FAST MONTE-CARLO GREEKS FOR FINANCIAL PRODUCTS WITH DISCONTINUOUS PAY-OFFS

JIUN HONG CHAN AND MARK JOSHI

ABSTRACT. We introduce a new class of numerical schemes for discretizing processes driven by Brownian motions. These allow the rapid computation of sensitivities of discontinuous integrals using pathwise methods even when the underlying densities post-discretization are singular. The two new methods presented in this paper allow Greeks for financial products with trigger features to be computed in the LIBOR market model with similar speed to that obtained by using the adjoint method for continuous pay-offs. The methods are generic with the main constraint being that the discontinuities at each step must be determined by a one-dimensional function: the proxy constraint. They are also generic with the sole interaction between the integrand and the scheme being the specification of this constraint.

1. Introduction

Whilst Monte Carlo simulation has long been a standard technique for the evaluation of high-dimensional integrals, there are still many subtleties to be explored. One particular issue of importance in financial and engineering applications is computing the sensitivities of such integrals to parameters. Much progress has been made on that question with the two most popular methods being the pathwise and likelihood ratio methods of Broadie and Glasserman (1996).

If we write our integral in the form

$$\int g(x,\theta)\Phi(x,\phi)dx,$$

where g is the integrand and Φ is a probability density, then the pathwise method relies on the differentiability of g (in fact, Lipschitz continuity is enough,) whereas the likelihood ratio method relies on the non-singularity of Φ. Generally by performing a change of variables one can shift the parameter dependence into either g or Φ and thus provided one of the two is well-behaved, the sensitivity can be calculated.

Giles and Glasserman (2006) showed that for the low-factor LIBOR market model it was possible to compute all the sensitivities with a computational complexity of number of rates times number of factors per step, provided the pay-off was Lipschitz continuous.

However, there are common financial applications with strong pay-off discontinuities, and the density is singular. In particular, when pricing a financial product which triggers if a rate crosses a target level, such as a target redemption note (TARN), using a low-factor LIBOR market model, g has strong discontinuities and, after discretization, Φ is supported on a low-dimensional submanifold which will depend on the parameters and the values of forward rates at the previous time step. Whilst one can extend the pathwise method to discontinuous pay-offs by using delta functions, Joshi–Kainth (2004), this approach is too product and model specific to resolve the problem satisfactorily.

We therefore present two new methods for sensitivity computation with several desirable features.

Date: November 4, 2009.

Key words and phrases. Price Sensitivities, Monte-Carlo Greeks, Partial Proxy Simulation Scheme, Minimal Partial Proxy Simulation Scheme, Pathwise Method, Trigger Product, Discontinuous Pay-off, Digital Option, Target Redemption Note, LIBOR Market Model.

- All sensitivities can be computed simultaneously in one simulation, with complexity equal to the number of factors times the number of rates per step.
- The integrand can be discontinuous across a hypersurface specified as the level set of a one-dimensional function determined at the previous time step.
- They apply to low-factor models for which the underlying density is singular.
- They are generic in that the scheme is determined by the one-dimensional proxy-constraint rather than the integrand.
- The variance of the resulting simulations are low.

The first of our two methods, the *pathwise partial proxy* (pathwise PP) method has slightly more general applicability in that the singularities can lie across multiple level sets of the proxy constraint, and also sometimes allows the avoidance of differentiating the pay-off which can yield more rapid implementation. The second method, the *pathwise minimal partial proxy* (pathwise MPP) method, is not quite as general and generic but has the virtue of even lower variances.

These two schemes can be viewed as limits of the partial proxy simulation scheme of Fries and Joshi (2008b) and the minimal partial proxy simulation scheme of Chan and Joshi (2009). Those schemes whilst effective had the disadvantage of requiring a simulation per sensitivity and therefore were not comparatively rapid when many sensitivities were required. The essential features of those schemes were to use a low-dimensional measure change to avoid the singularities of the integrand; the minimal partial proxy simulation scheme chose the measure change to minimize variance whereas the original scheme did so to fix the proxy constraint function. Our new schemes can therefore be viewed as using the pathwise method in the non-singular directions, and the likelihood ratio method in the singular one. Whilst our motivations are financial, our techniques are not and we believe that the approach will have wide applicability to many cases where the sensitivities of a discontinuous integral must be evaluated by Monte Carlo simulation.

The presentation of the minimal partial proxy scheme in Chan and Joshi (2009) was restricted to linear proxy constraint functions. Here before passing to the limit, we show how it can be generalized to work with non-linear proxy constraints. Our limiting method therefore also works with non-linear proxy constraints. In financial applications, the proxy constraints will typically be on the value of a single swap or forward rate at each time step, so these cases can be implemented once and for all. The adaptation to a new product then simply involves choosing the rate for each step.

We note that various papers have been written on the computation of sensitivities in the LMM. Fries and Kampen (2007) studied proxy simulation methods but their techniques required a full-factor model to avoid singular densities. Piterbarg (2004) and Fries and Joshi (2008a) use conditional analyticity to remove the singularity but the techniques still require one simulation per sensitivity and a great deal of handcrafting. Giles (2008) suggests another approach using the likelihood ratio method across the time-step before a cash-flow but this again necessitates the use of a full-factor model. The use of a low-factor model is desirable in that the computational complexity of the LMM is proportional to the number of factors per step, and such models are typically used by practitioners.

The paper is organized as follow: in section 2, we outline the numerical schemes and the notations used in this paper. In section 3, we introduce a class of numerical schemes known as the quasi mean-shifted proxy schemes. A brief review of the partial proxy simulation scheme is also presented together with the generalized version of the minimal partial proxy simulation scheme. Section 4 shows that under the quasi mean-shifted proxy schemes, we can interchange the expectation and differentiation when computing price sensitivities. We derive the pathwise derivatives for the partial proxy simulation scheme and the minimal partial proxy simulation scheme in section 5. In section 6 and 7, we will demonstrate how pathwise derivatives for both the partial proxy simulation schemes and the minimal partial proxy simulation schemes can be evaluated using a naive method and the adjoint method. Section 8 shows that, under the LIBOR Market Model, all deltas and vegas can be evaluated with a computational order proportional to the number of rates times the number of factors at each step of the simulation using the pathwise partial proxy method and the pathwise minimal partial proxy method. In section 9, a brief discussion of the products and model used for our numerical tests is provided with numerical results presented in section 10. Section 11 shows that the pathwise partial proxy method can be used to compute price sensitivities without having to differentiate the pay-off function for certain classes of pay-off function. We conclude in section 12.

2. Numerical Scheme and Greeks

2.1. **Numerical Scheme Specification.** Suppose that the underlying quantities of a financial product after a change of coordinates (e.g in log-coordinates), $K = (K_1, K_2, \ldots, K_n)^{\mathrm{T}}$, satisfy the following stochastic differential equation (SDE),

$$dK(t) = \mu\left(K, \theta, t\right) dt + \Sigma\left(K, \theta, t\right) dW(t), \tag{2.1}$$

where θ denotes a parametric vector of initial inputs, $\mu(K, \theta, t)$ is an n-dimensional column vector, $W(t)$ is an n-dimensional column vector of correlated Brownian motions and $\Sigma(K, \theta, t)$ is an $n \times n$ diagonal matrix.

Given a vector of inputs θ, we define $K_i^G(\theta)$ to be the discretization of K at time t_i under a numerical scheme G. We also have the j^{th} element of the discretization of K at time t_i given by $K_{i,j}^G(\theta)$. Hence, under an Euler discretization scheme, E, we have

$$K_{i+1}^E(\theta) = K_i^E(\theta) + \mu(K_i^E(\theta), \theta, t_i)\Delta t_i + A(K_i^E(\theta), \theta, t_i)Z_{i+1}, \tag{2.2}$$

where Z_{i+1} is a d-dimensional vector of independent standard normal random variables and $\Delta t_i = t_{i+1} - t_i$. The pseudo-square root

$$A(K_i^E(\theta), \theta, t_i) = \left(a_{jl}(K_i^E(\theta), \theta, t_i)\right)$$

is an $n \times d$ matrix satisfying

$$A(K_i^E(\theta), \theta, t_i)A(K_i^E(\theta), \theta, t_i)^{\mathrm{T}} = C(K_i^E(\theta), \theta, t_i) \tag{2.3}$$

where $C(K_i^E(\theta), \theta, t_i)$ is the $n \times n$ covariance matrix of $K_i^E(\theta)$ from time t_i to t_{i+1}. In general, the matrix $A(K_i^E(\theta), \theta, t_i)$ is not unique and there are many different pseudo-square roots that satisfy equation (2.3) (see Joshi, 2003a). Under a reduced-factor model, we also have $d < n$ to reduce the computational time (see Joshi, 2003b). For simplicity, equation (2.2) will be written as

$$K_{i+1}^E(\theta) = F_i(K_i^E(\theta), Z_{i+1}, \theta) \tag{2.4}$$

where F_i is a smooth mapping function. The discretized process $K_i^E(\theta)$ is said to be adapted to $\mathcal{F}_{t_i}$ where $\mathcal{F}_{t_i}$ is a filtration generated by $Z_1, Z_2, \ldots, Z_i$ i.e.

$$\mathcal{F}_{t_i} := \sigma(Z_1, Z_2, \ldots, Z_i).$$

2.2. **Monte-Carlo Greeks -The Bump and Revalue Approach.** When computing Greeks, we will have a vector of base inputs θ_0 and we wish to find the change in value arising from a shift to a vector of perturbed inputs θ_B. We define E^A to be the Euler scheme E with a vector of initial input θ_A. Therefore, under E^0, we have

$$K_{i+1}^E(\theta_0) \quad = \quad F_i(K_i^E(\theta_0), Z_{i+1}, \theta_0). \tag{2.5}$$

with the initial state variables given by $K_0^E(\theta_0)$. Similarly, under E^B, we have

$$K_{i+1}^E(\theta_B) \quad = \quad F_i(K_i^E(\theta_B), Z_{i+1}, \theta_B). \tag{2.6}$$

with the initial state variables given by $K_0^E(\theta_B)$.

Under the usual bump-and-revalue approach, the Monte-Carlo Greeks are calculated by applying finite differences to prices obtained using $K^E(\theta_0)$ and $K^E(\theta_B)$ where both are generated using a common Z. Although, in general, this method works well for financial products with continuous pay-offs, the Monte-Carlo Greeks of financial products with digital-type pay-offs calculated using such an approach can have high variances due to the pathwise discontinuities (For example, see Fries and Joshi (2008b)).

3. Quasi Mean-Shifted Proxy Simulation Schemes

3.1. **Quasi Mean-Shifted Proxy Simulation Schemes.**

Definition 1. *A numerical scheme Q is said to belong to the class of quasi mean-shifted proxy simulation schemes if Q has the following properties*

$$\begin{aligned} K_0^Q(\theta) &= K_0^E(\theta) \\ K_{i+1}^Q(\theta) &= F_i(K_i^Q(\theta), Z_{i+1} - \nu_{i+1}^Q(\theta, Z_{i+1}), \theta), \end{aligned} \tag{3.1}$$

where F_i is the same mapping function defined in (2.4).

The quasi mean-shifted proxy simulation schemes can be viewed as measure changes performed on Z such that in this new measure, Z has a drift of $-\nu^Q$. In order to produce the same expectation as in the original measure, we have to compensate the measure changes with a likelihood ratio term (i.e. Monte-Carlo weight or Radon-Nikodym derivative).

Since we are now considering a post-discretization measure change, we are no longer constrained by Girsanov's theorem and our restrictions on measure changes are now weaker. Whilst the conventional mean-shift requires ν_{i+1}^Q to be $\mathcal{F}_{t_i}$-measurable, here, ν_{i+1}^Q is allowed to depend on the realization of Z_{i+1} (i.e $\mathcal{F}_{t_{i+1}}$-measurable). In other words, the solution for ν_{i+1}^Q can be selected accordingly based on the outcome of Z_{i+1} and this provides us with a great deal of flexibility on how measure changes can be performed. Our ability to depart from the conventional mean-shift comes from the fact that the Monte-Carlo weight for such $\mathcal{F}_{t_{i+1}}$-measurable mean-shifts can be derived easily from the discretization scheme (see Fries and Joshi, 2008b).

Proposition 1. *Under the quasi mean-shifted proxy simulation schemes, the Monte-Carlo weight from t_i to t_{i+1} is given by*

$$w_{i+1}^Q(\theta) = \left| \det \left(\frac{\partial \tilde{Z}_{i+1}}{\partial Z_{i+1}} \right) \right| \frac{\phi^*(\tilde{Z}_{i+1})}{\phi^*(Z_{i+1})} \tag{3.2}$$

where

$$\tilde{Z}_{i+1} = Z_{i+1} - \nu_{i+1}^Q(\theta, Z_{i+1})$$

and ϕ^* is the density function of the joint normal distributions.

Proof. Suppose that $\tilde{Z} = Z - \nu(Z)$ and $g : \mathbb{R}^d \to \mathbb{R}$, we have

$$\begin{aligned} & \int_{\mathbb{R}^d} g(\tilde{Z})\phi^*(\tilde{Z})d\tilde{Z}, \\ = & \int_{\mathbb{R}^d} g(\tilde{Z})\phi^*(\tilde{Z}) \left| \det \left(\frac{\partial \tilde{Z}}{\partial Z} \right) \right| dZ, \\ = & \int_{\mathbb{R}^d} \left| \det \left(\frac{\partial \tilde{Z}}{\partial Z} \right) \right| \frac{\phi^*(\tilde{Z})}{\phi^*(Z)} g(\tilde{Z})\phi^*(Z)dZ. \end{aligned}$$

□

As Monte-Carlo simulations usually involve multiple steps, expectations obtained using the quasi mean-shifted proxy simulation schemes must be weighted by the accumulated Monte-Carlo weight (i.e $\prod w_i^Q(\theta)$).

3.1.1. *Computation of Monte-Carlo Greeks using Quasi Mean-Shifted Proxy Simulation Schemes.* When computing Monte-Carlo Greeks, instead of simulating $K^E(\theta_B)$ directly from E^B, we can evolve $K^Q(\theta_B)$ using Q^B (i.e Q with a vector of perturbed inputs θ_B) given by

$$\begin{aligned} K_0^Q(\theta_B) &= K_0^E(\theta_B), \\ K_{i+1}^Q(\theta_B) &= F_i(K_i^Q(\theta_B), Z_{i+1} - \nu_{i+1}^Q(\theta_B, Z_{i+1}), \theta_B). \end{aligned} \tag{3.3}$$

Specifically, we evolve the processes $K^Q(\theta_B)$ and $K^E(\theta_0)$ simultaneously using a common Z. At each step of the simulation, the value of $\nu_{i+1}^Q(\theta_B, Z_{i+1})$ is chosen so that the realizations of $K_{i+1}^Q(\theta_B)$ and $K_{i+1}^E(\theta_0)$ will not result in pathwise discontinuities. As pathwise discontinuities are eliminated, the Monte-Carlo Greeks obtained by applying finite differences to the weighted prices from Q^B and E^0 will have much lower variance and stable Monte-Carlo Greeks can be obtained. Note that, an important observation is that, by construction, pricing under Q^0 (i.e Q with a vector of base inputs θ_0) is exactly the same as pricing under E^0 i.e.

$$K_i^Q(\theta_0) = K_i^E(\theta_0)$$

with $w_i^Q(\theta_0) = 1$ for all i as no measure change is required.

A number of specific quasi mean-shifted proxy simulation schemes have been introduced for the purpose of Greek computations. These include the full proxy scheme (Fries and Kampen, 2007), the partial proxy scheme (Fries and Joshi, 2008b), the localized partial proxy scheme (Fries, 2007a) and the minimal partial proxy simulation scheme (Chan and Joshi, 2009). In general, all these schemes have the same properties described above. They only differ in how the solution for the mean-shift $\nu_i^Q(\theta_B, Z_i)$ is selected. In this paper, we focus on the partial proxy simulation scheme, P, and the minimal partial proxy simulation scheme, M.

3.2. **The Partial Proxy Simulation Scheme.** In this section, we will replace Q with P to indicate that we are working with the partial proxy simulation scheme. The partial proxy simulation scheme, P, was introduced by Fries and Joshi (2008b). They define a proxy constraint function, p_i, [1]

$$p_i : \mathbb{R}^n \to \mathbb{R},$$

where p_i represents the quantity that will give raise to pathwise discontinuities at t_i. Typically in financial applications, the proxy constraint function, p_i, will be either a swap-rate or a forward-rate on reset.

In order to prevent pathwise discontinuities, the scheme P^B (i.e P with a vector of perturbed inputs θ_B) is constructed by selecting $\nu_i^P(\theta_B, Z_i)$ such that for any given path, we have

$$p_i(K_i^P(\theta_B)) = p_i(K_i^E(\theta_0)) \tag{3.4}$$

for each i. This ensures that the quantity that gives rise to pathwise discontinuities are the same under P^B and E^0 at each t_i and hence, pathwise discontinuities are eliminated.

