Algorithmic Finance 4 (2015) 81–87
DOI 10.3233/AF-150045
IOS Press

Likelihood Ratio Method and Algorithmic Differentiation: Fast Second Order Greeks

Luca Capriotti*
Quantitative Strategies, Investment Banking Division, Credit Suisse Group, One Cabot Square, London E14 4QJ, United Kingdom

Abstract. We show how Adjoint Algorithmic Differentiation can be combined with the so-called Pathwise Derivative and Likelihood Ratio Method to construct efficient Monte Carlo estimators of second order price sensitivities of derivative portfolios. We demonstrate with a numerical example how the proposed technique can be straightforwardly implemented to greatly reduce the computation time of second order risk.

Keywords: Adjoint Algorithmic Differentiation, Monte Carlo, derivatives securities, risk management

1. Introduction

Monte Carlo (MC) simulations are a tool of paramount importance for risk management, especially in the context of counterparty risk borne by derivative securities (Capriotti and Lee, 2014).

The main drawback of MC methods is that they are generally computationally expensive. This is especially true when MC simulations are used for the calculation of price sensitivities, i.e., the derivatives of a security price with respect to the parameters of the model underlying its valuation: the so-called *Greeks* (Hull, 2002). Indeed, standard approaches for the calculation of the Greeks require repeating the calculation of the Profit & Loss of a portfolio under multiple perturbations of the market parameters in order to compute finite-difference estimators of the price sensitivities.

A recently introduced technology for real time risk is *Adjoint Algorithmic Differentiation* (AAD) (Capriotti and Giles, 2010; Capriotti, 2011; Capriotti, Lee, and Peacock, 2011; Capriotti and Giles, 2012). This powerful technique allows the fast computation of *first order* sensitivities without the necessity of repeating the valuation of the portfolio multiple times as in traditional finite-difference approaches. In the context of MC simulation, AAD can be used to implement efficiently the so-called Pathwise Derivative Method (Broadie and Glasserman, 1996; Glasserman, 2004).

In this paper we address *second order* sensitivities and will show how AAD can be combined with the so-called Likelihood Ratio Method (LRM) (Broadie and Glasserman, 1996; Glasserman, 2004) in order to reduce their computational cost by orders of magnitude. This paper is a contribution in the literature around AAD and the calculation of second order sensitivities, for a review see e.g. (Caplan, 2011).

2. Pathwise Derivative Method

Option pricing problems can be typically formulated in terms of the calculation of expectation values of the form

$$V = \mathbb{E}\Big[P(X(T_1), \ldots, X(T_M))\Big]. \qquad (1)$$

*Corresponding author: Luca Capriotti, Quantitative Strategies, Investment Banking Division, Credit Suisse Group, One Cabot Square, London E14 4QJ, United Kingdom. E-mail: luca.capriotti@gmail.com.

Here $X(t)$ is a N-dimensional vector and represents the value of a set of underlying market factors (e.g., stock prices, interest rates, foreign exchange pairs, etc.) at time t.

$$P(X(T_1), \ldots, X(T_M))$$

is the discounted payout function of the priced security, and depends in general on M observations of those factors. In the following, we will indicate the collection of such observations with a $d = N \times M$ dimensional state vector $X = (X(T_1), \ldots, X(T_M))^t$. Here and below the expectation values are assumed conditional on the filtration at time $t = 0$.

As a first example, we consider the case in which the underlying factors follow multi dimensional diffusion processes of the form

$$dX(t) = \mu(X(t), t, \theta)\, dt + \sigma(X(t), t, \theta)\, dW_t, \qquad (2)$$

with $X(t_0) = X_0$. Here the drift $\mu(X, t, \theta)$ and volatility $\sigma(X, t, \theta)$ are N-dimensional vectors, and W_t is a N-dimensional Brownian motion with instantaneous correlation matrix $\rho(t)$ defined by $\rho(t)\, dt = \mathbb{E}\left[dW_t dW_t^T\right]$. Here, the vector $\theta = (\theta_1, \ldots, \theta_{N_\theta})$ represents a set of N_θ parameters the model is dependent on.

