MLMC for inaccurate SDE calculations

Mike Giles, Mario Hefter, Lukas Mayer,
Steffen Omland, Klaus Ritter

Mathematical Institute, University of Oxford

Dept. of Mathematics, TU Kaiserslautern

Rhein-Main Arbeitskreis, Feb 5, 2016

Outline

- introduction to multilevel Monte Carlo (MLMC)
- overview of current collaboration into a new kind of MLMC application

In presenting MLMC, I hope to emphasise the simplicity of the idea – the approach can be used in different ways in different contexts.

There are lots of people working on a very wide variety of applications – I won't have time to cover this but more details are available from my webpages.

Monte Carlo method

In stochastic models, we often have

$$\underset{\text{random input}}{\omega} \longrightarrow \underset{\text{intermediate variables}}{S} \longrightarrow \underset{\text{scalar output}}{P}$$

The Monte Carlo estimate for $\mathbb{E}[P]$ is an average of N independent samples $\omega^{(n)}$:

$$Y = N^{-1} \sum_{n=1}^{N} P(\omega^{(n)}).$$

This is unbiased, $\mathbb{E}[Y] = \mathbb{E}[P]$, and the Central Limit Theorem proves that as $N \to \infty$ the error becomes Normally distributed with variance $N^{-1}\mathbb{V}[P]$.

For ε r.m.s. error, need $N = O(\varepsilon^{-2})$ samples, so $O(\varepsilon^{-2})$ total cost if each one has $O(1)$ cost.

Monte Carlo method

In many cases, this is modified to

$$\underset{\text{random input}}{\omega} \longrightarrow \underset{\text{intermediate variables}}{\widehat{S}} \longrightarrow \underset{\text{scalar output}}{\widehat{P}}$$

where $\widehat{S}, \widehat{P}$ are approximations to S, P, in which case the MC estimate

$$\widehat{Y} = N^{-1} \sum_{n=1}^{N} \widehat{P}(\omega^{(n)})$$

is biased, and the Mean Square Error is

$$\mathbb{E}[\,(\widehat{Y} - \mathbb{E}[P])^2] \;=\; N^{-1}\,\mathbb{V}[\widehat{P}] + \Big(\mathbb{E}[\widehat{P}] - \mathbb{E}[P]\Big)^2$$

Greater accuracy requires larger N and smaller weak error $\mathbb{E}[\widehat{P}] - \mathbb{E}[P]$.

SDE Path Simulation

My interest was in SDEs (stochastic differential equations) for finance, which in a simple one-dimensional case has the form

$$\mathrm{d}S_t = a(S_t, t)\,\mathrm{d}t + b(S_t, t)\,\mathrm{d}W_t$$

Here $\mathrm{d}W_t$ is the increment of a Brownian motion – Normally distributed with variance $\mathrm{d}t$.

This is usually approximated by the simple Euler-Maruyama method

$$\widehat{S}_{t_{n+1}} = \widehat{S}_{t_n} + a(\widehat{S}_{t_n}, t_n)\,h + b(\widehat{S}_{t_n}, t_n)\,\Delta W_n$$