In general, equation (3.4) alone will not uniquely determine the solution for $\nu_i^P(\theta_B, Z_i)$ as $\nu_i^P(\theta_B, Z_i)$ is a d-dimensional vector. One possible solution suggested by Fries and Joshi is that they restrict the solution of $\nu_i^P(\theta_B, Z_i)$ to the following form:

$$\nu_{i,s}^P(\theta_B, Z_i) = 0 \quad \text{for} \quad s = 2, 3, \ldots, d$$

[1] Here, we focus on the case where $p_i : \mathbb{R}^n \to \mathbb{R}$. In Fries and Joshi (2008b), they defined p_i to be $p_i : \mathbb{R}^n \to \mathbb{R}^k$ with $k < n$.

where the index s denotes the s^{th} element in $\nu_i^P(\theta_B, Z_i)$. This makes solving for $\nu_{i,1}^P(\theta_B, Z_i)$ straight-forward. As the mean-shift is applied only to $Z_{i,1}$ (i.e. the first factor), we require that

$$\frac{\partial p_i(K_i^P(\theta_B))}{\partial Z_{i,1}} = \frac{\partial p_i(K_i^P(\theta_B))}{\partial K_i^P(\theta_B)} \frac{\partial K_i^P(\theta_B)}{\partial Z_{i,1}} \neq 0, \tag{3.5}$$

where

$$\frac{\partial K_i^P(\theta_B)}{\partial Z_{i,1}} = \left(a_{11}(K_{i-1}^P(\theta_B), \theta_B, t_{i-1}), \ldots, a_{n1}(K_{i-1}^P(\theta_B), \theta_B, t_{i-1})\right)^{\mathrm{T}},$$

and

$$\frac{\partial p_i(K_i^P(\theta_B))}{\partial K_i^P(\theta_B)} = \left(\frac{\partial p_i}{\partial K_{i,1}^P}, \frac{\partial p_i}{\partial K_{i,2}^P}, \ldots, \frac{\partial p_i}{\partial K_{i,n}^P}\right) \neq \mathbf{0}.$$

Since many different pseudo-square roots exist and

$$\frac{\partial K_i^P(\theta_B)}{\partial Z_{i,1}}$$

depends on the first column of $A(K_{i-1}^E(\theta_B), \theta_B, t_{i-1})$, satisfying the requirement (3.5) is trivial as long as we select the pseudo-square root for the simulation sensibly. In general, the variance of the Monte-Carlo Greeks obtained under the partial proxy simulation scheme can be further reduced by using $A(K_{i-1}^E(\theta_B), \theta_B, t_{i-1})$ such that

$$\left|\frac{\partial p_i(K_i^P(\theta_B))}{\partial Z_{i,1}}\right|$$

is maximized.

Under the setup above, the Monte-Carlo weight, $w_i^P(\theta_B)$, is given by

$$w_i^P(\theta_B) = \left(1 - \frac{\partial \nu_{i,1}^P(\theta_B, Z_i)}{\partial Z_{i,1}}\right) \exp\left(Z_{i,1}\nu_{i,1}^P(\theta_B, Z_i) - \frac{1}{2}\nu_{i,1}^P(\theta_B, Z_i)^2\right), \tag{3.6}$$

from Proposition 1. As all the $Z_{i,s}$'s are independent and based on the restriction imposed on the solution for $\nu_i^P(\theta_B, Z_i)$, we clearly have

$$\det\left(\frac{\partial \tilde{Z}_i}{\partial Z_i}\right) = \left(1 - \frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right),$$

and

$$\begin{aligned} \frac{\phi^*(\tilde{Z}_i)}{\phi^*(Z_i)} &= \prod_{l=1}^{F} \frac{\phi(\tilde{Z}_{i,l})}{\phi(Z_{i,l})} \\ &= \frac{\phi(\tilde{Z}_{i,1})}{\phi(Z_{i,1})} \end{aligned}$$

where ϕ is the density function of the normal distribution.

3.3. **The Minimal Partial Proxy Simulation Scheme.** The minimal partial proxy simulation scheme was introduced by Chan and Joshi (2009). The essential idea of the minimal partial proxy simulation scheme is that the measure change at each time step is selected optimally such that it minimises the variance of the Monte-Carlo weights. This idea is driven by the fact that a naive measure change may result in an unstable Monte-Carlo weight (i.e sensitive to the outcome of Z) and hence increase the variance of the Monte-Carlo Greeks (Glasserman, 2003). In order to do so, a weaker constraint is made. In particular, we no longer select the proxy constraint to make the perturbed rate equal to the unperturbed rate at the end of the step, but instead make the weaker requirement that both rates are on the same side of the discontinuity. This yields an extra degree of freedom and so we are able to selected the measure change which minimizes the variance of the Monte-Carlo weight across the step.

The minimal partial proxy simulation scheme presented by Chan and Joshi (2009) focuses on a special case of the proxy constraint functions: a linear proxy constraint function. Here, we will present a generalized version of the minimal partial proxy simulation scheme. In particular, we extend the ideas from Chan and Joshi (2009) to cases with non-linear proxy constraint functions, and hence, the minimal partial proxy simulation scheme can now be applied to a larger class of financial products.

3.3.1. *Financial Product Specification.* Consider a financial product which depends on the underlying quantities K at $t_1, t_2, \ldots, t_m$ and pays a sequence of cash-flows C_i at time t_i. We assume that C_i is $\mathcal{F}_{t_i}$-measurable and is a continuous function of the inputs. We shall say that the product triggers at time t_i if some event E_i occurs such that a rebate R_i ($\mathcal{F}_{t_i}$-measurable) is received and no further cash flows are received.

Under the generalized minimal partial proxy simulation scheme, we require E_i to be defined as [2]

$$p_i(K_i(\theta)) > H_i(\theta),$$

for some $\mathcal{F}_{t_{i-1}}$-measurable trigger level, $H_i(\theta)$, and p_i is the proxy constraint function. Although, this is a restrictive requirement on how the event E_i has to be defined, many path-dependent trigger products can be rewritten in the above form as $H_i(\theta)$ is an $\mathcal{F}_{t_{i-1}}$-measurable function. We will also consider more general products such as autocaps where the product can trigger several times and on triggering the product becomes another product of the same type but with one fewer trigger left (e.g the rebate is another autocap with one fewer trigger left).

3.3.2. *The Minimal Partial Proxy Simulation Scheme.* We let M be the minimal partial proxy simulation scheme. The essential difference from the partial proxy simulation scheme is that here, we pick the mean-shift in an optimal way. In particular, given a vector of perturbed inputs θ_B, we restrict the solution of $\nu_i^M(\theta_B, Z_i)$ to the following form

$$\begin{aligned}\nu_{i,1}^M(\theta_B, Z_i) &= (1-\alpha_{i,1}^M(\theta_B))Z_{i,1} - \beta_{i,1}^M(\theta_B)\\ \nu_{i,s}^M(\theta_B, Z_i) &= 0\end{aligned}$$

for $s = 2, \ldots, d$ with $\alpha_{i,1}^M(\theta_B)$ and $\beta_{i,1}^M(\theta_B)$ independent of $Z_{i,1}$. The numerical scheme, M^B, is then constructed by selecting $\nu_{i,1}^M(\theta_B, Z_i)$ such that the following 2 conditions are satisfied,

(1) $p_i(K_i^E(\theta_0)) > H_i^E(\theta_0) \iff p_i(K_i^M(\theta_B)) > H_i^M(\theta_B)$,

(2) minimises the variance of the Monte-Carlo weight across that step.

Similar to the partial proxy simulation scheme, we also require

$$\frac{\partial p_i(K_i^M(\theta_B))}{\partial Z_{i,1}} \neq 0$$

[2]For cases where E_i is defined to be $p_i < H_i$, all the following results will still hold by taking the negative of p_i and $H(t_i)$.

and the Monte-Carlo weight is given by

$$w_i^M(\theta_B) = \alpha_{i,1}^M(\theta_B) \exp\left(Z_{i,1}\nu_{i,1}^M(\theta_B, Z_i) - \frac{1}{2}\nu_{i,1}^M(\theta_B, Z_i)^2 \right), \tag{3.7}$$

from Proposition 1.

3.3.3. *Eliminating Pathwise Discontinuities.* Condition (1) ensures that pathwise discontinuities are eliminated at t_i. However, solving for $\nu_{i,1}^M(\theta_B)$ (i.e $\alpha_{i,1}^M(\theta_B, Z_i)$ and $\beta_{i,1}^M(\theta_B)$) such that condition (1) holds can be complicated as at each time step, both numerical schemes (i.e M^B and E^0) are driven by d standard normal random variables.

Here, we will assume that we observe the realization of

$$Z_{i,s} \quad \text{for} \quad s = 2, 3 \ldots, d$$

before $Z_{i,1}$. The advantage of making such assumption is that once we know the realization of $Z_{i,s}$ for $s = 2, 3 \ldots, d$, both

$$p_i(K_i^E(\theta_0)) \quad \text{and} \quad p_i(K_i^M(\theta_B))$$

in condition (1) will only depend on $Z_{i,1}$. We will also assume that the proxy constraint function, p_i, is a monotone increasing function of $Z_{i,1}$ [3].

Definition 2. *We define $Z_{i,1}^*$ to be the critical point such that*

$$p_i(K_i^E(\theta_0)) > H_i^E(\theta_0) \iff Z_{i,1} > Z_{i,1}^*.$$

In other words, the financial product priced under E^0 will trigger if and only if $Z_{i,1} > Z_{i,1}^*$

Proposition 2. *The solution for $Z_{i,1}^*$ must satisfy the following equation*

$$p_i(\hat{K}_i^E(\theta_0)) = H_i^E(\theta_0), \tag{3.8}$$

where

$$\hat{K}_i^E(\theta_0) := F_{i-1}(K_{i-1}^E(\theta_0), \hat{Z}_i, \theta_0) \tag{3.9}$$

with the elements of $\hat{Z}_i$ given by

$$\begin{aligned} \hat{Z}_{i,1} &= Z_{i,1}^* \\ \hat{Z}_{i,s} &= Z_{i,s} \quad \textit{for} \quad s = 2, \ldots, d. \end{aligned}$$

Proof. Based on the definition of $Z_{i,1}^*$ the financial product priced under E^0 will trigger if and only if $Z_{i,1} > Z_{i,1}^*$. By setting $Z_{i,1} = Z_{i,1}^*$, we must therefore have $p_i(K_i^E(\theta_0)) = H_i^E(\theta_0)$. Hence the result in Proposition 2 clearly holds based on the definition of $\hat{K}_i^E(\theta_0)$ in equation (3.9). □

In general, once the exact form of the proxy constraint function, p_i, is known, solving for $Z_{i,1}^*$ is straight-forward as $p_i(K_i^E(\theta_0))$ is a function of $Z_{i,1}$ and the value of $Z_{i,1}$ which results in

$$p_i(K_i^E(\theta_0)) = H_i^E(\theta_0)$$

will be the critical point $Z_{i,1}^*$. For cases where the proxy constraint function p_i is non-linear, a numerical root search is usually required to solve for $Z_{i,1}^*$. Summarizing,

[3]For cases where the proxy constraint function, p_i, is a monotone decreasing function of $Z_{i,1}$, the same final result can be obtained using the same argument presented in this section.

Proposition 3. *Based on the assumptions above, condition (1) is satisfied as long as the solutions for $\alpha^M_{i,1}(\theta_B)$ and $\beta^M_{i,1}(\theta_B)$ are chosen such that the following equation holds*

$$p_i(\hat{K}^M_i(\theta_B)) = H^M_i(\theta_B), \tag{3.10}$$

where

$$\hat{K}^M_i(\theta_B) := F_{i-1}(K^M_{i-1}(\theta_B), \hat{Z}_i - \nu^M_i(\theta_B, \hat{Z}_i), \theta_B) \tag{3.11}$$

with the elements of $\hat{Z}_i$ given by

$$\begin{aligned} \hat{Z}_{i,1} &= Z^*_{i,1} \\ \hat{Z}_{i,s} &= Z_{i,s} \quad \textit{for} \quad s = 2, \dots, d \end{aligned}$$

and

$$\begin{aligned} \nu^M_{i,1}(\theta_B, \hat{Z}_i) &= (1 - \alpha^M_{i,1}(\theta_B)) Z^*_{i,1} - \beta^M_{i,1}(\theta_B), \\ \nu^M_{i,s}(\theta_B, \hat{Z}_i) &= 0 \quad \textit{for} \quad s = 2, \dots, d. \end{aligned}$$

3.3.4. *Linearization of Proxy Constraints.* For non-linear proxy constraint function, p_i, the numerical implementation of the minimal partial proxy simulation scheme can be expensive. This is because at each time step, we must solve for $Z^*_{i,1}$ such that equation (3.8) holds and both $\alpha^M_{i,1}(\theta_B)$ and $\beta^M_{i,1}(\theta_B)$ such that equation (3.10) is satisfied. For efficient practical application, we may linearize the proxy constraint function in equation (3.8) and (3.10). We define

$$\tilde{K}^E_i(\theta_0) := F_{i-1}(K^E_{i-1}(\theta_0), \tilde{Z}_i, \theta_0), \tag{3.12}$$

$$\bar{K}^M_i(\theta_B) := F_{i-1}(K^M_{i-1}(\theta_B), \tilde{Z}_i, \theta_B), \tag{3.13}$$

with the elements in $\tilde{Z}_i$ given by

$$\begin{aligned} \tilde{Z}_{i,1} &= 0 \\ \tilde{Z}_{i,s} &= Z_{i,s} \quad \text{for} \quad s = 2, \dots, d. \end{aligned}$$

Since the mapping function, F_{i-1}, is linear in Z, equation (3.12) and (3.13) can be written as

$$\tilde{K}^E_i(\theta_0) = \hat{K}^E_i(\theta_0) - Z^*_{i,1} S_{i-1}(K^E_{i-1}(\theta_0), \theta_0) \tag{3.14}$$

$$\bar{K}^M_i(\theta_B) = \hat{K}^M_i(\theta_B) - \left(\alpha^M_{i,1}(\theta_B) Z^*_{i,1} + \beta^M_{i,1}(\theta_B)\right) S_{i-1}(K^M_{i-1}(\theta_B), \theta_B) \tag{3.15}$$

where

$$S_i(K_i(\theta), \theta) := (a_{11}(K_i(\theta), \theta, t_i), \dots, a_{n1}(K_i(\theta), \theta, t_i))^{\mathrm{T}}.$$

Proposition 4. *By linearizing the proxy constraint function in equation (3.8) around $\tilde{K}^E_i(\theta_0)$, the approximation for $Z^*_{i,1}$ is given by*

$$\frac{H^E_i(\theta_0) - p_i(\tilde{K}^E_i(\theta_0))}{\nabla p_i(\tilde{K}^E_i(\theta_0)) \cdot S_{i-1}(K^E_{i-1}(\theta_0), \theta_0)}. \tag{3.16}$$

assuming that

$$\nabla p_i(\tilde{K}^E_i(\theta_0)) \cdot S_{i-1}(K^E_{i-1}(\theta_0), \theta_0) \neq 0.$$

Proof. We have

$$p_i(\tilde{K}_i^E(\theta_0)) + \nabla p_i(\tilde{K}_i^E(\theta_0)) \left(\hat{K}_i^E(\theta_0) - \tilde{K}_i^E(\theta_0)\right) = H_i^E(\theta_0).$$

by linearizing equation (3.8) around $\tilde{K}_i^E(\theta_0)$ and by substituting (3.14) into the equation above, the result in Proposition 4 clearly holds □

Proposition 5. *By linearizing the proxy constraint function in equation (3.10) around $\hat{K}_i^E(\theta_0)$, the solutions of $\alpha_{i,1}^M(\theta_B)$ and $\beta_{i,1}^M(\theta_B)$ must satisfy*

$$\beta_{i,1}^M(\theta_B) = X_{i,1}^M(\theta_B) - \alpha_{i,1}^M(\theta_B) Z_{i,1}^* \tag{3.17}$$

to prevent pathwise discontinuities, where

$$X_{i,1}^M(\theta_B) = \frac{H_i^M(\theta_B) - p_i(\hat{K}_i^E(\theta_0)) - \nabla p_i(\hat{K}_i^E(\theta_0)) \cdot (\bar{K}_i^M(\theta_B) - \hat{K}_i^E(\theta_0))}{\nabla p_i(\hat{K}_i^E(\theta_0)) \cdot S_{i-1}(K_{i-1}^M(\theta_B), \theta_B)}, \tag{3.18}$$

again assuming that

$$\nabla p_i(\hat{K}_i^E(\theta_0)) \cdot S_{i-1}(K_{i-1}^M(\theta_B), \theta_B) \neq 0.$$

Proof. By linearizing equation (3.10) around $\hat{K}_i^E(\theta_0)$ and we have

$$p_i(\hat{K}_i^E(\theta_0)) + \nabla p_i(\hat{K}_i^E(\theta_0)) \cdot (\hat{K}_i^M(\theta_B) - \hat{K}_i^E(\theta_0)) = H_i^M(\theta_B). \tag{3.19}$$

By rearranging (3.19) and using the result from (3.15), the result in Proposition 5 clearly holds. □

Note that, equation (3.17) can also be written as

$$X_{i,1}^M(\theta_B) = \alpha_{i,1}^M(\theta_B) Z_{i,1}^* + \beta_{i,1}^M(\theta_B). \tag{3.20}$$

Observe that, once we have linearized the proxy constraint function, condition (1) will not longer hold with probability of 1. That is, we might still have pathwise discontinuities. However, the probability of such events occurring is close to zero for 2 reasons. The first reason is that we have the flexibility to select the proxy constraint function. As long as the proxy constraint function is selected such that the effect of higher order derivatives is insignificant compared to the first order derivative, then our approximation for $Z_{i,1}^*$ using equation (3.16) will be adequate. For example, instead of using a CMS swap-rate as our proxy constraint function, we can use the log of a CMS swap-rate. The second reason is that the perturbed inputs θ_B are, in general, close to the base inputs θ_0. Hence we will also have that $\hat{K}_i^M(\theta_B)$ is close to $\hat{K}_i^E(\theta_0)$. Therefore, approximating $p_i(\hat{K}_i^M(\theta_B))$ in equation (3.10) up to the first order term at $\hat{K}_i^E(\theta_0)$ will be sufficiently good.

3.3.5. *Minimising the Variance of the Monte-Carlo Weight.* Based on the linearized proxy constraint function, solutions for $\alpha_{i,1}^M(\theta_B)$ and $\beta_{i,1}^M(\theta_B)$ are selected dynamically to ensure that the condition (2) holds. Observe that, if the probability of the product triggering at next time step is close to 0 or 1, the trigger effectively does not exist and there are no pathwise discontinuities. Under such scenarios, no measure change needs to be performed as performing a measure change will generally increase the variance of the Monte Carlo Greeks. Therefore, whenever $|Z_{i,1}^*|$ is greater than 4 (i.e four standard deviations[4]), we set $\alpha_{i,1}^M(\theta_B) = 1$ and $\beta_{i,1}^M(\theta_B) = 0$ (i.e no measure change) and we have zero variance for the Monte-Carlo weight.

