Advanced Monte Carlo Methods: Computing Greeks

Prof. Mike Giles

mike.giles@maths.ox.ac.uk

Oxford University Mathematical Institute

Outline

Computing Greeks

- finite differences
- likelihood ratio method
- pathwise sensitivities
- “Smoking adjoints” implementation

SDE path simulation

For the generic stochastic differential equation

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

an Euler approximation with timestep h is

$$\widehat{S}_{n+1} = \widehat{S}_n + a(\widehat{S}_n)\,h + b(\widehat{S}_n)\,Z_n\,\sqrt{h},$$

where Z is a $N(0,1)$ random variable. To estimate the value of a European option

$$V = \mathbb{E}[f(S(T))],$$

we take the average of N paths with M timesteps:

$$\widehat{V} = N^{-1} \sum_i f(\widehat{S}_M^{(i)}).$$

Greeks

As in Module 2, in addition to estimating the expected value

$$V = \mathbb{E}[f(S(T)],$$

we also want to know a whole range of “Greeks” corresponding to first and second derivatives of V with respect to various parameters:

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

$$\rho = \frac{\partial V}{\partial r}, \qquad \text{Vega} = \frac{\partial V}{\partial \sigma}.$$

These are needed for hedging and for risk analysis.

Finite difference sensitivities

If $V(\theta) = \mathbb{E}[f(S(T))]$ for a particular value of an input parameter θ, then the sensitivity $\dfrac{\partial V}{\partial \theta}$ can be approximated by one-sided finite difference

$$\frac{\partial V}{\partial \theta} = \frac{V(\theta+\Delta\theta) - V(\theta)}{\Delta\theta} \ + \ O(\Delta\theta)$$

or by central finite difference

$$\frac{\partial V}{\partial \theta} = \frac{V(\theta+\Delta\theta) - V(\theta-\Delta\theta)}{2\Delta\theta} \ + \ O((\Delta\theta)^2)$$

Nothing changes here from Module 2 because of the path simulation.

Finite difference sensitivities

As before, the clear advantage of this approach is that it is very simple to implement (hence the most popular in practice?)

However, the disadvantages are:

- expensive (2 extra sets of calculations for central differences)
- significant bias error if $\Delta\theta$ too large
- large variance if $f(S(T))$ discontinuous and $\Delta\theta$ small

Also, very important to use the same random numbers for the "bumped" path simulations to minimise the variance.

Likelihood ratio method

As a recap from Module 2, if we define $p(S)$ to the probability density function for the final state $S(T)$, then

$$V = \mathbb{E}[f(S(T))] = \int f(S)\, p(S)\, \mathrm{d}S,$$

$$\Longrightarrow \quad \frac{\partial V}{\partial \theta} = \int f\, \frac{\partial p}{\partial \theta}\, \mathrm{d}S = \int f\, \frac{\partial (\log p)}{\partial \theta}\, p\, \mathrm{d}S = \mathbb{E}\left[f\, \frac{\partial (\log p)}{\partial \theta} \right]$$

The quantity $\dfrac{\partial (\log p)}{\partial \theta}$ is sometimes called the “score function”.

Likelihood ratio method

Extending LRM to a SDE path simulation with M timesteps, with the payoff a function purely of the discrete states $\widehat{S}_n$, we have the M-dimensional integral

$$V = \mathbb{E}[f(\widehat{S})] = \int f(\widehat{S})\, p(\widehat{S})\, \mathrm{d}\widehat{S},$$

where $$\mathrm{d}\widehat{S} \equiv \mathrm{d}\widehat{S}_1\ \mathrm{d}\widehat{S}_2\ \mathrm{d}\widehat{S}_3\ \ldots\ \mathrm{d}\widehat{S}_M$$

and $p(\widehat{S})$ is the product of the p.d.f.s for each timestep

$$p(\widehat{S}) = \prod_n p_n(\widehat{S}_{n+1}|\widehat{S}_n)$$

$$\log p(\widehat{S}) = \sum_n \log p_n(\widehat{S}_{n+1}|\widehat{S}_n)$$

Likelihood ratio method

For the Euler approximation of GBM,

$$\log p_n = -\log \widehat{S}_n - \log \sigma - \tfrac{1}{2}\log(2\pi h) - \tfrac{1}{2}\,\frac{\left(\widehat{S}_{n+1} - \widehat{S}_n(1{+}r\,h)\right)^2}{\sigma^2\,\widehat{S}_n^2\,h}$$