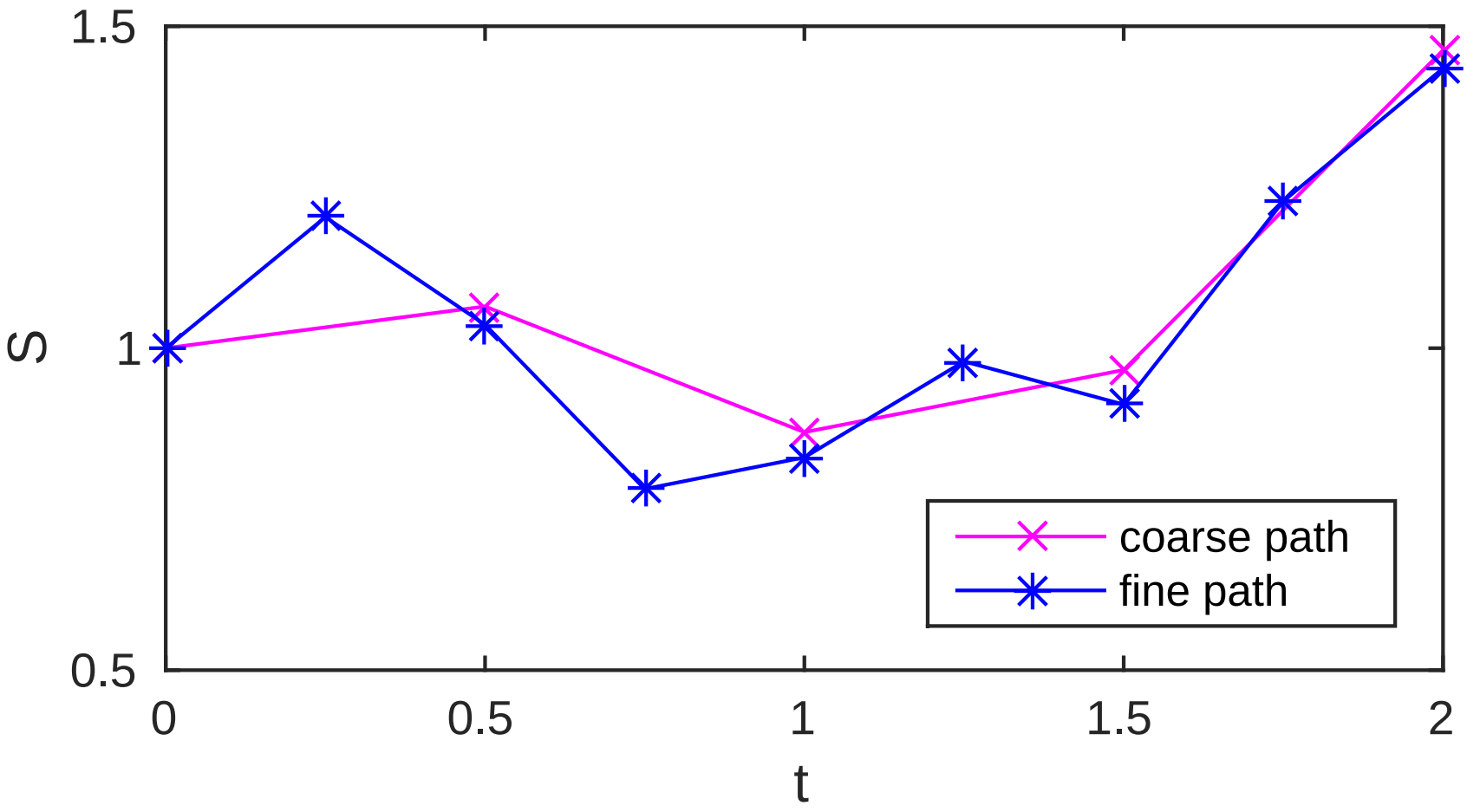
with uniform timestep h, and increments ΔW_n with variance h.

In simple applications, the output of interest is a function of the final value:

$$\widehat{P} \equiv f(\widehat{S}_T)$$

SDE Path Simulation

Geometric Brownian Motion: $\mathrm{d}S_t = r\,S_t\,\mathrm{d}t + \sigma\,S_t\,\mathrm{d}W_t$



SDE Path Simulation

Two kinds of discretisation error:

Weak error:

$$\mathbb{E}[\widehat{P}] - \mathbb{E}[P] = O(h)$$

Strong error:

$$\left(\mathbb{E}\left[\sup_{[0,T]}\left(\widehat{S}_t - S_t\right)^2\right]\right)^{1/2} = O(h^{1/2})$$

For reasons which will become clear, I prefer to use the Milstein discretisation for which the weak and strong errors are both $O(h)$.

SDE Path Simulation

The Mean Square Error is

$$N^{-1}\,\mathbb{V}[\widehat{P}] + \Big(\mathbb{E}[\widehat{P}] - \mathbb{E}[P]\Big)^2 \;\approx\; a\,N^{-1} + b\,h^2$$

If we want this to be ε^2, then we need

$$N = O(\varepsilon^{-2}), \qquad h = O(\varepsilon)$$

so the total computational cost is $O(\varepsilon^{-3})$.

To improve this cost we need to

- reduce N – variance reduction or Quasi-Monte Carlo methods
- reduce the cost of each path (on average) – MLMC

Two-level Monte Carlo

If we want to estimate $\mathbb{E}[\widehat{P}_1]$ but it is much cheaper to simulate $\widehat{P}_0 \approx \widehat{P}_1$, then since

$$\mathbb{E}[\widehat{P}_1] = \mathbb{E}[\widehat{P}_0] + \mathbb{E}[\widehat{P}_1 - \widehat{P}_0]$$

we can use the estimator

$$N_0^{-1} \sum_{n=1}^{N_0} \widehat{P}_0^{(0,n)} \;+\; N_1^{-1} \sum_{n=1}^{N_1} \left(\widehat{P}_1^{(1,n)} - \widehat{P}_0^{(1,n)} \right)$$

Benefit: if $\widehat{P}_1 - \widehat{P}_0$ is small, its variance will be small, so won't need many samples to accurately estimate $\mathbb{E}[\widehat{P}_1 - \widehat{P}_0]$, so cost will be reduced greatly.