[4] In Chan and Joshi (2009), they suggested that no measure change is required if we are 6 standard deviations away from triggering or not triggering. However, based on further numerical tests, we conclude that, by using 4 standard deviations instead of 6, the variance of the Monte-Carlo Greeks can be further reduced without resulting pathwise discontinuities.

Otherwise, $\alpha^M_{i,1}(\theta_B)$ is selected such that it minimises the variance of the Monte-Carlo weight and the solution for $\beta^M_{i,1}(\theta_B)$ is determined based on the equation (3.17) to prevent pathwise discontinuities. It turns out that $\alpha^M_{i,1}(\theta_B)$ which minimizes the variance of the Monte-Carlo weight satisfies the following equation [5] (see Chan and Joshi, 2009)

$$a\alpha^4 + b\alpha^3 + c\alpha^2 + d\alpha + e = 0, \tag{3.21}$$

where

$$\begin{aligned} \alpha &= \alpha^M_{i,1}(\theta_B) \\ a &= 2, \\ b &= 2 \cdot X^M_{i,1}(\theta_B) Z^*_{i,1}, \\ c &= -(3 + 2X^M_{i,1}(\theta_B)^2 + {Z^*_{i,1}}^2), \\ d &= X^M_{i,1}(\theta_B) Z^*_{i,1}, \\ e &= 1, \end{aligned} \tag{3.22}$$

and $X^M_{i,1}(\theta_B)$ is from equation (3.18). Four distinct real roots always exist for equation (3.21), only one real root is greater than $\frac{1}{\sqrt{2}}$ and this root is the solution for $\alpha^M_{i,1}(\theta_B)$ which minimises the variance of the Monte-Carlo weight. The derivations and the proofs of these results were given by Chan and Joshi (2009). As closed form solutions exist for the roots of a 4^{th} order polynomial, there is no need to perform a numerical root search, hence adopting our approach to calculating Monte-Carlo Greeks will not increase the computational time substantially and, at the same time, we can potentially achieve a significant variance reduction for Monte-Carlo Greeks (see, Chan and Joshi 2009). In section 6.2, we will also show that, under the pathwise approach, we no longer have to solve this 4^{th} order polynomial.

4. Expectation of Pathwise Derivatives - Unbiased Estimators of Monte-Carlo Greeks

In this section, we show that under the quasi mean-shifted proxy simulation schemes, Q, we can interchange differentiation and expectation when computing price sensitivities. Hence, price sensitivities can be obtained by evaluating the expectation of the pathwise derivative and this result holds even for financial products with discontinuous pay-off functions.

Suppose we have a smooth family of discretizations of the evolution of the state variables $K(\theta)$ with θ being a vector of inputs. We define $g(K(\theta))$ to be the deflated cash-flow generated by a financial product based on the realization of $K(\theta)$ and define $W(\theta)$ to be the accumulated Monte-Carlo weight. Note that, under Euler discretization scheme, $W(\theta)$ is equal to 1 for $\theta \in \Theta$. Therefore, the price of the financial product is

$$\mathbb{E}\left[W(\theta)g(K(\theta))\right],$$

and the price sensitivity with respect to θ in the direction of $\mathbf{u}$, is given by

$$\begin{aligned} &D_{\mathbf{u}}\mathbb{E}\left[W(\theta)g(K(\theta))\right] \\ = \; &\lim_{h\to 0}\mathbb{E}\left[\frac{W(\theta+h\mathbf{u})g(K(\theta+h\mathbf{u})) - W(\theta)g(K(\theta))}{h}\right]. \end{aligned} \tag{4.1}$$

We adapt a result from Glasserman 2003.

[5] Note that, the equation presented here is slightly different from the equation presented in Chan and Joshi (2009). In Chan and Joshi (2009), the equation is derived based on a numerical scheme driven by uncorrelated Brownian increments, while here, the equation is derived based on a numerical scheme driven by uncorrelated standard normal random variables.

Theorem 1. *The limit and the expectation in equation (4.1) can be interchanged i.e.*

$$\begin{aligned} & D_{\boldsymbol{u}}\mathbb{E}\left[W(\theta)g(K(\theta))\right] \\ = & \lim_{h\to 0}\mathbb{E}\left[\frac{W(\theta+h\boldsymbol{u})g(K(\theta+h\boldsymbol{u}))-W(\theta)g(K(\theta))}{h}\right] \\ = & \mathbb{E}\left[\lim_{h\to 0}\frac{W(\theta+h\boldsymbol{u})g(K(\theta+h\boldsymbol{u}))-W(\theta)g(K(\theta))}{h}\right] \\ = & \left\langle\mathbb{E}\left[\frac{\partial(W(\theta)g(K(\theta)))}{\partial\theta}\right],\boldsymbol{u}\right\rangle \end{aligned} \tag{4.2}$$

as long as the following conditions are satisfied:

1. *At each $\theta \in \Theta$, the derivative of $K_{i,j}(\theta)$ in the direction of $\boldsymbol{u}$ exists with probability 1 for all i and j*
2. *At each $\theta \in \Theta$, the derivative of $W(\theta)$ in the direction of $\boldsymbol{u}$ exists with probability 1*
3. $\mathrm{P}(K(\theta)\in D_g)=1$ *where D_g denotes the points where g is differentiable.*
4. *For any given vector of Z, there exists a constant C_g such that,*
$$|g(K(\theta_2))-g(K(\theta_1))| = C_g||K(\theta_2)-K(\theta_1)||$$
where $\theta_1, \theta_2 \in \Theta$.
5. *For any given vector of Z, there exists a constant $C_{i,j}$ such that*
$$|K_{i,j}(\theta_2)-K_{i,j}(\theta_1)| = C_{i,j}||\theta_2-\theta_1||$$
for all i and j and $\theta_1, \theta_2 \in \Theta$
6. *For any given vector of Z, there exists a constant C_w, such that*
$$|W(\theta_2)-W(\theta_1)| = C_w||\theta_2-\theta_1||$$

Proof. The conditions (1), (2) and (3) ensure that

$$\lim_{h\to 0}\frac{W(\theta+h\mathbf{u})g(K(\theta+h\mathbf{u}))-W(\theta)g(K(\theta))}{h}$$

exists with probability of 1 for $\theta \in \Theta$, while the conditions (4), (5) and (6) ensures that, for any given vector of Z,

$$W(\theta)g(K(\theta))$$

is Lipschitz in θ as the Lipschitz property is preserved under composition. Therefore, we can now apply the Dominated Convergence theorem to interchange the limit and the expectation and conclude that

$$\begin{aligned} & \lim_{h\to 0}\mathbb{E}\left[\frac{W(\theta+h\mathbf{u})g(K(\theta+h\mathbf{u}))-W(\theta)g(K(\theta))}{h}\right] \\ = & \mathbb{E}\left[\lim_{h\to 0}\frac{W(\theta+h\mathbf{u})g(K(\theta+h\mathbf{u}))-W(\theta)g(K(\theta))}{h}\right] \\ = & \left\langle\mathbb{E}\left[\frac{\partial(W(\theta)g(K(\theta)))}{\partial\theta}\right],\mathbf{u}\right\rangle \end{aligned} \tag{4.3}$$

□

Provided that all the conditions above are satisfied, price sensitivities can be obtained by evaluating the expectation of the pathwise derivative

$$\text{i.e.}\quad \mathbb{E}\left[\frac{\partial(W(\theta)g(K(\theta)))}{\partial\theta}\right]$$

and this approach is known as the pathwise method (see, Glasserman 2003). Similar to pricing, the expectation of the pathwise derivative is usually evaluated using Monte-Carlo simulation due to high-dimensional state space. At each path, we compute the pathwise derivative

$$\text{i.e.} \quad \frac{\partial (W(\theta) g(K(\theta)))}{\partial \theta},$$

and average it across a large number of paths. In general, the pathwise method is not applicable to financial product with discontinuous pay-offs as condition (4) does not hold.

However, under the quasi mean-shifted proxy simulation schemes, Q, price sensitivities can be evaluated using the pathwise method even for financial products with discontinuous pay-offs. By construction of Q, for any given vector Z, the pay-off function is a smooth function of inputs θ. Therefore, we have

$$g(K^Q(\theta))$$

Lipschitz in θ for any given vector Z. The accumulated Monte-Carlo weight under Q is also Lipschitz in θ as $W^Q(\theta)$ is the product of the ratio of joint normal density functions which are smooth functions. Therefore,

$$W^Q(\theta) g(K^Q(\theta))$$

is Lipschitz in θ for any given vector Z. The major advantage for being able to apply the pathwise method to Q, is that we can now compute price sensitivities with respect to all inputs simultaneously even for financial products with discontinuous pay-offs whereas previously, such methods could only be applied to products with continuous pay-off functions.

Note that, one obvious consequence of linearizing the proxy constraint function is that the condition (4) will no longer hold with probability of 1. Therefore, price sensitivities evaluated using the pathwise method can be biased. However, our numerical results in section 10 show that such bias is insignificant using the linearization method proposed in this paper.

5. Derivation of Pathwise Derivatives

In this section, we derive the pathwise derivative for the partial proxy simulation scheme, P and the minimal partial proxy simulation scheme, M. In particular, we assume that we are interested in computing the pathwise derivative with respect to a vector of initial inputs, θ_0, for a financial contract, D. The financial contract, D, pays a stream of cash-flow at times that are a subset of $t_1 < t_2 < \ldots < t_m$: The cash-flows at time t_i are an $\mathcal{F}_{t_i}$-measurable function. We define

$$g(K_1^A(\theta), K_2^A(\theta), \ldots, K_m^A(\theta))$$

to be the accumulated deflated cash-flows generated by D from t_1 up to t_m based on the realization of $K(\theta)$ under a numerical scheme A. For convenience, we write

$$g^A(\theta) \equiv g(K_1^A(\theta), K_2^A(\theta), \ldots, K_m^A(\theta)).$$

Since θ is a vector of inputs, we will therefore have a vector of price sensitivities (i.e. corresponding to each element in θ). For clarity of notation, θ is assumed to be an r-dimensional vector with the j^{th} element given by θ_j. Suppose that B_i denotes an n-dimensional vector at time t_i with the j^{th} element given by $B_{i,j}$, we define

$$\frac{\partial B_i}{\partial \theta} = \begin{pmatrix} \frac{\partial B_{i,1}}{\partial \theta_1} & \cdots & \frac{\partial B_{i,1}}{\partial \theta_r} \\ \vdots & \ddots & \vdots \\ \frac{\partial B_{i,n}}{\partial \theta_1} & \cdots & \frac{\partial B_{i,n}}{\partial \theta_r} \end{pmatrix}$$

to be an $n \times r$ matrix and

$$\frac{\partial B_{i,j}}{\partial \theta} = \left(\frac{\partial B_{i,j}}{\partial \theta_1}, \frac{\partial B_{i,j}}{\partial \theta_2}, \ldots, \frac{\partial B_{i,j}}{\partial \theta_r}\right)$$

to be an r-dimensional row vector. Therefore, the pathwise derivative is given by an r-dimensional row vector with the j^{th} element corresponding to the pathwise price sensitivity with respect to θ_j.

Proposition 6. *Under the partial proxy simulation scheme, the pathwise derivative with respect to θ_0 for the financial contract D is given by*

$$\sum_{i=1}^{m} \left[\frac{\partial g^P(\theta)}{\partial K_i^P} \frac{\partial K_i^P(\theta)}{\partial \theta} + g^P(\theta) \left(Z_{i,1} \frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial \theta} - \frac{\partial}{\partial \theta} \left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}} \right) \right) \right] \Bigg|_{\theta=\theta_0} \tag{5.1}$$

Proof. Given the inputs θ, the price of the financial contract, D, is given by

$$\mathbb{E}\left[g^P(\theta) \prod_{i=1}^{m} w_i^P(\theta) \right]. \tag{5.2}$$

By differentiating the inner expression of the expectation with respect to θ and setting $\theta = \theta_0$, we have

$$\begin{aligned}
& \left[\frac{\partial g^P(\theta)}{\partial \theta} \prod_{i=1}^{m} w_i^P(\theta) + g^P(\theta) \prod_{i=1}^{m} w_i^P(\theta) \left(\sum_{i=1}^{m} \frac{1}{w_i^P(\theta)} \frac{\partial w_i^P(\theta)}{\partial \theta} \right) \right] \Bigg|_{\theta=\theta_0} \\
= & \left[\frac{\partial g^P(\theta)}{\partial \theta} + g^P(\theta) \sum_{i=1}^{m} \frac{\partial \log w_i^P(\theta)}{\partial \theta} \right] \Bigg|_{\theta=\theta_0} \\
= & \sum_{i=1}^{m} \left[\frac{\partial g^P(\theta)}{\partial K_i^P} \frac{\partial K_i^P(\theta)}{\partial \theta} + g^P(\theta) \left(Z_{i,1} \frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial \theta} - \frac{\partial}{\partial \theta} \left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}} \right) \right) \right] \Bigg|_{\theta=\theta_0}
\end{aligned}$$

as, by construction,

$$\begin{aligned}
\nu_{i,1}^P(\theta_0, Z_i) &= 0, \\
w_i^P(\theta_0) &= 1, \\
\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}} \Bigg|_{\theta=\theta_0} &= 0,
\end{aligned} \tag{5.3}$$

and from equation (3.6), we have

$$w_i^P(\theta) = \left(1 - \frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}} \right) \exp\left(Z_{i,1} \nu_{i,1}^P(\theta, Z_i) - \frac{1}{2} \nu_{i,1}^P(\theta, Z_i)^2 \right). \tag{5.4}$$

□

Proposition 7. *Under the minimal partial proxy simulation scheme, the pathwise derivative with respect to θ_0 for the financial contract D is given by*

$$\sum_{i=1}^{m} \left[\frac{\partial g^M(\theta)}{\partial K_i^M} \frac{\partial K_i^M(\theta)}{\partial \theta} + g^M(\theta) \left(Z_{i,1} \frac{\partial \nu_{i,1}^M(\theta, Z_i)}{\partial \theta} + \frac{\partial \alpha_{i,1}^M(\theta)}{\partial \theta} \right) \right] \Bigg|_{\theta=\theta_0} \tag{5.5}$$

Proof. Given a vector of inputs θ, the price of the financial contract, D, is given by

$$\mathbb{E}\left[g^M(\theta)\prod_{i=1}^{m} w_i^M(\theta)\right]. \tag{5.6}$$

By differentiating the inner expression of the expectation with respect to θ and setting $\theta = \theta_0$, we have

$$\begin{aligned}
&\left[\frac{\partial g^M(\theta)}{\partial\theta}\prod_{i=1}^{m} w_i^M(\theta) + g^M(\theta)\prod_{i=1}^{m} w_i^M(\theta)\left(\sum_{i=1}^{m}\frac{1}{w_i^M(\theta)}\frac{\partial w_i^M(\theta)}{\partial\theta}\right)\right]\Bigg|_{\theta=\theta_0}\\
= &\sum_{i=1}^{m}\left[\frac{\partial g^M(\theta)}{\partial K_i^M}\frac{\partial K_i^M(\theta)}{\partial\theta} + g^M(\theta)\left(Z_{i,1}\frac{\partial \nu_{i,1}^M(\theta, Z_i)}{\partial\theta} + \frac{\partial \alpha_{i,1}^M(\theta)}{\partial\theta}\right)\right]\Bigg|_{\theta=\theta_0}
\end{aligned}$$

as we have

$$\begin{aligned}
\alpha_{i,1}^M(\theta_0) &= 1\\
\nu_{i,1}^M(\theta_0, Z_i) &= 0
\end{aligned}$$

and

$$w_i^M(\theta) = \alpha_{i,1}^M(\theta)\exp\left(Z_{i,1}\nu_{i,1}^M(\theta, Z_i) - \frac{1}{2}\nu_{i,1}^M(\theta, Z_i)^2\right). \tag{5.7}$$

□

6. Evaluating Pathwise Derivatives - A Naive Approach

In this section, we present a naive algorithm to evaluate the pathwise derivative derived in the previous section. Under this approach, the partial derivatives in equation (5.1) and (5.5) are calculated using matrix recursions as we evolve the state variables, K. The major problem with matrix recursions is that for a large number of inputs and state variables, the computational cost can be very high as we have to perform matrix multiplications at each step of the simulation. For example, suppose that θ_0 is an n-dimensional vector (e.g initial forward rates) and we have n state variables, the computational order will be at least $\mathcal{O}(n^3)$ per step. Whilst this approach is inefficient, especially when we have a large number of inputs and state variables, we still provide detailed explanations on its implementation, since we will see in sections 7 and 8 that the results shown in this section are important for developing more efficient methods of computation.

6.1. **Evaluating Partial Proxy Pathwise Derivatives - a Naive Approach.** In order to compute the pathwise derivative for the partial proxy simulation scheme in (5.1), we must be able to compute

$$\frac{\partial g^P(\theta)}{\partial K_i^P},\quad \frac{\partial K_i^P(\theta)}{\partial\theta},\quad \frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial\theta}\quad\text{and}\quad \frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right) \tag{6.1}$$

at $\theta = \theta_0$. Here, we will look at how these individual partial derivatives can be calculated.

6.1.1. *Computation of* $\frac{\partial g^P}{\partial K_i^P}$. Evaluating $\frac{\partial g^P}{\partial K_i^P}$ is straight-forward and it can be done easily by differentiating the deflated pay-offs with respect to K_i^P.

