= \frac{N}{2} \\
w_k^{(\mathrm{QE})} &= \frac{1}{N}
\end{aligned}$$

which means

$$\widehat{\langle\cdot\rangle}_N^{(\mathrm{QE})} = \langle\cdot\rangle_N \qquad (149)$$

For the Shannon/Kullback-Leibler entropy:

$$\begin{aligned}
\Psi^{(\mathrm{KL})}(p) &= p\ln p \\
\Psi'^{(\mathrm{KL})}(p) &= 1+\ln p \\
\phi^{(\mathrm{KL})}(y) &= \mathrm{e}^{y-1} \\
\phi'^{(\mathrm{KL})}(y) &= \phi^{(\mathrm{KL})}(y)
\end{aligned}$$

Using equation (123), this means

$$\phi'^{(\mathrm{KL})}_k = \phi^{(\mathrm{KL})}\Big(\sum_{j=1}^{M} g_{kj}\lambda_j^{(\mathrm{KL})} + \mu^{(\mathrm{KL})}\Big) = p_k^{(\mathrm{KL})}$$

and therefore

$$\begin{aligned}
s^{(\mathrm{KL})} &= 1 \\
w_k^{(\mathrm{KL})} &= p_k^{(\mathrm{KL})}\,,
\end{aligned}$$

which ultimately results in

$$\widehat{\langle\cdot\rangle}_N^{(\mathrm{KL})} \quad = \quad \widetilde{\langle\cdot\rangle}_N^{(\mathrm{KL})} \ . \tag{150}$$

$\Rightarrow$

$$\nabla_{\boldsymbol{g}^*} \cdot \widetilde{\langle v\rangle}_N^{(\mathrm{KL})} \quad = \quad \widetilde{\langle \boldsymbol{g}, \boldsymbol{g}^\top\rangle}_N^{(\mathrm{KL})\,-1} \cdot \widetilde{\langle \boldsymbol{g}, v\rangle}_N^{(\mathrm{KL})} \tag{151}$$

Under the Shannon/Kullback-Leibler entropy, the hedge ratios are equal to the regression coefficients of the payoff of the target contract onto the payoff-vector of the calibration instruments under the pricing measure [AG02].

Note: reverse projection onto price changes with respect to observable market variables (such as spot levels) can be accomplished if the sensitivities of the calibration instruments with respect to the market variables are available by independent, e.g. analytic, means:

$$\partial_{S_j} \cdot \quad = \quad \left(\partial_{S_j} \boldsymbol{g}^{*\top}\right) \cdot \nabla_{\boldsymbol{g}^*} \cdot \tag{152}$$

The benefits of Weighted Monte Carlo schemes are:-

- Exact matching of calibration instruments
- Hedge ratios are computed as a *free by-product*.

The potential downsides are:-

- Negative probablities $\Rightarrow$ destruction of equivalent martingale measure property.
- Numerical fitting after N iterations $\Rightarrow$ convergence diagrams change meaning.

Question: does the system of SDEs

$$\begin{aligned}
\mathrm{d}x &= -\kappa \cdot \frac{x}{s} \cdot (s-l)\mathrm{d}t + \sigma \mathrm{d}W_x \\
\mathrm{d}y &= -\kappa \cdot \frac{y}{s} \cdot (s-l)\mathrm{d}t + \sigma \mathrm{d}W_y \\
\mathrm{d}z &= -\kappa \cdot \frac{z}{s} \cdot (s-l)\mathrm{d}t + \sigma \mathrm{d}W_z
\end{aligned}$$

with $s = \sqrt{x^2+y^2+z^2}$ and $l = 1$, $\sigma = 1$, and $\kappa = 350$, have a stationary distribution?

What shape is it?

References

[Ack00] P.J. Acklam. An algorithm for computing the inverse normal cumulative distribution function. home.online.no/~pjacklam/notes/invnorm/index.html, June 2000. University of Oslo, Statistics Division.

[AG02] M. Avellaneda and R. Gamba. Conquering the Greeks in Monte Carlo: Efficient Calculation of the Market Sensitivities and Hedge-Ratios of Financial Assets by Direct Numerical Simulation. In Geman et al. [GMPV02], pages 93–109. www.math.nyu.edu/faculty/avellane/ConqueringTheGreeks.pdf.

[ALT03] A.N. Avramidis, P. L'ecuyer, and P.-A. Tremblay. Efficient simulation of gamma and variance-gamma processes. In Chick et al. [CSF+03], pages 319–326. www.informs-cs.org/wsc03papers/039.pdf.

[And07] L. B. Andersen. Efficient Simulation of the Heston Stochastic Volatility Model. *SSRN eLibrary*, 2007. ssrn.com/paper=946405.

[AS84] M. Abramowitz and I.A. Stegun. *Pocketbook of Mathematical Functions*. Harri Deutsch, 1984. ISBN 3-87144-818-4.