Multilevel Monte Carlo

Natural generalisation: given a sequence $\widehat{P}_0, \widehat{P}_1, \ldots, \widehat{P}_L$

$$\mathbb{E}[\widehat{P}_L] = \mathbb{E}[\widehat{P}_0] + \sum_{\ell=1}^{L} \mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}]$$

we can use the estimator

$$N_0^{-1} \sum_{n=1}^{N_0} \widehat{P}_0^{(0,n)} \; + \; \sum_{\ell=1}^{L} \left\{ N_\ell^{-1} \sum_{n=1}^{N_\ell} \left(\widehat{P}_\ell^{(\ell,n)} - \widehat{P}_{\ell-1}^{(\ell,n)} \right) \right\}$$

with independent estimation for each level of correction

Multilevel Monte Carlo

If we define

- C_0, V_0 to be cost and variance of $\widehat{P}_0$
- C_ℓ, V_ℓ to be cost and variance of $\widehat{P}_\ell - \widehat{P}_{\ell-1}$

then the total cost is $\displaystyle\sum_{\ell=0}^{L} N_\ell\, C_\ell$ and the variance is $\displaystyle\sum_{\ell=0}^{L} N_\ell^{-1} V_\ell$.

Using a Lagrange multiplier μ^2 to minimise the cost for a fixed variance

$$\frac{\partial}{\partial N_\ell} \sum_{k=0}^{L} \left(N_k\, C_k + \mu^2 N_k^{-1} V_k \right) = 0$$

gives

$$N_\ell = \mu \sqrt{V_\ell / C_\ell} \quad \Longrightarrow \quad N_\ell\, C_\ell = \mu \sqrt{V_\ell\, C_\ell}$$

Multilevel Monte Carlo

Setting the total variance equal to ε^2 gives

$$\mu = \varepsilon^{-2} \left(\sum_{\ell=0}^{L} \sqrt{V_\ell \, C_\ell} \right)$$

and hence, the total cost is

$$\sum_{\ell=0}^{L} N_\ell \; C_\ell \;=\; \varepsilon^{-2} \left(\sum_{\ell=0}^{L} \sqrt{V_\ell C_\ell} \right)^2$$

in contrast to the standard cost which is approximately $\varepsilon^{-2} \, V_0 \, C_L$.

The MLMC cost savings are therefore approximately:

- V_L/V_0, if $\sqrt{V_\ell C_\ell}$ increases with level
- C_0/C_L, if $\sqrt{V_\ell C_\ell}$ decreases with level

Multilevel Path Simulation

With SDEs, level ℓ corresponds to approximation using M^ℓ timesteps, giving approximate payoff $\widehat{P}_\ell$ at cost $C_\ell = O(h_\ell^{-1})$.

Simplest estimator for $\mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}]$ for $\ell > 0$ is

$$\widehat{Y}_\ell = N_\ell^{-1} \sum_{n=1}^{N_\ell} \left(\widehat{P}_\ell^{(n)} - \widehat{P}_{\ell-1}^{(n)} \right)$$

using same driving Brownian path for both levels.

Analysis gives MSE $= \displaystyle\sum_{\ell=0}^{L} N_\ell^{-1} V_\ell + \Big(\mathbb{E}[\widehat{P}_L] - \mathbb{E}[P] \Big)^2$

To make RMS error less than ε

- choose $N_\ell \propto \sqrt{V_\ell / C_\ell}$ so total variance is less than $\frac{1}{2}\,\varepsilon^2$
- choose L so that $\Big(\mathbb{E}[\widehat{P}_L] - \mathbb{E}[P] \Big)^2 < \frac{1}{2}\,\varepsilon^2$

Multilevel Path Simulation

For Lipschitz payoff functions $P \equiv f(S_T)$, we have

$$\begin{aligned} V_\ell \;\equiv\; \mathbb{V}\left[\widehat{P}_\ell - \widehat{P}_{\ell-1}\right] \;&\leq\; \mathbb{E}\left[(\widehat{P}_\ell - \widehat{P}_{\ell-1})^2\right] \\ &\leq\; K^2\,\mathbb{E}\left[(\widehat{S}_{T,\ell} - \widehat{S}_{T,\ell-1})^2\right] \\ &=\; \begin{cases} O(h_\ell), & \text{Euler-Maruyama} \\ O(h_\ell^2), & \text{Milstein} \end{cases} \end{aligned}$$

and hence

$$V_\ell\, C_\ell = \begin{cases} O(1), & \text{Euler-Maruyama} \\ O(h_\ell), & \text{Milstein} \end{cases}$$