6.1.2. *Computation of* $\frac{\partial K_i^P}{\partial\theta}$. Based on the setup of the partial proxy simulation scheme, we have that

$$\frac{\partial K_i^P}{\partial\theta} = \frac{\partial K_i^P}{\partial K_{i-1}^P}\frac{\partial K_{i-1}^P}{\partial\theta} - \frac{\partial K_i^P}{\partial Z_{i,1}}\frac{\partial \nu_{i,1}^P}{\partial\theta} + \frac{\partial F_{i-1}}{\partial\theta}, \tag{6.2}$$

as

$$\frac{\partial K_i^P}{\partial \nu_{i,1}^P} = -\frac{\partial K_i^P}{\partial Z_{i,1}}.$$

6.1.3. *Computation of* $\frac{\partial \nu_{i,1}^P}{\partial\theta}$. Here,

$$\left.\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial\theta}\right|_{\theta=\theta_0}$$

can be derived using equation (3.4) as the solution of $\nu_{i,1}^P(\theta, Z_i)$ must satisfy equation (3.4) to prevent pathwise discontinuities. Suppose that the perturbed inputs θ_B, are given by

$$\theta_B = \theta_0 + h\mathbf{u}$$

where $\mathbf{u}$ is a directional vector and $h > 0$, the partial proxy simulation scheme requires that

$$p_i(K_i^P(\theta_B)) = p_i(K_i^P(\theta_0)) \tag{6.3}$$

for all i to prevent pathwise discontinuities. Using Taylor's theorem, we have

$$p_i(K_i^P(\theta_0)) + \nabla(p_i \circ K_i^P)(\theta_0)h\mathbf{u} + \mathcal{O}(h^2) = p_i(K_i^P(\theta_0)), \tag{6.4}$$

and taking the coefficient of h, we have

$$\nabla(p_i \circ K_i^P)(\theta_0)\cdot\mathbf{u} = 0. \tag{6.5}$$

Given that this result must hold for any arbitrary directional vector, $\mathbf{u}$, we must therefore have

$$\left.\left[\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial\theta}\right]\right|_{\theta=\theta_0} = \mathbf{0}. \tag{6.6}$$

By substituting (6.2) into the equation above and rearranging, we have

$$\left.\frac{\partial \nu_{i,1}^P}{\partial\theta}\right|_{\theta=\theta_0} = \left.\left[\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\frac{\partial p_i}{\partial K_i^P}\left(\frac{\partial K_i^P}{\partial K_{i-1}^P}\frac{\partial K_{i-1}^P}{\partial\theta} + \frac{\partial F_{i-1}}{\partial\theta}\right)\right]\right|_{\theta=\theta_0}. \tag{6.7}$$

6.1.4. *Computation of* $\frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P}{\partial Z_{i,1}}\right)$. Similarly,

$$\left.\frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right)\right|_{\theta=\theta_0}$$

can be derived from equation (6.3). By differentiating both sides of equation (6.3) with respect to $Z_{i,1}$, we have

$$\left.\left(\frac{\partial p_i}{\partial K_i^P}\left(\frac{\partial K_i^P}{\partial Z_{i,1}} + \frac{\partial K_i^P}{\partial \nu_{i,1}^P}\frac{\partial \nu_{i,1}^P}{\partial Z_{i,1}}\right)\right)\right|_{\theta=\theta_B} = \left.\left(\frac{\partial p_i}{\partial K_i^P}\left(\frac{\partial K_i^P}{\partial Z_{i,1}} + \frac{\partial K_i^P}{\partial \nu_{i,1}^P}\frac{\partial \nu_{i,1}^P}{\partial Z_{i,1}}\right)\right)\right|_{\theta=\theta_0} \tag{6.8}$$

where $\theta_B = \theta_0 + h\mathbf{u}$. We define

$$\begin{aligned} q_i(\theta) &:= \left(\frac{\partial p_i(K_i^P(\theta))}{\partial K_i^P}\frac{\partial K_i^P(\theta)}{\partial Z_{i,1}}\right), \\ r_i(\theta) &:= \left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right). \end{aligned}$$

Since we know that

$$\begin{aligned} \frac{\partial K_i^P}{\partial Z_{i,1}} &= -\frac{\partial K_i^P}{\partial \nu_{i,1}^P}, \\ r_i(\theta_0) &= \left.\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right|_{\theta=\theta_0} = 0, \end{aligned}$$

from equation (5.3), so equation (6.8) can be expressed as

$$q_i(\theta_B)(1 - r_i(\theta_B)) = q_i(\theta_0). \tag{6.9}$$

Using Taylor's theorem, equation (6.9) becomes

$$\left(q_i(\theta_0) + \nabla q_i(\theta_0)h\mathbf{u} + \mathcal{O}(h^2)\right)\left(1 - \nabla r_i(\theta_0)h\mathbf{u} - \mathcal{O}(h^2)\right) = q_i(\theta_0), \tag{6.10}$$

and taking the coefficient of h, we have

$$\nabla q_i(\theta_0)\mathbf{u} - q_i(\theta_0)\nabla r_i(\theta_0)\mathbf{u} = 0. \tag{6.11}$$

As we are interested in $\nabla r_i(\theta_0)$ i.e.

$$\frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right)\Big|_{\theta=\theta_0},$$

by rearranging and expanding equation (6.11), we have

$$\begin{aligned} &\frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P}{\partial Z_{i,1}}\right)\Big|_{\theta=\theta_0} \\ = &\left[\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\frac{\partial}{\partial\theta}\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)\right]\Big|_{\theta=\theta_0} \\ = &\left[\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\left(\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{\mathrm{T}}\frac{\partial}{\partial\theta}\left(\frac{\partial p_i}{\partial K_i^P}\right) + \frac{\partial p_i}{\partial K_i^P}\frac{\partial}{\partial\theta}\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)\right)\right]\Big|_{\theta=\theta_0} \\ = &\left[\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\left(\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{\mathrm{T}}\frac{\partial^2 p_i}{\partial K_i^P\partial K_i^P}\frac{\partial K_i^P}{\partial\theta} + \frac{\partial p_i}{\partial K_i^P}\frac{\partial}{\partial\theta}\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)\right)\right]\Big|_{\theta=\theta_0}. \end{aligned} \tag{6.12}$$

Since we have to differentiate $\frac{\partial K_i^P(\theta)}{\partial Z_{i,1}}$ with respect to θ, we study the structure of $\frac{\partial K_i^P(\theta)}{\partial Z_{i,1}}$ in detail. From the partial proxy simulation scheme, we know that

$$\frac{\partial K_i^P(\theta)}{\partial Z_{i,1}} = \left(a_{11}(K_{i-1}^P(\theta), \theta, t_{i-1}), \ldots, a_{n1}(K_{i-1}^P(\theta), \theta, t_{i-1})\right)^{\mathrm{T}},$$

and based on the definition of $S_i(K_i(\theta), \theta)$ in section 3.3.4, we clearly have

$$S_{i-1}(K_{i-1}^P(\theta), \theta) = \frac{\partial K_i^P(\theta)}{\partial Z_i}.$$

Thus, we have

$$\frac{\partial}{\partial\theta}\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right) = \frac{\partial S_{i-1}}{\partial K_{i-1}^P}\frac{\partial K_{i-1}^P}{\partial\theta} + \frac{\partial S_{i-1}}{\partial\theta}$$

and by substituting the equation above into equation (6.12), we have

$$\begin{aligned}&\frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P}{\partial Z_{i,1}}\right)\Bigg|_{\theta=\theta_0}\\ = \;&\left[\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\left(\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{\mathrm{T}}\frac{\partial^2 p_i}{\partial K_i^P\partial K_i^P}\frac{\partial K_i^P}{\partial\theta} + \frac{\partial p_i}{\partial K_i^P}\frac{\partial S_{i-1}}{\partial K_{i-1}^P}\frac{\partial K_{i-1}^P}{\partial\theta} + \frac{\partial p_i}{\partial K_i^P}\frac{\partial S_{i-1}}{\partial\theta}\right)\right]\Bigg|_{\theta=\theta_0}.\end{aligned} \tag{6.13}$$

6.1.5. *Algorithm for the Naive Approach.* Based on equations (6.2), (6.7) and (6.13), the pathwise derivative for the partial proxy simulation scheme can be easily calculated. Observe that the partial derivatives

$$\frac{\partial K_i^P}{\partial K_{i-1}^P},\quad \frac{\partial K_i^P}{\partial Z_{i,1}},\quad \frac{\partial F_{i-1}}{\partial\theta},\quad \frac{\partial S_{i-1}}{\partial K_{i-1}^P}\quad \text{and}\quad \frac{\partial S_{i-1}}{\partial\theta}$$

can be easily evaluated from the underlying numerical scheme while

$$\frac{\partial p_i}{\partial K_i^P}\quad \text{and}\quad \frac{\partial^2 p_i}{\partial K_i^P\partial K_i^P}$$

can be calculated from the proxy constraint function. Starting at t_0, we evolve the process K. At each time step, we will compute

$$\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial\theta}\Bigg|_{\theta=\theta_0}$$

using equation (6.7) followed by

$$\frac{\partial K_i^P(\theta)}{\partial\theta}\Bigg|_{\theta=\theta_0}\quad \text{and}\quad \frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right)\Bigg|_{\theta=\theta_0}$$

using equation (6.2) and equation (6.13) respectively. We then compute

$$\left[\frac{\partial g^P(\theta)}{\partial K_i^P}\frac{\partial K_i^P(\theta)}{\partial\theta} + g^P(\theta)\left(Z_{i,1}\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial\theta} - \frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P(\theta, Z_i)}{\partial Z_{i,1}}\right)\right)\right]\Bigg|_{\theta=\theta_0}. \tag{6.14}$$

The sum of (6.14) at each time step up to maturity will be the pathwise derivative for the partial proxy simulation scheme.

6.2. **Evaluating Minimal Partial Proxy Pathwise Derivatives - A Naive Approach.** Similarly, to compute the pathwise derivative for the minimal partial proxy simulation scheme in equation (5.5), we must be able to compute

$$\frac{\partial g^M(\theta)}{\partial K_i^M},\quad \frac{\partial K_i^M(\theta)}{\partial\theta},\quad \frac{\partial \nu_{i,1}^M(\theta, Z_i)}{\partial\theta}\quad \text{and}\quad \frac{\partial \alpha_{i,1}^M(\theta)}{\partial\theta} \tag{6.15}$$

at $\theta = \theta_0$.

6.2.1. *Computation of* $\frac{\partial g^M}{\partial K_i^M}$. Evaluating $\frac{\partial g^M}{\partial K_i^M}$ is straight-forward. It can be calculated easily by differentiating the deflated product payoffs with respect to K_i^M.

6.2.2. *Computation of* $\frac{\partial K_i^M}{\partial \theta}$. Based on the setup of the minimal partial proxy simulation scheme, we have that

$$\frac{\partial K_i^M}{\partial \theta} = \frac{\partial K_i^M}{\partial K_{i-1}^M}\frac{\partial K_{i-1}^M}{\partial \theta} - \frac{\partial K_i^M}{\partial Z_{i,1}}\frac{\partial \nu_{i,1}^M}{\partial \theta} + \frac{\partial F_{i-1}}{\partial \theta}, \tag{6.16}$$

as we know that

$$\frac{\partial K_i^M}{\partial \nu_{i,1}^M} = -\frac{\partial K_i^M}{\partial Z_{i,1}}.$$

6.2.3. *Computation of* $\frac{\partial \alpha_{i,1}^M}{\partial \theta}$. The minimal partial proxy scheme requires $\alpha_{i,1}^M(\theta_B)$ to satisfy the following equation

$$\begin{aligned} &2\alpha_{i,1}^M(\theta_B)^4 + 2X_{i,1}^M(\theta_B)Z_{i,1}^*\alpha_{i,1}^M(\theta_B)^3 - (3 + 2X_{i,1}^M(\theta_B)^2 + {Z_{i,1}^*}^2)\alpha_{i,1}^M(\theta_B)^2 \\ &+X_{i,1}^M(\theta_B)Z_{i,1}^*\alpha_{i,1}^M(\theta_B) + 1 = 0 \end{aligned} \tag{6.17}$$

to minimise the variance of the Monte-Carlo weight across that time step. Using Taylor's theorem, we have

$$\alpha_{i,1}^M(\theta_B) = \alpha_{i,1}^M(\theta_0) + \nabla\alpha_{i,1}^M(\theta_0)h\mathbf{u} + \mathcal{O}(h^2) \tag{6.18}$$

and

$$X_{i,1}^M(\theta_B) = X_{i,1}^M(\theta_0) + \nabla X_{i,1}^M(\theta_0)h\mathbf{u} + \mathcal{O}(h^2) \tag{6.19}$$

We also know that

$$X_{i,1}^M(\theta_0) = \alpha_{i,1}^M(\theta_0)Z_{i,1}^* + \beta_{i,1}^M(\theta_0) = Z_{i,1}^*$$

from equation (3.20). By substituting equation (6.18) and equation (6.19) into (6.17), we conclude that

$$\left.\frac{\partial \alpha_{i,1}^M}{\partial \theta}\right|_{\theta=\theta_0} = \left(\frac{Z_{i,1}^*}{2 + (Z_{i,1}^*)^2}\right)\left.\frac{\partial X_{i,1}^M}{\partial \theta}\right|_{\theta=\theta_0} \tag{6.20}$$

by taking the coefficient of h.

6.2.4. *Computation of* $\frac{\partial X_{i,1}^M}{\partial \theta}$. The partial derivative

$$\left.\frac{\partial X_{i,1}^M(\theta)}{\partial \theta}\right|_{\theta=\theta_0},$$

can be derived using the facts that $\hat{K}_i^M(\theta)$ is a function of $X_{i,1}^M(\theta)$ and $\hat{K}_i^M(\theta)$ must satisfy equation (3.19) to prevent pathwise discontinuities. Therefore, by differentiating equation (3.19) with respect to θ, we can obtain $\frac{\partial X_{i,1}^M}{\partial \theta}$.

Recall that, from equation (3.11), we have

$$\hat{K}_i^M(\theta_B) := F_{i-1}(K_{i-1}^M(\theta_B), \hat{Z}_i - \nu_i^M(\theta_B, \hat{Z}_i), \theta_B) \tag{6.21}$$

with the elements of $\hat{Z}_i$ given by

$$\begin{aligned} \hat{Z}_{i,1} &= Z_{i,1}^* \\ \hat{Z}_{i,s} &= Z_{i,s} \quad \text{for} \quad s = 2, \dots, d. \end{aligned}$$

Since, we can write

$$\begin{aligned}\hat{Z}_{i,1} - \nu_{i,1}^M(\theta_B, \hat{Z}) &= Z_{i,1}^* - \left((1-\alpha_{i,1}^M(\theta_B))Z_{i,1}^* - \beta_{i,1}^M(\theta_B)\right)\\ &= \alpha_{i,1}^M(\theta_B)Z_{i,1}^* + \beta_{i,1}^M(\theta_B)\\ &= X_{i,1}^M(\theta_B)\end{aligned}$$

using results from (3.20), we conclude that $\hat{K}_i^M$ is a function of $X_{i,1}^M$.

By construction, we have

$$\hat{K}_i^M(\theta_0) = \hat{K}_i^E(\theta_0),$$

so equation (3.19) can be written as

$$p_i(\hat{K}_i^M(\theta_0)) + \nabla p_i(\hat{K}_i^M(\theta_0))(\hat{K}_i^M(\theta_B) - \hat{K}_i^M(\theta_0)) = H_i^M(\theta_B). \tag{6.22}$$

Since we have,

$$p_i(\hat{K}_i^M(\theta_0)) = H_i^M(\theta_0),$$

and applying Taylor's theorem to equation (6.22), we have

$$\left[\frac{\partial p_i}{\partial \hat{K}_i^M}\frac{\partial \hat{K}_i^M}{\partial \theta}\right]\Bigg|_{\theta=\theta_0} = \frac{\partial H_i^M}{\partial \theta}\Bigg|_{\theta=\theta_0}. \tag{6.23}$$

by taking the coefficient of h. By expanding the LHS of the equation (6.23), we get

$$\left[\frac{\partial p_i}{\partial \hat{K}_i^M}\left(\frac{\partial \hat{K}_i^M}{\partial K_{i-1}^M}\frac{\partial K_{i-1}^M}{\partial \theta} + \frac{\partial \hat{K}_i^M}{\partial X_{i,1}^M}\frac{\partial X_{i,1}^M}{\partial \theta} + \frac{\partial \hat{F}_{i-1}}{\partial \theta}\right)\right]\Bigg|_{\theta=\theta_0} = \frac{\partial H_i^M}{\partial \theta}\Bigg|_{\theta=\theta_0},$$

and here, instead of using F_{i-1}, we use $\hat{F}_{i-1}$ to denote the mapping function conditioning on $Z_{i,1} = Z_{i,1}^*$. Hence, by rearranging the equation above, we have

$$\frac{\partial X_{i,1}^M}{\partial \theta}\Bigg|_{\theta=\theta_0} = \left[\left(\frac{\partial p_i}{\partial \hat{K}_i^M}\frac{\partial K_i^M}{\partial Z_{i,1}^M}\right)^{-1}\left(\frac{\partial H_i^M}{\partial \theta} - \frac{\partial p_i}{\partial \hat{K}_i^M}\left(\frac{\partial \hat{K}_i^M}{\partial K_{i-1}^M}\frac{\partial K_{i-1}^M}{\partial \theta} + \frac{\partial \hat{F}_{i-1}}{\partial \theta}\right)\right)\right]\Bigg|_{\theta=\theta_0} \tag{6.24}$$

as

$$\frac{\partial K_i^M}{\partial Z_{i,1}} = \frac{\partial \hat{K}_i^M}{\partial X_{i,1}^M}.$$

We also note that

$$\frac{\partial X_{i,1}^M}{\partial \theta} = Z_{i,1}^*\frac{\partial \alpha_{i,1}^M}{\partial \theta} + \frac{\partial \beta_{i,1}^M}{\partial \theta}. \tag{6.25}$$

6.2.5. *Computation of* $\frac{\partial H_i^M}{\partial \theta}$. Since H_i^M is an $\mathcal{F}_{t_{i-1}}$-measurable function, in general, we have

$$\frac{\partial H_i^M}{\partial \theta} = \sum_{l=0}^{i-1}\left(\frac{\partial H_i^M}{\partial K_l^M}\frac{\partial K_l^M}{\partial \theta}\right). \tag{6.26}$$

However, in order to compute pathwise derivative efficiently, we will assume the trigger level, H_i^M is a function of previous trigger level, H_{i-1}^M and underlying quantities, K^M, at time t_{i-1}. So, we have

$$\frac{\partial H_i^M}{\partial \theta} = \frac{\partial H_i^M}{\partial H_{i-1}^M}\frac{\partial H_{i-1}^M}{\partial \theta} + \frac{\partial H_i^M}{\partial K_{i-1}^M}\frac{\partial K_{i-1}^M}{\partial \theta}. \tag{6.27}$$

Such assumptions are not unreasonable as the trigger functions for complicated path-dependent financial products such as LIBOR TARNs and CMS TARNs (see section 10) can be expressed in this form.