In this case, the evolution of the process X is usually approximated by sampling $X(t)$ on a discrete grid of points $0 = t_0 < t_1 < \ldots < t_n < \cdots < t_{N_s}$, a superset of the observation times $(T_1, \ldots, T_M)$, by means, for instance, of an Euler scheme:

$$\begin{aligned} X(t_{n+1}) = X(t_n) &+ \mu(X(t_n), t_n, \theta)\, h_n \\ &+ \sigma(X(t_n), t_n, \theta)\, \sqrt{h_n}\, Z(t_n), \end{aligned} \qquad (3)$$

where $h_n = t_{n+1} - t_n$, and $Z(t_n)$ is a N-dimensional vector of correlated unit normal random variables, with $\rho(t_n) = \mathbb{E}\left[Z(t_n) Z(t_n)^T\right]$.

The expectation value in (1) can be estimated by means of MC by sampling a number N_{MC} of random replicas of the underlying state vector $X[1], \ldots, X[N_{\mathrm{MC}}]$ and evaluating the payout $P(X)$ for each of them. This leads to the estimate of the option value V as

$$V \simeq \frac{1}{N_{\mathrm{MC}}} \sum_{i_{\mathrm{MC}}=1}^{N_{\mathrm{MC}}} P\left(X[i_{\mathrm{MC}}]\right). \qquad (4)$$

The Pathwise Derivative Method (Broadie and Glasserman, 1996) allows the calculation of the sensitivities of the option price V (1) with respect to the N_θ parameters θ, with a single simulation. Indeed, whenever the payout function is regular enough, e.g., Lipschitz-continuous, and under additional conditions that are often satisfied in financial pricing (see, e.g., (Glasserman, 2004)), one can write the sensitivity $\langle \bar{\theta}_k \rangle \equiv \partial V / \partial \theta_k$ as

$$\langle \bar{\theta}_k \rangle = \mathbb{E}\Big[\frac{\partial P(X)}{\partial \theta_k} \Big]. \qquad (5)$$

In general, the calculation of Equation (5) can be performed by applying the chain rule and averaging on each MC path the so-called pathwise derivative estimator

$$\bar{\theta}_k \equiv \frac{\partial P(X)}{\partial \theta_k} = \sum_{j=1}^{d} \frac{\partial P(X)}{\partial X_j} \times \frac{\partial X_j(\theta)}{\partial \theta_k}. \qquad (6)$$

In the following we will show how the Pathwise Derivative Method can be implemented efficiently by means of Adjoint Algorithmic Differentiation (AAD).

3. Adjoint Algorithmic Differentiation

The main idea underlying Algorithmic differentiation (AD) (Griewank and Walther, 2008) is that any computer implemented function – no matter how complicated – can be interpreted as a composition of basic arithmetic and intrinsic operations that are easy to differentiate. What makes AD particularly attractive, when compared to finite-difference methods is its computational efficiency. In fact, AD exploits the information on the structure of the computer code in order to optimize the calculation which – as an added benefit – are also not affected by any finite-difference error. In particular, when one requires the derivatives of a small number of outputs with respect to a large number of inputs, the calculation can be highly optimized by applying the chain rule through the instructions of the program in opposite order with respect to their original evaluation. This gives rise to Adjoint Algorithmic Differentiation (AAD).

Griewank (Griewank and Walther, 2008) contains a detailed discussion of the computational cost of AAD. In this section, we will only recall the main results in order to clarify how this technique can be beneficial in a financial setting. The interested reader can find in (Capriotti, 2011) several simple examples illustrating the intuition behind these results.

To this end, consider a function

$$Y = \texttt{FUNCTION}(X) \qquad (7)$$

mapping a vector X in $\mathbb{R}^n$ in a vector Y in $\mathbb{R}^m$ through a sequence of steps

$$X \to \ldots \to U \to V \to \ldots \to Y. \quad (8)$$

Here, the real vectors U and V represent intermediate variables used in the calculation and each step can be a distinct high-level function or even an individual instruction.