MLMC Theorem

(Slight generalisation of version in 2008 *Operations Research* paper)

If there exist independent estimators $\widehat{Y}_\ell$ based on N_ℓ Monte Carlo samples, each costing C_ℓ, and positive constants $\alpha, \beta, \gamma, c_1, c_2, c_3$ such that $\alpha \geq \frac{1}{2}\min(\beta, \gamma)$ and

i) $\left|\mathbb{E}[\widehat{P}_\ell - P]\right| \leq c_1\, 2^{-\alpha\,\ell}$

ii) $\mathbb{E}[\widehat{Y}_\ell] = \begin{cases} \mathbb{E}[\widehat{P}_0], & \ell = 0 \\ \mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}], & \ell > 0 \end{cases}$

iii) $\mathbb{V}[\widehat{Y}_\ell] \leq c_2\, N_\ell^{-1} 2^{-\beta\,\ell}$

iv) $\mathbb{E}[C_\ell] \leq c_3\, 2^{\gamma\,\ell}$

MLMC Theorem

then there exists a positive constant c_4 such that for any $\varepsilon < 1$ there exist L and N_ℓ for which the multilevel estimator

$$\widehat{Y} = \sum_{\ell=0}^{L} \widehat{Y}_\ell,$$

has a mean-square-error with bound $\mathbb{E}\left[\left(\widehat{Y} - \mathbb{E}[P]\right)^2\right] < \varepsilon^2$

with an expected computational cost C with bound

$$C \leq \begin{cases} c_4\, \varepsilon^{-2}, & \beta > \gamma, \\ c_4\, \varepsilon^{-2} (\log \varepsilon)^2, & \beta = \gamma, \\ c_4\, \varepsilon^{-2-(\gamma-\beta)/\alpha}, & 0 < \beta < \gamma. \end{cases}$$

MLMC Theorem

Two observations of optimality:

- MC simulation needs $O(\varepsilon^{-2})$ samples to achieve RMS accuracy ε. When $\beta > \gamma$, the cost is optimal — $O(1)$ cost per sample on average.

  (Would need multilevel QMC to further reduce costs)

- When $\beta < \gamma$, another interesting case is when $\beta = 2\alpha$, which corresponds to $\mathbb{E}[\widehat{Y}_\ell]$ and $\sqrt{\mathbb{E}[\widehat{Y}_\ell^2]}$ being of the same order as $\ell \to \infty$.

  In this case, the total cost is $O(\varepsilon^{-\gamma/\alpha})$, which is the cost of a single sample on the finest level — again optimal.

New research

We are interested in three sources of inaccuracy (quite different to the usual stochastic uncertainty):

- reduced precision in performing path calculations
- reduced number of bits in uniform random number generation
- reduced accuracy in converting uniform r.v.'s to Normal

Objectives:

- design good MLMC estimators
- analyse variance behaviour and determine complexity (based on some appropriate cost model)

Low precision computations

Standard C/C++ computations have a choice of two levels of floating point precision:

- single (`float`) – 32-bit with 24-bit mantissa
- double (`double`) – 64-bit with 54-bit mantissa

In each case, only finitely many numbers can be represented – other numbers are rounded to nearest representable value. 1 ulp (unit of least precision) is the difference between one representable value and its nearest neighbour

$$\text{relative error} \equiv 1 \text{ ulp/value} \approx \begin{cases} 10^{-7}, & \text{single} \\ 10^{-16}, & \text{double} \end{cases}$$

Rounding error introduced by addition/multiplication can be modelled as Normally distributed with magnitude 1 ulp.

Low precision computations

Most people do calculations in double precision, so why should we care?

- on CPUs, single precision is twice as fast as double, if code is vectorised.
- on GPUs, the difference can be a factor 5–10, and there is an extra factor 2 on the latest NVIDIA GPUs if you use half-precision with a 10-bit mantissa.
- FPGAs (field-programmable gate arrays) allow arbitrary fixed-point and floating point arithmetic, with huge potential savings.

The first work was particularly motivated by FPGAs for which "exact" Normal r.v.'s can be generated easily.

