From CFD to computational finance (and back again?)

Mike Giles

University of Oxford
Mathematical Institute

MIT Center for Computational Engineering Seminar

March 14th, 2013

Outline

- a short personal history
- computational finance
- "smoking adjoints"
- multilevel Monte Carlo

Short personal history

- BA (Maths) at Cambridge, 1978–81
- SM/PhD (Aero/Astro) at MIT, 1981–85
- Assistant/Associate Prof in Aero/Astro, 1985–92
- Reader/Prof in Computing Laboratory, Oxford, 1992–2004
- moved to Mathematical Institute in 2004
- acting head of the Mathematical and Computational Finance Group

Research history

CFD: 1981 – 1992

- compressible flow CFD
- turbomachinery applications (GTL and Rolls-Royce)
- non-reflecting boundary conditions
- vector/multi-threaded parallel computing, and visualisation

CFD: 1992 – 2004

- distributed-memory parallel computing
- adjoint methods for design
- HYDRA CFD code – now the main CFD code used by Rolls-Royce for all turbomachinery design
- started getting into uncertainty quantification

Research history

Developing HYDRA was too much like hard work, not enough fun — time for a change

Considered computational biology (but I know nothing about biology)

Instead moved to computational finance:

- lots of opportunities (at least there were in 2004!)
- close to London (one of top two international finance centres)
- excellent colleagues in Mathematics working on modelling side
- thought I could exploit my computational PDE knowledge

Currently:

- 50% effort on Monte Carlo methods – half in finance
- 50% effort on HPC, primarily using GPUs

Computational Finance

Options pricing – investment banks

- Monte Carlo methods (60%)
- PDEs / finite difference methods (30%)
- other semi-analytic methods (10%)

High-frequency algorithmic trading – hedge funds

Might seem a bad time to be in this business, but as an academic it's fine:

- clear need for better models
- regulators and internal risk management demanding more simulation
- computational finance accounts for 10% of Top500 "supercomputers"
- only problem is lack of research funding

Computational Finance

Computational finance reminds me of CFD about 20 years ago

- not many academics working on numerical methods
- codes are small – my biggest is probably 1000 lines
- lots of low-hanging fruit, surprisingly more on the Monte Carlo side than on the PDE side
- Olivier Pironneau, Peter Forsyth and others moved earlier from CFD to finance, but kept to PDEs
- Monte Carlo researchers have mainly come from theoretical physics (and don't know about design optimisation)
- in the past, each product group within a bank often had its own codes — past 5 years have seen consolidation, often resulting in a single corporate Monte Carlo system for both London and New York

SDEs in Finance

In computational finance, stochastic differential equations are used to model the behaviour of

- stocks
- interest rates
- exchange rates
- weather
- electricity/gas demand
- crude oil prices
- ...

SDEs in Finance

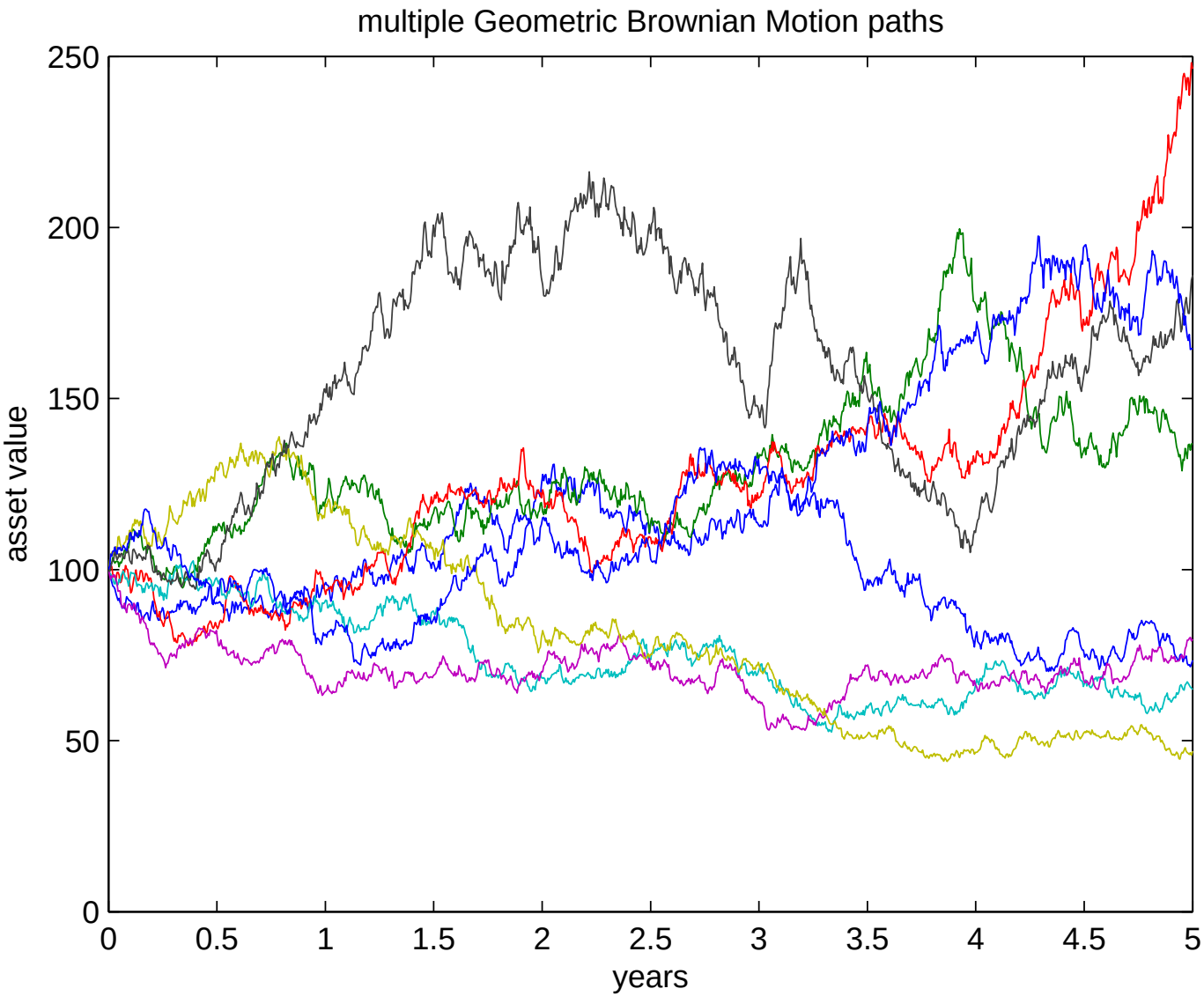
Stochastic differential equations are just ordinary differential equations plus an additional random source term.

