Integrating Feynman-Kac Equations Using Hermite Quintic Finite Elements

A White Paper by Rogue Wave Software.

November 2008
Integrating Feynman-Kac Equations Using Hermite Quintic Finite Elements

by Rogue Wave Software and Richard Hanson\(^1\)

© 2012 by Rogue Wave Software. All Rights Reserved

Printed in the United States of America

Publishing History:

April 2012 - Update

November 2008 – First Publication

Trademark Information

The Rogue Wave Software name and logo, SourcePro, Stingray, HostAccess, IMSL and PV-WAVE are registered trademarks of Rogue Wave Software, Inc. or its subsidiaries in the US and other countries. JMSL, JWAVE, TS-WAVE, PyIMSL and Knowledge in Motion are trademarks of Rogue Wave Software, Inc. or its subsidiaries. All other company, product or brand names are the property of their respective owners.

IMPORTANT NOTICE: The information contained in this document is subject to change without notice. Rogue Wave Software, Inc. makes no warranty of any kind with regards to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Rogue Wave Software, Inc. shall not be liable for errors contained herein or for incidental, consequential, or other indirect damages in connection with the furnishing, performance, or use of this material.

\(^1\) The C version of the VNI software [12] was converted from Fortran 95 by Martin Preiss, Rogue Wave Software, Germany. Preiss re-edited this white paper based on an early draft that used a Fortran 95 implementation for development of the algorithm.
# TABLE OF CONTENTS

Abstract.................................................................................................................................................. 4
Introduction ........................................................................................................................................... 4
Section 1 – Finite Elements and Galerkin Methods............................................................................. 5
  The Finite Elements ......................................................................................................................... 5
  The Galerkin Process and Matrix Assembly..................................................................................... 8
  The Boundary Conditions ................................................................................................................. 9
Section 2 – Some Black-Scholes Examples......................................................................................... 11
Section 3 – A Non-Linear Forcing Term for American Options......................................................... 16
About the Author ................................................................................................................................. 20
References ........................................................................................................................................... 20
Abstract

Methods are given for numerically solving a generalized version of the Feynman-Kac partial differential equation. The solution is expressed as a linear combination of piece-wise Hermite quintic polynomials. This twice-differentiable representation has the attributes of being a high-order method that allows easy evaluation of the solution and certain of its partial derivatives. The time-dependent solution coefficients are computed by finite element Galerkin procedures. Boundary values and initial conditions are required.

Many significant problems in financial modeling can be expressed as particular choices of coefficients, initial conditions, and boundary values. That fact allows this Feynman-Kac solver to be used in many financial modeling applications. These include the Black-Scholes models with European type or American type exercise opportunities on Calls or Puts.

Introduction

This report describes mathematical and numerical software methods for solving the generalized Feynman-Kac partial differential equation

\[ f_t + \mu(x,t)f_x + \frac{\sigma^2(x,t)}{2}f_{xx} - \kappa(x,t)f = \phi(f,x,t), \quad f(x,T) = p(x), \{f_t = \frac{\partial f}{\partial t}, \text{etc.}\} \]

Equation 1. Feynman-Kac (F-K) Equation with Additional Terms \( \kappa f \) and \( \phi \)

The equation is generally solved by integrating backwards, starting at \( t = T \). Thus the function \( p(x) \) is a terminal condition. Numerical values are computed on a compact interval \([x_{\min}, x_{\max}]\). Boundary conditions are required:

\[ a(x,t)f + b(x,t)f_x + c(x,t)f_{xx} = d(x,t), \quad x = x_{\min} \& x_{\max} \]

Equation 2. F-K Vector Boundary Conditions

The boundary conditions can be vectors of dimension 1, 2 or 3. Details on their form and smoothness requirements are given below in the section on boundary conditions.

