CREDIT SUISSE




ALGORITHMIC DIFFERENTIATION: ADJOINT GREEKS MADE EASY

Luca Capriotti
Quantitative Strategies
Credit Suisse - Investment Banking Division

Global Derivatives 2011
Paris, April 11–14, 2011

Outline

- Calculating Risk with Monte Carlo … Fast
- Adjoint Formulations of the Pathwise Derivative Method
- Limitation of "Algebraic" Adjoint Methods
- Algorithmic Adjoint Approaches
- Adjoint Algorithmic Differentiation (AAD) in a nutshell
- AAD and the Pathwise Derivative Method: Adjoints made easy
- A preview of \`Real Time Counterparty Credit Risk Management'
- Conclusions

The Setting: Pathwise Derivative Method

- Monte Carlo Expectation Values

$$V(\theta) = \mathbb{E}_{\mathbb{Q}}\Big[g(X(T_1),\ldots,X(T_M))\Big]$$

... and sensitivities

$$\frac{\partial V(\theta)}{\partial \theta_k} = \mathbb{E}_{\mathbb{Q}}\Big[\frac{\partial}{\partial \theta_k} g(X(\theta))\Big]$$

Lipschitz

- Pathwise Derivative Estimator

Chain Rule

$$\bar{\theta}_k \equiv \frac{\partial g(X(\theta))}{\partial \theta_k} = \sum_{j=1}^{N\times M} \frac{\partial g(X)}{\partial X_j} \times \frac{\partial X_j}{\partial \theta_k}$$

Payout Derivatives

Tangent Process

Pathwise Derivative Method: Challenges

$$\bar{\theta}_k \equiv \frac{\partial g(X(\theta))}{\partial \theta_k} = \sum_{j=1}^{N\times M} \frac{\partial g(X)}{\partial X_j} \times \frac{\partial X_j}{\partial \theta_k}$$

Payout Derivatives

Tangent Process

Since the variance of the estimator is comparable to the one of finite differences, all this is worth the hassle if the resulting computational time is significantly lower than the one of Bumping

We need an efficient way to calculate:

1. Simulation of the Tangent Process
2. Derivatives of the Payout

“Algebraic” Adjoint Methods

Giles and Glasserman’s `Smoking Adjoint’, Risk Magazine 2006

Leclerc et al., Risk Magazine 2009

Joshi et al., several preprints

⇨ Libor Market Model & Swaptions

⇨ Concentrate on the efficient Simulation of the Tangent Process

In a nutshell:

1. Formulate the propagation of the Tangent process in terms of Linear Algebra Ops
2. Optimize the computation time by rearranging the order of the computations
3. Implement the rearranged sequence of operations