The adjoint mode of AD results from propagating the derivatives of the final output with respect to all the intermediate variables – the so called *adjoints* – until the derivatives with respect to the independent variables are formed. Using the standard AD notation, the adjoint of any intermediate variable V_k is defined as

$$\bar{V}_k = \sum_{j=1}^{m} \bar{Y}_j \frac{\partial Y_j}{\partial V_k}, \quad (9)$$

where $\bar{Y}$ is vector in $\mathbb{R}^m$. In particular, for each of the intermediate variables U_i, using the chain rule we get,

$$\bar{U}_i = \sum_{j=1}^{m} \bar{Y}_j \frac{\partial Y_j}{\partial U_i} = \sum_{j=1}^{m} \bar{Y}_j \sum_k \frac{\partial Y_j}{\partial V_k} \frac{\partial V_k}{\partial U_i},$$

which corresponds to the adjoint mode equation for the intermediate step represented by the function $V = V(U)$

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

namely a function of the form $\bar{U} = \bar{V}(U, \bar{V})$. Starting from the adjoint of the outputs, $\bar{Y}$, we can apply this rule to each step in the calculation, working from right to left,

$$\bar{X} \leftarrow \ldots \leftarrow \bar{U} \leftarrow \bar{V} \leftarrow \ldots \leftarrow barY \quad (10)$$

until we obtain $\bar{X}$, i.e., the following linear combination of the rows of the Jacobian of the function $X \to Y$:

$$\bar{X}_i = \sum_{j=1}^{m} \bar{Y}_j \frac{\partial Y_j}{\partial X_i}, \quad (11)$$

with $i = 1, \ldots, n$.

In the adjoint mode, the cost does not increase with the number of inputs, but it is linear in the number of (linear combinations of the) rows of the Jacobian that need to be evaluated independently. In particular, if the full Jacobian is required, one needs to repeat the adjoint calculation m times, setting the vector $\bar{Y}$ equal to each of the elements of the canonical basis in $\mathbb{R}^m$.

One particularly important theoretical result is that given a computer program performing some high-level function (7), the execution time of its adjoint counterpart

$$\bar{X} = \texttt{FUNCTION\_b}(X, \bar{Y}) \quad (12)$$

(with suffix _b for "backward" or "bar") calculating the linear combination (11) is bounded by approximatively 4 times the cost of execution of the original one, namely

$$\frac{\text{Cost}[\texttt{FUNCTION\_b}]}{\text{Cost}[\texttt{FUNCTION}]} \leq \omega_A \quad (13)$$

with $\omega_A \in [3, 4]$ (Griewank and Walther, 2008).

4. AAD and the Pathwise Derivative Method

AAD provides a general design and programming paradigm for the efficient implementation of the Pathwise Derivative Method. This stems from the observation that the pathwise estimator in (6) is a linear combination of the rows of the Jacobian of the map $\theta \to X(\theta)$, describing the simulated state vector, with weights given by the X gradient of the payout function $P(X)$. Both the calculation of the derivatives of the payout and of the linear combination of the rows of $\partial X/\partial\theta$ are tasks that can be performed efficiently by AAD. In particular, as discussed above, one can obtain all the pathwise sensitivities with respect to θ, $\bar{\theta}$, at a cost that is at most roughly 4 times the cost of calculating the payout estimator itself.