This F-K equation includes a number of interesting models as special cases. One of our primary interests is the Black-Scholes equation for the value \( f(S,t) \) of an option [1]:

---

2 This work benefited by discussions with Fred Krogh, Gerald Hanweck, John Brophy and Martin Preiss.
3 http://en.wikipedia.org/wiki/Feynman-Kac_formula
\[
\frac{\partial f}{\partial t} + (r - D) S \frac{\partial f}{\partial S} + \frac{\sigma^2}{2} S^2 \frac{\partial^2 f}{\partial S^2} - rf = 0,
\]
\[
S_{\text{min}} \leq S \leq S_{\text{max}}, \quad t_{\text{invest}} \leq t \leq t_{\text{expiry}}
\]

Equation 3. Black-Scholes Model, a Special Case of F-K

Here \( r \) is the instantaneous risk-free interest rate, \( D \) is the continuous dividend of the instrument, and \( \sigma \) is the continuous positive volatility\(^4\) of that instrument. There are also boundary conditions, as in Equation 2, which will be defined and used as required.

A part of this report gives numerical results for particular choices occurring in the Black-Scholes model. The CNL functions \texttt{imsl\_f\_feyman\_kac} and \texttt{imsl\_f\_feynman\_kac\_evaluate} (documented in [12]) are used to compute these models.

**Section 1 – Finite Elements and Galerkin Methods**

The mathematical methods we use for solving F-K equations are well-known and tested (See [2], p.89). We give a detailed description of the finite elements and the construction of the matrices that arise when solving the Differential-Algebraic equations (DAE) for the solution.

**The Finite Elements**

This development begins with a grid of points or a partition of the interval \([x_{\text{min}}, x_{\text{max}}]\). Write this grid as \((x_{\text{min}} = x_1 < x_2 < \ldots < x_m (= x_{\text{max}})\). We see that it is a decomposition of the interval into \(N_{\text{int}} = m - 1\) abutting subintervals. For each fixed \(t\), on each subinterval, the unknown solution is given by a Hermite quintic polynomial. An issue that we do not discuss is placement of these grid points. The development given here is a more general method than that found in [11].

The polynomial is represented by six basis polynomials. Thus in the interval

\[
x \in [x_i, x_{i+1}],
\]

the function and its first three derivatives are given by

\[
f(x,t) = f_i b_0(z) + f_{i+1} b_0(1-z) + h_i f_i' b_1(z) - h_i f_{i+1}' b_1(1-z) + h_i^2 f_i'' b_2(z) + h_i^2 f_{i+1}'' b_2(1-z)
\]
\[
f_x(x,t) = h_i^{-1} (f_i b_0'(z) - f_{i+1} b_0'(1-z) + h_i f_i' b_1'(z) + h_i f_{i+1}' b_1'(1-z) + h_i^2 f_i'' b_2'(z) - h_i^2 f_{i+1}'' b_2'(1-z))
\]
\[
f_{xx}(x,t) = h_i^{-2} (f_i b_0''(z) + f_{i+1} b_0''(1-z) + h_i f_i'' b_1''(z) - h_i f_{i+1}'' b_1''(1-z) + h_i^2 f_i''' b_2''(z) + h_i^2 f_{i+1}''' b_2''(1-z))
\]
\[
f_{xxx}(x,t) = h_i^{-3} (f_i b_0'''(z) - f_{i+1} b_0'''(1-z) + h_i f_i''' b_1'''(z) + h_i f_{i+1}''' b_1'''(1-z) + h_i^2 f_i'''' b_2'''(z) - h_i^2 f_{i+1}'''' b_2'''(1-z))
\]

Equation 4. Partial Derivatives of the Piecewise Solution

\(^4\) In the F-K equation identify the general term \(\sigma(x,t)\) with the Black-Scholes volatility factor multiplied by the price—i.e. \(\sigma S\).
For our purpose the values $f_i, f_i', f_i'', f_{i+1}, f_{i+1}', f_{i+1}''$ are time-dependent coefficients associated with that interval. The local error term is $O(h^6)$. The basis functions and their partial derivatives are given by the formulas

\[
\begin{align*}
    b_0(z) &= -6z^3 + 15z^2 - 10z + 1 = (1-z)^3(6z^2 + 3z + 1), \\
    b_1(z) &= -3z^3 + 8z^2 - 6z + z = (1-z)^3 z(3z + 1), \\
    b_2(z) &= \frac{1}{2}(-z^4 + 3z^3 - 3z^2 + z^2) = \frac{1}{2}(1-z)^3 z^2, \\
    b_3'(z) &= -30(1-z)^3 z^2; \quad (b_3'' = ) b_0^{(2)} (z) = -60z(2z^2 - 3z + 1); \quad b_0^{(3)} (z) = -60(6z^2 - 6z + 1) \\
    b_1'(z) &= -(1-z)^3(15z^2 - 2z - 1); \quad b_1^{(2)} (z) = 12z(1-z)(5z - 3); \quad b_1^{(3)} (z) = -12(15z^2 - 16z + 3) \\
    b_2'(z) &= -\frac{1}{2}(1-z)^3 z(5z - 2); \quad b_2^{(2)} = (1-z)(10z^2 - 8z + 1); \quad b_2^{(3)} (z) = -3(10z^2 - 12z + 3)
\end{align*}
\]