"Analytic" Adjoint Methods

Tangent Process

$$\bar{\theta}_k = \frac{\partial P(X(N))}{\partial X(N)}^T \Delta(N)$$

Matrix Recursion

$$\Delta(n+1) = D(n)\Delta(n) \quad \Delta(0) = I$$

Matrix Matrix **Forward** Recursion

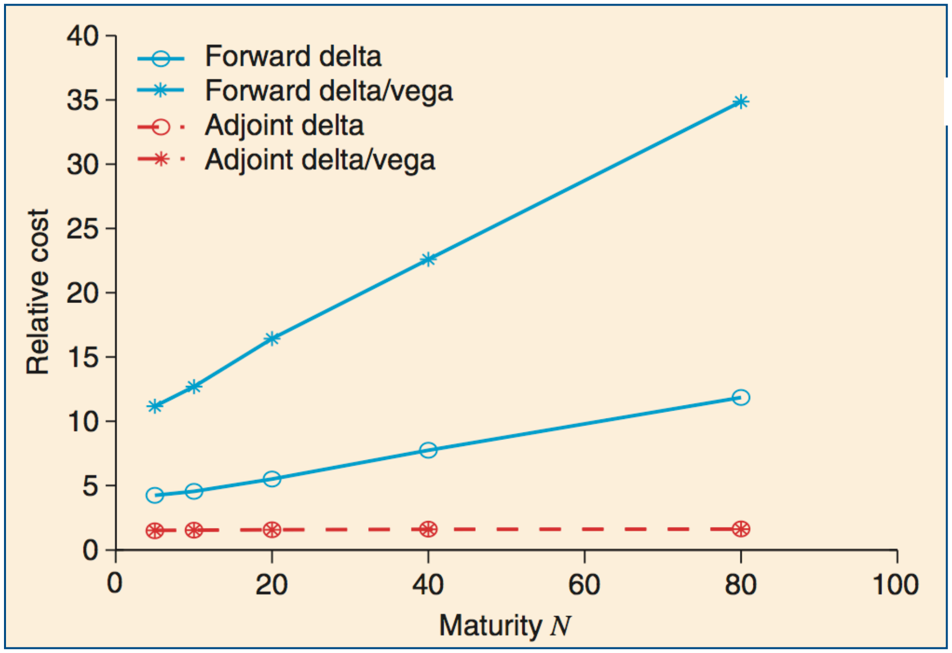
$$O(N^3)$$

$$\bar{\theta}_k = \frac{\partial P(X(N))}{\partial X(N))}^T D(N-1)\cdots D(1)\Delta(0)$$

Matrix Vector **Backward** Recursion

$$O(N^2)$$

“Analytic” Adjoint Methods



Giles and Glasserman, Risk Magazine 2006

Arbitrary number of sensitivity at a **fixed small cost**

Limitations of Analytic Adjoint Methods

- LMM is bit of an ad-hoc application …
  - Difficult to generalize to Path Dependent Options
  - The required Algebraic Analysis is in general cumbersome
  - Not general enough for all the applications in Finance
  - The derivatives required are often not available in closed form
  - What about the derivatives of the Payout?

Algorithmic Adjoint Approaches: AAD

- Adjoint implementations can be seen as instances of a programming technique known as Adjoint Algorithmic Differentiation (AAD)

- In general AAD allows the calculation of the gradient of an algorithm at a cost that a small constant (~ 4) times the cost of evaluating the function itself, independent of the number of input variables.

The Payoff estimator is a mapping of the form

$$\theta \longrightarrow g(X(\theta))$$

AAD gives all the Risk estimators for a small fixed cost

$$\bar{\theta}_k \equiv \frac{\partial g(X(\theta))}{\partial \theta_k}$$

Q How do AAD work anyway?

$$Y = \texttt{FUNCTION}(X)$$

$$X \;\rightarrow\; \ldots \;\rightarrow\; U \;\rightarrow\; V \;\rightarrow\; \ldots \;\rightarrow\; Y$$

Adjoints

$$\boxed{\bar{V}_k = \sum_{j=1}^{m} \bar{Y}_j \frac{\partial Y_j}{\partial V_k}}$$

$$\boxed{\bar{U}_i = \sum_k \bar{V}_k \frac{\partial V_k}{\partial U_i}}$$

Propagation Rule

$$\bar{X} \;\leftarrow\; \ldots \;\leftarrow\; \bar{U} \;\leftarrow\; \bar{V} \;\leftarrow\; \ldots \;\leftarrow\; \bar{Y}$$

$$\bar{X} = \texttt{FUNCTION\_B}(X, \bar{Y})$$

Main Result

$$\boxed{\bar{X}_i = \sum_{j=1}^{m} \bar{Y}_j \frac{\partial Y_j}{\partial X_i}}$$

Lin. Comb. Jacobian Rows
at a small **fixed** cost

AAD as a Design Paradigm

- AAD can be used as a **design paradigm** even for large inhomogeneous algorithms
- Address both aspects of the implementation of the Pathwise Derivative Method