The stochastic term accounts for the uncertainty of unpredictable day-to-day events.

The aim is **not** to predict exactly what will happen in the future, but to predict the probability of a range of possible things that **might** happen, and compute some averages, or the probability of an excessive loss.

This is really just uncertainty quantification, and they've been doing it for quite a while because they have so much uncertainty.

SDEs in Finance



SDEs in Finance

Examples:

- Geometric Brownian motion (Black-Scholes model for stock prices)

$$\mathrm{d}S = r\,S\,\mathrm{d}t + \sigma\,S\,\mathrm{d}W$$

- Cox-Ingersoll-Ross model (interest rates)

$$\mathrm{d}r = \alpha(b-r)\,\mathrm{d}t + \sigma\sqrt{r}\,\mathrm{d}W$$

- Heston stochastic volatility model (stock prices)

$$\begin{aligned} \mathrm{d}S &= r\,S\,\mathrm{d}t + \sqrt{V}\,S\,\mathrm{d}W_1 \\ \mathrm{d}V &= \lambda\,(\sigma^2 - V)\,\mathrm{d}t + \xi\sqrt{V}\;\mathrm{d}W_2 \end{aligned}$$

with correlation ρ between $\mathrm{d}W_1$ and $\mathrm{d}W_2$

Generic Problem

Stochastic differential equation with general drift and volatility terms:

$$\mathrm{d}S(t) = a(S,t)\,\mathrm{d}t + b(S,t)\,\mathrm{d}W(t)$$

$W(t)$ is a Wiener variable with the properties that for any $q<r<s<t$, $W(t)-W(s)$ is Normally distributed with mean 0 and variance $t-s$, independent of $W(r)-W(q)$.

In many finance applications, we want to compute the expected value of an option dependent on the terminal state $P(S(T))$

Other options depend on the average, minimum or maximum over the whole time interval.

Euler Discretisation

Euler-Maruyama discretisation with timestep h:

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

In the scalar case, each ΔW_n is a Normal random variable with mean 0 and variance h.

May seem very simple-minded but it's hard to improve on the Euler discretisation, and many codes are this simple.

Two different ways of measuring the discretisation error:

$$\text{Strong error:} \qquad \mathbb{E}\left[(S(t_n) - \widehat{S}_n)^2\right] = O(h)$$

$$\text{Weak error:} \qquad \mathbb{E}\left[P(S(t_n)) - P(\widehat{S}_n)\right] = O(h)$$

The "Greeks"

As well as estimating the value $V = \mathbb{E}[P]$, also very important to estimate various first and second derivatives:

$$\Delta = \frac{\partial V}{\partial S_0}, \qquad \Gamma = \frac{\partial^2 V}{\partial S_0^2}, \qquad \text{Vega} = \frac{\partial V}{\partial \sigma}$$

These are needed for "hedging" – banks try to hold a portfolio of different financial products so that the effects of the random terms all cancel.

(Contrary to the public perception, banks generally do not intentionally "gamble" with their assets.)

In some cases, can need 100 or more first order derivatives
$\Longrightarrow$ use of adjoints is natural

Monte Carlo simulation

If we want the expected value of a payoff function P which depends on an underlying quantity S, so that

$$V \;=\; \mathbb{E}[P(S)] \;=\; \int P(S)\; p_S(\theta; S)\; \mathrm{d}S$$

where p_S is the probability distribution for S which depends on an input parameter θ, then the Monte Carlo estimate is

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

where the $S^{(n)}$ are generated independently, so $\mathbb{V}[Y] = N^{-1}\,\mathbb{V}[P(S)]$ giving an $O(N^{-1/2})$ sampling error.

To achieve an RMS error of ε requires $O(\varepsilon^{-2})$ samples.

LRM sensitivity analysis

If p_S is a differentiable function of θ, then

$$\frac{\partial V}{\partial \theta} = \int P(S) \frac{\partial p_S}{\partial \theta} \, \mathrm{d}S = \int P(S) \frac{\partial \log p_S}{\partial \theta} p_S(S) \, \mathrm{d}S$$
$$= \mathbb{E}\left[P(S) \frac{\partial \log p_S}{\partial \theta} \right]$$

which can be estimated as

$$N^{-1} \sum_{n=1}^{N} P(S^{(n)}) \frac{\partial \log p_S^{(n)}}{\partial \theta}$$

This is the Likelihood Ratio Method for computing sensitivities:

- can handle discontinuous payoffs $P(S)$
- requires a known distribution p_S (e.g. log-normal)
- usually has a much larger variance than alternatives

Pathwise sensitivity analysis

Alternatively, if S depends on θ and an independent random variable Z, then

$$V \;=\; \mathbb{E}[P(\,S(\theta;Z)\,)] \;=\; \int P(\,S(\theta;Z)\,)\;p_Z(Z)\;\mathrm{d}Z$$

and

$$\frac{\partial V}{\partial \theta} \;=\; \int \frac{\partial P}{\partial S}\;\frac{\partial S}{\partial \theta}\;p_Z(Z)\;\mathrm{d}Z \;=\; \mathbb{E}\left[\frac{\partial P}{\partial S}\;\frac{\partial S}{\partial \theta}\right],$$

which gives the pathwise sensitivity estimate

$$N^{-1}\sum_{n=1}^{N}\frac{\partial P}{\partial S}\;\dot{S}^{(n)}$$

where $\dot{S}$ is the path sensitivity keeping fixed all of the random numbers.

This is the "natural" sensitivity analysis which comes from differentiating the original Monte Carlo estimate – this derivation shows the need for $P(S)$ to be at least continuous.

"Smoking Adjoints"

My first finance paper in 2006 was with Paul Glasserman from Columbia Business School:

- "Smoking Adjoints: fast Monte Carlo Greeks" in Risk, the leading monthly publication for the finance industry
- explains how to do a discrete adjoint implementation of pathwise sensitivity analysis for an application needing 100's of Greeks, at a cost less than double the original MC cost
- Yves Achdou and Olivier Pironneau had previously used adjoints for finance PDEs, but the technique hadn't been transferred over to the Monte Carlo side
- absolutely nothing novel from an academic point of view, but has had an impact in the industry – I think many banks now use it

Unsteady adjoints

Suppose that we have

$$S_{n+1} = f_n(S_n, \theta), \qquad n = 0, 1, \ldots, N-1$$

given starting value S_0 and parameter θ, and we want to compute $P(S_N)$.