**Equation 5. Hermite Quintic Basis Functions and Three Derivatives**

![Graphs of Hermite Basis Functions and Two Derivatives](image.png)
During progression from one interval to the next, there is overlap with three coefficients from the adjacent interval. This observation and the formulas for the derivatives show that for each fixed \( t \), the function \( f(x,t) \) is continuous at the interior grid points and has two continuous derivatives with respect to \( x \). Formulas for the piece-wise quadratic third derivatives are included here, even though that function is no longer generally continuous at the interior points. This third derivative function may be of interest to users. The total number of parameters on this grid is \( 3(N_{\text{int}} + 1) \). The rationale for choosing this set of elements is based on these observations:

- Since the local error term is of order \( O(h^6) \), this translates into fewer unknowns than a finite difference method. Typically there is high accuracy when using even a modest-sized grid. This should result in efficient computational algorithms since the size of the system of differential-algebraic equations and thus the associated linear algebra will be of a smaller dimension. There is also special banded structure for the matrices of the linear algebra. This allows for the use of tailored banded matrix linear system solvers and matrix-vector products.

- The functions and intrinsic continuous partial derivatives \( \{f, f_x, f_{xx}, f_{t}, f_{tx}, f_{txx} \} \) are available by direct evaluation of the solution and its derivative functions. These additional partial derivative values are often required by users and they can be obtained at essentially no extra cost in computing time. In the case of the Black-Scholes model these functions include many of The Greeks\(^5\). The CNL function \texttt{imsl\_f\_feynman\_kac\_evaluate} can compute these derivatives, which we call the Intrinsic Greeks. Evaluation can occur for any set of values of \( x \) at a given \( t \). There is no resulting sub-problem such as interpolation. The evaluator does that for the user.

- For some applications it is necessary to interpolate in the orthogonal direction for the solution as a function of \( t \) with \( x \) fixed. A way to do this is assemble values \( \{f, f_t\} \) at a grid of \( t \) values at a fixed \( x \). Then perform Hermite cubic interpolation at any set of values within this grid of \( t \) values. This \( C^1 \) cubic has a local error of \( O(\Delta t^4) \) and is given by the formulas

\[
\begin{align*}
\hat{b}_0(z) &= (1 + 2z)(1 - z)^2, \\
\hat{b}_1(z) &= z(1 - z)^2 \\
\hat{b}_0'(z) &= -6(1 - z)z, \\
\hat{b}_1'(z) &= 3z^2 - 4z + 1
\end{align*}
\]

To interpolate at any \( t \in [t_i, t_{i+1}] \) compute

\[
\begin{align*}
f(x,t) &= f(x,t_i) + f_t(x,t_i)\Delta t + f_{xx}(x,t_i)\hat{b}_0(1 - z) + \Delta t f'_{t}(x,t_i)\hat{b}_1(1 - z) - f_{txx}(x,t_i)\hat{b}_0'(1 - z), \\
z &= (t - t_i)/\Delta t, \\
f'_t &= \frac{\partial f}{\partial t}(x,t_i), \text{ etc.}
\end{align*}
\]

\(^5\) http://en.wikipedia.org/wiki/The_Greeks
The complexity of these representations and the quintic basis functions are a concern. Verifying the formulas was aided by the use of algebraic manipulation software. The Hermite quintic formulas have been coded in \texttt{imsl\_f\_feynman\_kac\_evaluate}. Testing with the Black-Scholes equation has yielded results consistent with analytic solutions.