MLMC approach 1

Consider one Euler-Maruyama timestep:

$$\widehat{S}_{n+1} = \widehat{S}_n + a(\widehat{S}_n)\, h + b(\widehat{S}_n)\, \Delta W_n.$$

The effects of rounding errors can be modelled as

$$\widehat{S}_{n+1} = \widehat{S}_n + a(\widehat{S}_n)\, h + b(\widehat{S}_n)\, \Delta W_n + \widehat{S}_n\, 2^{-p}\, Z_n$$

where p is number of bits of precision, and Z_n is a standard Normal r.v. After T/h timesteps,

$$\widehat{S}_{T/h} - \widehat{S}^{exact}_{T/h} \approx \sum_{n=0}^{T/h-1} c_n\, 2^{-p}\, Z_n$$

where $c_n = O(1)$, and therefore for Lipschitz functionals

$$\mathbb{V}[\widehat{S}_{T/h} - \widehat{S}^{exact}_{T/h}] = O(2^{-2p} h^{-1}) \quad \Longrightarrow \quad \mathbb{V}[\widehat{P} - \widehat{P}^{exact}] = O(2^{-2p} h^{-1})$$

MLMC approach 1

We want to estimate $\mathbb{E}[P]$, and P is approximated by $\widehat{P}_\ell$ on level ℓ with $2^{2\ell}$ timesteps and precision p_ℓ.

$$\mathbb{E}[\widehat{P}_L] = \mathbb{E}[\widehat{P}_0] + \sum_{\ell=1}^{L} \mathbb{E}[\,\widehat{P}_\ell - \widehat{P}_{\ell-1}]$$

and so we use the MLMC estimator

$$N_0^{-1} \sum_{i=1}^{N_0} \widehat{P}_0^{(i,0)} + \sum_{\ell=1}^{L} N_\ell^{-1} \sum_{i=1}^{N_\ell} (\widehat{P}_\ell^{(i,\ell)} - \widehat{P}_{\ell-1}^{(i,\ell)})$$

with $\widehat{P}_\ell^{(i,\ell)} - \widehat{P}_{\ell-1}^{(i,\ell)}$ coming from same Brownian path.

This means we generate "exact" Brownian increments ΔW for the fine path, then sum them to get the increments for the coarse path

MLMC approach 1

On level ℓ the variance is

$$\mathbb{V}[\widehat{P}_\ell - \widehat{P}_{\ell-1}] \;\approx\; \mathbb{V}[\widehat{P}_\ell - \widehat{P}_\ell^{exact}] + \mathbb{V}[\widehat{P}_\ell^{exact} - \widehat{P}_{\ell-1}^{exact}] + \mathbb{V}[\widehat{P}_{\ell-1} - \widehat{P}_{\ell-1}^{exact}]$$

so empirically choose precision $p_{\ell-1}$ so that

$$\mathbb{V}[\widehat{P}_{\ell-1} - \widehat{P}_{\ell-1}^{exact}] \approx 0.1\; \mathbb{V}[\widehat{P}_\ell^{exact} - \widehat{P}_{\ell-1}^{exact}]$$

for low cost with minimal extra variance. We expect that this requires

$$2^{-2p_{\ell-1}} h_{\ell-1}^{-1} = O(h_{\ell-1}) \quad \Longrightarrow \quad 2^{-p_\ell} = O(h_\ell) = O(2^{-2\ell})$$

Numerical results confirm this, with $p_\ell = 2\,\ell + \text{const}$

On FPGAs achieve good results – overall, same accuracy with factor 2 reduction in run-time compared to "single" precision.

The savings would have been even greater with the Milstein discretisation, as this leads to most work on coarse levels.

Uniform random number generation

Standard theory is based on uniform random variables U in $(0,1)$.

Computational implementations are close to this ideal with most uniform generators giving values of the form

$$U = k/m$$

where m is fixed and large (e.g. $m = 2^{32}$) and k is uniformly distributed in $\{0, 1, 2, \ldots, m-1\}$.

We are concerned with what happens when m is small (e.g. $m = 8$) which will clearly lead to errors.

Random number generation

To avoid problems at $U=0$, we will use

$$U = (k+\tfrac{1}{2})/m \quad \Longrightarrow \quad U \in (0,1)$$

or

$$U = (2k+1-m)/m \quad \Longrightarrow \quad U \in (-1,1)$$

If $U \in (0,1)$, it can be converted to a standard Normal random variable by

$$Z = \Phi^{-1}(U).$$

Alternatively, if $U \in (-1,+1)$, then it can be converted by

$$Z \;=\; \Phi^{-1}(\tfrac{1}{2}(U+1)) \;\equiv\; \sqrt{2}\,\mathrm{erfinv}(U),$$

where $\mathrm{erfinv}()$ is the inverse error function (part of all standard mathematical libraries).

