Multilevel Monte Carlo methods

Mike Giles

Mathematical Institute, University of Oxford

University of Manchester UQ Seminar

March 15th, 2017

Acknowledgements: many collaborators inc. Abdul-Lateef Haji-Ali

Outline

- Part I: an overview of the multilevel Monte Carlo method, emphasising
  - the simplicity of the idea
  - its flexibility – it's not prescriptive, more an approach
  - there are lots of people working on a variety of applications
- Part II: current research
  - mixed precision arithmetic
  - VaR (Value-at-Risk)

Monte Carlo method

In stochastic models, we often have

$$\begin{array}{ccccc} \omega & \longrightarrow & S & \longrightarrow & P \\ \text{random input} & & \text{intermediate variables} & & \text{scalar output} \end{array}$$

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 $N^{-1}\mathbb{V}[P]$.

Monte Carlo method

In many cases, this is modified to

$$\begin{array}{ccccc} \omega & \longrightarrow & \widehat{S} & \longrightarrow & \widehat{P} \\ \text{random input} & & \text{intermediate variables} & & \text{scalar output} \end{array}$$

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

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

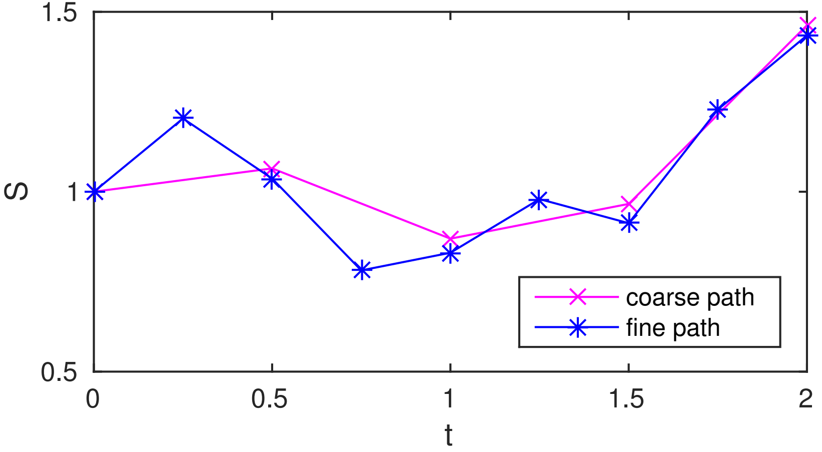
with uniform timestep h, and increments ΔW_n with variance h.

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

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

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

(For those familiar with multigrid methods, this is in the spirit of "fine grid accuracy with coarse grid cost" and going from 2 levels to many is natural)

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

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 $\left(\mathbb{E}[\widehat{P}_L] - \mathbb{E}[P] \right)^2 < \frac{1}{2}\,\varepsilon^2$

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.

MLMC work on SDEs

- Milstein discretisation for path-dependent options – G (2008)
- numerical analysis – G, Higham, Mao (2009), Avikainen (2009), G, Debrabant, Rößler (2012)
- financial sensitivities ("Greeks") – Burgos (2011)
- jump-diffusion models – Xia (2011)
- Lévy processes – Dereich (2010), Marxen (2010), Dereich & Heidenreich (2011), Xia (2013), Kyprianou (2014)
- American options – Belomestny & Schoenmakers (2011)
- Milstein in higher dimensions without Lévy areas – G, Szpruch (2014)
- adaptive timesteps – Hoel, von Schwerin, Szepessy, Tempone (2012), G, Lester, Whittle (2014)

MLMC work on random/stochastic PDEs

Quite natural application, with better cost savings than SDEs due to higher dimensionality

- Graubner & Ritter (Darmstadt, 2008) – parabolic
- Cliffe, G, Scheichl, Teckentrup (Bath/Nottingham, 2011-16) – elliptic
- Barth, Jenny, Lang, Meyer, Mishra, Müller, Schwab, Sukys, Zollinger (ETH Zürich, 2011-16) – elliptic, parabolic, hyperbolic
- G, Reisinger (Oxford, 2012) – parabolic
- Harbrecht, Peters (Basel, 2013-16) – elliptic
- Efendiev (Texas A&M, 2015) – numerical homogenization
- Vidal-Codina, G, Peraire (MIT, 2015-6) – reduced basis approximation