In a MC simulation, the sampling of the market risk factor X on the observation dates $(T_1, \ldots, T_M)$ according to the discretization scheme (3) could be implemented for instance by means of a method of the form

$$(X(T_1), \ldots, X(T_M)) = \texttt{PROP}[\theta, Z], \quad (14)$$

where $Z = (Z(t_0), \ldots, Z(t_{N_s}))$. The payout estimator $P(X)$ is then evaluated on the random sample X

$$P = P(X(T_1), \ldots, X(T_M)). \quad (15)$$

The first step of the adjoint algorithm is the adjoint of the payout evaluation. This is a function of the form

$$\bar{X} = \bar{P}(X(T_1), \ldots, X(T_M), \bar{P}) \quad (16)$$

where $\bar{X} = (\bar{X}(T_1), \ldots, X(T_M))$ is the adjoint of the state vector on the observation dates,

$$\bar{X}(T_m) = \frac{\partial P(X)}{\partial X(T_m)} \bar{P}, \tag{17}$$

for $m = 1, \ldots, M$, and $\bar{P} = 1$. The adjoints of the state vector on the simulation dates corresponding to the observation dates are initialized at this stage. The adjoint state vector is then propagated backwards in time through the adjoint of the propagation method (14), namely

$$\bar{\theta} = \texttt{PROP\_b}[\theta, Z, \bar{X}], \tag{18}$$

computing internally the adjoint of the propagation rule (3)

$$\bar{X}(t_n) \mathrel{+}= \sum_{j=1}^{N} \bar{X}_j(t_{n+1}) \frac{\partial X_j(t_{n+1})}{\partial X(t_n)}, \tag{19}$$

$$\bar{\theta} \mathrel{+}= \sum_{j=1}^{N} \bar{X}_j(t_{n+1}) \frac{\partial X_j(t_{n+1})}{\partial \theta}, \tag{20}$$

for $n = N_s - 1, \ldots 0$, where we have used the notation $+=$ for the standard addition assignment operator. Here $\bar{\theta}$ and $\bar{X}(t_n)$ for all t_n not corresponding to an observation date are assumed initialized to zero upon entering the function PROP_b. It is easy to verify that the output $\bar{\theta}$ is the pathwise derivative estimator in Equation (6). In the following, as it is customary among practitioners (Capriotti and Giles, 2012), we will refer to this method simply as AAD.

5. Likelihood Ratio Method

An alternative method for the calculation of sensitivities is the Likelihood Ratio Method (LRM) (Broadie and Glasserman, 1996). Under mild regularity conditions on the probability density $\varphi_\theta(X)$, the sensitivity of the option price (1)

$$V(\theta) = \mathbb{E}[P(X)] = \int dX \;\; P(X)\, \varphi_\theta(X)\,, \tag{21}$$

with respect to any parameter θ_k can be obtained as

$$\bar{\theta}_k = \mathbb{E}[P(X)\Omega_k(X)]\,, \tag{22}$$

i.e., by calculating the expectation value of the original payout function multiplied by the so-called *likelihood ratio* weight

$$\Omega_k(x) = \frac{\partial \log \varphi_\theta(X)}{\partial \theta_k}\,, \tag{23}$$

giving as MC estimator:

$$\bar{\theta}_k \simeq \frac{1}{N_{\mathrm{MC}}} \sum_{i_{\mathrm{MC}}=1}^{N_{\mathrm{MC}}} P(X[i_{\mathrm{MC}}])\Omega_k(X[i_{\mathrm{MC}}])\,. \tag{24}$$

In the diffusive setting (2), using the Markov property, the probability density function $\varphi_\theta(X)$ can be factored as

$$\varphi_\theta(X) = \prod_{m=0}^{M-1} \varphi_\theta(X(T_{m+1})|X(T_m)), \tag{25}$$

where $\varphi_\theta(X(T_{m+1})|X(T_m))$ represents the probability density for the random variable $X(t)$ at time T_{m+1} conditional on assuming the value $X(T_m)$ at time $t = T_m$. When such transition probabilities are not known in closed form they can be approximated using a discretization scheme. For instance, under the Euler scheme of Equation (3),

$$\varphi_\theta(X(T_{m+1})|X(T_m)) = \int \prod_{t_n \in (T_m, T_{m+1})} dX(t_n) \tag{26}$$

$$\varphi_\theta(X(t_{n+1})|X(t_n)) \simeq \left(\prod_{i=1}^{N} \frac{1}{\sigma_i(X(t_n), t_n, \theta)\sqrt{h_n}} \right) \phi(Z_\theta(t_n); \rho), \tag{27}$$

where the N-dimensional vector $Z_\theta(t_n)$ is

$$Z_\theta(t_n) = \frac{X(t_{n+1}) - X(t_n) - \mu(X(t_n), t_n, \theta)h_n}{\sigma(X(t_n), t_n, \theta)\sqrt{h_n}}, \tag{28}$$