6.2.6. *Computation of* $\frac{\partial \nu_{i,1}^M}{\partial \theta}$. In order to compute

$$\left.\frac{\partial \nu_{i,1}^M(\theta, Z_i)}{\partial \theta}\right|_{\theta=\theta_0},$$

we will use the results from section 6.2.4. Since $\nu_{i,1}^M(\theta, Z_i) = (1 - \alpha_{i,1}^M(\theta))Z_{i,1} - \beta_{i,1}^M(\theta)$, the sensitivity of $\nu_{i,1}^M$ with respect to θ_0 can be expressed in the following form

$$\begin{aligned}
\left.\frac{\partial \nu_{i,1}^M}{\partial \theta}\right|_{\theta=\theta_0} &= \left.\left[-Z_{i,1}\frac{\partial \alpha_{i,1}^M}{\partial \theta} - \frac{\partial \beta_{i,1}^M}{\partial \theta}\right]\right|_{\theta=\theta_0} \\
&= \left.\left[(Z_{i,1}^* - Z_{i,1})\frac{\partial \alpha_{i,1}^M}{\partial \theta} - \frac{\partial X_{i,1}^M}{\partial \theta}\right]\right|_{\theta=\theta_0} \\
&= \left.\left[\left(-\frac{2 + Z_{i,1}Z_{i,1}^*}{2 + (Z_{i,1}^*)^2}\right)\frac{\partial X_{i,1}^M}{\partial \theta}\right]\right|_{\theta=\theta_0}, \qquad (6.28)
\end{aligned}$$

using results for (6.20) and (6.25). Therefore, computation of $\frac{\partial \nu_{i,1}^M}{\partial \theta}$ is straight-forward once we have $\frac{\partial X_{i,1}^M}{\partial \theta}$.

6.2.7. *Algorithm for the Naive Approach.* Observe that the partial derivatives

$$\frac{\partial K_i^M}{\partial K_{i-1}^M}, \quad \frac{\partial K_i^M}{\partial Z_{i,1}}, \quad \frac{\partial F_{i-1}}{\partial \theta}, \quad \frac{\partial \hat{K}_i^M}{\partial K_{i-1}^M} \quad \text{and} \quad \frac{\partial \hat{F}_{i-1}}{\partial \theta}$$

can be obtained easily from the underlying numerical scheme while

$$\frac{\partial p_i}{\partial \hat{K}_i^M}$$

can easily be calculated from the proxy constraint function. Similarly,

$$\frac{\partial H_i^M}{\partial H_{i-1}^M} \quad \text{and} \quad \frac{\partial H_i^M}{\partial K_{i-1}^M}$$

can be easily calculated from the product trigger.

Starting at t_0, we evolve the process K. At each step, we compute $Z_{i,1}^*$ and $\frac{\partial H_i^M}{\partial \theta}$ using equation (3.16) and (6.27) respectively. If $Z_{i,1}^*$ is greater than ± 4, we set $\frac{\partial X_{i,1}^M(\theta)}{\partial \theta}$ **=0** (i.e no measure change) else we compute

$$\left.\frac{\partial X_{i,1}^M(\theta)}{\partial \theta}\right|_{\theta=\theta_0}$$

using (6.24). We then proceed to compute

$$\left.\frac{\partial K_i^M(\theta)}{\partial \theta}\right|_{\theta=\theta_0}, \quad \left.\frac{\partial \nu_{i,1}^M(\theta, Z_i)}{\partial \theta}\right|_{\theta=\theta_0}, \quad \text{and} \quad \left.\frac{\partial \alpha_{i,1}^M(\theta)}{\partial \theta}\right|_{\theta=\theta_0}$$

using (6.16), (6.28) and (6.20). By accumulating

$$\left[\frac{\partial g^M(\theta)}{\partial K_i^M}\frac{\partial K_i^M(\theta)}{\partial\theta} + g^M(\theta)\left(Z_{i,1}\frac{\partial \nu_{i,1}^M(\theta, Z_i)}{\partial\theta} - \frac{\partial \alpha_{i,1}^M(\theta)}{\partial\theta}\right)\right]\Bigg|_{\theta=\theta_0}. \tag{6.29}$$

from t_1 up to t_m, we will obtain the pathwise derivative under the minimal partial proxy simulation scheme.

6.3. **Evolving the State Variables.** One subtle but important issue that is worth discussing is the evolution of our state variables, K. By construction, given the initial inputs θ_0 and a vector of Z, we have

$$K_i^E(\theta_0) = K_i^P(\theta_0) = K_i^M(\theta_0)$$

for all i as no measure change is required (i.e $\nu_i^P(\theta_0, Z_i) = \nu_i^M(\theta_0, Z_i) = \mathbf{0}$). Since all the partial derivatives are evaluated at θ_0, we can instead evolve the state variables using E^0 and compute all the partial derivatives based on the realization of $K_i^E(\theta_0)$ at each time step.

7. Evaluating Pathwise Derivatives - The Adjoint Method

As explained in the previous section, the naive approach involves matrix recursions. Hence, the computational cost can be very high when we have a large number of state variables and inputs. By using the adjoint method, we can reduce the computational order. The use of the adjoint method in computing price sensitivities was first introduced by Giles and Glasserman (2006). Instead of using matrix recursions, the pathwise derivative can be evaluated using vector recursions starting from time step t_m up to t_0 (i.e backward summation). The vector recursion results for the partial proxy simulation scheme and the minimal partial proxy simulation scheme can be easily derived using mathematical inductions.

In this section, instead of explicitly stating that a partial derivative is evaluated at θ_0, we assume all the partial derivatives are evaluated at θ_0. Any new vector notations introduced in this section and section 8 will take the following form

$$B^A(i),$$

where A represents the specific numerical scheme and i is the time index. Whilst previously notations in the form of $B_i^A(\theta)$ were useful in emphasizing the dependency on θ for the purpose of deriving the pathwise derivative, here, in contrast, we emphasize the dependency on the time t_i.

7.1. **Evaluating Partial Proxy Pathwise Derivatives - The Adjoint Method.** Recall that, the partial proxy pathwise derivative is given by

$$\sum_{l=1}^{m}\left[\frac{\partial g^P}{\partial K_l^P}\frac{\partial K_l^P}{\partial\theta} + g^P.\left(Z_{l,1}\frac{\partial \nu_{l,1}^P}{\partial\theta} - \frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{l,1}^P}{\partial Z_{l,1}}\right)\right)\right]. \tag{7.1}$$

We will see that the backward summation from $l = m$ to $i + 1$ for the expression above can be expressed in the following form

$$V^P(i)\frac{\partial K_i^P}{\partial\theta} + R^P(i), \tag{7.2}$$

where both $V^P(i)$ and $R^P(i)$ are row vectors. Hence, the backward summation from $l = m$ to i for the expression in (7.1) is given by

$$V^P(i)\frac{\partial K_i^P}{\partial\theta} + R^P(i) + \frac{\partial g^P}{\partial K_i^P}\frac{\partial K_i^P}{\partial\theta} + g^P.\left(Z_{i,1}\frac{\partial \nu_{i,1}^P}{\partial\theta} - \frac{\partial}{\partial\theta}\left(\frac{\partial \nu_{i,1}^P}{\partial Z_{i,1}}\right)\right) \tag{7.3}$$

By substituting the result from equation (6.13) into equation (7.3), we have

$$D^P(i)\frac{\partial K_i^P}{\partial \theta} + g^P.\left(Z_{i,1}\frac{\partial \nu_{i,1}^P}{\partial \theta} - C^P(i)\frac{\partial S_{i-1}}{\partial K_{i-1}^P}\frac{\partial K_{i-1}^P}{\partial \theta}\right) - B^P(i) + R^P(i), \tag{7.4}$$

where

$$\begin{aligned}
C^P(i) &= \left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\frac{\partial p_i}{\partial K_i^P},\\
D^P(i) &= V^P(i) + \frac{\partial g^P}{\partial K_i^P} - g^P.\left(\frac{\partial p_i}{\partial K_i^P}\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{-1}\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{\mathrm{T}}\frac{\partial^2 p_i}{\partial K_i^P \partial K_i^P},\\
B^P(i) &= g^P C^P(i)\frac{\partial S_{i-1}}{\partial \theta}.
\end{aligned} \tag{7.5}$$

By substituting equation (6.2) into equation (7.4), equation (7.4) becomes

$$\begin{aligned}
&\left(D^P(i)\frac{\partial K_i^P}{\partial K_{i-1}^P} - g^P C^P(i)\frac{\partial S_{i-1}^P}{\partial K_{i-1}^P}\right)\frac{\partial K_{i-1}^P}{\partial \theta} + \left(g^P Z_{i,1} - D^P(i)\frac{\partial K_i^P}{\partial Z_{i,1}}\right)\frac{\partial \nu_{i,1}^P}{\partial \theta}\\
&+D^P(i)\frac{\partial F_{i-1}}{\partial \theta} - B^P(i) + R^P(i).
\end{aligned} \tag{7.6}$$

Similarly, by substituting equation (6.7) into equation (7.6), we obtain

$$\left(\tilde{V}^P(i)\frac{\partial K_i^P}{\partial K_{i-1}^P} - g^P C(i)\frac{\partial S_{i-1}}{\partial K_{i-1}^P}\right)\frac{\partial K_{i-1}^P}{\partial \theta} + \tilde{V}^P(i)\frac{\partial F_{i-1}}{\partial \theta} - B^P(i) + R^P(i) \tag{7.7}$$

where

$$\tilde{V}^P(i) = D^P(i) + \left(g^P Z_{i,1} - D^P(i)\frac{\partial K_i^P}{\partial Z_{i,1}}\right)C^P(i).$$

By setting

$$\begin{aligned}
V^P(i-1) &= \tilde{V}^P(i)\frac{\partial K_i^P}{\partial K_{i-1}^P} - g^P C^P(i)\frac{\partial S_{i-1}}{\partial K_{i-1}^P},\\
R^P(i-1) &= \tilde{V}^P(i)\frac{\partial F_{i-1}}{\partial \theta} - B^P(i) + R^P(i),
\end{aligned}$$

we will also have

$$V^P(i-1)\frac{\partial K_{i-1}^P}{\partial \theta} + R^P(i-1) \tag{7.8}$$

for the summation from $l = m$ to i for the expression in (7.1). Such result will clearly hold at $l = m$, i.e.

$$\frac{\partial g^P}{\partial K_m^P}\frac{\partial K_m^P}{\partial \theta} + g^P.\left(Z_{m,1}\frac{\partial \nu_{m,1}^P}{\partial \theta} - \frac{\partial}{\partial \theta}\left(\frac{\partial \nu_{m,1}^P}{\partial Z_{m,1}}\right)\right) = V^P(m-1)\frac{\partial K_{m-1}^P}{\partial \theta} + R^P(m-1)$$

with $V^P(m)=\mathbf{0}$ and $R^P(m)=\mathbf{0}$. Hence, under the adjoint method, the partial proxy pathwise derivative can be evaluated using

$$\begin{aligned} & V^P(0)\frac{\partial K_0^P}{\partial\theta}+R^P(0) \\ = \;& V^P(0)\frac{\partial K_0^P}{\partial\theta}+\sum_{i=1}^{m}\left(\tilde{V}^P(i)\frac{\partial F_{i-1}}{\partial\theta}-B^P(i)\right). \end{aligned} \tag{7.9}$$

The adjoint algorithm to compute the partial proxy pathwise derivative for any given path is as follows:

(1) We compute and store all the relevant partial derivatives as we evolve K and we set $V^P(m)=\mathbf{0}$ and $R^P(m)=\mathbf{0}$.
(2) At step i, we compute $C^P(i)$ followed by $D^P(i)$, $B^P(i)$ and $\tilde{V}^P(i)$.
(3) We then compute $R^P(i-1)$ and $V^P(i-1)$.
(4) We repeat step (2) and (3) from $i=m$ to $i=1$ and the pathwise derivative is given by the equation (7.9).

7.2. **Evaluating Minimal Partial Proxy Pathwise Derivatives - The Adjoint Method.** Recall that, the minimal partial proxy pathwise derivative is given by

$$\sum_{l=1}^{m}\left[\frac{\partial g^M}{\partial K_l^M}\frac{\partial K_l^M}{\partial\theta}+g^M.\left(Z_{l,1}\frac{\partial\nu_{l,1}^M}{\partial\theta}+\frac{\partial\alpha_{l,1}^M}{\partial\theta}\right)\right]. \tag{7.10}$$

We assume that the backward summation from $l=m$ to $i+1$ for the expression above can be expressed in following form

$$V^M(i)\frac{\partial K_i^M}{\partial\theta}+B^M(i)\frac{\partial H_i^M}{\partial\theta}+R^M(i), \tag{7.11}$$

where both $V^M(i)$, $B^M(i)$ and $R^M(i)$ are row vectors. Hence, the backward summation from $l=m$ to i for the expression in (7.10) is given by

$$V^M(i)\frac{\partial K_i^M}{\partial\theta}+B^M(i)\frac{\partial H_i^M}{\partial\theta}+R^M(i)+\frac{\partial g^M}{\partial K_i^M}\frac{\partial K_i^M}{\partial\theta}+g^M.\left(Z_{i,1}\frac{\partial\nu_{i,1}^M}{\partial\theta}+\frac{\partial\alpha_{i,1}^M}{\partial\theta}\right) \tag{7.12}$$

By substituting equation (6.16) into equation (7.12), equation (7.12) becomes

$$\begin{aligned} & \tilde{V}^M(i)\frac{\partial K_i^M}{\partial K_{i-1}^M}\frac{\partial K_{i-1}^M}{\partial\theta}+B^M(i)\frac{\partial H_i^M}{\partial\theta}+R^M(i)+\tilde{V}^M(i)\frac{\partial F_{i-1}}{\partial\theta} \\ & +\left(g^M Z_{i,1}-\tilde{V}^M(i)\frac{\partial K_i^M}{\partial Z_{i,1}}\right)\frac{\partial\nu_{i,1}^M}{\partial\theta}+g^M\frac{\partial\alpha_{i,1}^M}{\partial\theta}, \end{aligned} \tag{7.13}$$

where

$$\tilde{V}^M(i)=V^M(i)+\frac{\partial g^M}{\partial K_i^M}.$$

By using the results from equation (6.20) and equation (6.28), equation (7.13) can be written as

$$\tilde{V}^M(i)\frac{\partial K_i^M}{\partial K_{i-1}^M}\frac{\partial K_{i-1}^M}{\partial\theta}+B^M(i)\frac{\partial H_i^M}{\partial\theta}+R^M(i)+\tilde{V}^M(i)\frac{\partial F_{i-1}}{\partial\theta}+C^M(i)\frac{\partial X_{i,1}^M}{\partial\theta}, \tag{7.14}$$

where

$$C^M(i) = \left[\frac{g^M Z^*_{i,1}}{2+(Z^*_{i,1})^2} + \left(\frac{2+Z_{i,1}Z^*_{i,1}}{2+(Z^*_{i,1})^2}\right)\left(\tilde{V}^M(i)\frac{\partial K_i^M}{\partial Z_{i,1}} - g^M Z_{i,1}\right)\right]\mathbf{1}_{\{|Z^*_{i,1}|<4\}}.$$

and $\mathbf{1}_{\{|Z^*_{i,1}|<4\}}$ is the indicator function indicating the measure change decision. In particular, whenever $|Z^*_{i,1}|$ is greater than 4, there will be no measure change and we set $C^M(i) = \mathbf{0}$.