Engineering Uncertainty Quantification

Simplest possible example:

- 3D elliptic PDE, with uncertain boundary data
- grid spacing proportional to $2^{-\ell}$ on level ℓ
- cost is $O(2^{+3\ell})$, if using an efficient multigrid solver
- 2nd order accuracy means that

$$
\begin{aligned}
& \widehat{P}_\ell(\omega) - P(\omega) &\approx\ & c(\omega)\, 2^{-2\ell} \\
\Longrightarrow\quad & \widehat{P}_{\ell-1}(\omega) - \widehat{P}_\ell(\omega) &\approx\ & 3\, c(\omega)\, 2^{-2\ell}
\end{aligned}
$$

- hence, $\alpha = 2,\ \beta = 4,\ \gamma = 3$
- cost is $O(\varepsilon^{-2})$ to obtain ε RMS accuracy
- this compares to $O(\varepsilon^{-3/2})$ cost for one sample on finest level, so $O(\varepsilon^{-7/2})$ for standard Monte Carlo

Non-geometric multilevel

So far, almost all MLMC applications use a geometric sequence of levels, refining the timestep (or grid spacing) by a constant factor when going from level $\ell-1$ to level ℓ.

Coming from a multigrid background, this is very natural, but it is **NOT** a requirement of the multilevel Monte Carlo approach.

All MLMC needs is a sequence of levels with

- increasing accuracy
- increasing cost
- increasingly small difference between outputs on successive levels

Reduced Basis PDE approximation

Vidal-Codina, Nguyen, G, Peraire (2015) take a fine FE discretisation:

$$A(\omega)\, u = f(\omega)$$

and use a reduced basis approximation

$$u \approx \sum_{k=1}^{K} v_k\, u_k$$

to obtain a low-dimensional reduced system

$$A_r(\omega)\, v = f_r(\omega)$$

- larger $K \implies$ greater accuracy at greater cost
- in multilevel treatment, K_ℓ varies with level
- brute force optimisation is used to determine the optimal number of levels, and reduced basis size on each level

Other MLMC applications

- parametric integration, integral equations (Heinrich)
- multilevel QMC (Dick, G, Kuo, Scheichl, Schwab, Sloan)
- stochastic chemical reactions (Anderson & Higham, Tempone)
- mixed precision computation on FPGAs (Korn, Ritter, Wehn)
- MLMC for MCMC (Scheichl, Schwab, Stuart, Teckentrup)
- Coulomb collisions in plasma (Caflisch)
- nested simulation (Haji-Ali & Tempone, Hambly & Reisinger)
- invariant distribution of contractive Markov process (Glynn & Rhee)
- invariant distribution of contractive SDEs (G, Lester & Whittle)

Three MLMC extensions

- unbiased estimation – Rhee & Glynn (2015)
  - randomly selects the level for each sample
  - no bias, and finite expected cost and variance if $\beta > \gamma$
- Richardson-Romberg extrapolation – Lemaire & Pagès (2014)
  - reduces the weak error, and hence the number of levels required
  - particularly helpful when $\beta < \gamma$
- Multi-Index Monte Carlo – Haji-Ali, Nobile, Tempone (2015)
  - important extension to MLMC approach, combining MLMC with sparse grid methods

Current MLMC projects in Oxford

- SDEs with highly nonlinear drift – Wei Fang
  adaptive timestepping and numerical analysis
- long-chain molecules in solution – Shenghan Ye, Endre Süli
- EVPPI – Wei Fang, Zhenru Wang, Howard Thom (Bristol)
  nested simulation problem in medical decision making
- SPDEs – Matteo Croci, Patrick Farrell, Marie Rognes (Simula)
  MLMC / MLQMC for SPDEs $\longrightarrow$ FEniCS software package
- stochastic reaction networks – Chris Lester, Casper Beentjes, Ruth Baker
- parabolic SPDEs in credit modelling – Zhenru Wang, Christoph Reisinger
- Big Data stochastic gradient method – Tigran Nagapetyan
- MLMC with mixed precision arithmetic – Oliver Sheridan-Methven
- Value-at-Risk – Abdul-Lateef Haji-Ali

Mixed 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.