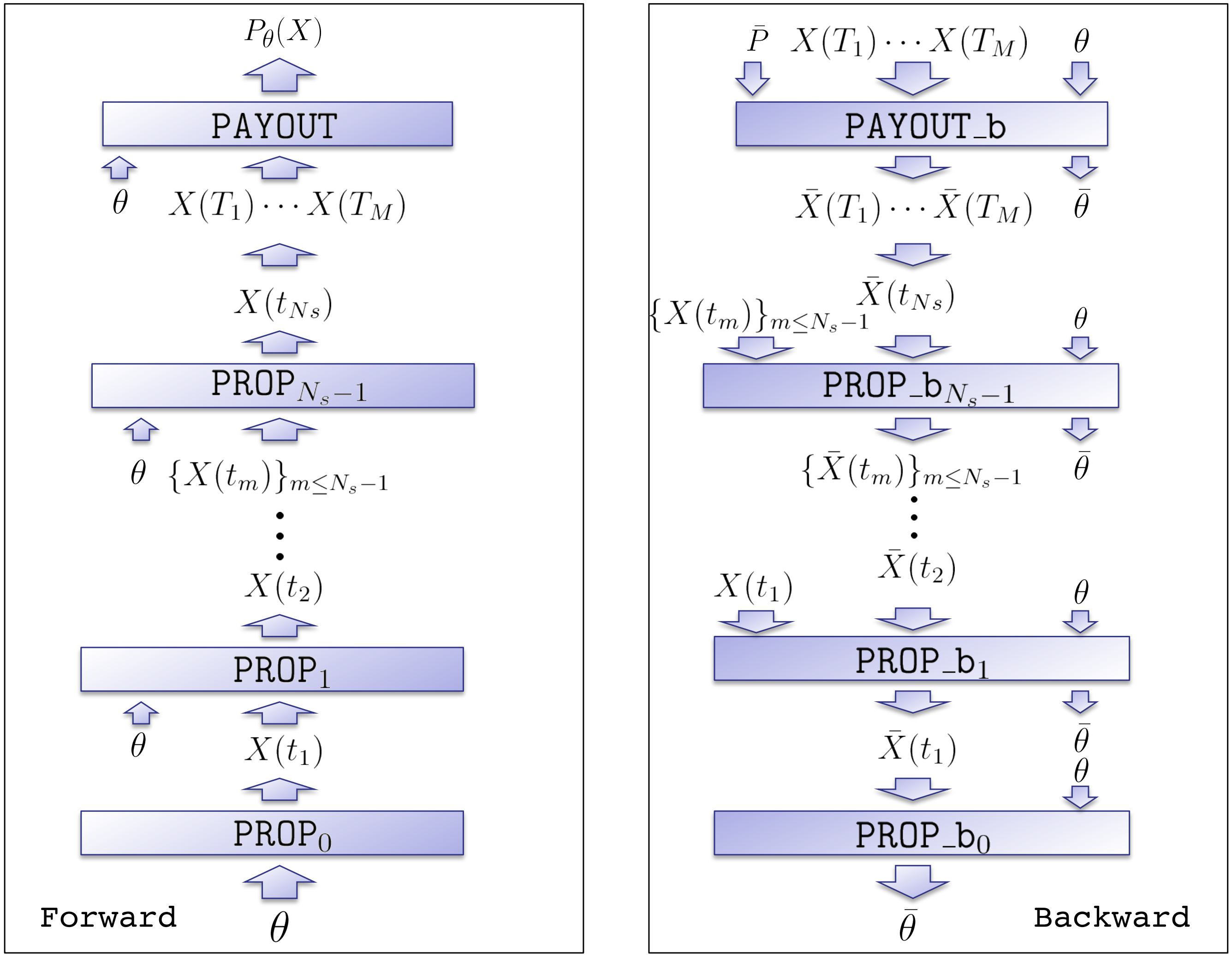
Derivative of the Payout $$\frac{\partial g(X)}{\partial X_j}$$ L.C., Journ. Comp. Fin. (2011)

Tangent Process $$\frac{\partial X_j(\theta)}{\partial \theta_k}$$

$$\bar{\theta}_k \equiv \frac{\partial g(X(\theta))}{\partial \theta_k} = \sum_{j=1}^{N\times M} \frac{\partial g(X)}{\partial X_j} \times \frac{\partial X_j}{\partial \theta_k}$$

- Linear combination of the columns of the Jacobian
- All the Greeks at a cost that is a small (~4) multiple of the PV estimator

