JFE Online Appendix: The QUAD Method

Part of the QUAD technique is the use of quadrature for numerical solution of option pricing problems. Andricopoulos et al. (2003, 2007) use quadrature as the only computational 'engine'; in this concluding paper a second type of engine is added, using closed-form approximations for transition density functions. However, in all three papers, quadrature is used in the same way. In this appendix we outline the technique.

The intuition behind QUAD is that although Equation (1) is not valid across all points in time, the valuation problem can be sliced into consecutive time intervals, during which it is locally applicable.

$$V(x,t) = A(x)\int_{-\infty}^{\infty} B(x,y)V(y,t+\Delta t)dy, \tag{1}$$

Imposition of the appropriate option features at their corresponding observation times provides a link between these consecutive intervals, and solution of complex problems becomes possible and is, by the nature of the technique, extremely fast. Any of the common quadrature methods is applicable, including higher-order schemes and Gaussian quadrature. The algorithm is readily modularized to permit easy interchange of quadrature engines. Calculation is no more complicated than calculating the option value using a tree or finite difference but in QUAD the contribution at the previous timestep comes from many nodes representing different levels of the underlying asset not just two or three and, unlike trees and finite difference grids, calculations are not required in regions between observation times.

Quadrature methods solve an integral equation of the form shown in Equation (1). These are widely described in Mathematics textbooks. We demonstrate for Simpson's rule, which is frequently the superior choice as the numerical engine due to its robustness, simplicity and fast convergence. Simpson's Rule for some integration $\int_a^b f(x)dx$ is:

$$\int_a^b f(x)dx = \int_a^{a+2h} f(x)dx \approx \frac{1}{3}h(f_0 + 4f_1 + f_2)$$

where

$$h = \frac{b-a}{2}$$

This can be derived (and the error bounds and rate of convergence determined) by using quadratic interpolation. Consider a Lagrange interpolating polynomial:

$$\begin{aligned} P(x) = & \frac{(x-a-h)(x-a)}{(a+2h-a-h)(a+2h-a)} f_0 \\ & + \frac{(x-a-2h)(x-a)}{(a+h-a-2h)(a+h-a)} f_1 \\ & + \frac{(x-a-2h)(x-a-h)}{(a-a-2h)(a-a-h)} f_2 \end{aligned}$$

Arranging and simplifying the polynomial, we find:

$$\begin{aligned} P(x) = & \frac{x^2}{h^2}\left(\frac{1}{2}f_0 - f_1 + \frac{1}{2}f_2\right) \\ & + \frac{x}{h^2}\left(-\frac{1}{2}(2a+3h)f_0 + (2a+3h)f_1 - \frac{1}{2}(2a+h)f_2\right) \\ & + \frac{1}{2h^2}\left(\frac{1}{2}(a+h)(a+2h)f_0 - a(a+2h)f_1 + \frac{1}{2}a(a+h)f_2\right) \end{aligned}$$

Integrating and simplifying:

$$\begin{aligned} \int_a^b f(x)dx &= \int_a^{a+2h} P(x)dx \\ &= \frac{1}{3}h(f_0 + 4f_1 + f_2) - \frac{1}{90}h^5 f^{(4)}(\epsilon) \end{aligned}$$

for some $\epsilon \in [a, b]$.

We can always split the region of integration into n smaller regions, approximate the integrand in each of the n regions and sum them back together. This gives the composite Simpson's rule:

$$\begin{aligned} \int_a^{a+2h} f(x)dx = & \frac{1}{3}h(f_0 + 4(f_1 + f_3 + .. + f_{2n-1}) + 2(f_2 + f_4 + .. + f_{2n-2}) + f_{2n}) \\ & - \frac{n}{90}h^5 f^{(4)}(\epsilon) \end{aligned} \tag{2}$$

where $n = \frac{b-a}{2h}$

We observe that the error is bounded by:

$$\frac{h^4}{180}(b-a)\max_{\epsilon\in[a,b]}\left|f^{(4)}(\epsilon)\right|$$

and has convergence rate of order h^4. Other numerical schemes and their error bounds can be derived in similar fashion.