$$\Longrightarrow \quad \frac{\partial(\log p_n)}{\partial\sigma} \;=\; -\frac{1}{\sigma} + \frac{\left(\widehat{S}_{n+1} - \widehat{S}_n(1{+}r\,h)\right)^2}{\sigma^3\,\widehat{S}_n^2\,h}$$

$$= \; \frac{Z_n^2 - 1}{\sigma}$$

where Z_n is the unit Normal defined by

$$\widehat{S}_{n+1} - \widehat{S}_n(1{+}r\,h) = \sigma\,\widehat{S}_n\,\sqrt{h}\,Z_n$$

Likelihood ratio method

Hence, the approximation of Vega is

$$\frac{\partial}{\partial\sigma}\,\mathbb{E}[f(\widehat{S}_M)] = \mathbb{E}\left[\left(\sum_n \frac{Z_n^2\!-\!1}{\sigma}\right) f(\widehat{S}_M)\right]$$

Note that again this gives zero for $f(S)\equiv 1$.

Note also that $\mathbb{V}[Z_n^2-1]=2$ and therefore

$$\mathbb{V}\left[\left(\sum_n \frac{Z_n^2\!-\!1}{\sigma}\right) f(\widehat{S}_M)\right] \;=\; O(M) \;=\; O(T/h)$$

This $O(h^{-1})$ blow-up is the great drawback of the LRM.

Pathwise sensitivities

Under certain conditions (e.g. $f(S), a(S,t), b(S,t)$ all continuous and piecewise differentiable)

$$\frac{\partial}{\partial\theta}\,\mathbb{E}[f(S(T))] = \mathbb{E}\left[\frac{\partial f(S(T))}{\partial\theta}\right] = \mathbb{E}\left[\frac{\partial f}{\partial S}\;\frac{\partial S(T)}{\partial\theta}\right].$$

with $\dfrac{\partial S(T)}{\partial\theta}$ computed by differentiating the path evolution.

Pros:

- less expensive (1 cheap calculation for each sensitivity)
- no bias

Cons:

- can't handle discontinuous payoffs

Pathwise sensitivities

In Module 2, when we could directly sample $S(T)$ this led to the estimator

$$\frac{1}{N}\sum_{i=1}^{N} \frac{\partial f}{\partial S}(S^{(i)})\ \frac{\partial S^{(i)}}{\partial \theta}$$

which is the derivative of the usual price estimator

$$\frac{1}{N}\sum_{i=1}^{N} f(S^{(i)})$$

Gives incorrect estimates when $f(S)$ is discontinuous.

e.g. for digital put $\dfrac{\partial f}{\partial S} = 0$ so estimated value of Greek is zero – clearly wrong.

Pathwise sensitivities

Returning to the generic stochastic differential equation

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

an Euler approximation with timestep h gives

$$\widehat{S}_{n+1} = F_n(\widehat{S}_n) \equiv \widehat{S}_n + a(\widehat{S}_n)\,h + b(\widehat{S}_n)\,Z_n\,\sqrt{h}.$$

Defining $\Delta_n = \dfrac{\partial \widehat{S}_n}{\partial S_0}$, then $\Delta_{n+1} = D_n\,\Delta_n$, where

$$D_n \;\equiv\; \frac{\partial F_n}{\partial \widehat{S}_n} \;=\; I + \frac{\partial a}{\partial S}\,h + \frac{\partial b}{\partial S}\,Z_n\,\sqrt{h}.$$

Pathwise sensitivities

The payoff sensitivity to the initial state (Deltas) is then

$$\frac{\partial f(\widehat{S}_N)}{\partial S_0} = \frac{\partial f(\widehat{S}_N)}{\partial \widehat{S}_N} \, \Delta_N$$

If $S(0)$ is a vector of dimension m, then each timestep

$$\Delta_{n+1} = D_n \, \Delta_n,$$

involves a $m \times m$ matrix multiplication, with $O(m^3)$ CPU cost – costly, but still cheaper than finite differences which are also $O(m^3)$ but with a larger coefficient.