Mixed 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 (and the future Intel "Knights Mill" Xeon Phi) 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.

Research at TU Kaiserslautern

In a collaboration between mathematicians (Ritter, Korn) and electrical engineers (Wehn), the first work was motivated by FPGAs which can easily generate "exact" Normal r.v.'s.

In an SDE approximation, 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 the number of bits of precision, and Z_n is a standard Normal r.v.

After T/h timesteps,

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

so for Lipschitz functionals

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

Research at TU Kaiserslautern

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 can 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.

Research at TU Kaiserslautern

On level ℓ the variance is

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

so for low cost with minimal extra variance choose precision $p_{\ell-1}$ so that

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

This suggests

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

FPGA implementation achieves good results – overall, same accuracy with factor 2 reduction in run-time compared to “single” precision.

New Oxford project

- funded by ARM, the focus is on processors with half/single/double precision vector arithmetic
- start with a standard MLMC expansion, e.g. for an SDE,

$$\mathbb{E}[\widehat{P}_L] = \sum_{\ell=0}^{L} \mathbb{E}[\Delta \widehat{P}_\ell], \qquad \Delta \widehat{P}_0 \equiv \widehat{P}_0$$

  then approximate each $\mathbb{E}[\Delta \widehat{P}_\ell]$ using mixed precision arithmetic

$$\mathbb{E}[\Delta \widehat{P}_\ell] \approx \mathbb{E}[\Delta \widehat{P}_\ell^{(1/2)}] + \mathbb{E}[\Delta \widehat{P}_\ell^{(1)} - \Delta \widehat{P}_\ell^{(1/2)}] + \mathbb{E}[\Delta \widehat{P}_\ell^{(2)} - \Delta \widehat{P}_\ell^{(1)}]$$

- will also look at random number generation and the inverse CDF method for transforming uniform r.v.'s into other distributions
- could split half precision computation into multiple levels with different transformations for Normals

Value-at-Risk

Banks, insurance companies, pension companies and regulators, are all concerned about the risk of a very large loss in a short time.

Given a risk horizon τ (1 week for banks, 1 year for pension/insurance companies?) with a given distribution for risk factors R_τ over that interval, the simplest question is

What is the probability of a portfolio loss L exceeding L_{max}?

This means estimating $\mathbb{P}[L > L_{max}] \equiv \mathbb{E}\left[\mathbf{1}(L > L_{max})\right]$ where

$$L(R_\tau) \ = \ \sum_{p=1}^{P} L_p(R_\tau) \ = \ \sum_{p=1}^{P} \mathbb{E}[f_p] - \mathbb{E}[f_p | R_\tau]$$

This is therefore a nested simulation problem, and the indicator function makes it even harder.

Value-at-Risk

The true VaR L_α is defined implicitly by

$$\mathbb{P}[L > L_\alpha] = \alpha$$

for some specified small α.

This involves either a root-finding process to determine L_α, or ordering multiple samples of L to find the appropriate quantile.

Another important risk measure is Conditional Value-at-Risk (CVaR), also known as Expected Shortfall,

$$\mathbb{E}\left[L \mid L > L_\alpha \right].$$

Value-at-Risk

What makes it expensive?   Where is the potential for MLMC?