Differentiation gives this recurrence for $\dot{S}_n \equiv \partial S_n / \partial \theta$,

$$\dot{S}_{n+1} \;=\; \frac{\partial f_n}{\partial S_n}\,\dot{S}_n + \frac{\partial f_n}{\partial \theta} \;\equiv\; A_n\,\dot{S}_n + b_n, \qquad\qquad \dot{P} = \frac{\partial P}{\partial S_N}\,\dot{S}_N,$$

and therefore, after some rearrangement,

$$\dot{P} = \frac{\partial P}{\partial S_N} \sum_{n=0}^{N-1} \left(A_{N-1} A_{N-2} \ldots A_{n+2} A_{n+1}\right) b_n.$$

Unsteady adjoints

Equivalently, we have

$$\dot{P} \;=\; \frac{\partial P}{\partial S_N} \sum_{n=0}^{N-1} \left(A_{N-1}A_{N-2}\ldots A_{n+2}A_{n+1}\right) b_n \;=\; \sum_{n=0}^{N-1} v_{n+1}^T\, b_n$$

where v_n are the adjoint variables defined by

$$v_N = \left(\frac{\partial P}{\partial S_N}\right)^T, \qquad v_n = A_n^T v_{n+1}.$$

Key points:

- separate $\dot{S}_n$ calculation required for each input parameter θ
- v_n doesn't depend on θ, so just one adjoint computation required
- nothing here depends on whether the application is CFD or finance

Unsteady adjoints

- need to store all of the S_n – this could be a problem in CFD (use "checkpointing") but not in finance
- automatic differentiation tools can automate the generation of the code (we used Tapenade for the Rolls-Royce HYDRA code) but finance applications are simple enough to do by hand

  (One of the early uses of adjoint AD tool was in EAPS in the early 1990's for an oceanographic application)
- automatic differentiation theory assures you that the cost for an unlimited number of first order sensitivities is no more than a factor 4 greater than the original simulation cost

"Vibrato" Monte Carlo

Pathwise sensitivity analysis fails if P is discontinuous.

"Vibrato" Monte Carlo combines LRM for the final simulation timestep with pathwise analysis for the rest of the path calculation.

Combines the strengths of the two methods:

- can handle discontinuous payoffs
- variance is always better than LRM, and similar to pathwise when the payoff is continuous
- has an efficient adjoint implementation when there are lots of sensitivities to be computed

"Binning"

This was introduced by a collaborator, Luca Capriotti at Credit Suisse.

Monte Carlo analysis computes an estimate and a confidence interval.

Differentiating everything gives the sensitivity of both the estimate and the confidence interval, but we want the confidence interval for the sensitivity.

In simple cases, compute the sensitivity analysis for each path, then calculate the confidence interval.

In more complex cases with expensive pre-computations (e.g. Cholesky factorisation of correlation matrix) need to group paths, compute the sensitivity for each group, then combine into a confidence interval.

More on adjoints for finance

- works naturally for finite difference applications
- Black-Scholes PDE goes backwards in time from known payoff to present value
- adjoint goes forward in time – potentially a bit confusing
- linked to natural duality between forward and backward Kolmogorov equations

Anyone interested to learn more, please contact me for lecture notes, and see

`http://people.maths.ox.ac.uk/gilesm/libor/`

Multilevel Monte Carlo

Coming from CFD, the use of adjoints was very natural.

What else is there? Multigrid!

But there's no iterative solver here – instead just keep the central ideas of

- a nested sequence of grids
- fine grid accuracy at coarse grid cost

Multilevel Monte Carlo

Consider multiple levels of simulations with different timesteps $h_\ell = 2^{-\ell}\, T,\ \ell = 0, 1, \ldots, L$, and payoff $\widehat{P}_\ell$

The expected value on the finest level, which is what we want, can be expressed as a telescoping sum:

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

The aim is to estimate the quantity on the left by independently estimating each of the expectations on the right, and do so in a way which minimises the overall variance for a fixed computational cost.

Multilevel Monte Carlo

Key idea: approximate $\mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}]$ using N_ℓ simulations with $\widehat{P}_\ell$ and $\widehat{P}_{\ell-1}$ obtained using same Brownian path:

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

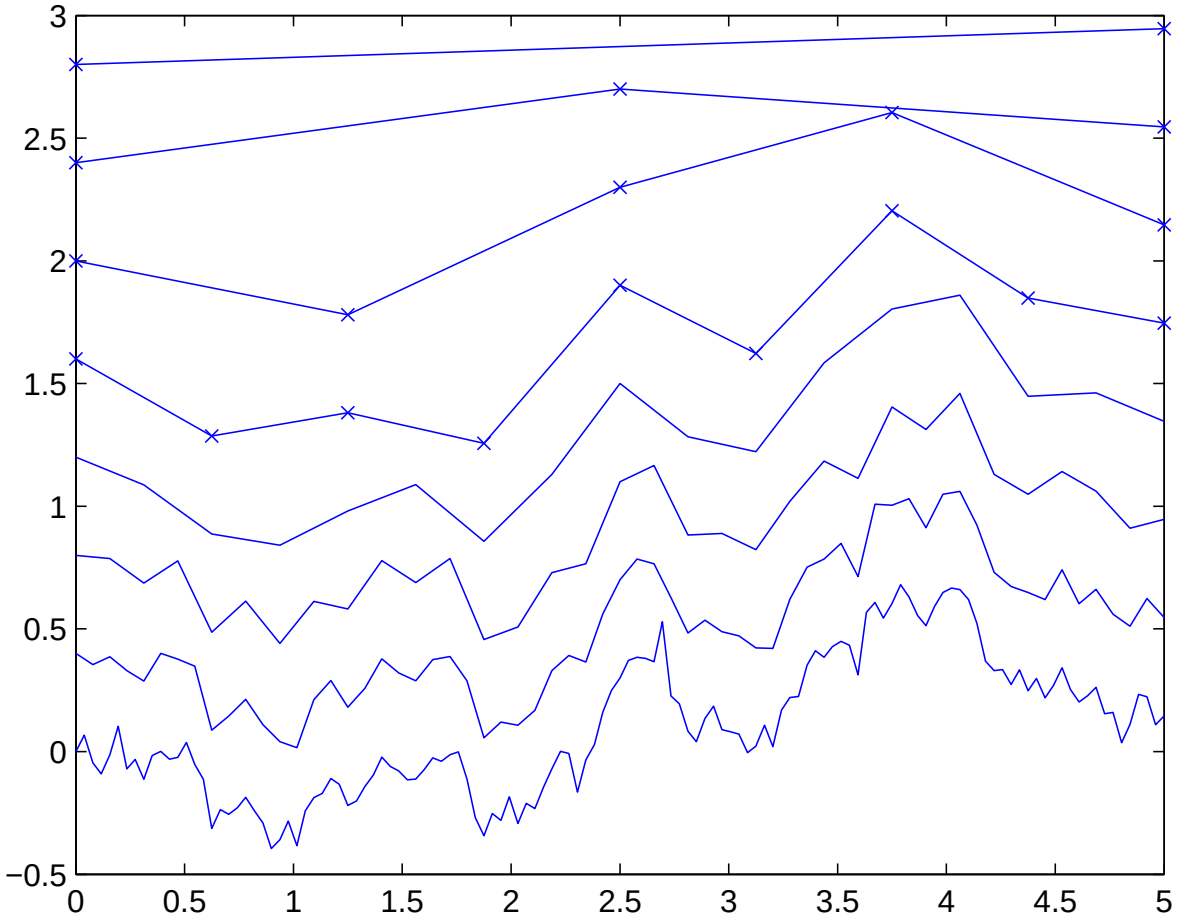
Why is this helpful?