(to be read component-wise) and $\phi(Z; \rho)$ is the N-dimensional multivariate Gaussian density with zero mean and covariance ρ.

In this case, the likelihood ratio weight takes the form

$$\Omega_{\theta_k}(X) = \sum_{n=0}^{N_s-1} \Omega_{\theta_k}(X(t_{n+1})|X(t_n)), \tag{29}$$

with

$$\Omega_{\theta_k}(X(t_{n+1})|X(t_n)) = \partial_{\theta_k} \log \phi(Z_\theta(t_n)) = -\partial_{\theta_k} \sum_{i=1}^{N} \log \sigma_i(X(t_n), t_n, \theta)$$

$$-Z_\theta(t_n)^T \rho^{-1}(t_n)\partial_{\theta_k} Z_\theta(t_n). \tag{30}$$

Although we will not take advantage of this in the simple examples discussed here, it is worth noting that one can apply the principles of AD to compute efficiently the derivatives contained in the expression of the likelihood ratio weights (30).

It is easy to see that the likelihood ratio weights above give the expected result in the case a multi-asset lognormal model of the form

$$\frac{dS_i(t)}{S_i(t)} = r\,dt + \sigma_i\,dW_t, \tag{31}$$

for $i = 1, \ldots N$, where S_i^0 and σ_i are the spot price and volatility of the i-th asset (the components of the vector of model parameters θ) and r is a constant instantaneous short rate of interest. In this case by performing the usual transformation $X_i = \log S_i$ (Karatzas and Shreve, 1988) one obtains the exact recursion

$$X_k(T_{m+1}) = X_k(T_m) + \left(r - \frac{\sigma_k^2}{2}\right)\Delta T_m + \sigma_k\sqrt{\Delta T_m}\,Z_k(T_m), \tag{32}$$

with $\Delta T_m = T_{m+1} - T_m$, so that the probability density function (25) reads

$$\varphi_\theta(X(T_{m+1})|X(T_m)) = \prod_{i=0}^{N_s-1} \frac{1}{S_i(T_{m+1})\sigma_i\sqrt{\Delta T_m}}\phi(Z_\theta(T_m)), \tag{33}$$

with

$$Z_\theta(T_m) = \frac{\log(S(T_{m+1})/S(T_m)) - (r - \sigma^2/2)\Delta T_m}{\sigma\sqrt{\Delta T_m}}, \tag{34}$$

and the likelihood ratio weights for the k-th Delta and Vega (Hull, 2002) are given by

$$\Omega_k^\Delta(X) = \frac{[\rho^{-1}Z(t_0)]_k}{\sigma_k\sqrt{\Delta t_0}S_k(t_0)}, \tag{35}$$

and

$$\Omega_k^V(X) = \sum_{m=0}^{M-1}\left[-\frac{1}{\sigma_k} + [\rho^{-1}Z(T_m)]_k\left(\frac{Z_k(T_m)}{\sigma_k} - \sqrt{\Delta T_m}\right)\right]. \tag{36}$$

In the special case of a single asset, the weights above simplify to the well-known expressions

$$\Omega^\Delta(X) = \frac{Z(t_0)}{\sigma\sqrt{\Delta t_0}S(t_0)}, \tag{37}$$

and

$$\Omega^V(X) = \sum_{m=0}^{M-1}\left(\frac{Z(T_m)^2 - 1}{\sigma} - Z(T_m)\sqrt{\Delta T}\right). \tag{38}$$