Cost may be less in practice because D_n may have a lot of zero entries.

Pathwise sensitivities

To calculate the sensitivity to other parameters (such as volatility $\Longrightarrow$ vegas) consider a generic parameter θ.

Defining $\Theta_n = \partial \widehat{S}_n / \partial \theta$, then

$$\Theta_{n+1} = \frac{\partial F_n}{\partial \widehat{S}_n} \Theta_n + \frac{\partial F_n}{\partial \theta} \equiv D_n \Theta_n + B_n,$$

and hence

$$\frac{\partial f}{\partial \theta} = \frac{\partial f(\widehat{S}_N)}{\partial \widehat{S}_N} \Theta_N$$

Vega example

Suppose we have a down-and-out barrier option based on a single GBM asset, and we want to compute vega.

Euler approximation with timestep h:

$$\widehat{S}_{n+1} = F_n(\widehat{S}_n) \equiv \widehat{S}_n + r\,\widehat{S}_n\,h + \sigma\,\widehat{S}_n\,Z_n\,\sqrt{h}$$

Differentiating this gives:

$$\frac{\partial \widehat{S}_{n+1}}{\partial \sigma} = \frac{\partial \widehat{S}_n}{\partial \sigma}\left(1 + r\ + \sigma Z_n\,\sqrt{h}\right) + \widehat{S}_n\,Z_n\,\sqrt{h}$$

with initial condition $\dfrac{\partial \widehat{S}_0}{\partial \sigma} = 0$.

Vega example

Using the treatment discussed in Module 4, where $p_n = p_n(\widehat{S}_n, \widehat{S}_{n+1}, \sigma)$ is conditional probability of being across the barrier in n^{th} timestep, the discounted payoff is

$$\exp(-rT)\ (\widehat{S}_N - K)^+\ P_N$$

where

$$P_n = \prod_{m=0}^{n-1} (1 - p_m),$$

is probability of not crossing the barrier in first n timesteps, and $P_0 = 0$.

Vega example

Since

$$P_{n+1} = P_n \, (1 - p_n)$$

then

$$\frac{\partial P_{n+1}}{\partial \sigma} = \frac{\partial P_n}{\partial \sigma}\, (1-p_n) - P_n \left(\frac{\partial p_n}{\partial \widehat{S}_n} \, \frac{\partial \widehat{S}_n}{\partial \sigma} + \frac{\partial p_n}{\partial \widehat{S}_{n+1}} \, \frac{\partial \widehat{S}_{n+1}}{\partial \sigma} + \frac{\partial p_n}{\partial \sigma} \right)$$

with initial condition $\dfrac{\partial P_0}{\partial \sigma} = 0$.

The payoff sensitivity is then

$$\exp(-rT) \left(\mathbf{1}_{\widehat{S}_N > K} \frac{\partial \widehat{S}_N}{\partial \sigma} \, P_N + (\widehat{S}_N - K)^+ \, \frac{\partial P_N}{\partial \sigma} \right)$$

Automatic Differentiation

Generating the pathwise sensitivity code is tedious, but straightforward, and can be automated:

- source-source code generation: takes an old code for payoff evaluation and produces a new code which also computes sensitivities
- operator overloading: defines new object (value + sensitivity), and re-defines operations appropriately e.g.

$$\begin{pmatrix} a \\ \dot{a} \end{pmatrix} * \begin{pmatrix} b \\ \dot{b} \end{pmatrix} \equiv \begin{pmatrix} a\, b \\ \dot{a}\, b + a\, \dot{b} \end{pmatrix}$$

For more information, see
`www.autodiff.org/`
`people.maths.ox.ac.uk/gilesm/libor/`

Discontinuous payoffs

Pathwise sensitivity needs the payoff to be continuous.

What can you do when it is not?

- for digital options, can use a crude piecewise linear approximation
- alternatively, use conditional expectations which effectively smooth the payoff
  - the barrier option is a good example of this, using the probability of crossing conditional on the path values at discrete times
  - Glasserman discusses a similar approach for digital options, stopping the path simulation one timestep early then taking a conditional expectation

Discontinuous payoffs

Glasserman's approach has problems in multiple dimensions (hard to evaluate expected value analytically) so I developed an approach I call “vibrato Monte Carlo”.