Finally, by substituting equation (6.27) and equation (6.24) into equation (7.14), we get

$$\begin{aligned}
&\left(\tilde{V}^M(i)\frac{\partial K_i^M}{\partial K_{i-1}^M} + \hat{V}^M(i)\frac{\partial \hat{K}_i^M}{\partial K_{i-1}^M} + D^M(i)\frac{\partial H_i^M}{\partial K_{i-1}^M}\right)\frac{\partial K_{i-1}^M}{\partial\theta} \\
&+D^M(i)\frac{\partial H_i^M}{\partial H_{i-1}^M}\frac{\partial H_{i-1}^M}{\partial\theta} + R^M(i) + \tilde{V}^M(i)\frac{\partial F_{i-1}}{\partial\theta} + \hat{V}^M(i)\frac{\partial \hat{F}_{i-1}}{\partial\theta},
\end{aligned} \tag{7.15}$$

where

$$\begin{aligned}
\hat{V}^M(i) &= -C^M(i)\left(\frac{\partial p_i}{\partial \hat{K}_i^M}\frac{\partial K_i^M}{\partial Z_{i,1}}\right)^{-1}\frac{\partial p_i}{\partial \hat{K}_i^M}, \\
D^M(i) &= C^M(i)\left(\frac{\partial p_i}{\partial \hat{K}_i^M}\frac{\partial K_i^M}{\partial Z_{i,1}}\right)^{-1} + B^M(i).
\end{aligned}$$

By letting

$$\begin{aligned}
V^M(i-1) &= \left(\tilde{V}^M(i)\frac{\partial K_i^M}{\partial K_{i-1}^M} + \hat{V}^M(i)\frac{\partial \hat{K}_i^M}{\partial K_{i-1}^M} + D^M(i)\frac{\partial H_i^M}{\partial K_{i-1}^M}\right), \\
B^M(i-1) &= D^M(i)\frac{\partial H_i^M}{\partial H_{i-1}^M}, \\
R^M(i-1) &= R^M(i) + \tilde{V}^M(i)\frac{\partial F_{i-1}}{\partial\theta} + \hat{V}^M(i)\frac{\partial \hat{F}_{i-1}}{\partial\theta},
\end{aligned}$$

then the summation from $l = m$ to i for the expression in (7.10) is given by

$$V^M(i-1)\frac{\partial K_{i-1}^M}{\partial\theta} + B^M(i-1)\frac{\partial H_{i-1}^M}{\partial\theta} + R^M(i-1). \tag{7.16}$$

This result clearly holds at $l = m$ i.e.

$$\frac{\partial g^M}{\partial K_m^M}\frac{\partial K_m^M}{\partial\theta} + g^M.\left(Z_{m,1}\frac{\partial \nu_{m,1}^M}{\partial\theta} + \frac{\partial \alpha_{m,1}^M}{\partial\theta}\right) = V^M(m-1)\frac{\partial K_{m-1}^M}{\partial\theta} + B^M(m-1)\frac{\partial H_{m-1}^M}{\partial\theta} + R^M(m-1).$$

where $V^M(m) = \mathbf{0}$, $B^M(m) = \mathbf{0}$ and $R^M(m) = \mathbf{0}$. As we know that

$$\frac{\partial H_0^M}{\partial\theta} = \mathbf{0},$$

the pathwise derivative of the minimal partial proxy simulation schemes can therefore be evaluated using

$$\begin{aligned} & V^M(0)\frac{\partial K_0^M}{\partial \theta} + R^M(0) \\ = \;& V^M(0)\frac{\partial K_0^M}{\partial \theta} + \sum_{i=1}^{m}\left(\tilde{V}^M(i)\frac{\partial F_{i-1}}{\partial \theta} + \hat{V}^M(i)\frac{\partial \hat{F}_{i-1}}{\partial \theta}\right), \end{aligned} \tag{7.17}$$

under the adjoint approach. The adjoint algorithm to compute the minimal partial proxy pathwise derivative for a given path is as follows:

(1) We compute and store all the relevant partial derivatives as we evolve K and we set $V^M(m) = \mathbf{0}$, $B^M(m) = \mathbf{0}$ and $R^M(m) = \mathbf{0}$.
(2) At step i, we compute $\tilde{V}^M(i)$ followed by $C^M(i)$, $\hat{V}^M(i)$ and $D^M(i)$.
(3) We then compute $V^M(i-1)$, $B^M(i-1)$ and $R^M(i-1)$.
(4) We repeat step (2) and (3) from $i = m$ to $i = 1$ and the pathwise derivative is given by the equation (7.17).

7.3. **Limitations of The Adjoint Method.** Using the adjoint method presented in this section, we cannot do better than $\mathcal{O}(n^2)$ per step (assuming θ is an n-dimensional vector and we have n state variables). This is because at each step of the simulation, we have to compute

$$[P1]\tilde{V}^P(i)\frac{\partial K_i^P}{\partial K_{i-1}^P}, \quad [P2]\left(\frac{\partial K_i^P}{\partial Z_{i,1}}\right)^{\mathrm{T}}\frac{\partial^2 p_i}{\partial K_i^P \partial K_i^P}, \quad [P3]C^P(i)\frac{\partial S_{i-1}}{\partial K_{i-1}^P}, \tag{7.18}$$

$$[P4]C^P(i)\frac{\partial S_{i-1}}{\partial \theta}, \quad [P5]\tilde{V}^P(i)\frac{\partial F_{i-1}}{\partial \theta},$$

for the partial proxy simulation scheme, and similarly,

$$[M1]\tilde{V}^M(i)\frac{\partial K_i^M}{\partial K_{i-1}^M}, \quad [M2]\hat{V}^M(i)\frac{\partial \hat{K}_i^M}{\partial K_{i-1}^M}, \tag{7.19}$$

$$[M3]\tilde{V}^M(i)\frac{\partial F_{i-1}}{\partial \theta}, \quad [M4]\hat{V}^M(i)\frac{\partial \hat{F}_{i-1}}{\partial \theta}$$

for the minimal partial proxy scheme. All these involve multiplications of a $(1 \times n)$ vector with a $(n \times n)$ matrix.

8. Fast Greeks under the LIBOR Market Model

For cases where the pay-off function is continuous, Giles and Glasserman (2006) have shown that, under the LIBOR Market Model, all the deltas and vegas can be evaluated with a computational order proportional to the number of rates times the number of factors at each step of the simulation (i.e $\mathcal{O}(nd)$ per step). Their approach involves studying the special drift structure of the LIBOR market model and they use the forward rates to carry out the differentiation.

Similar results were also presented by Denson and Joshi (2009). Instead of evolving the forward rates directly, they evolved the log of the forward rates. They show that by using the log of the forward rates as the states variables to carry out the differentiation, we can achieve an additional 20% speed improvement when computing deltas and vegas.

In this section, we will work with the log of the forward rates and show that, for cases where we have linear or linearized proxy constraint functions, all deltas and elementary vegas can be evaluated with a computational order of $\mathcal{O}(nd)$ per step under the LIBOR Market Model using the pathwise partial proxy method and the pathwise minimal partial proxy method. We wish to emphasize that this result also holds for the Displaced Diffusion LIBOR Market Model.

8.1. **LIBOR Market Model.** Under the LIBOR Market Model, we have n forward rates $f_1, f_2, \ldots, f_n$ with the corresponding tenor structure of $t_0 = 0 < t_1 < \ldots < t_{n+1}$ and forward rates are assumed to follow

$$\frac{df_j(t)}{f_j(t)} = \mu_j(f,t)dt + \sigma_j(t)dW(t), \tag{8.1}$$

where $\sigma_j(t)$ is a deterministic d-dimensional row vector and $W(t)$ is a d-dimensional column vector of uncorrelated Brownian motions. Under the spot-measure, which corresponds to using the discretely compounded money market account as numeraire, the drift term is given by no arbitrage arguments as

$$\mu_j(f,t) = \sum_{h=\eta(t)}^{j} \frac{f_h(t)\tau_h}{1 + f_h(t)\tau_h} \sigma_j(t)\sigma_h(t)^{\mathrm{T}}, \tag{8.2}$$

where

$$\tau_i = t_{i+1} - t_i$$

and $\eta(t)$ gives the index of the next forward rate to reset at time t. Under the usual log-Euler discretization (see Joshi, 2003a) with $f_j(t_i) \equiv f_j(i)$, we have

$$\log f_j(i+1) = \log f_j(i) + \tilde{\mu}_j(f(i)) - \frac{1}{2}C_{jj}(i) + a_j^r(i)Z(i+1) \tag{8.3}$$

with

$$\tilde{\mu}_j(f(i)) = \sum_{h=\eta(t)}^{j} \frac{f_h(i)\tau_h}{1 + f_h(i)\tau_h} C_{jh}(i)$$

where $C_{jh}(i)$ is the covariance between $\log f_j$ and $\log f_h$ from t_i to t_{i+1}, $Z(i+1)$ is a d-dimensional column vector of standard normal random variables and $a_j^r(i)$ is the j^{th} row of the $n \times d$ pseudo-root, $A(i) = (a_{js}(i))$ such that

$$A(i)A(i)^{\mathrm{T}} = C(i).$$

8.2. **Fast LMM Deltas.** In this section, we show that all deltas can be evaluated with a computational order of $\mathcal{O}(nd)$ per step under the LMM.

8.2.1. *Fast Deltas under the Pathwise Partial Proxy Method.* Under the Pathwise Partial Proxy Method, pathwise deltas are given by

$$V^P(0)\frac{\partial K_0^P}{\partial \theta}$$

where

$$\frac{\partial K_0^P}{\partial \theta} = \operatorname{diag}\left(\frac{1}{f_1(0)}, \frac{1}{f_2(0)}, \ldots, \frac{1}{f_n(0)}\right)$$

as

$$\frac{\partial F_{i-1}}{\partial \theta} = \mathbf{0}, \quad \frac{\partial S_{i-1}}{\partial \theta} = \mathbf{0}. \tag{8.4}$$

We also have

$$\frac{\partial S_{i-1}}{\partial K_{i-1}^P} = \mathbf{0}.$$

as pseudo-square roots at each time step are state independent. Therefore, as long as we can evaluate expression $[P1]$ and $[P2]$ from equation (7.18) with a computational order of $\mathcal{O}(nd)$, then the overall computational order for deltas will be $\mathcal{O}(nd)$ per step, Observe that, the expression $[P2]$ exists due to the second order partial derivative of the proxy constraint function p_i with respect to K_i^P i.e

$$\frac{\partial^2 p_i}{\partial K_i^P \partial K_i^P}.$$

Clearly, if we have a linear proxy constraint function, $[P2]$ will be zero. Here, our approach is to linearize the proxy constraint function. In particular, we will linearize both sides of equation (3.4) at $\tilde{K}_i^E(\theta_0)$ and this turns out to be a good linearization point based on our numerical results. Since we have now linearized the proxy constrain function, we have

$$\left.\frac{\partial p_i}{\partial \tilde{K}_i^P}\right|_{\theta=\theta_0}$$

instead of

$$\left.\frac{\partial p_i}{\partial K_i^P}\right|_{\theta=\theta_0}$$

in section 6.1 and 7.1. Note that, by construction,

$$\left.\frac{\partial p_i}{\partial \tilde{K}_i^P}\right|_{\theta=\theta_0} = \left.\frac{\partial p_i}{\partial \tilde{K}_i^E}\right|_{\theta=\theta_0}.$$

By studying the special drift structure of the forward rates under the LIBOR Market Model, the computational order of $[P1]$ can be reduced to $\mathcal{O}(nd)$ (For example, see Denson and Joshi (2009)). Instead of evaluating

$$\tilde{V}^P(i+1)\frac{\partial K_{i+1}^P}{\partial K_i^P} \equiv \tilde{V}^P(i+1)\frac{\partial \log f(i+1)}{\partial \log f(i)}. \tag{8.5}$$

directly using matrix multiplications, we split this computation into two parts where each part only requires a computation order of $\mathcal{O}(nd)$. We first compute

$$Y_j(i) \quad = \quad \sum_{h=j}^{n} \tilde{V}_h^P(i+1) a_h^r(i)^{\mathrm{T}} \tag{8.6}$$

for all j where $Y_j(i)$ is a d-dimensional row vector and $\tilde{V}_h^P(i+1)$ represents the h elements in the vector $\tilde{V}^P(i+1)$. The j^{th} element of the expression in (8.5) is then calculated using

$$\tilde{V}_j^P(i+1) + \frac{f_j(i)\tau_j a_j^r(i)}{(1+f_j(i)\tau_j)^2} Y_j(i). \tag{8.7}$$

Hence, the expression in (8.5) can be evaluated with a computational order of $\mathcal{O}(nd)$.

8.2.2. *Fast Deltas under the Pathwise Minimal Partial Proxy Method.* Similarly, under the Pathwise Minimal Partial Proxy Method, pathwise deltas are given by

$$V^M(0)\frac{\partial K_0^M}{\partial \theta}$$

where

$$\frac{\partial K_0^M}{\partial \theta} = \text{diag}\left(\frac{1}{f_1(0)}, \frac{1}{f_2(0)}, \ldots, \frac{1}{f_n(0)}\right)$$

as

$$\frac{\partial F_{i-1}}{\partial \theta} = \mathbf{0} \quad \text{and} \quad \frac{\partial \hat{F}_{i-1}}{\partial \theta} = \mathbf{0} \tag{8.8}$$

for all i.

Under the LMM, we have

$$\frac{\partial \hat{K}_i}{\partial K_{i-1}} = \frac{\partial K_i}{\partial K_{i-1}}$$

as diffusion coefficients (i.e pseudo-roots) do not depend on the state variables. Hence, as long as the following expression

$$\left[\tilde{V}^M(i) + \hat{V}(i)\right] \frac{\partial K_i^M}{\partial K_{i-1}^M}$$

can be evaluated with a computational order of $\mathcal{O}(nd)$ then the overall computational order for deltas will be $\mathcal{O}(nd)$ per step. In the previous sections, we have shown how this can be done.

8.3. **Fast Elementary Vegas.** For the purpose of the LMM, we are usually interested in the elementary vegas, which is defined to be price sensitivity with respect to the pseudo-square root element a_{js}. We have n steps in our simulation and, at each step of our simulation, there are $n \times d$ pseudo-square root elements. Hence, in total, there are $n^2 d$ price sensitivities to be evaluated (i.e with respect to all entries of the pseudo-square roots). Elementary vegas are usually converted to market vegas (see, Joshi and Kwon (2009)) for hedging purposes. Under the LMM, all elementary vegas can be evaluated with a computational order of $\mathcal{O}(nd)$ per step.

8.3.1. *Fast Elementary Vegas under the Pathwise Partial Proxy Method.* Under the Pathwise Partial Proxy Method, the pathwise price sensitivity with respect to $a_{js}(i)$ is given by

$$\tilde{V}^P(i+1)\frac{\partial F_i}{\partial a_{js}(i)} - B^P(i+1) \tag{8.9}$$

where

$$B^P(i+1) = g^P C^P(i+1)\frac{\partial S_i}{\partial a_{js}(i)}$$

Note that, once pathwise deltas are calculated, we have $\tilde{V}^P(i+1)$ and $C^P(i+1)$.

In order to show that all elementary vegas can be evaluated with a computational order of $\mathcal{O}(nd)$ per step, we will first look at the computation of $B^P(i+1)$. Based on the definition of S_i, we have

$$S_i = (a_{11}(i), a_{21}(i), \ldots, a_{n1}(i))^{\mathrm{T}}$$

Hence, if $s = 1$

$$\begin{aligned} B^P(i+1) &= g^P C^P(i+1)\frac{\partial S_i}{\partial a_{j1}(i)}, \\ &= g^P C_j^P(i+1), \end{aligned} \tag{8.10}$$

else

$$B^P(i+1) = 0,$$

where $C_j^P(i+1)$ represent the j^{th} element in the row vector $C^P(i+1)$. Hence, the computation of $B^P(i+1)$ is trivial once $C^P(i+1)$ is known.

Using the results from Denson and Joshi (2009), we have

$$\tilde{V}^P(i+1)\frac{\partial F_i}{\partial a_{js}(i)} = (L_{js}(i) - a_{js}(i) + Z_s(i+1))\tilde{V}_j^P(i+1) + \frac{f_j(i)\tau_j}{1+f_j(i)\tau_j}M_{js}(i) \tag{8.11}$$

where

$$\begin{aligned} M_{js}(i) &= \sum_{h=j}^{n} \tilde{V}_h^P(i+1)a_{hs}(i) \\ L_{js}(i) &= \sum_{h=\eta(i)}^{j} \frac{f_h(i)\tau_h}{1+f_h(i)\tau_h} a_{hs}(i) \\ Z_s(i+1) &\equiv Z_{i+1,s} \end{aligned} \tag{8.12}$$

As $M_{js}(i)$ and $L_{js}(i)$ can be calculated recursively, the computation order of $M_{js}(i)$ and $L_{js}(i)$ for all j and s at time t_i is at most $\mathcal{O}(nd)$. Therefore, price sensitivities with respect to all entries of the pseudo-square root at time t_i can be evaluated with a computational order of $\mathcal{O}(nd)$ and we conclude that all elementary vegas can be evaluated with a computational order of $\mathcal{O}(nd)$ per step.

8.3.2. *Fast Elementary vegas under the Pathwise Minimal Partial Proxy Method.* Similarly, under the Pathwise Minimal Partial Proxy Method, the pathwise price sensitivity with respect to $a_{js}(i)$ is given by

$$\tilde{V}(i+1)\frac{\partial F_i}{\partial a_{js}(i)} + \hat{V}(i+1)\frac{\partial \hat{F}_i}{\partial a_{js}(i)}$$

and, as usual, once pathwise deltas are calculated, we have $\tilde{V}(i+1)$ and $\hat{V}(i+1)$. Recall that the only difference between F_i and $\hat{F}_i$ is that under $\hat{F}_i$, we condition on $Z_{i+1,1} = Z^*_{i+1,1}$. Therefore, as long as $s \neq 1$, the pathwise price sensitivity is given by

$$\left[\tilde{V}(i+1) + \hat{V}(i+1)\right] \frac{\partial F_i}{\partial a_{js}(i)}. \tag{8.13}$$

else, if $s = 1$, the pathwise price sensitivity is given by

$$\left[\tilde{V}(i+1) + \hat{V}(i+1)\right] \frac{\partial F_i}{\partial a_{js}(i)} + \left(Z_1^*(i+1) - Z_1(i+1)\right)\hat{V}_j(i+1). \tag{8.14}$$

where

$$Z_1^*(i+1) - Z_1(i+1) \equiv Z^*_{i+1,1} - Z_{i+1,1}.$$

Using the results from section 8.3.1, we conclude that all elementary vegas can be calculated with a computational order of $\mathcal{O}(nd)$ per step.

9. Numerical Tests: Model and Product Specifications

For our numerical tests, we consider computing deltas and vegas for digital caplets, digital CMS, target redemption notes with LIBOR floater (LIBOR TARN), and target redemption notes with CMS floater (CMS TARN) and we use the LMM as the benchmark model.

9.1. **LIBOR Market Model Specification.** For our numerical tests, we model semi-annual forward rates with a flat volatility structure of 20% and a tenor structure of

$$t_0 = 0 < t_1 < \ldots < t_{n+1}$$

where $t_j = 0.5 \times j$. We also assume that the Brownian driver of LMM has a correlation of $\rho_{ij} = 0.5 + 0.5\exp(-0.2|t_i - t_j|)$ and reduced to the first 5 factors. Instead of evolving the forward rates directly, the log of the forward rates will be evolved across each tenor date.