A general formula for QUAD under one dimensional integration is expressed as follows: Given step size $\delta x = \frac{b-a}{N}$,

$$\int_a^b f(x)dx \approx \delta x\, \boldsymbol{q}\cdot\boldsymbol{f}(x)$$

where $\boldsymbol{f}(x)$ is a $\{N+1\}$-dimensional vector with its ith term defined as:

$$\boldsymbol{f}_i(x) = f(a+(i-1)*\delta x) \qquad \forall i\in[1,N+1]$$

and $\boldsymbol{q}$ is $\{N+1\}$-dimensional weighting vector. For example, $\boldsymbol{q}_S$ under Simpson's rule for even N is:

$$\boldsymbol{q}_S = \frac{1}{3}\begin{pmatrix}1\\4\\2\\\vdots\\2\\4\\1\end{pmatrix} \tag{3}$$

For comparison, under the Trapezoidal rule (see Mathematics texts),

$$\boldsymbol{q}_T = \frac{1}{2}\begin{pmatrix} 1 \\ 2 \\ \vdots \\ 2 \\ 1 \end{pmatrix} \tag{4}$$

In order to integrate via quadrature, the integrand should have continuous derivatives. The first derivative of the payoff on a call option is clearly discontinuous at the strike price. If this discontinuity is avoided, convergence will be smooth. We split the integration range into two smaller parts with boundary defined on the discontinuity; the derivatives of the integrand are continuous in both parts of the integration range: $-\infty < y < 0$ and $0 < y < +\infty$. For a European call option, clearly the valuation problem is reduced to evaluating the integral corresponding to the range $0 < y < +\infty$.

For other options this discontinuity may occur in other ways; for a barrier option the discontinuity is in the payoff function itself at the barrier, creating a degree of nonlinearity error, and for an American put it occurs in the second derivative at the free boundary. If the discontinuities are not taken into account, then convergence will not be smooth (though reasonable solutions may still be had if discontinuities are not precisely located, in quick and dirty programming!). The location of discontinuities in the payoff function will be known a priori for some classes of options (such as a vanilla call or put option or a discrete barrier option). For other classes of options (such as a Bermudan option) the position of the discontinuity is calculated at every observation time via Newton-Raphson iteration, which converges rapidly. Options that are awkward to deal with using other lattice/grid techniques are handled via QUAD with relative ease (examples: lookback options in three dimensions and moving barrier options) because quadrature points can be easily and precisely placed.

Practically, the infinite integration range needs to be truncated but truncation error quickly tends to zero with increasing multiples of standard deviation of the underlying asset. If the integration range of y is set corresponding to the movement of asset price within ten standard deviations away from initial price, the truncation error is small enough to be neglected under even the most extreme σ setups. The integration range is then reduced to $0 < y < \xi\sigma\sqrt{\tau} + x_0$ where ξ denotes the number of standard deviations considered.

With the integrand and integration range properly set up, as above, the valuation problem can be solved by Equation (2) as

$$\begin{aligned} V(x,t) &\approx A(x) \int_0^{N\delta y} f(x,y)dy \\ &\approx \frac{A(x)\delta y}{3} \Bigg(f(x,0) + 4 \sum_{i=0}^{N-1} f\left(x, (i+\frac{1}{2})\delta y\right) \\ &+ 2 \sum_{i=1}^{N-1} f\left(x, i\delta y\right) + f(x, N\delta y) \Bigg) \end{aligned}$$

where number of steps, N, and step size, δy, are defined as:

$$N = \frac{\xi\sigma\sqrt{\tau} + x_0}{\delta y} \tag{5}$$

Consequently, the error term is bounded by:

$$\frac{\delta y^4}{180}(\xi\sigma\sqrt{\tau} + x_0) \max_{\epsilon\in[0,N\delta y]} \left| f^{(4)}(\epsilon) \right|$$

which quickly converges to zero as δy becomes small, with a rate of convergence of four. Richardson Extrapolation can be used to improve the rate of convergence further:

$$V_{ext} = \frac{\delta y_1^d V_2 - \delta y_2^d V_1}{\delta y_1^d - \delta y_2^d}$$

where $d = 4$ is the rate of convergence of Simpson's rule, and V_1 and V_2 are prices evaluated with corresponding step sizes δy_1 and δy_2.