Diffusive Setting



Lognormal Example

$$X(t_{n+1}) = \mathtt{PROP}_n(X(t_n), \theta)$$

- Step 1

$$\mu = rX(t_n)$$

- Step 2

$$\Sigma = \sigma X(t_n)$$

- Step 3

$$R = e^{(\mu-\Sigma^2/2)\Delta t+\Sigma\sqrt{\Delta t}Z}$$

- Step 4

$$X(t_{n+1}) = X(t_n)\,R$$

$$(\bar{X}(t_n), \bar{\theta}) {+}{=} \mathtt{PROP\_b}_n(., \bar{X}(t_{n+1}))$$

- Step $\bar{1}$

$$\bar{X}(t_n) {+}{=} \bar{\mu} r \quad \bar{\theta}_r {+}{=} \bar{\mu} X(t_n)$$

- Step $\bar{2}$

$$\bar{X}(t_n) {+}{=} \bar{\Sigma}\sigma \quad \bar{\theta}_\sigma {+}{=} \bar{\Sigma} X(t_n)$$

- Step $\bar{3}$

$$\bar{\mu} = \bar{R}\,R\Delta t$$

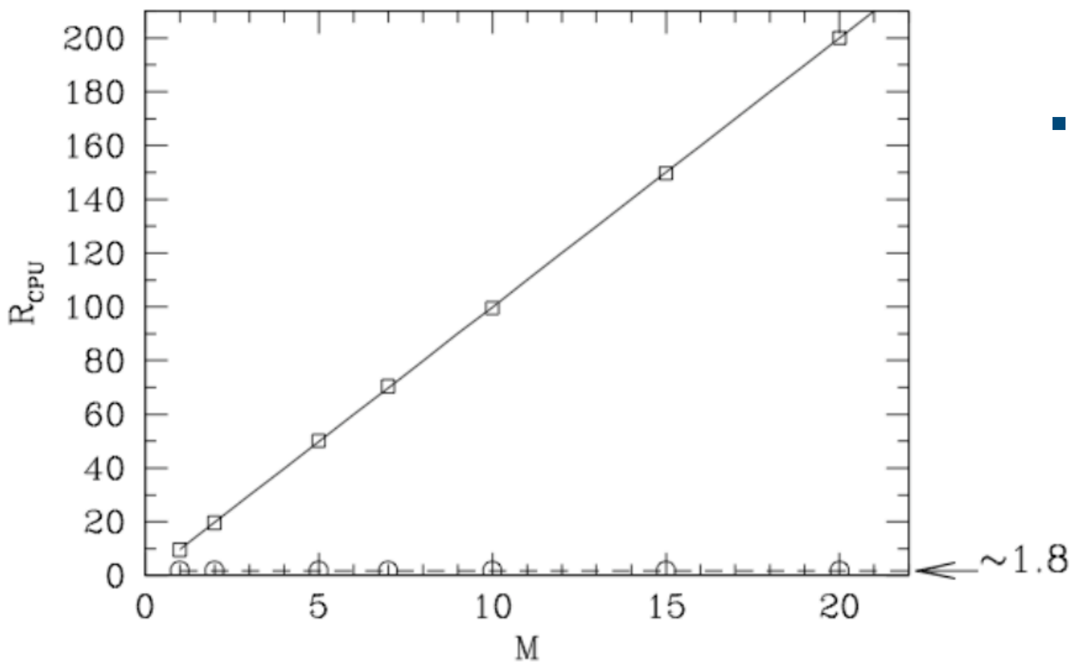
$$\bar{\Sigma} = \bar{R}\,R(-\Sigma\Delta t + \sqrt{\Delta t}Z)$$