9.2. **Product Descriptions and Specifications.** Here, we provide a brief description of the products used for the numerical tests (For detailed explanations, see Fries (2007b)).

9.2.1. *Digital Caplets.* A digital caplet will pay 1 if the underlying forward rate and finishes above the strike or zero otherwise. A small perturbation of the forward rate can shift a digital caplet from finishing out-of-money to in-the-money resulting a discontinuous pathwise value.

For our numerical test, we use a 5-year digital caplet with a forward rate resetting on year 5 as the underlying (i.e $f_{10}(t_{10})$). We assume that this digital caplet has a strike of 10% and the initial LIBOR curve is flat at 10%. As for the proxy constraint function, we will use the log of the forward rate (i.e a linear proxy constraint function) resetting at year 5.

9.2.2. *Digital CMS.* A digital CMS is exactly the same as a digital caplet except that, instead of using a forward rate, a constant maturity swap-rate (CMS) will be used as the underlying quantity of the contract. For the purpose of our numerical tests, we compute the price sensitivities of a 5-year digital CMS with a 5-year constant maturity swap-rate as the underlying. We further assume that this digital CMS has a strike of 10% and the initial LIBOR curve is flat at 10%. Here, the log of the CMS resetting at year 5 will be used as the proxy constraint function.

9.2.3. *LIBOR TARN.* A LIBOR TARN has the following features,

- similar to a callable bond,
- pays large initial coupon followed by inverse floating coupons (e.g. $\max(10\% - 2f_j(t_j), 0)$) determined by the forward rate on reset. [6]
- will be redeemed once the total coupon paid reaches the target coupon or at maturity whichever is earlier.

After the initial coupon, subsequent coupons will only be paid if the underlying forward rate is low and no coupon payments will be made if the underlying forward rate is high. Hence, in an upward sloping forward curve environment, LIBOR TARN will either be redeemed very early in the life of the contact or at maturity. Small changes to the model inputs can change the timing of redemption and causes pathwise discontinuities.

For our numerical tests, we consider a 5.5-year LIBOR TARN with zero initial coupon and has a target coupon of 9%. We further assume that this TARN pays semi-annual inverse floating coupons with a rate of $\max(10\% - 2f_j(t_j), 0)$. We assume that coupons are determined at the beginning of the reset date and payable at the end of the reset period. Since LIBOR TARNs are known to have strong pay-off discontinuity effects in an upward sloping interest-rate environment, we assume that the LIBOR forward rates increase linearly from $f_1 = 2.5\%$ to $f_{10} = 11.5\%$ so that the probability of early redemption is close to 0.5 As for the proxy constraint function, we will use the log of the forward rate (i.e a linear proxy constraint function) resetting at each tenor date.

9.2.4. *CMS TARN.* Similar to the LIBOR TARN, a CMS TARN will have all the same features except that a constant maturity swap-rate (CMS) on each reset date will be used to determine the coupon payments. Therefore, under an upward sloping interest-rate environment, a small change to model inputs can shift the timing of the redemption results in pathwise discontinuities.

For the purpose of our numerical test, we consider a 5.5-year CMS TARN with zero initial coupon and has a target coupon of 9%. We further assume that this TARN pays semi-annual inverse floating coupons of $\max(10\% - 2\,\mathrm{CMS}_j, 0)$ where CMS_j is the 5-year constant maturity swap-rate resetting at time t_j. We assume that coupons are determined at the beginning of the reset date and payable at the end of the reset period. Since CMS TARNs are known to have strong pay-off discontinuity effects in an upward sloping interest rate environment, we assume that the LIBOR forward rates

[6] For our numerical test, we assume that coupons are determined at the beginning of the reset date and payable at the end of the reset period.

increase linearly from $f_1 = 1\%$ to $f_{19} = 10\%$. To prevent pathwise discontinuities, the log of the constant maturity swap rate resetting at each tenor date will be used as the proxy constraint.

10. Numerical Results

In this section, we present numerical results. Deltas and vegas are evaluated using the naive bump and revalue method, the pathwise partial proxy method and the pathwise minimal partial proxy method (pathwise MPP). We do not present results for the partial proxy and minimal partial proxy methods; their standard errors would agree with the path-wise methods (which are their limits) up to a small discretization bias, and their timings would be similar to the bump and revalue method but slightly slower due to the extra computations required. Here, instead of presenting the numerical results for all elementary vegas, we present the price sensitivities with respect to the volatility of each forward rate. These can be obtained by summing the weighted elementary vegas

$$\frac{\partial\,\mathrm{Price}}{\partial\sigma_j} = \sum_s \sum_i \frac{a_{js}(i)}{\sigma_j}\frac{\partial\,\mathrm{Price}}{\partial a_{js}(i)},$$

where σ_j is the volatility for f_j and, as specify earlier, $\sigma_j = 0.2$ for all j in our numerical tests. For cases where the log of a swap-rate (i.e. non-linear proxy constraint) is used as a proxy constraint function, two sets of numerical results will be presented for the pathwise partial proxy method - using the exact non-linear proxy constraint function (pathwise PP) and using the linearized proxy constraint function (pathwise PP(L)). For the bump and revalue method, deltas and vegas are calculated applying the finite differences to prices obtained using the based inputs and perturbed inputs. For deltas, the perturbed inputs are obtained by shifting the relevant initial forward rate in base inputs by 1 basis point, while the perturbed inputs for vegas are obtained by shifting the base volatility by 10 basis points (i.e. $\sigma_j + 0.1\%$). Note that, under the bump and revalue method, vegas are calculated directly instead of taking the weighted sum of the elementary vegas. The numerical results are presented in the following order: we first compare the standard errors of deltas and vegas; we then consider the time taken to compute all the deltas and vegas; and lastly, we show that there is no significant bias due to linearizing the proxy constraint function.

10.1. Standard Errors.

In this section, we compare the standard errors of deltas and vegas calculated using the bump and revalue method, the pathwise partial proxy method and the pathwise minimal partial proxy method. The means and the standard errors of deltas and vegas for all the benchmark products are calculated using $10,000$ batches of simulations, each with 5000 paths and the results are presented in table 10.1, 10.2, 10.3 and 10.4.

Overall, deltas and vegas evaluated using the pathwise MPP method have the lowest standard errors followed by the pathwise PP method and the naive bump and revalue method. In particular, the pathwise MPP method performs significantly better than both the pathwise PP method and the naive bump and revalue method when it comes to the computation of vegas. As we can see from table 10.1, the vega for digital caplet (corresponding to f_{10}) evaluated using the pathwise MPP method has a variance approximately 36 times and 256 times lower than the pathwise PP method and the naive bump and revalue method, respectively. Variance reductions of similar magnitude for vegas evaluated using the pathwise MPP method can also be observed for the digital CMS, the LIBOR TARN and the CMS TARN. As for deltas, the results vary depending on the benchmark product. For the digital caplet, the digital CMS and the CMS TARN, both the pathwise PP method and the pathwise MPP method give similar standard errors for deltas. However, for the LIBOR TARN, the standard errors of deltas evaluated using the pathwise MPP method are significantly lower than the pathwise PP method. From the numerical results, we can also see that there is no significant differences between using an exact proxy constraint function and a linearized proxy constraint function since both the pathwise PP method and the pathwise PP(L) method give similar means and standard errors.

Another interesting observation is that, for the LIBOR TARN and the CMS TARN, the naive bump and revalue method produces stable deltas and vegas for forward rates resetting close to the maturity of the contract (see table 10.2 and 10.4). This is because, as explained earlier, in an upward sloping interest-rate environment, the LIBOR TARN and the CMS TARN will either be redeemed very early in the life of the contact or at maturity. Pathwise discontinuities are the results of early redemption, which in turn depends only on forward rates or swap-rates with early reset date. Any forward rate or swap-rate with reset date close to the maturity of the contract will not have a significant impact on the early redemption and hence, bumping and shifting such forward rates will not result in pathwise discontinuities.

10.2. **Timing Consideration.** In this section, we will consider the time taken to compute all deltas and vegas for the benchmark products. Tables 10.5 and table 10.6 show the time required to compute all deltas and both deltas and vegas combined for all the benchmark products using 2^{16} paths. This numerical test was carried out using a 3.16Ghz Intel Core 2 Duo PC with 4 Gb of RAM, with single threaded C++ code.

Under the naive bump and revalue method, the time taken is directly proportional to the number of deltas and vegas. This is because every price sensitivity has to be evaluated individually. Therefore, the computational time for the naive bump and revalue method is significantly higher compared to the pathwise PP method and the pathwise MPP method. Note that we have restricted to only computing one vegas per forward rate, in a full application there may well be many more: for example, one might want sensitivities to co-terminal swaption implied volatilities as well as caplets. As we can see from tables 10.5 and 10.6, the time taken to computes all deltas and vegas is similar between the pathwise MPP method and the pathwise PP method with a linear or a linearized proxy constraint function. For cases where a non-linear proxy constraint function is used, the computational time is slightly higher as expected.

10.3. **Bias due to the Linearization of the Proxy Constraint Functions.** For the digital CMS and the CMS TARN, deltas and vegas calculated using the linearized proxy constraint function may be biased. By construction of the minimal partial proxy simulation scheme, we have to linearize the proxy constraint function and under the partial proxy simulation scheme, we choose to linearize the proxy constraint function to reduce the computational order at each step. In order to study the magnitude of the bias, deltas and vegas for the digital CMS and the CMS TARN are evaluated using 2^{27} paths of Sobol random numbers. We compare the deltas and vegas obtained using the pathwise MPP method and the pathwise PP method with linearized proxy constraint function (pathwise PP(L)) against the pathwise PP method using the exact proxy constraint function (pathwise PP). Results obtained from the pathwise PP approach using the exact proxy constraint function are unbiased estimators of deltas and vegas as the condition (4) in Theorem 1 holds with probability of 1. From tables 10.7 and 10.8, we see that the linearization of the proxy constraint function causes no significant biases to the deltas and vegas.

11. Further Discussion

11.1. **Advantages of the Pathwise Partial Proxy Method.** While our numerical results indicate that the pathwise MPP method is better than the pathwise PP method, we must remember that the pathwise MPP method can only be applied to financial products with one discontinuity at each time step, and so for products such as the double digital caplet, it is not applicable.

Another significant practical advantage offered by the pathwise PP method is that we can eliminate the need to differentiate the pay-off function. As explained by Giles (2009), the use of scripting in

	Delta						Vega					
	Bump and Revalue		Pathwise PP		Pathwise MPP		Bump and Revalue		Pathwise PP		Pathwise MPP	
Forward Rate	mean	std error	mean	std error	mean	std error	mean	std error	mean	std error	mean	std error
f_1	-11.48%	6.14%	-11.46%	0.08%	-11.46%	0.08%	0.00%	1.43%	0.00%	0.02%	0.00%	0.02%
f_2	-11.58%	8.21%	-11.46%	0.10%	-11.46%	0.09%	-0.02%	2.01%	0.00%	0.04%	0.00%	0.03%
f_3	-11.55%	10.64%	-11.46%	0.11%	-11.46%	0.10%	-0.02%	2.49%	0.01%	0.05%	0.01%	0.04%
f_4	-11.52%	12.86%	-11.45%	0.14%	-11.45%	0.11%	-0.02%	2.94%	0.01%	0.07%	0.01%	0.06%
f_5	-11.56%	14.54%	-11.45%	0.17%	-11.45%	0.13%	-0.03%	3.30%	0.01%	0.09%	0.01%	0.07%
f_6	-11.54%	16.35%	-11.45%	0.20%	-11.45%	0.15%	-0.03%	3.69%	0.01%	0.11%	0.01%	0.08%
f_7	-11.62%	17.78%	-11.45%	0.24%	-11.45%	0.17%	-0.04%	4.05%	0.02%	0.14%	0.02%	0.10%
f_8	-11.66%	19.44%	-11.44%	0.29%	-11.44%	0.20%	-0.05%	4.44%	0.02%	0.17%	0.02%	0.12%
f_9	-11.60%	21.32%	-11.44%	0.34%	-11.44%	0.24%	-0.06%	4.82%	0.02%	0.21%	0.02%	0.14%
f_{10}	496.15%	245.65%	498.15%	35.39%	498.21%	24.06%	-25.53%	16.50%	-25.41%	6.31%	-25.46%	1.07%

Table 10.1. *Digital Caplet: Means and standard errors for deltas and vegas evaluated using the bump and revalue approach, the pathwise partial proxy approach (pathwise PP) and the pathwise minimal partial proxy approach (pathwise MPP).*

f_2	-12.45%	9.70%	-12.51%	0.08%	-12.51%	0.08%	-12.51%	0.06%	0.01%	2.12%	0.01%	0.04%	0.01%	0.04%	0.01%	0.03%
f_3	-12.43%	11.83%	-12.51%	0.10%	-12.51%	0.10%	-12.51%	0.08%	0.02%	2.61%	0.01%	0.06%	0.01%	0.06%	0.01%	0.04%
f_4	-12.49%	13.30%	-12.51%	0.13%	-12.51%	0.13%	-12.51%	0.10%	0.02%	3.06%	0.01%	0.08%	0.01%	0.08%	0.01%	0.06%
f_5	-12.39%	15.29%	-12.51%	0.16%	-12.51%	0.16%	-12.50%	0.11%	0.02%	3.42%	0.02%	0.10%	0.02%	0.10%	0.02%	0.07%
f_6	-12.38%	16.81%	-12.50%	0.20%	-12.50%	0.20%	-12.50%	0.14%	0.02%	3.75%	0.02%	0.12%	0.02%	0.12%	0.02%	0.08%
f_7	-12.38%	18.27%	-12.50%	0.24%	-12.50%	0.24%	-12.50%	0.16%	0.02%	4.09%	0.02%	0.14%	0.02%	0.14%	0.02%	0.10%
f_8	-12.39%	19.74%	-12.50%	0.29%	-12.50%	0.29%	-12.50%	0.19%	0.04%	4.45%	0.02%	0.17%	0.02%	0.17%	0.03%	0.11%
f_9	-12.41%	21.26%	-12.50%	0.34%	-12.50%	0.34%	-12.49%	0.22%	0.05%	4.81%	0.03%	0.21%	0.03%	0.21%	0.03%	0.13%
f_{10}	51.81%	90.71%	51.98%	5.38%	51.92%	5.45%	51.96%	3.26%	-3.80%	7.41%	-3.78%	1.14%	-3.78%	1.16%	-3.76%	0.29%
f_{11}	65.49%	88.78%	65.61%	5.10%	65.58%	5.16%	65.61%	3.11%	-1.59%	6.71%	-1.62%	1.08%	-1.62%	1.10%	-1.61%	0.24%
f_{12}	62.21%	86.52%	62.61%	4.81%	62.60%	4.85%	62.63%	2.96%	-1.63%	6.01%	-1.59%	1.01%	-1.58%	1.02%	-1.58%	0.19%
f_{13}	59.32%	84.91%	59.76%	4.53%	59.76%	4.55%	59.78%	2.83%	-1.54%	5.57%	-1.50%	0.94%	-1.50%	0.95%	-1.49%	0.16%
f_{14}	57.00%	83.27%	57.04%	4.26%	57.05%	4.27%	57.07%	2.70%	-1.36%	5.38%	-1.37%	0.88%	-1.37%	0.88%	-1.37%	0.15%
f_{15}	54.63%	81.83%	54.43%	4.01%	54.45%	4.01%	54.47%	2.58%	-1.14%	5.32%	-1.23%	0.83%	-1.22%	0.82%	-1.22%	0.15%
f_{16}	52.10%	80.00%	51.93%	3.78%	51.96%	3.76%	51.98%	2.47%	-1.01%	5.44%	-1.08%	0.79%	-1.07%	0.78%	-1.07%	0.17%
f_{17}	49.70%	78.14%	49.54%	3.56%	49.57%	3.52%	49.58%	2.37%	-0.85%	5.79%	-0.92%	0.75%	-0.91%	0.73%	-0.91%	0.18%
f_{18}	47.04%	76.08%	47.26%	3.35%	47.28%	3.29%	47.29%	2.27%	-0.63%	6.23%	-0.73%	0.72%	-0.72%	0.70%	-0.72%	0.21%
f_{19}	44.87%	74.46%	45.07%	3.15%	45.08%	3.07%	45.08%	2.17%	-0.53%	6.57%	-0.57%	0.68%	-0.57%	0.66%	-0.57%	0.24%

Table 10.2. *Digital CMS: Means and standard errors for deltas and vegas evaluated using the naive bump and revalue method, the pathwise partial proxy method with linearized constraint (pathwise PP(L)), the pathwise partial proxy method (pathwise PP) and the pathwise minimal partial proxy method (pathwise MPP).*

	Delta						Vega					
	Bump and Revalue		Pathwise PP		Pathwise MPP		Bump and Revalue		Pathwise PP		Pathwise MPP	
Forward Rate	mean	std error	mean	std error	mean	std error	mean	std error	mean	std error	mean	std error
f_1	-585.36%	150.62%	-587.70%	126.07%	-587.56%	43.12%	1.32%	4.86%	1.27%	2.35%	1.28%	0.59%
f_2	-580.88%	150.91%	-583.11%	86.43%	-583.48%	39.39%	2.90%	5.11%	2.90%	3.19%	2.89%	1.39%
f_3	-400.46%	131.60%	-402.25%	64.19%	-402.01%	27.28%	1.95%	4.48%	1.91%	3.96%	1.92%	0.83%
f_4	-74.38%	45.60%	-74.60%	52.77%	-74.64%	12.30%	2.28%	3.39%	2.26%	4.50%	2.25%	0.90%
f_5	-28.40%	18.22%	-28.34%	44.36%	-28.27%	6.98%	0.46%	1.96%	0.48%	4.95%	0.47%	0.95%
f_6	-19.82%	8.67%	-19.95%	37.84%	-19.78%	4.36%	-0.31%	1.21%	-0.32%	5.35%	-0.32%	0.92%
f_7	-17.09%	4.24%	-17.14%	32.83%	-17.13%	2.98%	-0.76%	0.73%	-0.76%	5.71%	-0.77%	0.88%
f_8	-16.05%	2.10%	-16.20%	28.94%	-16.06%	2.21%	-1.05%	0.41%	-1.04%	5.98%	-1.05%	0.85%
f_9	-15.48%	0.86%	-15.47%	25.60%	-15.49%	1.74%	-1.26%	0.20%	-1.25%	6.21%	-1.25%	0.84%
f_{10}	-15.10%	0.16%	-15.09%	16.56%	-15.11%	1.37%	-1.42%	0.09%	-1.43%	4.74%	-1.42%	0.80%