It is a hybrid method. Conditional on the path value $\widehat{S}_{N-1}$ one timestep before the end, the value value $\widehat{S}_N$ has a Normal distribution, if using an Euler discretisation.

Hence, can use LRM for the final timetsep to get the sensitivity to changes in $\widehat{S}_{N-1}$, and combine this with pathwise to get sensitivity of $\widehat{S}_{N-1}$ to the input parameters.

M.B. Giles, ’Vibrato Monte Carlo sensitivities’, pp. 369-392 in Monte Carlo and Quasi Monte Carlo Methods 2008, Springer, 2009.

Adjoint approach

The adjoint (or reverse mode AD) approach computes the same values as the standard (forward) pathwise approach, but much more efficiently for the sensitivity of a single output to multiple inputs.

The approach has a long history in applied math and engineering:

- optimal control theory (find control which achieves target and minimizes cost)
- design optimization (find shape which maximizes performance)

Adjoint approach

Returning to the generic stochastic o.d.e.

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

with Euler approximation

$$\widehat{S}_{n+1} = F_n(\widehat{S}_n) \equiv \widehat{S}_n + a(\widehat{S}_n)\,h + b(\widehat{S}_n)\,Z_n\,\sqrt{h}$$

if $\Delta_n = \dfrac{\partial \widehat{S}_n}{\partial S_0}$, then $\Delta_{n+1} = D_n\,\Delta_n, \quad D_n \equiv \dfrac{\partial F_n(\widehat{S}_n)}{\partial \widehat{S}_n},$

and hence

$$\frac{\partial f(\widehat{S}_N)}{\partial S_0} = \frac{\partial f(\widehat{S}_N)}{\partial \widehat{S}_N} \Delta_N = \frac{\partial f}{\partial S}\, D_{N-1}\, D_{N-2} \ldots\; D_0\, \Delta_0$$

Adjoint approach

If S is m-dimensional, then D_n is an $m\times m$ matrix, and the computational cost per timestep is $O(m^3)$.

Alternatively,

$$\frac{\partial f(\widehat{S}_N)}{\partial S_0} = \frac{\partial f}{\partial S}\, D_{N-1}\, D_{N-2} \cdots D_0\, \Delta_0 = V_0^T \Delta_0,$$

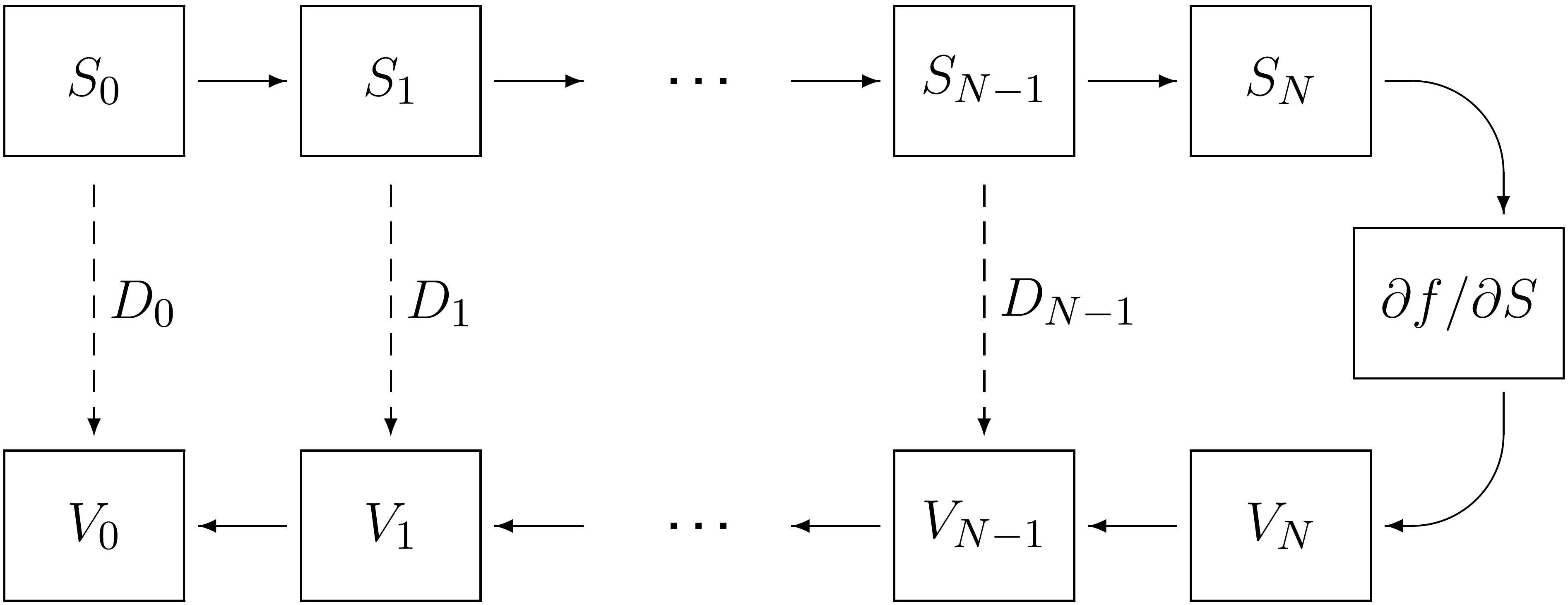
where adjoint $V_n = \left(\dfrac{\partial f(\widehat{S}_N)}{\partial \widehat{S}_n} \right)^T$ is calculated from