- Step $\bar{4}$

$$\bar{X}(t_n) {+}{=} \bar{X}(t_{n+1})\,R$$

$$\bar{R} = \bar{X}(t_{n+1})$$

Best of Asian Option

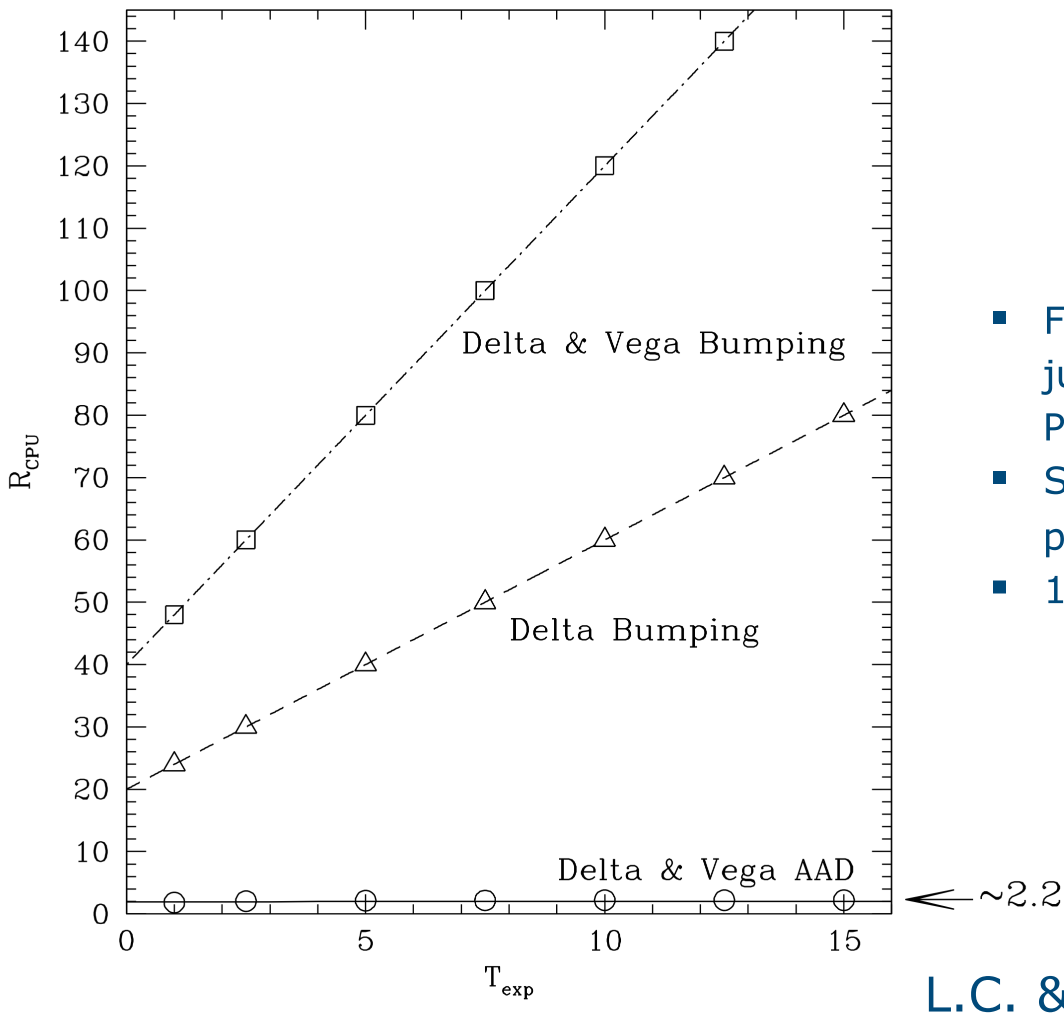


- Full Delta and Vega calculation for just twice the cost to calculate the PV

L.C., Journ. Comp. Fin. (2011)

L.C. & Mike Giles, preprint (2011)

Back to the LMM test ground



- Full Delta and Vega calculation for just twice the cost to calculate the PV
- Similar results for both Euler and predictor corrector discretizations
- 100x savings in typical applications

L.C. & Mike Giles, preprint (2011)

A preview of:

Q "Real Time Counterparty Risk Management in Monte Carlo"

- Risk manage CVA/DVA is challenging because all the trades facing the same counterparty must be valued at the same time, typically with Monte Carlo

- AAD is naturally suited for this task

$$
\begin{aligned}
V_{\mathrm{CVA}} \simeq & \sum_{i=1}^{N_O} \mathbb{E}\Big[\mathbb{I}(T_{i-1} < \tau_c \leq T_i) D(0, T_i) \\
& \times L_{\mathrm{GD}}(T_i) \Big(NPV(T_i) - C\left(R(T_i^-)\right) \Big)^+ \Big]
\end{aligned}
$$

Cpty Rating

A preview of:

“Real Time Counterparty Risk Management in Monte Carlo”

- A new challenge: Rating Dependent Payoffs

$$P\left(T_i, R(T_i), X(T_i)\right) = \sum_{r=0}^{N_R} \tilde{P}_i(X(T_i); r)\, \delta_{r, R(T_i)}$$