\textbf{The Galerkin Process and Matrix Assembly}

Using a development similar to that of [2], p. 116, write the F-K equation in divergence form

\[
\Lambda(f) \equiv \left( \frac{\partial}{\partial t} + (\mu(x,t) - \sigma \sigma') \frac{\partial}{\partial x} + \left( \frac{\sigma^2(x,t)}{2} \right) \frac{\partial}{\partial x} \left( \frac{\partial}{\partial x} \right) - \kappa(x,t) \right) f - \phi = 0
\]

\[
R(x,t) \equiv f_t + (\mu(x,t) - \sigma \sigma') f_x + \left( \frac{\sigma^2(x,t)}{2} \right) f_{xx} - \kappa(x,t) f - \phi = 0, \{ \sigma' = \frac{\partial \sigma}{\partial x} \}
\]

By adding the piece-wise definitions we may arrange the unknown solution function as the series \( f(x,t) = \sum_{j=1}^{3(N_{\text{int}}+1)} y_j(t) \beta_j(x) \). The coefficient functions are defined by re-labeling:

\[ y_j(t) = f(x_j,t), y_{j+1}(t) = f_x(x_j,t), y_{j+2}(t) = f_{xx}(x_j,t), \text{ etc.} \]

Then impose the Galerkin orthogonality conditions \( \int_{x_{\text{min}}}^{x_{\text{max}}} R(x,t) \beta_j(x) \, dx = 0, j = 1, \ldots, 3(N_{\text{int}}+1) \).

Expand and integrate by parts the third term in each expression to yield

\[
\int_{x_{\text{min}}}^{x_{\text{max}}} \left( \frac{\sigma^2(x,t)}{2} f_x \right) \beta_j(x) \, dx = \left. \frac{\sigma^2(x,t)}{2} f_x \beta_j(x) \right|_{x_{\text{min}}}^{x_{\text{max}}} - \int_{x_{\text{min}}}^{x_{\text{max}}} \left( \frac{\sigma^2(x,t)}{2} f_x \beta_j'(x) \right) \, dx.
\]

Collect all terms after performing this integration to get five matrices contributing to the DAE

\[
M \tilde{y}' + N \tilde{y} - R \tilde{y} - K \tilde{y} + \tilde{p} = M \tilde{y}' + \left( N - R - K + Q \right) \tilde{y} - \tilde{p},
\]

where

\[
\tilde{y} := \tilde{y}(t) = \left[ y_1(t), \ldots, y_{3(N_{\text{int}}+1)}(t) \right]^T, \quad \tilde{y}' = \frac{d\tilde{y}}{dt},
\]

\[
\tilde{\beta} := \tilde{\beta}(x) = \left[ \beta_1(x), \ldots, \beta_{3(N_{\text{int}}+1)}(x) \right]^T, \quad \tilde{\beta}' = \frac{d\tilde{\beta}}{dx},
\]

\[
M = \int \tilde{\beta} \tilde{\beta}'^T, \quad N = \int (\mu - \sigma \sigma') \tilde{\beta} \tilde{\beta}'^T, \quad R = \int \frac{\sigma^2}{2} \tilde{\beta} \tilde{\beta}'^T,
\]

\[
K = \int \kappa \tilde{\beta} \tilde{\beta}'^T, \quad Q = \int \left( \frac{\sigma^2}{2} \beta(x_{\text{max}}) \tilde{\beta}'(x_{\text{max}})^T - \frac{\sigma^2}{2} \beta(x_{\text{min}}) \tilde{\beta}'(x_{\text{min}})^T \right),
\]

\[
\tilde{p} = \int \phi \tilde{\beta}.
\]

The integration is approximated using Gauss-Legendre (G-L) quadrature normalized to the interval \([0,1]\), see [3], p. 390. The leading matrix term \( M \) will be the integral of products of resulting degree 10. It will not be dependent on \( t \) nor will it depend
on any coefficient in the F-K equation. This means it will only have to be computed once. To
exactly integrate this symmetric matrix an order 6 G-L formula is used on each subinterval. This
is because the error term for G-L quadrature has a factor involving the 12-th derivative. This
will be identically zero on each subinterval. However using just an order 5 formula gives an
error term involving the 10-th derivative which is a non-zero constant. [See [3], (8.5.6), p. 391.]
The user can reset the order of the G-L formula to a higher or lower value to deal with
particular behavior in the remaining coefficients. Note that the exact form of the polynomial
representation for \( \beta_j(x) \) is not an issue. Only the 6 evaluations are required to integrate the
leading term \( M \) exactly, apart from rounding errors.