For a given number of MC iterations, the calculation of the Greeks by means of the LRM is generally fast when compared to finite differences (Broadie and Glasserman, 1996). However, the speed of convergence of the LRM estimators is difficult to predict *a priori*, as it is payout and parameters dependent, and can be in some instances particularly slow (Glasserman, 2004). In fact, since the LRM weight has in general zero mean as a result of the identity

$$\partial_\theta \int dX\ \varphi_\theta(X) = 0, \tag{39}$$

the LRM estimators for the Greeks have no definite sign. This can give rise to poor variance properties whenever the configurations with opposite sign have similar weight in the MC average (24) so that the final outcome is the result of the cancellation of two comparable and not necessarily highly correlated quantities. While variance reduction techniques can be used in many specific cases to reduce the variance of the LRM estimators (Capriotti, 2008), generally speaking, the pathwise derivative estimators have a smaller variance - often much smaller. As a results AAD is generally the preferred method for the calculation of first order sensitivities (Capriotti and Giles, 2012).

6. Mixed AAD LRM Estimators for Second Order Greeks

Both AAD and LRM can be generalized to compute second order sensitivities. However, the computational cost associated with second order AAD pathwise derivative sensitivities of $V(\theta)$ can be shown to scale linearly with the number of model parameters θ, N_θ (Griewank and Walther, 2008; Capriotti and Giles, 2012). This has the same computational cost of forming finite-difference estimators of the form

$$\frac{\partial^2 V(\theta)}{\partial\theta_l\partial\theta_k} \simeq \frac{\bar\theta_k(\theta + \delta e_l) - \bar\theta_k(\theta)}{\delta}, \tag{40}$$

where e_l is the l-th canonical vector in $\mathbb{R}^{N_\theta}$, by computing N_θ 'bumped' AAD estimators $\bar{\theta}_k(\theta + \delta e_l)$, for $l = 1, \ldots, N_\theta$, in addition to the base estimator $\bar{\theta}_k(\theta)$.

On the other hand, second order LRM estimators are know to be characterized in many cases by poor variance properties (Glasserman, 2004).

It is however possible to combine the AAD and the LRM estimators for first order sensitivities to obtain a mixed LRMAAD estimator for second order sensitivities. In order to do so, one needs to notice that the expectation of the pathwise estimator $\bar{\theta}_k$ in Equation (6),

$$\langle \bar{\theta}_k \rangle = \int dX \varphi_\theta(X) \sum_{i=1}^{d} \frac{\partial P(X)}{\partial X_i} \frac{\partial X_i(\theta)}{\partial \theta_k}, \tag{41}$$

has both a functional dependence as well as a distributional dependence on θ_k. Multiplying the estimator $\bar{\theta}_k = \partial P / \partial \theta_k$ by the LRM weight in Equation (29) captures the distributional dependence but for the functional dependence we need to differentiate the estimator itself. This gives

$$\frac{\partial \langle \bar{\theta}_k \rangle}{\partial \theta_l} = \int dX \varphi_\theta(X) \Bigg[\Omega_{\theta_k}(X) \frac{\partial P(X)}{\partial \theta_k} + \sum_{i=1}^{d} \frac{\partial P(X)}{\partial X_i} \frac{\partial}{\partial \theta_l} \frac{\partial X_i(\theta)}{\partial \theta_k} \Bigg], \tag{42}$$

so that, recalling that $\bar{\theta}_k = \partial P(X) / \partial \theta_k$, the LRMAAD estimator reads

$$\Gamma_{kl}(X) = \Omega_{\theta_l}(X) \bar{\theta}_k + \sum_{i=1}^{d} \frac{\partial P(X)}{\partial X_i} \frac{\partial}{\partial \theta_l} \frac{\partial X_i(\theta)}{\partial \theta_k}. \tag{43}$$