- $\widehat{P}_\ell \approx \widehat{P}_{\ell-1}$ since both approximate same P
- $V_\ell \equiv \mathbb{V}[\widehat{P}_\ell - \widehat{P}_{\ell-1}]$ is small, especially on finer levels
- fewer samples needed to estimate $\mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}]$
- end up using many, cheap samples on coarse levels, and a few, expensive samples on fine levels

Easy implementation: generate Brownian increments ΔW for level ℓ, then sum them pairwise to get increments for level $\ell-1$

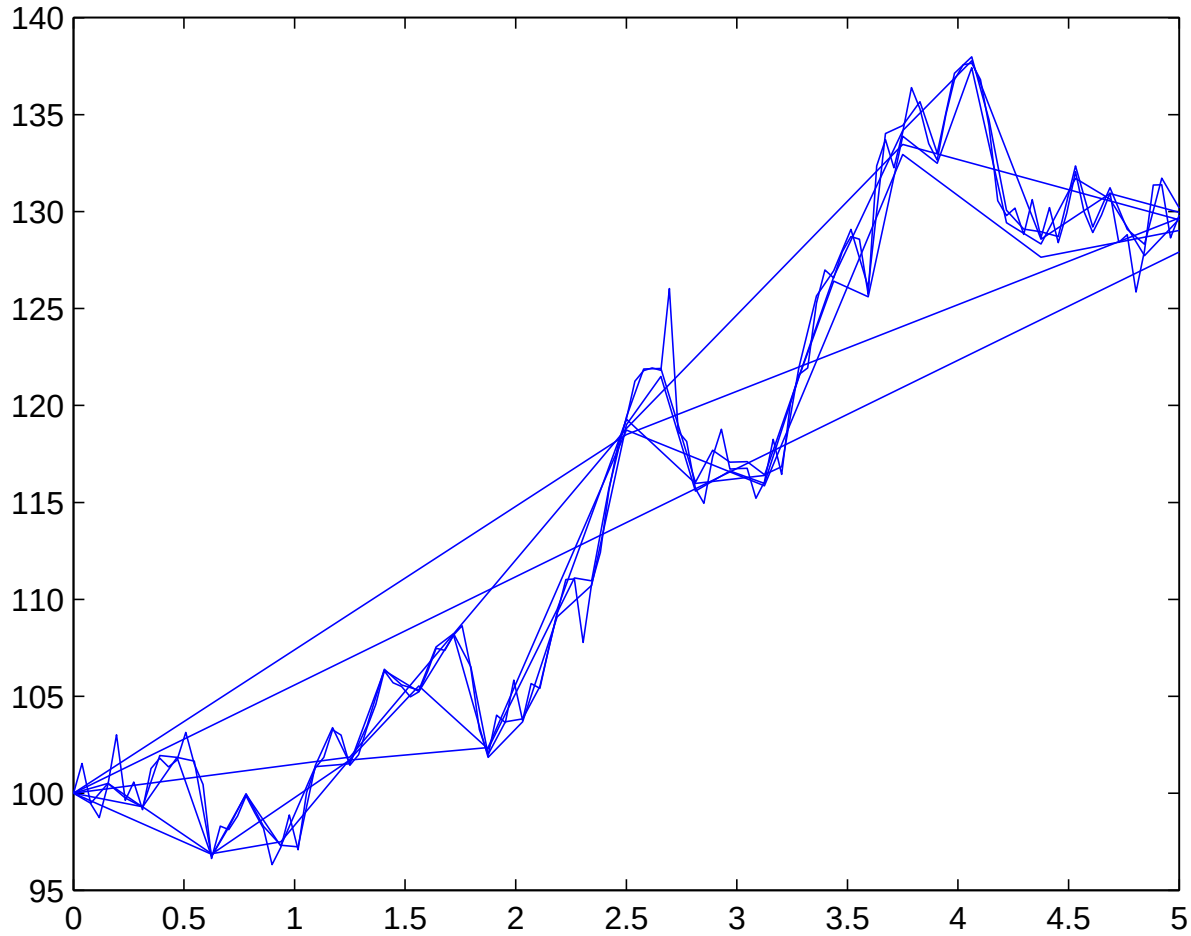
Multilevel Monte Carlo

Discrete Brownian path at different levels (with offset for clarity)



Multilevel Monte Carlo

GBM paths at different levels (without offsets)



Multilevel Monte Carlo

If C_ℓ is cost of a sample on level ℓ, the variance of the combined estimator is

$$\sum_{\ell=0}^{L} N_\ell^{-1} V_\ell$$

and its computational cost is

$$\sum_{\ell=0}^{L} N_\ell \; C_\ell$$

so the variance is minimised for fixed cost by choosing N_ℓ proportional to $\sqrt{V_\ell / C_\ell}$, and then the cost on level ℓ is proportional to $\sqrt{V_\ell \, C_\ell}$.

In an SDE application with timestep h_ℓ, typically have $C_\ell = O(h_\ell^{-1})$ and $V_\ell = O(h_\ell)$, so the computational effort is spread evenly across all levels.