- large number of financial products in the portfolio (P)
- often needs lots of Monte Carlo samples for inner conditional expectation (M)
- sometimes needs lots of timesteps for SDE approximation (T)

P, M and T all offer possibilities for MLMC treatment

Prior research on VaR

Gordy & Juneja (2010) considered

$$\mathbb{P}\left[L\!>\!L_{max}\right] \;\equiv\; \mathbb{E}\left[\mathbf{1}\left(L\!>\!L_{max}\right)\right]$$

using N outer samples for R_τ, and M inner samples to estimate $L(R_\tau)$.

The variance for the estimator for $L(R_\tau)$ is $O(M^{-1}P^{-1})$, and Gordy & Juneja prove this produces a bias in the outer estimate of the same order.

Hence, for ε RMS accuracy require

- $M = \max(1, O(\varepsilon^{-1}P^{-1}))$
- $N = O(\varepsilon^{-2})$

and so the complexity is $O(M\,N\,P) = O(\max(\varepsilon^{-2}P, \varepsilon^{-3}))$.

Prior research on VaR

Broadie, Du & Moallemi (2011) improved on Gordy & Juneja by noting that we don't need many samples to determine whether $L > L_{max}$ unless $L - L_{max}$ is small.

Switching to an adaptive strategy if we use

$$M = \left\lceil \min\left(c\,\varepsilon^{-1} P^{-1}, 9\,\sigma^2(R_\tau)/d^2(R_\tau)\right) \right\rceil$$

then the average number of inner samples is

$$\overline{M} = \max(1, O(\varepsilon^{-1/2} P^{-1})),$$

reducing the overall complexity to $O(\overline{M}\,N\,P) = O(\max(\varepsilon^{-2} P, \varepsilon^{-5/2}))$.

This is better, but still not the $O(\varepsilon^{-2})$ that we aim for.

Also, the issue of timestepping approximation hasn't been addressed yet.

Oxford VaR project

One key idea: conditional on R_τ, the total loss is

$$L((R_\tau) \;\equiv\; \sum_{p=1}^{P} L_p(R_\tau) \;=\; P\;\mathbb{E}[L_p(R_\tau)]$$

where p is uniformly distributed in $\{1,2,\ldots,P\}$ in the r.h.s. expectation

Combining this with expectations over the Brownian paths, and Rhee & Glynn's random MLMC technique for the timestepping, we end up with an inner conditional expectation of the form

$$L(R_\tau) = \mathbb{E}[g(p,W,\ell')].$$

We then combine MLMC with Broadie *et al*'s adaptive algorithm to use $M_\ell(R_\tau)$ inner samples on level ℓ, with

$$M_\ell(R_\tau) = \max\left(c_1\, 2^\ell, \min\left(c_2\, 4^\ell, 9\,\sigma^2(R_\tau)/\widehat{d}^2(R_\tau)\right)\right)$$

with $\widehat{d} \approx L(R_\tau) - L_{max}$.

Oxford VaR project

Doing this, we get $\alpha \approx 2$, $\beta \approx 1$, $\gamma \approx 1$ and hence the complexity is roughly $O(\varepsilon^{-2})$, independent of P.

Current status:

- most of this is working numerically, but we are still fine-tuning the adaptive strategy, especially with Rhee/Glynn randomisation which gives a function $g(p, W, \ell')$ with limited finite moments
- currently working on the numerical analysis – a fun challenge!
- will then work on making portfolio testcase more realistic, and the calculation of VaR (by stochastic root-finding?) and CVar

Conclusions

- multilevel idea is very simple; key question is how to apply it in new situations, and perform the numerical analysis
- discontinuous output functions can cause problems, but there is a lot of experience now in coping with this
- there are also "tricks" which can be used in situations with poor strong convergence
- being used for an increasingly wide range of applications; biggest computational savings when coarsest (reasonable) approximation is much cheaper than finest
- currently, getting at least $100\times$ savings for SPDEs and stochastic chemical reaction simulations

References

Webpages for my research papers and talks:

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

70-page *Acta Numerica 2015* review and MATLAB test codes:

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

– contains references to almost all MLMC research

MLMC community webpage:

`people.maths.ox.ac.uk/gilesm/mlmc_community.html`