The second term in the equation above can be implemented by differentiation of the function `PROP_b` in Equation (18). The corresponding adjoint reads,

$$\bar{\theta}^{(2)} = \texttt{PROP\_b\_b}(\theta, Z, \bar{X}, \bar{\bar{\theta}}), \tag{44}$$

and computes for $l = 1, \ldots N_\theta$ the quantity

$$\bar{\theta}^{(2)}_l = \sum_{k=1}^{N_\theta} \frac{\partial}{\partial \theta_l} \Big[\sum_{i=1}^{d} \frac{\partial P(X)}{\partial X_i(\theta)} \frac{\partial X_i(\theta)}{\partial \theta_k} \Big] \bar{\bar{\theta}}_k, \tag{45}$$

at a cost which is, according to Equation (13), a small constant (of order 4) times the cost of computing executing `PROP_b`. By setting the vector $\bar{\bar{\theta}}$ equal to the canonical vector e_l in $\mathbb{R}^{N_\theta}$, for $l = 1, \ldots, N_\theta$, and repeating the execution of function (44) one obtains therefore the second term in Equation (43).

Alternatively, one can compute the AAD derivative of the LRM estimator for the first order sensitivities giving

$$\Gamma_{kl}(X) = \Omega_{\theta_k}(X) \bar{\theta}_l + P(X) \frac{\partial \Omega_{\theta_k}(X)}{\partial \theta_l}, \tag{46}$$

where the first term is identical to the one in Equation (43) and the second can be computed by means of the adjoint of the function implementing Equation (29),

$$\Omega_\theta(X) = \texttt{OMEGA}(Z, \theta, \rho), \tag{47}$$

with $\Omega = (\Omega_{\theta_1}, \ldots, \Omega_{\theta_{N_s}})^T$. Such adjoint is a function of the form

$$\bar{\theta}^{\Omega} = \texttt{OMEGA\_b}(\theta, Z, \rho, \bar{\Omega}), \tag{48}$$

computing, according to the definition (12),

$$\bar{\theta}^{\Omega}_l = \sum_{k=1}^{N_\theta} \frac{\partial \Omega_{\theta_k}}{\partial \theta_l} \bar{\Omega}_{\theta_k}, \tag{49}$$

for $l = 1, \ldots, N_\theta$. By setting the vector $\bar{\Omega} = (\bar{\Omega}_{\theta_1}, \ldots, \bar{\Omega}_{\theta_{N_s}})^t$ equal to the canonical vector e_l in $\mathbb{R}^{N_\theta}$, for $l = 1, \ldots, N_\theta$, times $P(X)$ and repeating execution of the function (48) one obtains therefore the second term in Equation (46), $P(X) \partial \Omega_{\theta_k}(X) / \partial \theta_l$.

While the computational cost of both mixed estimator scales linearly with the number of sensitivities N_θ the computational cost is generally much smaller than computing the finite-difference estimators (40) because only a part of the estimator is differentiated explicitly and because the LRM estimators are generally cheap to compute (Broadie and Glasserman, 1996). This will be illustrated in the following section.

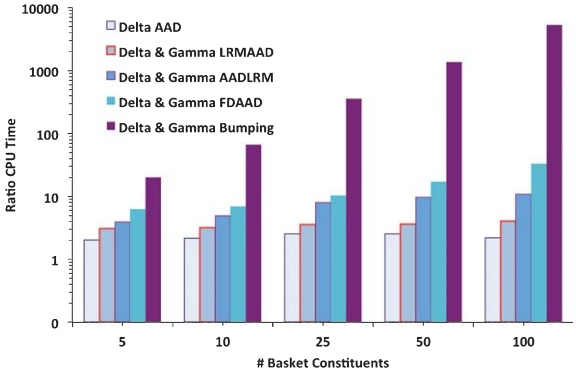

Fig. 1. Cost of computing Delta and Gamma relative to the cost of computing the value for the Basket Option (50) as a function of the number of assets.