$$V_n = D_n^T V_{n+1}, \quad V_N = \left(\frac{\partial f}{\partial \widehat{S}_N} \right)^T,$$

at a computational cost which is $O(m^2)$ per timestep.

Adjoint approach

Note the flow of data within the path calculation:



– memory requirements are not significant because data only needs to be stored for the current path.

Adjoint approach

To calculate the sensitivity to other parameters, consider a generic parameter θ. Defining $\Theta_n = \partial \widehat{S}_n / \partial \theta$, then

$$\Theta_{n+1} = \frac{\partial F_n}{\partial S}\,\Theta_n + \frac{\partial F_n}{\partial \theta} \equiv D_n\,\Theta_n + B_n,$$

and hence

$$\begin{aligned}
\frac{\partial f}{\partial \theta} &= \frac{\partial f}{\partial \widehat{S}_N}\,\Theta_N \\
&= \frac{\partial f}{\partial \widehat{S}_N}\Big\{ B_{N-1} + D_{N-1} B_{N-2} + \ldots \\
&\qquad\qquad + D_{N-1} D_{N-2} \ldots D_1 B_0 \Big\} \\
&= \sum_{n=0}^{N-1} V_{n+1}^T B_n .
\end{aligned}$$

Adjoint approach

Different θ's have different B's, but same V's

$$\Longrightarrow \text{ Computational cost} \simeq m^2 \; + \; m \times \text{\# parameters},$$