To achieve an $O(\varepsilon)$ RMS error ends up requiring a cost which is $O(\varepsilon^{-2}(\log \varepsilon)^2)$, instead of usual $O(\varepsilon^{-3})$.

MLMC Theorem

Theorem: If there exist independent estimators $\widehat{Y}_\ell$ based on N_ℓ Monte Carlo samples, each costing C_ℓ, and positive constants α, β, γ, 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) $\mathbb{E}[\widehat{Y}_\ell] = \begin{cases} \mathbb{E}[\widehat{P}_0], & l = 0 \\ \mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}], & l > 0 \end{cases}$

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

iv) $C_\ell \leq c_3\, 2^{\gamma\,\ell}$

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 $\mathbb{E}\left[\left(\widehat{Y} - \mathbb{E}[P]\right)^2\right] < \varepsilon^2$

and a computational cost C with bound

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

MLMC Theorem

This is a powerful, general theorem:

- applies to a wide range of stochastic processes (SDEs, SPDEs, Lévy processes, Poisson processes, etc.)
- applies also to a wide choice of numerical approximations, leaving lots of flexibility for creating multilevel estimators with a variance V_ℓ which converges rapidly to zero

. . . and also a total "cheat":

- hard bit is constructing numerical approximations with good properties
- even harder bit is proving it – the numerical analysis can be really tough

Multilevel Monte Carlo

Finance applications:

- SDEs with Euler and Milstein discretisations
- jump-diffusion and Lévy processes, either by directly simulating increments, or by simulating all but the smallest jumps
- support for a variety of path-dependent options
  - digital – discontinuous function of final value
  - Asian – based on an average over time interval
  - lookback – based on minimum / maximum over interval
  - barrier – knocked out if path crosses a certain level
  - American options – much tougher because of optional early exercise
- numerical analysis now exists for most of this
- also some work on Multilevel quasi-Monte Carlo (using Sobol points or rank-1 lattices instead of pseudo-random numbers)
- 10-15 groups worldwide working on different aspects

Back to CFD

I'm now working with Rob Scheichl (Bath) and Andrew Cliffe (Nottingham) applying these ideas to the modelling of oil reservoirs and groundwater contamination in nuclear waste repositories.

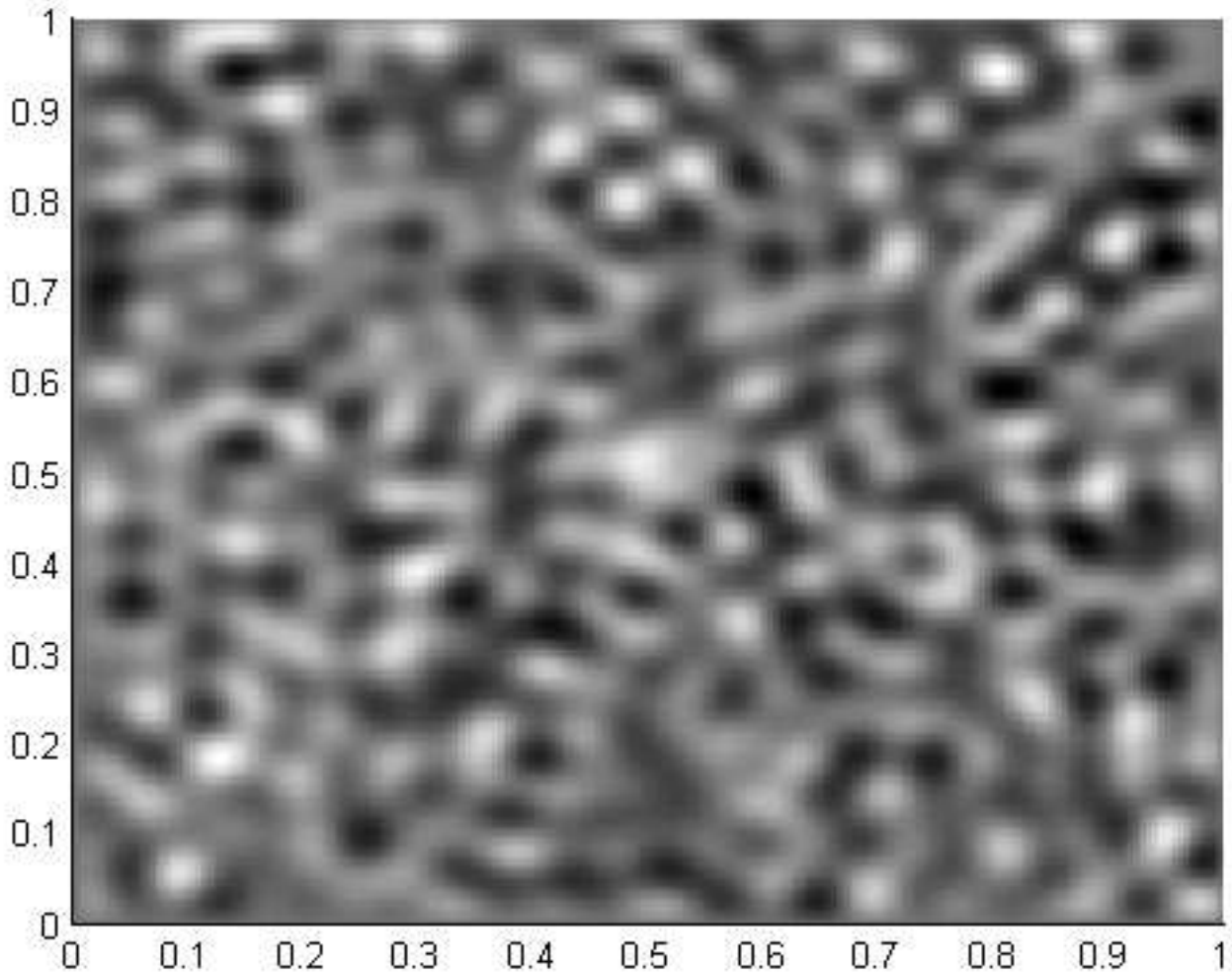
Here we have an elliptic SPDE coming from Darcy's law:

$$\nabla \cdot \Big(\kappa(x) \nabla p \Big) = 0$$

where the permeability $\kappa(x)$ is uncertain, and $\log \kappa(x)$ is often modelled as being Normally distributed with a spatial covariance such as

$$\mathrm{cov}(\log \kappa(x_1), \log \kappa(x_2)) = \sigma^2 \exp(-\|x_1 - x_2\|/\lambda)$$

Elliptic SPDE



A typical realisation of κ for $\lambda = 0.001,\ \sigma = 1$.