Rating Transition Markov Chain model (Jarrow, Lando and Turnbull ‘97)

$$R(T_i) = \sum_{r=1}^{N_R} \mathbb{I}\left(\tilde{Z}_i^R > Q(T_i, r)\right)$$

Quantile Threshold

- The Rating state space is discrete (hence the Payoff is non Lipschitz)
- The Pathwise Derivative method gives only part of the Risk

A preview of:
Q "Real Time Counterparty Risk Management in Monte Carlo"

Singular Contribution:

$$\partial_{\theta_k} P\big(T_i, \tilde{Z}_i, X(T_i)\big) = -\sum_{r=1}^{N_R} \Big(\tilde{P}_i(X(T_i); r) - \tilde{P}_i(X(T_i); r-1)\Big)\delta\left(\tilde{Z}_i^R = Q(T_i, r; \theta)\right)\, \partial_{\theta_k} Q(T_i, r; \theta)$$

Can be integrated out analytically

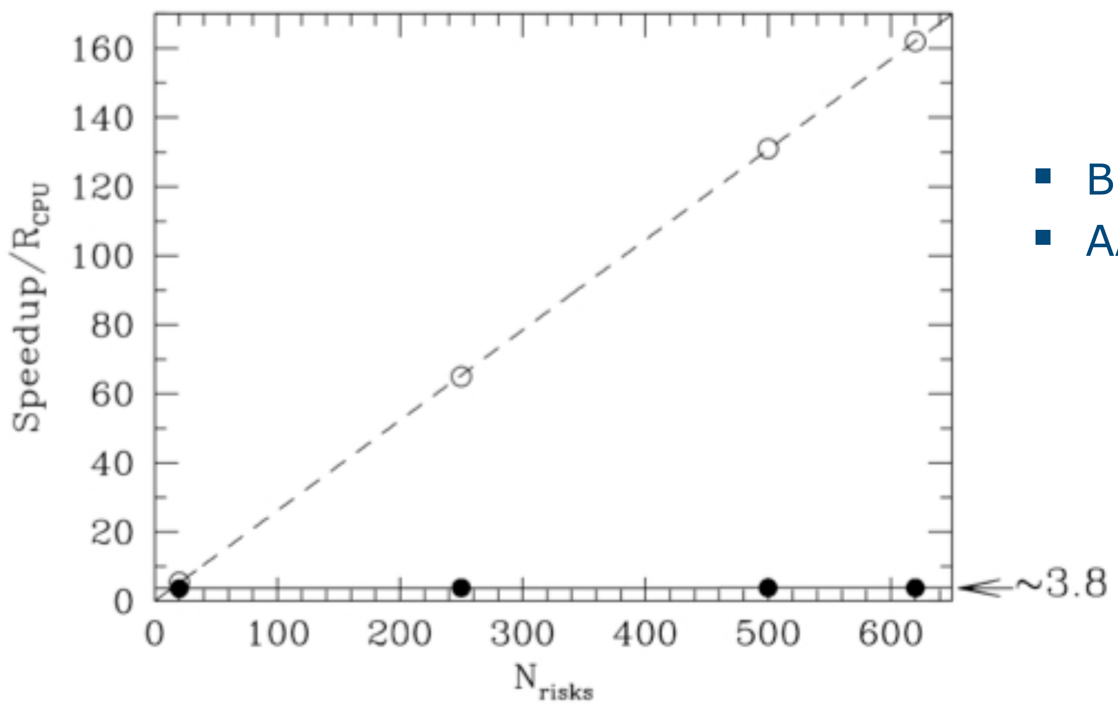
$$\bar{\theta}_k = -\sum_{r=1}^{N_R} \frac{\phi(Z^\star, Z_i^X, \rho_i)}{\sqrt{i}\,\phi(Z_i^X, \rho_i^X)} \partial_{\theta_k} Q(T_i, r; \theta) \times \Big(\tilde{P}_i(X(T_i); r) - \tilde{P}_i(X(T_i); r-1)\Big)$$

Variance Reduction vs. Bumping:

δ	VR[Q(1,1)]	VR[Q(1,2)]	VR[Q(1,3)]
0.1	24	16	12
0.01	245	165	125
0.001	2490	1640	1350

A preview of:
"Real Time Counterparty Risk Management in Monte Carlo"

- Test Application: Calculation of risk for the CVA of a portfolio of 5 commodity swaps over a 5 years horizon (over 600 risks)



- Bumping: ~ 1h 40 min
- AAD: ~ 10 sec

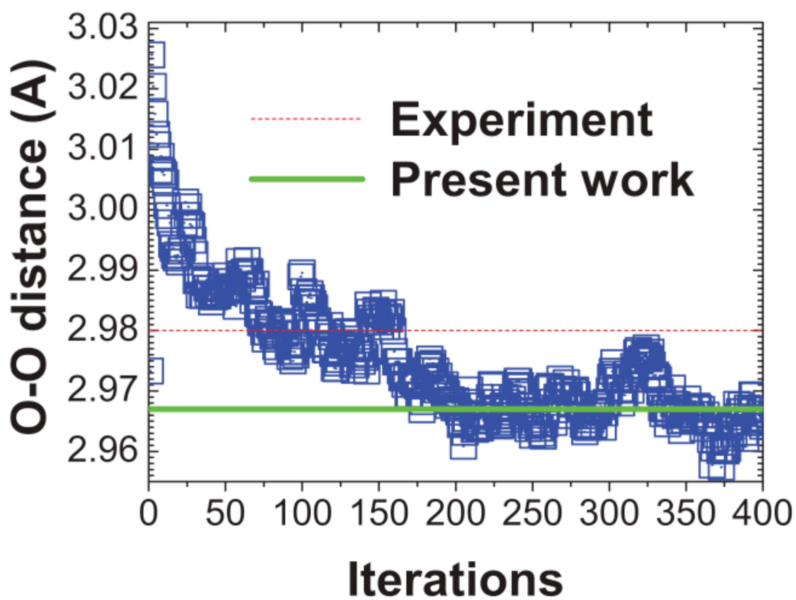
L.C., J. Lee and M. Peacock, preprint (2011)

“Giving Back” to Natural Sciences

THE JOURNAL OF CHEMICAL PHYSICS **133**, 234111 (2010)

Algorithmic differentiation and the calculation of forces by quantum Monte Carlo

Sandro Sorella[1,2,a)] and Luca Capriotti[3,b)]
[1] *SISSA, International School for Advanced Studies, 34151, Trieste, Italy*
[2] *DEMOCRITOS Simulation Center, CNR-IOM Istituto Officina dei Materiali, 34151, Trieste, Italy*
[3] *Quantitative Strategies, Investment Banking Division, Credit Suisse Group, Eleven Madison Avenue, New York City, New York 10010-3086, USA*



Opens the way to Monte Carlo simulations based on first principles Quantum Mechanics

Conclusions

- Algebraic Adjoint approaches can be seen as specific instances of a more general paradigm: Adjoint Algorithmic Differentiation (AAD)
- AAD can be employed to evaluate efficiently option sensitivities for virtually any model and financial security encountered in practice
- AAD allows the calculation of the Greeks in at most 4 times the time necessary for calculation of the P&L of the portfolio
- Risk is calculated orders of magnitude faster than standard bumping, thus producing a significant reduction in infrastructure costs, and allowing "real time" monitoring of Risk and more effective hedging strategies
- Preview of: Real Time CCRM
    - Analytical Integration of the Rating Singular Contribution
    - Additional significant Speed Up coming from Variance Reduction

References

- L.C. and M. Giles, *Fast Correlation Greeks by Adjoint Algorithmic Differentiation*, Risk Magazine, April 2010
- L.C., *Fast Greeks by Algorithmic Differentiation,* Journal of Computational Finance **14**, 3 (2011)
- L.C., Jacky Lee, and Matthew Peacock, *Real Time Counterparty Credit Risk Management in Monte Carlo,* preprint 2011
- S. Sorella and L.C., *Algorithmic Differentiation and the calculation of forces in Quantum Monte Carlo*, Journal of Chemical Physics **3**, 234111 (2010)

Also available at **www.luca-capriotti.net**