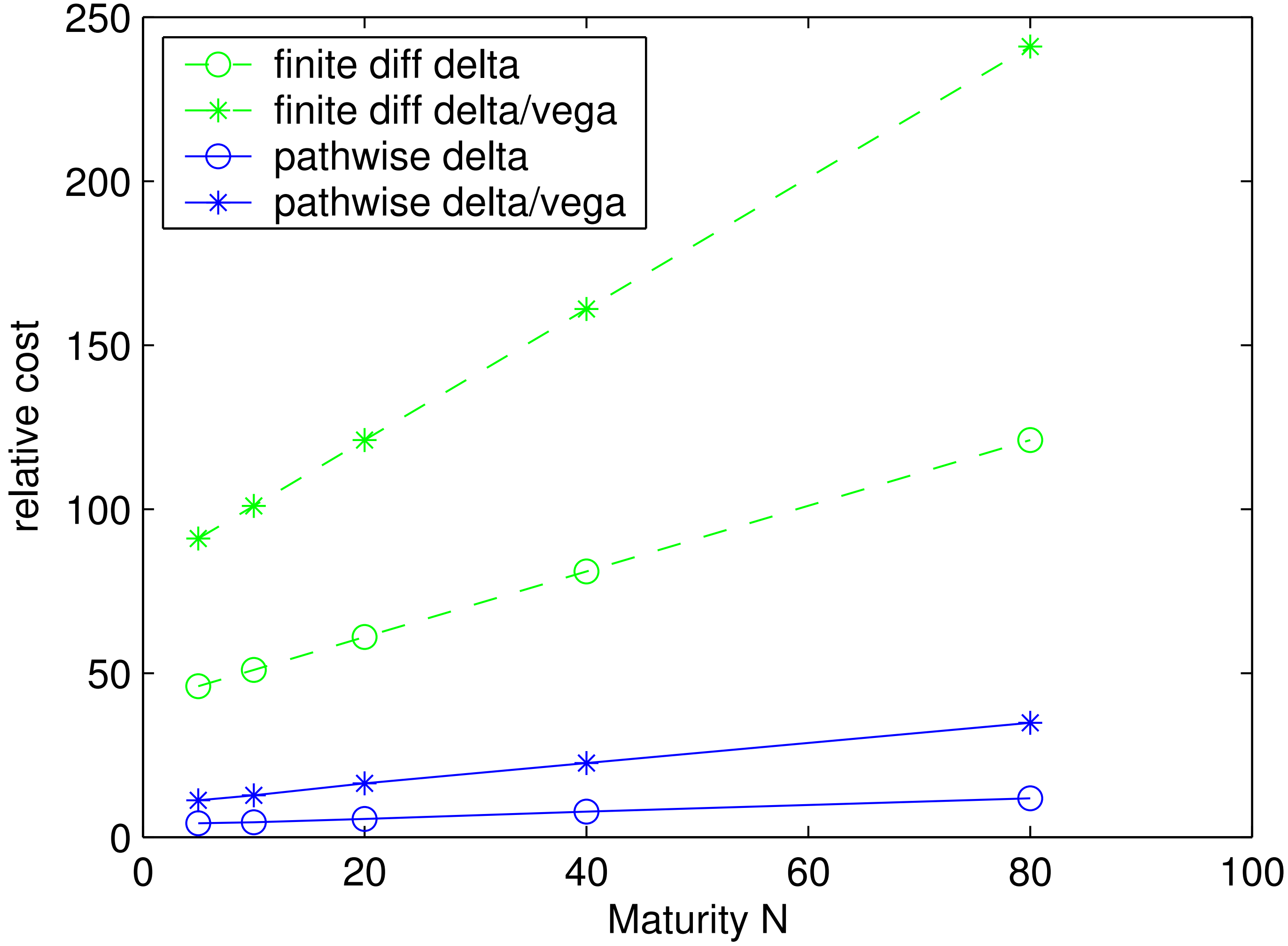
compared to the standard forward approach for which

$$\text{Computational cost} \simeq m^2 \times \text{\# parameters}.$$

However, the adjoint approach only gives the sensitivity of one output, whereas the forward approach can give the sensitivities of multiple outputs for little additional cost.