The Greeks

Calculation of Greeks is straightforward, via finite differencing on option values for several neighboring values of x and v; for example:

$$
\begin{aligned}
\Delta \equiv V_x(x,v) &\approx \frac{V(x+\delta x, v) - V(x-\delta x, v)}{2\,\delta x} \\
\nu \equiv V_v(x,v) &\approx \frac{V(x, v+\delta v) - V(x, v-\delta v)}{2\,\delta v} \\
\Gamma \equiv V_{xx}(x,v) &\approx \frac{V(x+\delta x, v) - 2V(x,v) + V(x-\delta x, v)}{\delta x^2} \\
\text{Vomma} \equiv V_{vv}(x,v) &\approx \frac{V(x, v+\delta v) - 2V(x,v) + V(x, v-\delta v)}{\delta v^2} \\
\text{Vanna} \equiv V_{xv}(x,v) & \\
&\approx \frac{V(x+\delta x, v+\delta v) - V(x+\delta x, v-\delta v) - V(x-\delta x, v+\delta v) + V(x-\delta x, v-\delta v)}{4\,\delta x\,\delta v}
\end{aligned}
$$

for small δx and δv (such that $\frac{\delta x}{x} \ll 1$ and $\frac{\delta v}{v} \ll 1$).

Other first and second order Greeks can also be calculated in the same fashion.

QUAD with Path-Dependent Options

A discrete path-dependent option is an option whose payoff function is dependent not only on the price of the underlying at the exercise date but also on the price of the underlying at discrete points in time before exercise. Consider a path-dependent option that matures at time T. Suppose that the option and its underlying are monitored M+1 times with M intervals Δt_m from present time t to maturity, such that:

$$T = t + \sum_{i=1}^{M} \Delta t_i$$

The valuation of this option is divided into evaluation of M separate options with maturity $t + \sum_{i=1}^{k+1} \Delta t_i$ at time $t + \sum_{i=1}^{k} \Delta t_i$ $\forall$ integer $k \in [0, M)$. Denote these observation points before maturity as:

$$t_k = t + \sum_{i=1}^{k} \Delta t_i$$

Working backwards in time with known final conditions at maturity, the value of option $V_M(x, t_{M-1})$ with maturity T at t_{M-1} can be priced for the entire range of underlying prices, x. Together with conditions imposed at this observation point, these option prices can then be used as final conditions for the valuation of option $V_M - 1(x, t_{M-2})$ with maturity t_{M-1} at t_{M-2}. This evaluation process is continued until the value of option price $V_1(x, t)$ is found.

The pricing of the option is effectively an M step multinomial tree with number of branches for each node equal to $2N + 1$ subject to discontinuous boundaries, and N is the number of QUAD steps defined in Equation 5. Note that in this multinomial tree the number of nodes does not change after initialization at the first time step and is not subject to discontinuity of the payoff function. Therefore it does not give rise to the saw-tooth effect observed with tree-based models.

In practice, path-dependent options are discretely monitored at discrete points in time and the payoff function of the option depends on a discrete set of underlying prices rather than the continuous pricing function. If the price of a discretely monitored option $V(M)$ converges at rate d to the price of its theoretical continuously monitored counterpart V^* as the number of observation point M tends to infinity, then:

$$V_M = V^* + a\frac{1}{M^d}$$

for some constant a. Richardson extrapolation can be used for calculating option prices with higher observation frequency:

$$\begin{aligned} V_{M_3} =& \frac{M_1^d(M_2^d - M_3^d)V_{M_1} + M_2^d(M_3^d - M_1^d)V_{M_2}}{M_3^d(M_2^d - M_1^d)} \\ M_3 >& M_1, M_2 \end{aligned} \tag{6}$$

Choice of K

In Section (3.4) of the main text, we introduce a proxy for accuracy, K, to allow comparisons between the various computations. Here, we justify our approach in more detail by deriving a strictly correct mathematical approach which then leads to a reasonable proxy.

