From CFD to computational finance (and back again?)

Mike Giles

mike.giles@maths.ox.ac.uk

Oxford University Mathematical Institute

Oxford-Man Institute of Quantitative Finance

21st Century Challenges in
Computational Engineering and Science

Princeton University, Nov 19-21, 2009

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

Computational Finance

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) are demanding more simulation
- computational finance accounts for 10% of Top500 supercomputers
- still plenty of MSc students willing/able to fund themselves
- only problem is lack of research funding

Computational Finance

Computational finance is where CFD was 20-25 years ago

- not many academics working on numerical methods
- codes are small – my biggest is probably 1000 lines
- still lots of low-hanging fruit
- in banks, each product group often has its own codes; consolidation into a single corporate Monte Carlo system for both London and New York is underway

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
- ...

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

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

$$P \equiv f(S(T))$$

Standard MC Approach

Euler 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.

Simplest estimator for expected payoff $\mathbb{E}[P]$ is an average from N independent path simulations:

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

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

The "Greeks"

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

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

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

“Smoking Adjoints” (2006)

First significant paper was with Paul Glasserman from Columbia Business School:

- “Smoking Adjoints: fast Monte Carlo Greeks” in Risk, a monthly publication for the finance industry
- explains how to use discrete adjoints for an important application which requires lots of Greeks
- 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 a number of banks now use it

“Smoking Adjoints”

The adjoint implementation is based on pathwise sensitivity analysis which relies on the identity

$$\frac{\partial}{\partial \theta}\,\mathbb{E}[P] = \mathbb{E}\left[\frac{\partial P}{\partial \theta}\right]$$

but this breaks down if P is discontinuous.

There are some other ways of treating this case, but they don’t have efficient adjoint implementations.

I’ve recently developed a new way of handling the discontinuity which does retain an efficient adjoint implementation.

. . . and for my next trick

Coming from CFD, the use of adjoints was quite natural

What else is there?   Multigrid!

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

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

Multilevel Monte Carlo

Consider multiple sets of simulations with different timesteps $h_l = 2^{-l}\,T,\ l = 0,1,\ldots,L$, and payoff $\widehat{P}_l$

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

Expected value is same – aim is to reduce variance of estimator for a fixed computational cost.

Key point: approximate $\mathbb{E}[\widehat{P}_l - \widehat{P}_{l-1}]$ using N_l simulations with $\widehat{P}_l$ and $\widehat{P}_{l-1}$ obtained using <u>same</u> Brownian path.

$$\widehat{Y}_l = N_l^{-1} \sum_{i=1}^{N_l} \left(\widehat{P}_l^{(i)} - \widehat{P}_{l-1}^{(i)} \right)$$

Multilevel Monte Carlo

This has led to a number of papers, covering both applications and numerical analysis. Main point is a big reduction in computational cost for many problems.

To achieve a root-mean-square accuracy of ε:

- cost of standard approach is $O(\varepsilon^{-3})$
- cost of multilevel approach is $O(\varepsilon^{-2})$
- cost is further reduced using quasi-random numbers

Back to CFD?

One new project uses multilevel Monte Carlo for oil reservoir and nuclear waste repository simulation,

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

where $\log\kappa$ is Normally distributed with a given spatial correlation.

Some people use “polynomial chaos” or Karhunen-Loeve expansions, but we think multilevel Monte Carlo may be better when there is minimal spatial correlation.

Back to CFD?

Another project concerns the use of GPUs for HPC

- current NVIDIA GPUs have up to 240 cores; next generation has 512
- 1 GPU is roughly $10\times$ faster than 2 CPUs, with similar cost and power consumption
- programmed in C with some extensions
- ideal for trivially-parallel Monte Carlo simulations
- also very effective for finite difference applications
- new project addresses the needs of unstructured grid applications through a general-purpose open-source library and program transformation tools

Further information

Web: `www.maths.ox.ac.uk/~gilesm/`

Email: `mike.giles@maths.ox.ac.uk`

. . . or talk to me here about GPU project