Conversion to Normals

In the latter case, we can approximate the conversion by

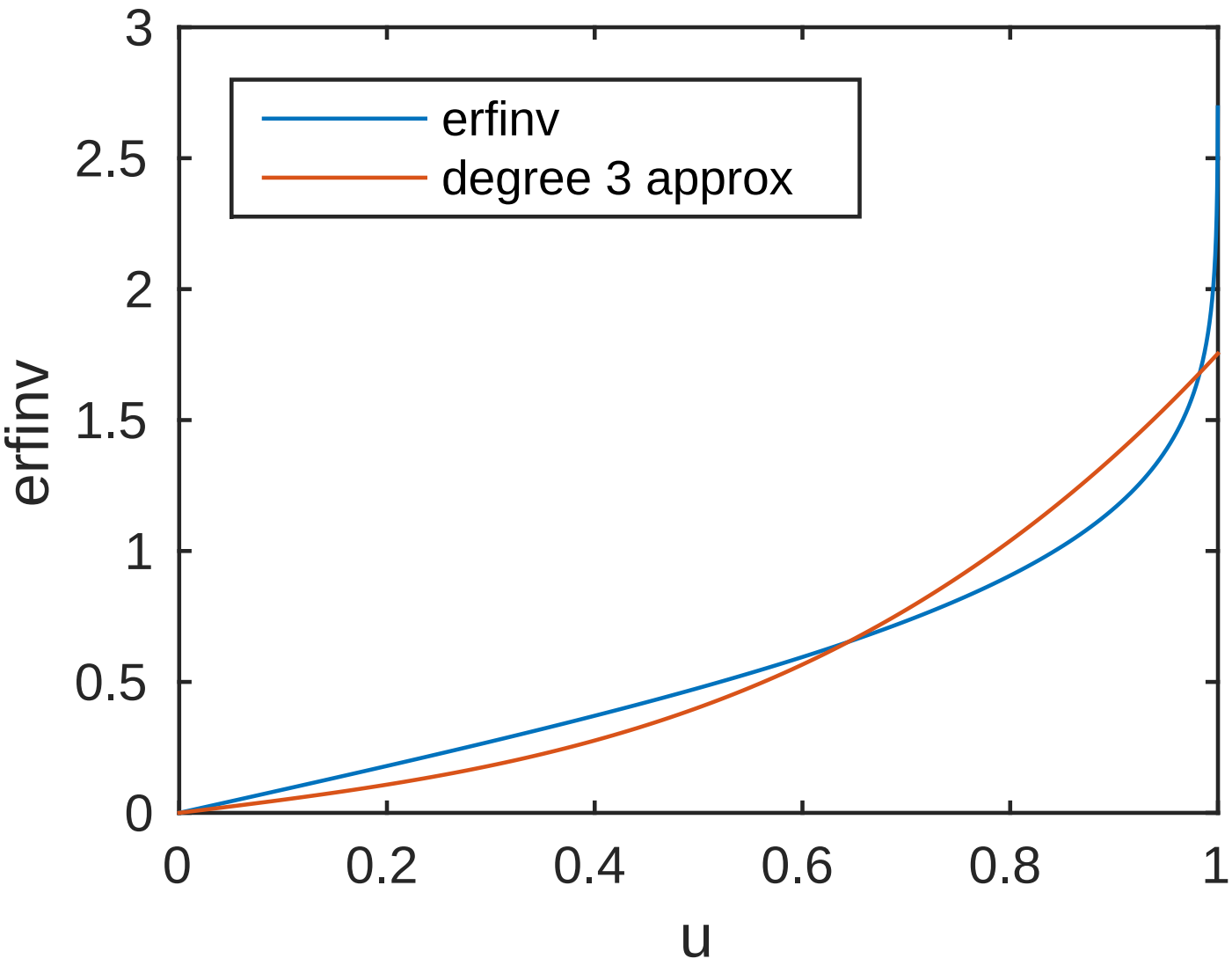
$$\widetilde{Z} = \sqrt{2}\,\widetilde{\mathrm{erfinv}}(U)$$

where $\widetilde{\mathrm{erfinv}}()$ is an (inaccurate) approximation to $\mathrm{erfinv}()$.

Note that if $\widetilde{\mathrm{erfinv}}()$ is an odd function (anti-symmetric around $U=0$) then $\mathbb{E}[\widetilde{Z}] = 0$.