Each potential QUAD engine has an error upper bound $O(\delta y^d)$ for an arbitrary order d. The computational complexity of the QUAD method for a discretely path dependent option with M observations to maturity, is of order $O(M \times N^b)$ where N is the number of QUAD steps such that:

$$N = \frac{y_{\text{max}} - y_{\text{min}}}{\delta y}$$

b is defined by the dimension of the underlying of the option D. The number of plain QUAD methods used when $D = 1$ is:

$$(N+1) + M \times (2 \sum_{\frac{N}{2}+1}^{N+1} i - (N+1))$$

This expression is a polynomial of N up to second order. It is then useful to define a vector as:

$$\boldsymbol{g} = \begin{pmatrix} \frac{N}{2}+1 & \frac{N}{2}+2 & \dots & N & N+1 & N & \dots & \frac{N}{2}+2 & \frac{N}{2}+1 \end{pmatrix}$$

We can next formulate the computational complexity of D-dimensional QUAD method as:

$$(N+1)^D + M \times \left(\boldsymbol{G}^{i_1,i_2,\cdots,i_{D-1}} \boldsymbol{G}_{i_1,i_2,\cdots,i_{D-1}}\right)$$

where

$$\boldsymbol{G}^{i_1,i_2,\cdots,i_D} = \boldsymbol{g} \otimes \boldsymbol{G}^{i_1,i_2,\cdots,i_{D-1}}$$

and this expression is polynomial of N up to the order of $2 \times D$. For example, the QUAD method using a local volatility model on a single asset would have a computation complexity of $O(M \times N^2)$, and for stochastic volatility models would have $O(MN^4)$.

If we construct the grid such that the overall error term is independent of M, we are to force the following expression to be independent of M:

$$\begin{aligned} f(N) &= \left((N+1)^D + M \times \left(\boldsymbol{G}^{i_1,i_2,\cdots,i_{D-1}} \boldsymbol{G}_{i_1,i_2,\cdots,i_{D-1}}\right)\right) \times O(\delta y^d) \\ &\equiv \left((N+1)^D + M \times \left(\boldsymbol{G}^{i_1,i_2,\cdots,i_{D-1}} \boldsymbol{G}_{i_1,i_2,\cdots,i_{D-1}}\right)\right) \times O(N^{-d}) \end{aligned}$$

we therefore choose our time step K as a function, of M, $S(M)$ such that

$$M = S^{-1}(K) = \frac{\text{constant}}{\left(\boldsymbol{G}^{i_1,i_2,\cdots,i_{D-1}}\boldsymbol{G}_{i_1,i_2,\cdots,i_{D-1}}\right) O(K^{-d}))}$$

such that $\frac{1}{M}$ is a polynomial of K with maximum order of $2D - d$ and minimum order of $-d$.

Now, this method of choosing our time step, although providing a precisely built, uniformly dense grid across the whole range of observation M, adds unnecessary computation cost to the algorithm and so we use a proxy which mimics the dynamics of $S(M)$. Note that, for Simpson's rule with a local volatility model:

$$S^{-1}(K) = \frac{\text{constant}}{\frac{3}{4}O(\delta y^2) + \frac{3}{2}O(\delta y^3) + O(\delta y^4)}$$

Conveniently, it then turns out in practice that there is no need to calculate explicitly the function $S(M)$. Both $\frac{1}{\log(M)}$ and $\frac{1}{\sqrt{M}}$ can be a good proxy for $S(M)$.

QUAD in Multiple Dimensions

Consider a three dimensional integral in $\mathbb{R}^3$ cuboid space y_1, y_2, y_3 over domain $y_i^{max} \leq y_i \leq y_i^{min}$:

$$\begin{aligned} I &= \int\int\int f(y_1, y_2, y_3)dy_1dy_2dy_3 \\ &= \int_{y_3^{max}}^{y_3^{min}} \left(\int_{y_2^{max}}^{y_2^{min}} \left(\int_{y_1^{max}}^{y_1^{min}} f(y_1, y_2, y_3)dy_1 \right) dy_2 \right) dy_3 \end{aligned} \tag{7}$$