Table 10.3. *LIBOR TARN: Means and standard errors for deltas and vegas evaluated using the bump and revalue approach, the pathwise partial proxy approach (pathwise PP) and the pathwise minimal partial proxy approach (pathwise MPP).*

f_2	-82.45%	25.99%	-83.07%	5.11%	-83.08%	5.12%	-83.07%	3.32%	0.09%	0.70%	0.09%	0.07%	0.09%	0.07%	0.09%	0.04%
f_3	-96.64%	31.66%	-97.19%	4.52%	-97.25%	4.54%	-97.19%	4.47%	0.08%	0.92%	0.07%	0.11%	0.08%	0.12%	0.08%	0.06%
f_4	-92.33%	34.15%	-92.88%	4.23%	-92.94%	4.24%	-92.90%	4.40%	-0.15%	1.05%	-0.16%	0.16%	-0.15%	0.16%	-0.16%	0.07%
f_5	-90.23%	34.62%	-90.88%	4.21%	-90.93%	4.23%	-90.92%	4.35%	-0.21%	1.13%	-0.23%	0.21%	-0.23%	0.21%	-0.23%	0.08%
f_6	-88.93%	34.50%	-89.64%	4.11%	-89.67%	4.12%	-89.63%	4.28%	-0.25%	1.16%	-0.27%	0.25%	-0.27%	0.26%	-0.27%	0.09%
f_7	-87.62%	34.25%	-88.25%	3.97%	-88.27%	3.98%	-88.23%	4.19%	-0.30%	1.14%	-0.31%	0.30%	-0.31%	0.31%	-0.31%	0.10%
f_8	-86.15%	33.83%	-86.79%	3.87%	-86.79%	3.87%	-86.75%	4.11%	-0.35%	1.15%	-0.36%	0.35%	-0.36%	0.36%	-0.36%	0.11%
f_9	-84.61%	33.40%	-85.26%	3.82%	-85.24%	3.81%	-85.20%	4.02%	-0.40%	1.19%	-0.42%	0.40%	-0.42%	0.41%	-0.42%	0.12%
f_{10}	-82.93%	33.08%	-83.69%	3.29%	-83.65%	3.28%	-83.62%	3.93%	-0.47%	1.26%	-0.48%	0.35%	-0.48%	0.36%	-0.48%	0.14%
f_{11}	-44.43%	28.07%	-45.19%	2.77%	-45.11%	2.75%	-45.10%	2.57%	0.72%	1.28%	0.70%	0.33%	0.70%	0.33%	0.70%	0.11%
f_{12}	-27.59%	22.48%	-28.14%	2.31%	-28.08%	2.30%	-28.06%	1.49%	0.56%	1.22%	0.54%	0.33%	0.55%	0.33%	0.55%	0.09%
f_{13}	-12.63%	14.67%	-12.66%	1.86%	-12.68%	1.85%	-12.70%	0.91%	0.47%	0.97%	0.47%	0.34%	0.47%	0.34%	0.47%	0.08%
f_{14}	-4.49%	7.67%	-4.39%	1.58%	-4.41%	1.57%	-4.43%	0.57%	0.36%	0.65%	0.35%	0.33%	0.35%	0.32%	0.35%	0.08%
f_{15}	-1.71%	4.03%	-1.65%	1.44%	-1.67%	1.43%	-1.66%	0.43%	0.20%	0.41%	0.20%	0.31%	0.20%	0.31%	0.21%	0.08%
f_{16}	-0.61%	2.03%	-0.63%	1.29%	-0.64%	1.28%	-0.65%	0.34%	0.11%	0.27%	0.11%	0.30%	0.11%	0.29%	0.11%	0.07%
f_{17}	-0.22%	1.02%	-0.22%	1.15%	-0.23%	1.14%	-0.24%	0.27%	0.05%	0.16%	0.05%	0.28%	0.05%	0.27%	0.05%	0.07%
f_{18}	-0.07%	0.46%	-0.04%	1.05%	-0.05%	1.05%	-0.06%	0.21%	0.02%	0.07%	0.02%	0.26%	0.02%	0.25%	0.01%	0.06%
f_{19}	0.00%	0.00%	0.01%	1.00%	0.01%	0.99%	0.00%	0.15%	0.00%	0.00%	0.00%	0.25%	0.00%	0.24%	0.00%	0.05%

Table 10.4. *CMS TARN: Means and standard errors for deltas and vegas evaluated using the naive bump and revalue method, the pathwise partial proxy method with linearized constraint (pathwise PP(L)), the pathwise partial proxy method (pathwise PP) and the pathwise minimal partial proxy method (pathwise MPP).*

	Bump and Revalue	Pathwise PP(L)	Pathwise PP	Pathwise MPP
Digital Caplet	9.65	n/a	2.35	2.42
Digital CMS	42.28	5.02	6.60	5.01
LIBOR TARN	7.03	n/a	2.08	2.25
CMS TARN	30.92	5.05	8.25	5.59

Table 10.5. *Time (in second) taken to compute all deltas using* 2^{16} *paths.*

	Bump and Revalue	Pathwise PP(L)	Pathwise PP	Pathwise MPP
Digital Caplet	18.42	n/a	4.31	4.37
Digital CMS	82.45	8.94	10.53	9.07
LIBOR TARN	13.42	n/a	3.28	3.57
CMS TARN	60.29	7.50	10.72	8.25

Table 10.6. *Time (in second) taken to compute both deltas and vegas combined using* 2^{16} *paths.*

	Pathwise PP(L)		Pathwise PP		Pathwise MPP	
Forward Rates	deltas	vegas	deltas	vegas	deltas	vegas
f_1	-12.517%	0.003%	-12.517%	0.003%	-12.517%	0.003%
f_2	-12.514%	0.006%	-12.514%	0.006%	-12.514%	0.006%
f_3	-12.511%	0.009%	-12.511%	0.009%	-12.511%	0.009%
f_4	-12.508%	0.012%	-12.508%	0.012%	-12.508%	0.012%
f_5	-12.505%	0.016%	-12.505%	0.015%	-12.505%	0.015%
f_6	-12.501%	0.019%	-12.502%	0.019%	-12.502%	0.018%
f_7	-12.498%	0.022%	-12.498%	0.022%	-12.499%	0.022%
f_8	-12.495%	0.026%	-12.495%	0.026%	-12.496%	0.025%
f_9	-12.492%	0.029%	-12.493%	0.029%	-12.493%	0.029%
f_{10}	52.041%	-3.770%	51.983%	-3.767%	51.973%	-3.766%
f_{11}	65.669%	-1.617%	65.636%	-1.614%	65.628%	-1.613%
f_{12}	62.664%	-1.584%	62.651%	-1.581%	62.642%	-1.581%
f_{13}	59.805%	-1.500%	59.810%	-1.498%	59.800%	-1.498%
f_{14}	57.083%	-1.370%	57.100%	-1.368%	57.091%	-1.368%
f_{15}	54.480%	-1.220%	54.504%	-1.218%	54.495%	-1.219%
f_{16}	51.985%	-1.066%	52.012%	-1.064%	52.003%	-1.066%
f_{17}	49.596%	-0.906%	49.621%	-0.904%	49.612%	-0.905%
f_{18}	47.312%	-0.717%	47.329%	-0.715%	47.320%	-0.717%
f_{19}	45.114%	-0.562%	45.117%	-0.560%	45.109%	-0.561%

Table 10.7. *Digital CMS: Deltas and vegas evaluated using* 2^{27} *paths. The unbiased estimators for deltas and vegas are given by the pathwise partial proxy approach using the exact proxy constraint function (pathwise PP).*

real-world implementations means it can be very difficult in practice to evaluate the pay-off derivative of very complex path-dependent financial products. Our approach here has some similarities to his method. However, while Giles' method can only be applied to a full-factor model, ours can be applied to a reduced-factor model. Our approach requires the pay-off at time t_i to be a function of the proxy constraint function from t_1 up to t_i. Since we have the freedom to choose the proxy constraint function, the pay-off function of most financial products can be written in this form (e.g. CMS TARN and LIBOR TARN). We also emphasize that this idea can also be applied to financial products with continuous pay-off functions.

Recall that, under the partial proxy simulation scheme, the accumulated deflated pay-off generated by a financial product is given by

$$g^P(\theta) := g(K_1^P(\theta), K_2^P(\theta), \ldots, K_m^P(\theta))$$

	Pathwise PP(L)		Pathwise PP		Pathwise MPP	
Forward Rates	deltas	vegas	deltas	vegas	deltas	vegas
f_1	-64.873%	0.022%	-64.878%	0.021%	-64.873%	0.021%
f_2	-83.150%	0.087%	-83.160%	0.088%	-83.151%	0.088%
f_3	-97.228%	0.076%	-97.282%	0.080%	-97.271%	0.080%
f_4	-92.946%	-0.161%	-92.998%	-0.157%	-92.988%	-0.156%
f_5	-90.969%	-0.231%	-91.011%	-0.228%	-90.999%	-0.228%
f_6	-89.689%	-0.274%	-89.719%	-0.271%	-89.706%	-0.270%
f_7	-88.309%	-0.315%	-88.327%	-0.313%	-88.318%	-0.312%
f_8	-86.837%	-0.362%	-86.844%	-0.361%	-86.834%	-0.360%
f_9	-85.307%	-0.418%	-85.296%	-0.416%	-85.284%	-0.416%
f_{10}	-83.749%	-0.484%	-83.711%	-0.481%	-83.700%	-0.481%
f_{11}	-45.205%	0.700%	-45.134%	0.702%	-45.127%	0.703%
f_{12}	-28.130%	0.543%	-28.062%	0.547%	-28.058%	0.547%
f_{13}	-12.673%	0.471%	-12.696%	0.474%	-12.694%	0.474%
f_{14}	-4.398%	0.347%	-4.424%	0.349%	-4.421%	0.349%
f_{15}	-1.641%	0.205%	-1.655%	0.206%	-1.654%	0.206%
f_{16}	-0.644%	0.109%	-0.651%	0.110%	-0.650%	0.110%
f_{17}	-0.235%	0.050%	-0.239%	0.051%	-0.237%	0.050%
f_{18}	-0.061%	0.016%	-0.062%	0.016%	-0.061%	0.015%
f_{19}	0.001%	0.000%	0.001%	0.000%	0.000%	0.000%

Table 10.8. *CMS TARN: Deltas and vegas evaluated using 2^{27} paths. The unbiased estimators for deltas and vegas are given by the pathwise partial proxy approach using the exact proxy constraint function (pathwise PP).*

and it can be expressed as

$$g^P(\theta) = \sum_{i=1}^{m} \frac{N_0^P(\theta)}{N_i^P(\theta)} \tilde{g}_i^P(\theta)$$

where $N_i^P(\theta)$ represents the value of the numeraire at time t_i and

$$\tilde{g}_i^P(\theta) := \tilde{g}_i(K_1^P(\theta), K_2^P(\theta), \ldots, K_i^P(\theta))$$

represents the pay-off at time t_i and is a $\mathcal{F}_{t_i}$-measurable function. Therefore, we have

$$\left.\frac{\partial g^P(\theta)}{\partial \theta}\right|_{\theta=\theta_0} = \left.\left[\sum_{i=1}^{m} \tilde{g}_i^P(\theta) \frac{\partial}{\partial \theta}\left(\frac{N_0^P(\theta)}{N_i^P(\theta)}\right) + \sum_{i=1}^{m} \frac{N_0^P(\theta)}{N_i^P(\theta)} \frac{\partial \tilde{g}_i^P(\theta)}{\partial \theta}\right]\right|_{\theta=\theta_0} \tag{11.1}$$

From an implementation perspective, it is the computation of

$$\left.\frac{\partial \tilde{g}_i^P(\theta)}{\partial \theta}\right|_{\theta=\theta_0}$$

that poses the greatest difficulty.

Suppose that the pay-off at time t_i can also be written in this form,

$$\tilde{g}_i^P(\theta) = \tilde{f}_i(p_1(K_1^P(\theta)), p_2(K_2^P(\theta)), \ldots, p_i(K_i^P(\theta)))$$

(i.e. the pay-off is a function of the proxy constraint functions.) where $p_i(K_i^P(\theta))$ is the proxy constraint function as defined earlier, then we know that

$$\left.\frac{\partial \tilde{g}_i^P(\theta)}{\partial \theta}\right|_{\theta=\theta_0} = \mathbf{0}.$$

This is because by construction of the partial proxy simulation scheme, we have

$$p_i(K_i^P(\theta_0)) = p_i(K_i^P(\theta_B))$$

where $\theta_B = \theta_0 + h\mathbf{u}$ for all i. Hence, equation (11.1) can be written as

$$\left.\frac{\partial g^P(\theta)}{\partial \theta}\right|_{\theta=\theta_0} = \sum_{i=1}^{m} \left[\tilde{g}_i^P(\theta) \frac{\partial}{\partial \theta}\left(\frac{N_0^P(\theta)}{N_i^P(\theta)}\right)\right]\Bigg|_{\theta=\theta_0}.$$

and we no longer have to compute the derivative of the pay-off function. Computing

$$\left.\frac{\partial}{\partial \theta}\left(\frac{N_0^P(\theta)}{N_i^P(\theta)}\right)\right|_{\theta=\theta_0}$$

is straight-forward as the numeraire is product independent.

12. Conclusion

By introducing a new class of new numerical schemes known as the quasi mean-shifted proxy simulations schemes, we have shown that the pathwise method can be applied to compute price sensitivities for financial products with discontinuous pay-off functions. Using this result, we developed two new methods known as the pathwise partial proxy method (pathwise PP) and the pathwise minimal partial proxy method (pathwise MPP). These methods can be used to compute a large number of price sensitivities simultaneously even for financial products with discontinuous pay-offs. Similar to cases where the pay-off function is continuous, under the LIBOR Market Model, the computational cost per step for all deltas and vegas is proportional to the number of rates times the number of factors using the pathwise PP method and the pathwise MPP method. While numerical results suggest that the pathwise MPP method performs significantly better than the pathwise PP method in term of variance reductions, one significant advantage of the pathwise PP method is that price sensitivities can be evaluated without having to differentiate the pay-off function for certain classes of pay-off functions.

References

[1] Brace, A., Gatarek, D., and Musiela, M. (1997). The market model of interest-rate dynamics, *Mathematical Finance* **7**, 127–155.
[2] Brace, A., (2007). *Engineering BGM,* Chapman and Hall.
[3] Broadie, M., and Glasserman, P. (1996). Estimating security price derivatives using simulation. *Management Science* **42**(2), 269-285.
[4] Denson, N., and Joshi, M. S. (2009). Flaming Logs. *Wilmott Journal* Volume **1** Issue **5-6**, 259-262.
[5] Chan, J. H., and Joshi, M. S. (2009). Minimal Partial Proxy Simulation Schemes for Generic and Robust Monte-Carlo Greeks. http://ssrn.com/abstract=1406368
[6] Fries, C. P (2007a). Localized Proxy Simulation Schemes for Generic and Robust Monte-Carlo Greeks. http://ssrn.com/abstract=984744
[7] Fries, C. P (2007b). *Mathematical Finance. Theory, Modelling, Implementation*. Wiley, New York.
[8] Fries, C. P., and Joshi, M. S. (2008a). Conditional Analytic Monte Carlo Pricing Scheme for Auto-Callable Products. http://ssrn.com/abstract=1125725.
[9] Fries, C. P., and Joshi, M. S. (2008b). Partial proxy simulation schemes for generic and robust Monte Carlo Greeks. *The Journal of Computational Finance* **11**(3),79-106.
[10] Fries, C. P., and Kampen, J. (2007) Proxy simulation schemes for generic robust Monte Carlo sensitivities, process oriented importance sampling and high accuracy drift approximation. *The Journal of Computational Finance* **10**(2), 97-128.
[11] Giles, M. (2008). Vibrato Monte Carlo sensitivities. *In Monte Carlo and Quasi-Monte Carlo Methods 2008*. Springer, New York.
[12] Giles, M., and Glasserman, P., (2006). Smoking Adjoints: fast Monte Carlo Greeks. *Risk, January 2006*, 92-96.
[13] Glasserman, P. (2003). *Monte Carlo Methods in Financial Engineering (Stochactic Modelling and Applied Probability)*. Springer, New York.
[14] Jäckel, P. (2002). *Monte Carlo Methods in Finance.* Wiley, New York.
[15] Joshi, M. S. (2003a). *The Concepts and Practice of Mathematical Finance.* Cambridge University Press, Cambridge.

[16] Joshi, M. S. (2003b). Rapid computation of drifts in a reduced factor LIBOR market model. *Wilmott Magazine*, May.
[17] Joshi, M.S., Kainth, D.S., (2004), Rapid computation of prices and deltas of nth to default swaps in the Li Model, *Quantitative Finance,* volume 4, issue 3, pages 266–275
[18] Joshi, M. S. and Kwon, O. K. (2009). Monte Carlo market Greeks in the displaced diffusion LIBOR market model. Forthcoming.
[19] Piterbarg V. V. (2004). TARNs: models, valuation, risk sensitivities. *Wilmott Magazine* **14**,62-71.

Centre for Actuarial Studies, Department of Economics, University of Melbourne, Victoria 3010, Australia
E-mail address: j.chan23@pgrad.unimelb.edu.au
E-mail address: mark@markjoshi.com