7. Numerical Results

As a numerical illustration of the computational efficiency of the method proposed here we consider a European option on a basket of N stocks with (undiscounted) payout given by

$$P(X) = (B(T_M) - K, 0)^+ , \tag{50}$$

where $B(T_M) = \sum_{i=1}^{N} w_i S_i(T_M)$, is the value of the Basket at time T_M, and w_i, $i = 1, \ldots N$, are some fixed weights. The adjoint of this payout is very simple to derive and was given in (Broadie and Glasserman, 1996). The stock prices are assumed to follow a log-normal dynamics and we want to address the problem of computing the first order (Deltas) and second order (Gammas) sensitivities with respect to the spot prices $S_i(t_0)$.

As illustrated in Fig. 6 the mixed AADLRM and LRMAAD estimators are far more efficient than using finite-differences and significatively more efficient than the mixed finite-difference AAD (FDAAD) approach of Equation (40). All the mixed estimators results in over one order of magnitude savings in computation time even for medium sized basket $N \sim 10$. As expected, while the pure finite-difference estimators display a computational complexity scaling as $O(N^2)$, both the AADLRM and the FDAAD approaches scale as $O(N)$. However, the dependence on N for the AADLRM approach is much weaker because only a part of the estimator scales linearly with the number of assets, namely the one arising from the second term in (46). This is generally true also for the LRMAAD estimator. However, for this simple problem the second term in Equation (43) can be computed more efficiently than it is possible in general exploiting the fact that in a lognormal model of the form (31), $\partial^2 S_i(T_m)/\partial S_k(t_0)\partial S_j(t_0)$ is zero unless $i = k = l$. This results in the remarkable outcome that the full Gamma matrix can be computed at a cost that approximately four times the cost of computing the value of the option *irrespective* to the number of underlying assets in the basket.

8. Conclusions

In conclusion we have shown how by combining Adjoint Algorithmic Differentiation (AAD) and the Likelihood Ratio Method (LRM) it is possible to construct efficient second order risk estimators that typically results in orders of magnitude savings in computation time.

Acknowledgments

The author would like to thank Jacky Lee for stimulating discussions and a careful reading of the manuscript. The opinions and views expressed in this paper are uniquely those of the author, and do not necessarily represent those of Credit Suisse Group.

References

Broadie, M., Glasserman, P., 1996. Estimating Security Price Derivatives Using Simulation. In: Management Science 42, pp. 269–285.

Caplan, P., 2011. Numerical Computation of Second Derivatives with Applications to Optimization Problems.

Capriotti, L., 2011. Fast Greeks by Algorithmic Differentiation. In: Journal of Computational Finance 3, 3–35.

Capriotti, L., Giles, M., 2010. Fast Correlation Risk by Adjoint Algorithmic Differentiation. In: Risk 23, 79–85.

Capriotti, L., Giles, M., 2012. Algorithmic Differentiation: Adjoint Greeks Made Easy. In: Risk 25, 92–98.

Capriotti, L., Lee, J., 2014. Adjoint Algorithmic Differentiation: Real-Time Counterparty Credit Risk Management in Monte Carlo Simulation's. In: Counterparty Risk Management. Ed. by Eduardo Canabarro and Michael Pykhtin. Risk Books.

Capriotti, L., 2008. Reducing the variance of likelihood ratio greeks in Monte Carlo. In: Winter Simulation Conference. Ed. by Scott J. Mason et al. IEEE press, pp. 587–593.

Capriotti, L., Shinghoi L., Matthew, P., 2011. Real time counterparty credit risk management in Monte Carlo. In: Risk 24, pp. 86–90.

Glasserman, P., 2004. Monte Carlo Methods in Financial Engineering. New York: Springer.

Griewank, A., Walther, A., 2008. Evaluating derivatives: principles and techniques of algorithmic differentiation (second edition). SIAM.

Hull, J.C., 2002. Options, Futures and Other Derivatives. New Jersey: Prentice Hall.

Karatzas, I., Shreve, S.E., 1988. Brownian motion and stochastic calculus. Graduate texts in mathematics. Springer-Verlag.