Define following the functions:

$$G(y_2, y_3) = \int_{y_1^{max}}^{y_1^{min}} f(y_1, y_2, y_3)dy_1 \tag{8}$$

and

$$H(y_3) = \int_{y_2^{max}}^{y_2^{min}} G(y_2, y_3)dy_2 \tag{9}$$

such that,

$$I = \int_{y_3^{max}}^{y_3^{min}} H(y_3) dy_3 \tag{10}$$

The integration problem (7) is equivalent to the three integrations 8 to 10. QUAD can be used recursively for each integration to find G, H and I. The D-dimensional QUAD over D-orthotope domain is formulated as follows:
Given quad steps in each dimension as $\delta y_k = \frac{y_k^{max} - y_k^{min}}{N}$, the integral can be approximated by:

$$I \approx \left(\prod_{k=1}^{D} \delta y_k \right) \boldsymbol{Q}_{i_1 \ldots i_D} \boldsymbol{F}_{i_1..i_D}(y_1, \cdots, y_D)$$

where $\boldsymbol{F}$ is a $(N{+}1)^D$ tensor of function value at quad points defined as:

$$\begin{aligned} \boldsymbol{F}_{i_1..i_D}(y_1, \cdots, y_D) = & f(y_1 + (i_1 - 1)\delta y_1, \cdots, y_D + (i_D - 1)\delta y_D) \\ & \forall i_1, i_2, \cdots, i_D \in [1, N+1] \end{aligned}$$

and $\boldsymbol{Q}$ is a $(N{+}1)$-dimension weighting tensor of order D such that, if the weights is defined in one dimension as $(N{+}1)$-dimension vector $\boldsymbol{q}$ with its ith entry denoted by q_i, the $\{i_1..i_D\}$-th entry of $\boldsymbol{Q}$ is defined as:

$$\boldsymbol{Q}_{i_1..i_D} = \prod_{k=1}^{D} q_{i_k}$$
$$\forall i_1, i_2, \cdots, i_D \in [1, N+1]$$

For the purpose of illustration, the weighting matrix $\boldsymbol{Q}_S$ of Simpson's rule in 2-d can be calculated from its 1-d weighting vector $\boldsymbol{q}_S$ defined in Equation (3) as:

$$\boldsymbol{Q}_S = \boldsymbol{q}_S \otimes \boldsymbol{q}_S = \frac{1}{9}\begin{pmatrix} 1 & 4 & 2 & \cdots & 2 & 4 & 1 \\ 4 & 16 & 8 & \cdots & 8 & 16 & 4 \\ 2 & 8 & 4 & \cdots & 4 & 8 & 2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 2 & 8 & 4 & \cdots & 4 & 8 & 2 \\ 4 & 16 & 8 & \cdots & 8 & 16 & 4 \\ 1 & 4 & 2 & \cdots & 2 & 4 & 1 \end{pmatrix}$$

and weighting matrix $\boldsymbol{Q}_T$ of Trapezoidal rule in two-dimension can be calculated from its one-dimension weighting vector $\boldsymbol{q}_T$ defined in Equation (4) as:

$$\boldsymbol{Q}_T = \boldsymbol{q}_T \otimes \boldsymbol{q}_T = \frac{1}{4}\begin{pmatrix} 1 & 2 & \cdots & 2 & 1 \\ 2 & 4 & \cdots & 4 & 2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 2 & 4 & \cdots & 4 & 2 \\ 1 & 2 & \cdots & 2 & 1 \end{pmatrix}$$

In fact, the numbers of QUAD steps N_k for each dimension k do not need to be consistent as long as they satisfy the conditions defined by the QUAD scheme, thus: N_k is even for Simpson's quadrature. In this work, N_k is set universally across dimensions in order to simplify the tracking of convergence rate. Again, in order to remove any non-linearity error, the integration domain is segmented along discontinuities of function f and the remaining error term of D-dimensional QUAD takes the form:

$$\sum_{k=1}^{D} o(\delta y_k^d)$$

where d is the rate of convergence of the chosen QUAD scheme in one dimension. It is readily observed that, for an arbitrary dimension D, the rate of convergence of any QUAD scheme inherits its one-dimensional value. Therefore (and importantly), Richardson's extrapolation can also be adapted in higher dimensions to improve the convergence of QUAD method.