Elliptic SPDE

Samples of $\log k$ are provided by a Karhunen-Loève expansion:

$$\log k(\mathbf{x},\omega) = \sum_{n=0}^{\infty} \sqrt{\theta_n}\ \xi_n(\omega)\ f_n(\mathbf{x}),$$

where θ_n, f_n are eigenvalues / eigenfunctions of the correlation function:

$$\int R(\mathbf{x},\mathbf{y})\ f_n(\mathbf{y})\ \mathrm{d}\mathbf{y} = \theta_n\, f_n(\mathbf{x})$$

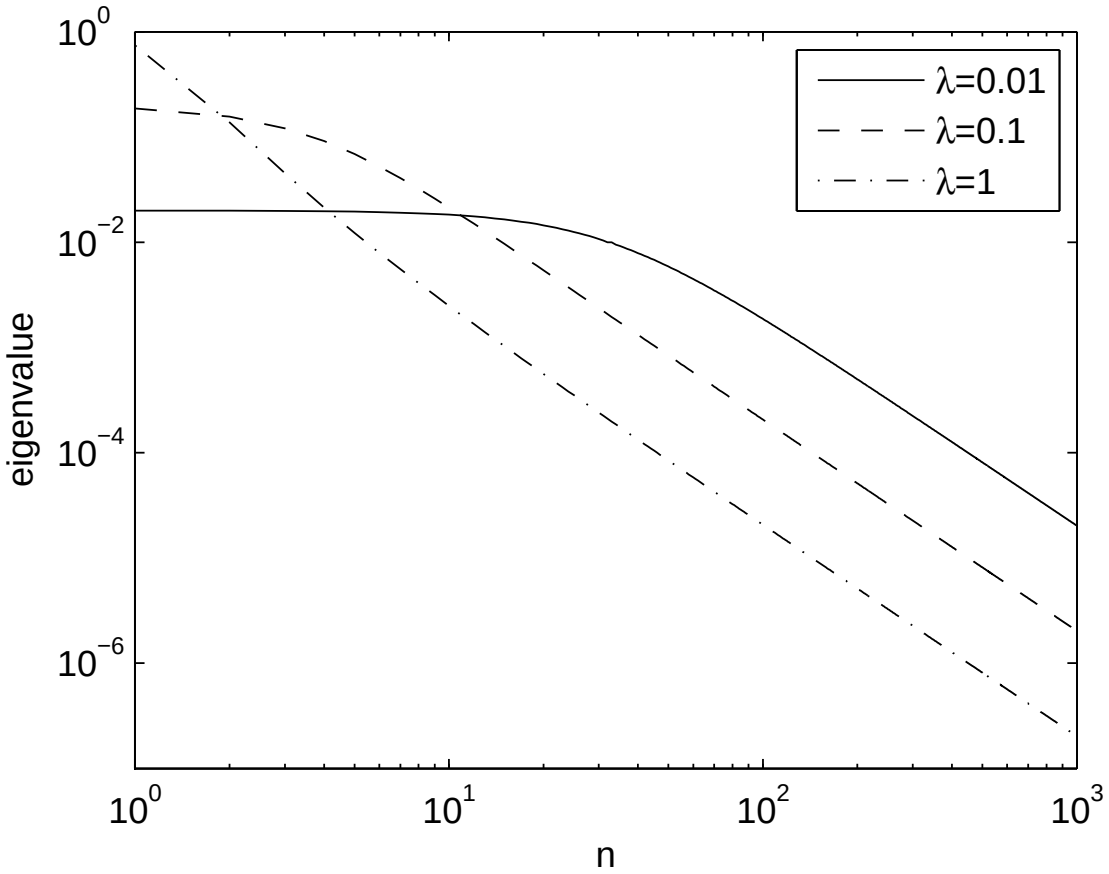
and $\xi_n(\omega)$ are standard Normal random variables.

Numerical experiments truncate the expansion.

(Latest 2D/3D work uses an efficient FFT construction based on a circulant embedding.)

Elliptic SPDE

Decay of 1D eigenvalues



When $\lambda = 1$, can use a low-dimensional polynomial chaos approach, but it's impractical for smaller λ.

Elliptic SPDE

Discretisation:

- cell-centred finite volume discretisation on a uniform grid – for rough coefficients we need to make grid spacing very small on finest grid
- each level of refinement has twice as many grid points in each direction
- current numerical experiments use a direct solver for simplicity, but in 3D will use an efficient AMG multigrid solver with a cost roughly proportional to the total number of grid points

2D Results

Boundary conditions for unit square $[0,1]^2$:
– fixed pressure: $p(0,x_2)=1,\ \ p(1,x_2)=0$
– Neumann b.c.: $\partial p/\partial x_2(x_1,0)=\partial p/\partial x_2(x_1,1)=0$

Output quantity – mass flux: $-\int k\,\dfrac{\partial p}{\partial x_1}\,\mathrm{d}x_2$

Correlation length: $\lambda = 0.2$

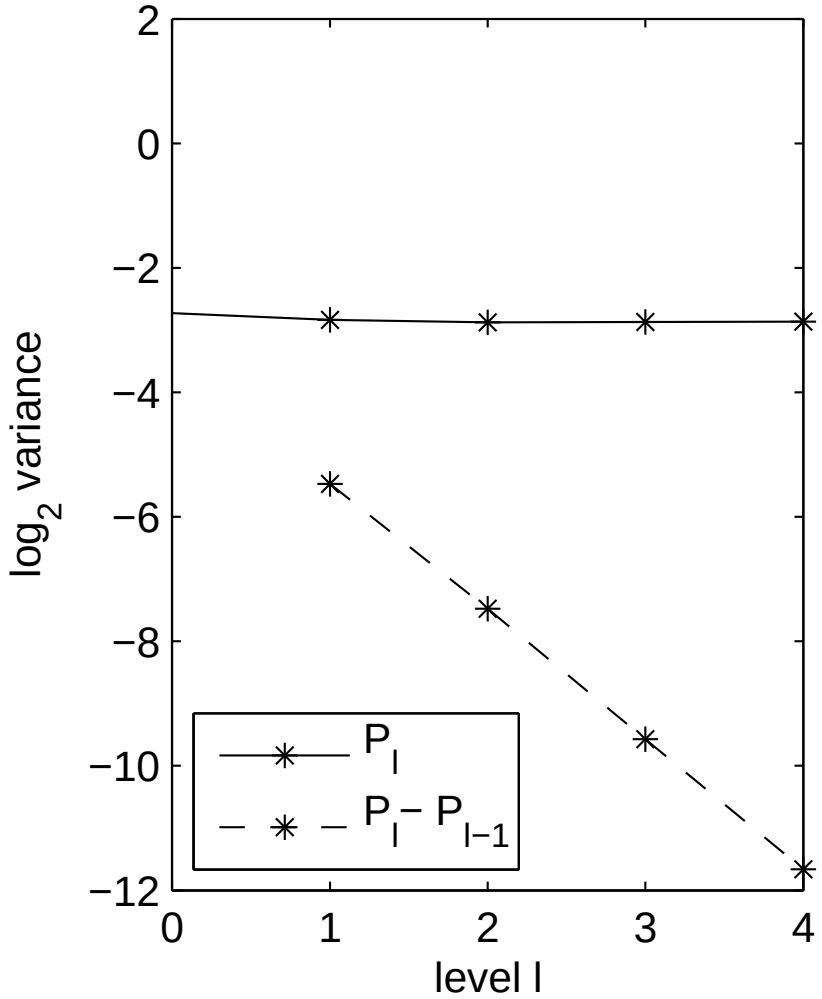
Coarsest grid: $h = 1/8$ (comparable to λ)