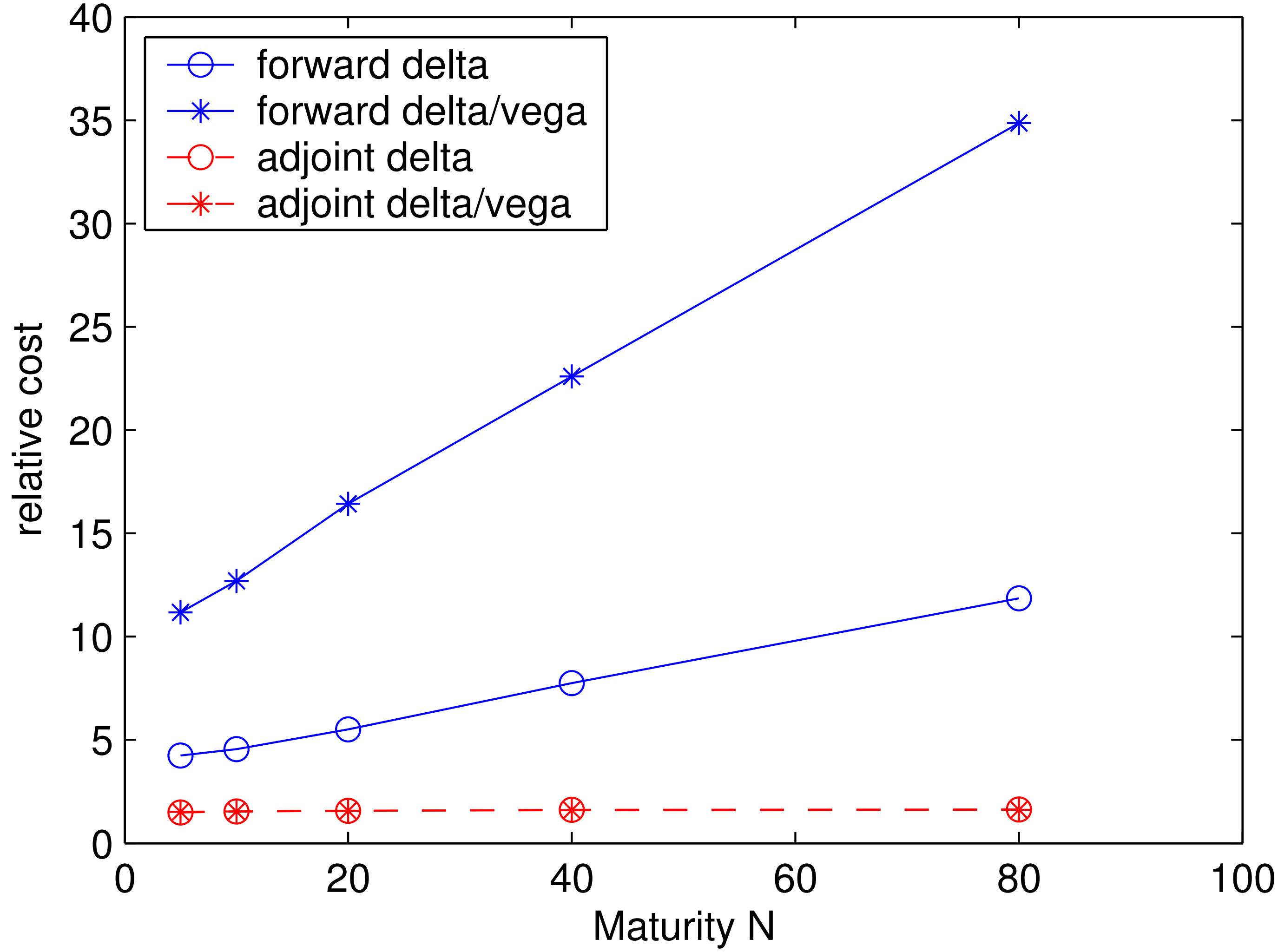
LIBOR Market Model

Finite differences versus forward pathwise sensitivities:



LIBOR Market Model

Forward versus adjoint pathwise sensitivities:



Conclusions

- Greeks are vital for hedging and risk analysis
- Finite difference approximation is simplest to implement, but far from ideal
- Likelihood ratio method for discontinuous payoffs
- In all other cases, pathwise sensitivities are best
- Payoff smoothing may handle the problem of discontinuous payoffs
- Adjoint pathwise approach gives an unlimited number of sensitivities for a cost comparable to the initial valuation

References

- M.B. Giles, P. Glasserman. 'Smoking adjoints: fast Monte Carlo Greeks', RISK, 19(1):88-92, January 2006.
- M. Leclerc, Q. Liang, I. Schneider, 'Fast Monte Carlo Bermudan Greeks', RISK, 22(7):84-88, 2009.
- L. Capriotti, M.B. Giles. 'Fast correlation Greeks by adjoint algorithmic differentiation', RISK, 23(4):77-83, 2010.
- L. Capriotti, J. Lee, M. Peacock, 'Real Time Counterparty Credit Risk Management in Monte Carlo', RISK 24(6):86-90, 2011.
- L. Capriotti, 'Fast Greeks by algorithmic differentiation', Journal of Computational Finance 14(3):3-35, 2011.
- L. Capriotti, M.B. Giles. 'Algorithmic differentiation: adjoint Greeks made easy', RISK, 25(10), 2012.