It is desirable to enforce $\mathbb{E}[\widetilde{Z}^2] = 1$, where the expectation is over the discrete set of possible U values.

Conversion to Normals



MLMC algorithm 2

For the second algorithm, we use approximate random number generation.

Problem: how do we couple the simulations on different levels?

In the standard MLMC algorithm for SDEs using 2^ℓ timesteps on level ℓ, we generate Brownian increments ΔW_n on the finer level, then sum them in pairs to get the Brownian increments for the coarser level.

Since

$$\sqrt{h}\; N(0,1) + \sqrt{h}\; N(0,1) \;\sim\; \sqrt{2h}\; N(0,1)$$

this is equivalent in distribution to directly generating the Brownian increments on the coarser level – hence the telescoping sum is OK.

MLMC algorithm 2

However, summing the Brownian increments in pairs does not result in an equivalent distribution when we have the new inaccuracies.

Thus, we would violate the MLMC telescoping sum if we did it this way.

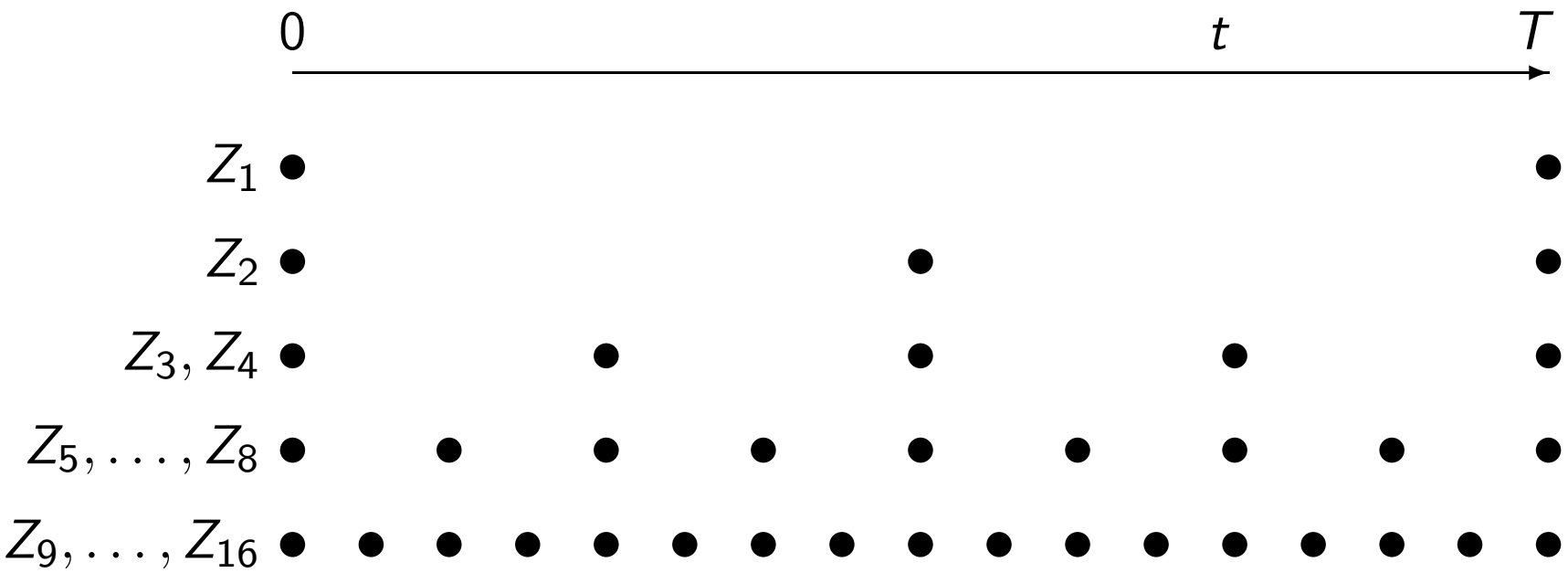
Solution: use a Brownian Bridge construction which works recursively, progressively refining the Brownian path representation.

$$\begin{array}{ccccccc} U_\ell & \longrightarrow & Z_\ell & \xrightarrow{\text{BB}} & W_\ell & \longrightarrow & \Delta W_\ell \\ \downarrow & \text{truncate} & & & & & \\ U_{\ell-1} & \longrightarrow & Z_{\ell-1} & \xrightarrow{\text{BB}} & W_{\ell-1} & \longrightarrow & \Delta W_{\ell-1} \end{array}$$

This guarantees the same distribution for Brownian increments on level ℓ, because the construction at level ℓ always uses the same U_ℓ uniform discrete distribution.