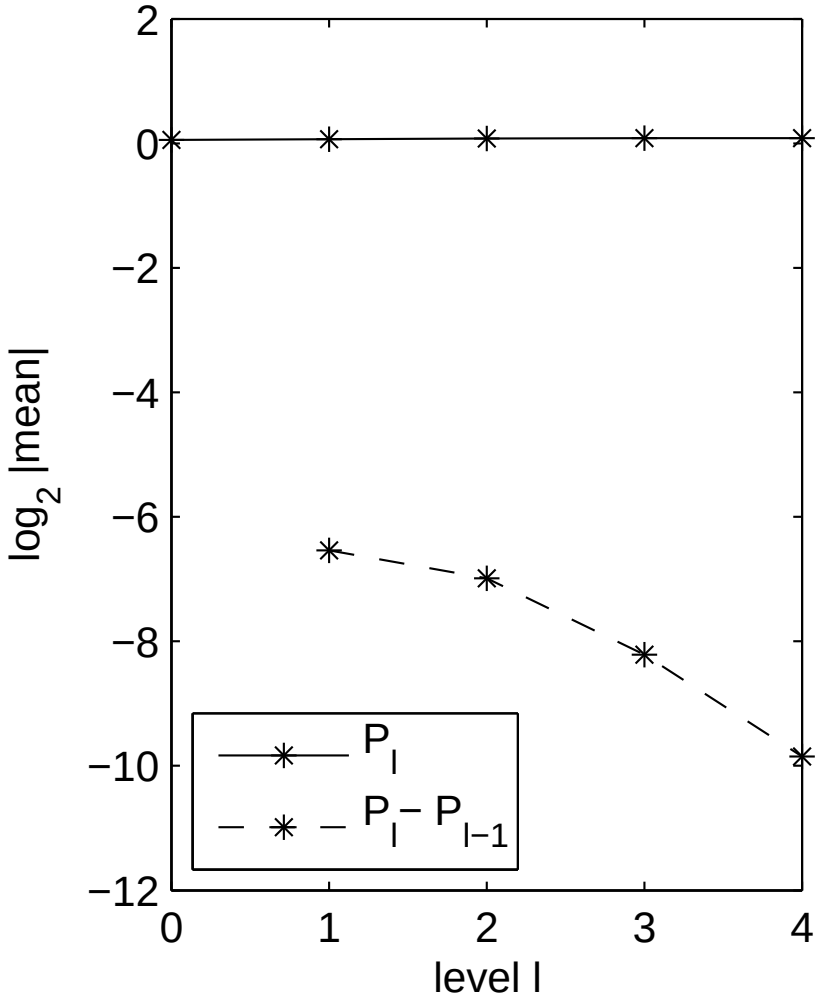
Finest grid: $h = 1/128$

Karhunen-Loève truncation: $m_{KL} = 4000$

Cost taken to be proportional to number of nodes

2D Results

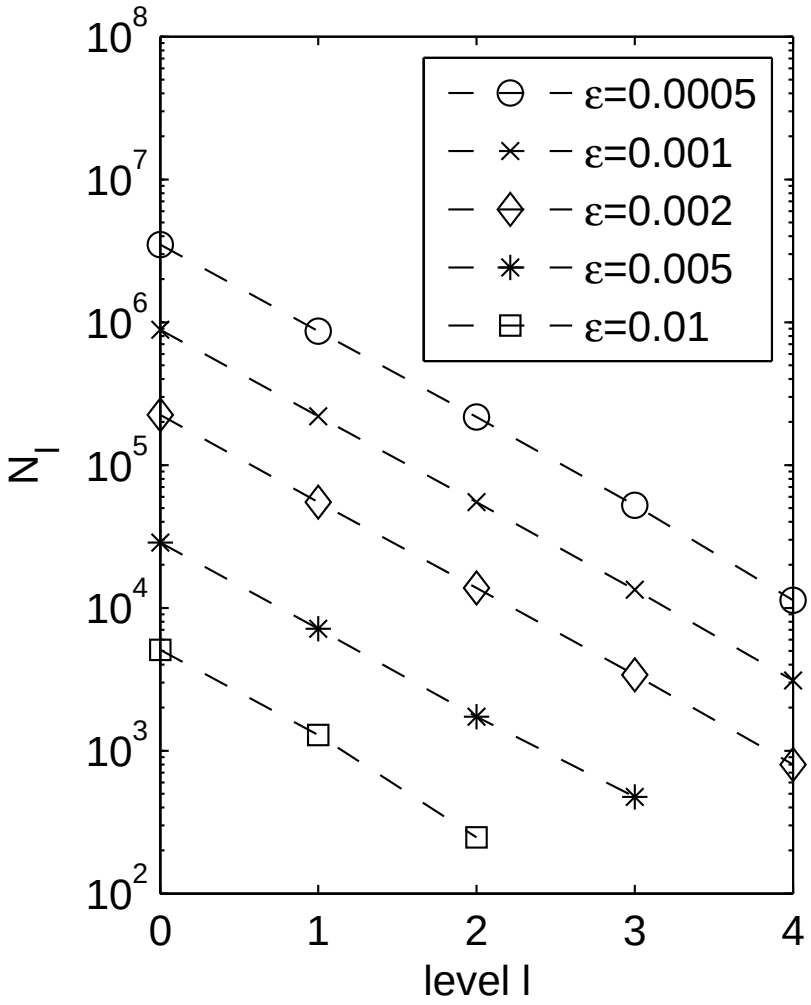


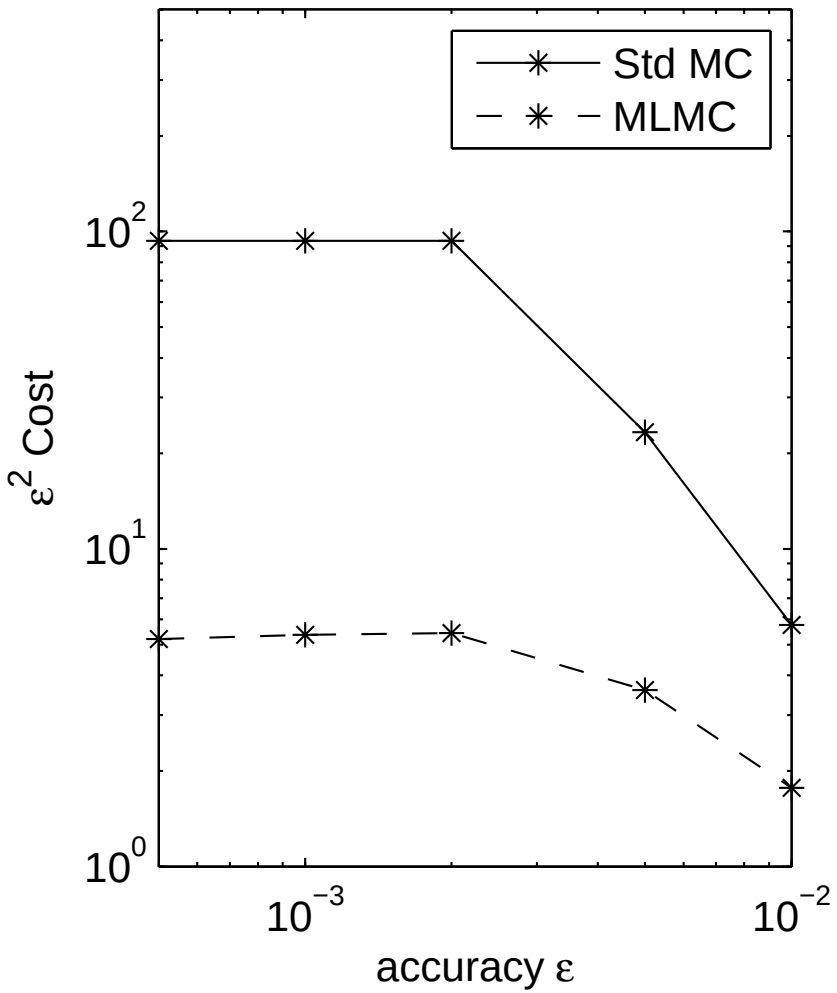


$$\mathbb{V}[\widehat{P}_\ell - \widehat{P}_{\ell-1}] \sim h_\ell^2$$

$$\mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}] \sim h_\ell^2$$

2D Results


10^8
10^7
10^6
10^5
10^4
10^3
10^2
N_l
0
1
2
3
4
level l
ε=0.0005
ε=0.001
ε=0.002
ε=0.005
ε=0.01


10^2
10^1
10^0
ε^2 Cost
10^{-3}
10^{-2}
accuracy ε
Std MC
MLMC

Complexity analysis

Relating things back to the MLMC theorem:

$$\begin{aligned} \mathbb{E}[\widehat{P}_\ell - P] &\sim 2^{-2\ell} \implies \alpha = 2 \\ V_\ell &\sim 2^{-2\ell} \implies \beta = 2 \\ C_\ell &\sim 2^{d\ell} \implies \gamma = d \quad \text{(dimension of PDE)} \end{aligned}$$

To achieve r.m.s. accuracy ε requires finest level grid spacing $h \sim \varepsilon^{1/2}$ and hence we get the following complexity:

dim	MC	MLMC
1	$\varepsilon^{-2.5}$	ε^{-2}
2	ε^{-3}	$\varepsilon^{-2}(\log \varepsilon)^2$
3	$\varepsilon^{-3.5}$	$\varepsilon^{-2.5}$

Other SPDE applications

For more on multilevel for SPDEs, see the work of Christoph Schwab and his group (ETH Zurich):

`http://www.math.ethz.ch/~schwab/`

- elliptic, parabolic and hyperbolic PDEs
- stochastic coefficients, initial data, boundary data

Schwab used to work on alternative techniques such as "polynomial chaos" so I think the fact that he has now switched to multilevel is significant.

For other papers on multilevel, see my MLMC community homepage:

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

Bio-chemical reactions