[BK04] M. Broadie and Ö. Kaya. Exact Simulation of Stochastic Volatility and other Affine Jump Diffusion Processes. Working paper, Columbia University, New York, 2004. www.orie.cornell.edu/~aberndt/FEseminar/papers04/exact_sim_200409.pdf.

[BM58] G. E. P. Box and M. E. Muller. A note on the generation of random normal deviates. *Annals Math. Statist.*, 29:610–611, 1958.

[CSF+03] S. Chick, P.J. Snchez, D. Ferrin, D.J. Morrice, and A.F. Seila, editors. *Proceedings of the 1998 Winter Simulation Conference*, 2003.

[Dos77] H. Doss. Liens entre équations différentielles stochastiques ordinaires. *Annales de l'Institut Henrie Poincaré. Probabilités et Statistiques*, 13:99–125, 1977.

[ELM01] P. Embrechts, F. Lindskog, and A. McNeil. Modelling Dependence with Copulas and Applications to Risk Management. Working paper, Department of Mathematics, ETHZ CH-8092, Zürich, Switzerland, 2001. www.risklab.ch/ftp/papers/DependenceWithCopulas.pdf.

[Fey48] R. Feynman. Space-time approach to non-relativistic quantum mechanics. *Rev. Mod. Phys.*, 20:367–387, 1948.

[Fox96] B. L. Fox. Generating Poisson processes by quasi-Monte Carlo. Technical report, SIM-OPT Consulting, 872 Timber Lane, Boulder, CO 80304, 1996. citeseer.nj.nec.com/cache/papers/cs/4246/http:zSzzSzwww.math.hkbu.edu.hkzSzqmczSzgpp.pdf/fox96generating.pdf.

[GMPV02] H. Geman, D. Madan, S. R. Pliska, and T. Vorst, editors. *Mathematical Finance - Bachelier Congress 2000*, 2002.

[Jäc02] P. Jäckel. *Monte Carlo methods in finance*. John Wiley and Sons, February 2002.

[Kac51] M. Kac. On some connections between probability theory and differential and integral equations. In *Proc. 2nd Berkeley Symposium on Math. Stat. & Probability*, pages 189–215, 1951.

[Kau01] R. Kaufmann. Copulas as an Integrated Risk Management Tool. Risk 2001 europe conference, april 10-11, paris, RiskLab, Departement Mathematik, ETH Zürich, Switzerland, 2001. www.math.ethz.ch/~kaufmann/Copulas/Paris2001.pdf.

[KL51] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22:79–86, 1951.

[KP99] P. E. Kloeden and E. Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1992, 1995, 1999.

[KS91] I. Karatzas and S. E. Shreve. *Brownian motion and Stochastic Calculus*. Springer, 1991.

[Kva00] B. Kvasov. *Methods of Shape-Preserving Spline Approximation*. World Scientific Publishing, 2000. ISBN 9810240104.

[L'E01] P. L'Ecuyer. Software for Uniform Random Number Generation: Distinguishing the Good and the Bad. In Peters et al. [PMR01]. www.iro.umontreal.ca/~lecuyer/myftp/papers/wsc01rng.pdf.

[LMS01] F. Lindskog, A. McNeil, and U. Schmock. Kendall's Tau for Elliptical Distributions. Working paper, RiskLab, Departement Mathematik, ETH Zürich, Switzerland, 2001. www.risklab.ch/ftp/papers/KendallsTau.pdf.

[MN97] M. Matsumoto and T. Nishimura. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. www.math.keio.ac.jp/~matumoto/emt.html, 1997.

[Nea73] H. R. Neave. On using the Box-Muller transformation with multiplicative congruential pseudo-random number generators. *Applied Statistics*, 22:92–97, 1973.

[NX95] H. Niederreiter and C. P. Xing. Low-discrepancy sequences obtained from algebraic function fields over finite fields. *Acta Arithmetica*, 72:281–298, 1995.

[NX96] H. Niederreiter and C. P. Xing. Low-discrepancy sequences and global function fields with many rational places. *Finite Fields and Their Applications*, 2:241–273, 1996. www.dismat.oeaw.ac.at/pirs/niedxing.html.

[PMR01] B. A. Peters, J. S. Smithand D. J. Medeiros, and M. W. Rohrer, editors. *Proceedings of the 2001 Winter Simulation Conference*, Dec 2001.

[PTVF92] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992. www.library.cornell.edu/nr/cbookcpdf.html.

[SB02] N. J. Smelser and P. B. Baltes, editors. *International Encyclopedia of the Social and Behavioral Sciences*, chapter Random Numbers (P. L'Ecuyer), pages 12735–12738. Pergamon, 2002. www.iro.umontreal.ca/~lecuyer/myftp/papers/ensbs.pdf.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27, 1948.

[Sob76] I. M. Sobol'. Uniformly Distributed Sequences with an Additional Uniform Property. *USSR Computational Mathematics and Mathematical Physics*, 16(5):236–242, 1976.

[Tez95] S. Tezuka. *Uniform Random Numbers: Theory and Practice*. Kluwer Academic Publishers, 1995.