MLMC algorithm 2

Brownian Bridge construction:



More bits are used for random number generation on coarse levels of Brownian Bridge, fewer on finer levels.

MLMC algorithm 2

Current status of research:

- cost model counts the number of random bits used
- L_2 error analysis for Brownian path construction has the same complexity/accuracy relationship as previous research by others on quantisation methods using non-uniform discrete distributions
- L_∞ error analysis is poorer – non-uniform sampling of U gives better approximation of the tails of the Normal distribution
- numerical results for SDE solutions based on the Brownian paths look encouraging, but analysis looks challenging
- now considering a different approach which may be easier to analyse: $2^{\ell/2}$ timesteps with "exact" Brownian increments, $2^{\ell/2}$ sub-intervals within each timestep for approximate Brownian Bridge interpolant

Conclusions

- multilevel idea is very simple; key question is how to apply it in new situations, and perform the numerical analysis
- new research on inaccurate computations illustrates the flexibility of the MLMC approach
- MLMC is being used for an increasingly wide range of applications; biggest computational savings are when coarsest (reasonable) approximation is much cheaper than finest

References

M.B. Giles. 'Multilevel Monte Carlo methods'. pp. 83-103 in Monte Carlo and Quasi-Monte Carlo Methods 2012, Springer, 2013.

C. Brugger, C. de Schryver, N. Wehn, S. Omland, M. Hefter, K. Ritter, A. Kostiuk, R. Korn. 'Mixed precision multilevel Monte Carlo on hybrid computing systems', pp. 215-222, IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr), 2104.

Webpages for my other research papers and talks:

`people.maths.ox.ac.uk/gilesm/mlmc.html`
`people.maths.ox.ac.uk/gilesm/slides.html`

Webpage for new 70-page *Acta Numerica* review and MATLAB test codes:

`people.maths.ox.ac.uk/gilesm/acta/`

MLMC Community

Webpage: people.maths.ox.ac.uk/gilesm/mlmc_community.html

Abo Academi (Avikainen) – numerical analysis
Basel (Harbrecht) – elliptic SPDEs, sparse grids
Bath (Kyprianou, Scheichl, Shardlow, Yates) – elliptic SPDEs, MCMC, Lévy-driven SDEs, stochastic chemical modelling
Chalmers (Lang) – SPDEs
Duisburg (Belomestny) – Bermudan and American options
Edinburgh (Davie, Szpruch) – SDEs, numerical analysis
EPFL (Abdulle) – stiff SDEs and SPDEs
ETH Zürich (Jenny, Jentzen, Schwab) – SPDEs, multilevel QMC
Frankfurt (Gerstner, Kloeden) – numerical analysis, fractional Brownian motion
Fraunhofer ITWM (Iliev) – SPDEs in engineering
Hong Kong (Chen) – Brownian meanders, nested simulation in finance
IIT Chicago (Hickernell) – SDEs, infinite-dimensional integration, complexity analysis
Kaiserslautern (Heinrich, Korn, Ritter) – finance, SDEs, parametric integration, complexity analysis
KAUST (Tempone, von Schwerin) – adaptive time-stepping, stochastic chemical modelling
Kiel (Gnewuch) – randomized multilevel QMC
LPMA (Frikha, Lemaire, Pagès) – numerical analysis, multilevel extrapolation, finance applications
Mannheim (Neuenkirch) – numerical analysis, fractional Brownian motion
MIT (Peraire) – uncertainty quantification, SPDEs
Munich (Hutzenthaler) – numerical analysis
Oxford (Baker, Giles, Hambly, Reisinger) – SDEs, SPDEs, numerical analysis, finance applications, stochastic chemical modelling
Passau (Müller-Gronbach) – infinite-dimensional integration, complexity analysis
Stanford (Glynn) – numerical analysis, randomized multilevel
Strathclyde (Higham, Mao) – numerical analysis, exit times, stochastic chemical modelling
Stuttgart (Barth) – SPDEs
Texas A&M (Efendiev) – SPDEs in engineering
UCLA (Caflisch) – Coulomb collisions in physics
UNSW (Dick, Kuo, Sloan) – multilevel QMC
UTS (Baldeaux) – multilevel QMC
Warwick (Stuart, Teckentrup) – MCMC for SPDEs
WIAS (Friz, Schoenmakers) – rough paths, fractional Brownian motion, Bermudan options
Wisconsin (Anderson) – numerical analysis, stochastic chemical modelling