Another interesting application area is bio-chemical reactions.

Chemical reactions are usually modelled by ODEs:

$$\dot{c}_s = \sum_r \lambda_r(c)\, \nu_{r,s}$$

where λ_r is the rate for reaction r, and $\nu_{r,s}$ is its effect on species s.

However, when concentrations are extremely small, the modelling is stochastic, at the level of individual molecular reactions:

$$\mathbb{P}\Big(\text{reaction } r \text{ occurs in time interval } [t, t+\mathrm{d}t]\Big) = \lambda_r(c)\ \mathrm{d}t$$

Bio-chemical reactions

This leads to a hierarchy of models:

- SSA (Stochastic Simulation Algorithm)
  - models each individual reaction
- τ-leaping method
  - time-stepping with Poisson model for reactions in each time interval
- Langevin equations
  - an SDE with Brownian noise
  - limit as Poisson distribution approaches Normal at high rates
- "standard" ODEs
  - limit as std. dev. of Normal distribution becoming negligible relative to mean

Anderson (Wisconsin) & Higham (Strathclyde) have used multilevel for SSA and τ-leaping, and obtained large computational savings.

Conclusions

- Moving from CFD to Monte Carlo simulation, I have been fortunate in being able to adapt ideas from CFD to new challenges:
  - adjoints for sensitivity calculations – very natural
  - multilevel Monte Carlo – not as natural because it is different from multigrid, but a similar philosophy
- The multilevel Monte Carlo development is now feeding back into CFD for uncertainty quantification in applications such as nuclear waste repositories and oil servoir simulation
- I think stochastic modelling and simulation is an important growth area in applied mathematics, engineering and science