With the particular basis of quintic splines, the only non-zero value of \( \beta_j(x_{\text{max}}) \) is at the right
end, or \( \beta_{3N_{\text{ur}}+3}(x_{\text{max}}) = 1 \), and for \( \beta'_{3N_{\text{ur}}+2}(x_{\text{max}}) = h_{N_{\text{ur}}} \). Similarly, the only non-zero value of
\( \beta_j(x_{\text{min}}) \) is at the left end, or \( \beta_1(x_{\text{min}}) = 1 \), and for \( \beta'_1(x_{\text{min}}) = h_1 \). Thus we have the last
component of \( Q\tilde{y} \) with the value \( h_{N_{ur}} \frac{\sigma^2(x_{\text{max}}, t)}{2} y_{3N_{ur}+2}(t) \) and the first component with the
value \(-h_1 \frac{\sigma^2(x_{\text{min}}, t)}{2} y_2(t)\). Notice that the boundary conditions have not yet been used in
formulating the DAE. We then consider the intermediate problem DAE

\[
M\tilde{y}' + N\tilde{y} - R\tilde{y} - K\tilde{y} + Q\tilde{y} - \tilde{p} \equiv M\tilde{y}' + P\tilde{y} - \tilde{p} = 0, \quad \tilde{y}' = \frac{d\tilde{y}}{dt}, \quad \tilde{p} = \int \phi \tilde{p}
\]

Equation 6. DAE of Coefficients -- No Boundary Conditions Provided

For computational efficiency it is important to note when the matrix \( P = N - R - K + Q \) has
dependence on \( t \). If there is no dependence this matrix can be computed once, at the outset of
integration. It is thereafter not necessary to re-compute it at every time step. If any coefficient
in the set \( \{\mu, \kappa, \sigma, \sigma'\} \) has time dependence then some terms of \( P \) must be re-computed at
each time step. The Black-Scholes Equation 3, with constant values of \( D, r, \sigma \) is a case where
there is no time dependence in the accumulated element matrices. Also note that in this
problem an order 6 G-L formula exactly integrates each term of the matrix \( P \) for the Black-
Scholes model with constant \( \{r, \sigma\} \). This follows from the fact that the polynomial degree in
each integrand is 10.

The Boundary Conditions

Following the derivation of the DAE system that follows from applying the Galerkin method, the
boundary value conditions of Equation 2 are applied. These end conditions then take
precedence over those in Equation 5. Thus we replace the first and last equations by the
appropriate boundary value equations. After this replacement the DAE system
can be written as \( U\tilde{y}' + V\tilde{y} - w = 0 \). The first and last rows of \( U \) are zero. The
remaining rows are those of \( M \). The first and last rows of \( V \) are replaced by the
values of the boundary value coefficients from Equation 2. The remaining rows are
those of $P$. The vector $w$ is $\bar{p}$ except in the first and last components that correspond to the values $d(x,t)$ at the boundaries. A technical point here is that the boundary conditions imply that the first and last rows and $V$ and $w$ may depend on $t$ even though the remaining rows or components do not.

During the integration of the DAE there is a requirement to solve a linear algebraic system with a coefficient matrix $A = \gamma U + V$. The value of $\gamma$ depends on details of the integration, including step-size and order of the backward differentiation formulas, [4]. The boundary values must be specified so that for some range of values for $\gamma$ the banded matrix $A$ is non-singular. This property defines the DAE to be an index 1 problem. Existing DAE software such as DDASLX, [5], translated into CNL, is used for its solution. According to the LAPACK definition and data structure\(^6\) for a banded matrix\(^7\), the upper and lower bandwidth of $A$ is 5 and the matrix bandwidth is 11. The fact that $A$ is banded implies that efficient algorithms for solving banded matrix linear system problems can be applied. There is also a requirement to compute the residual function of the DAE $F(\bar{y}', \bar{y}, t) = U\bar{y}' + V\bar{y} - w = 0$. Since $A$, $U$ and $V$ have the same banded properties, Level-2 BLAS\(^8\), namely the CLAPACK version of SGBMV, are used for computing these matrix-vector products in the right-hand side of $F(\bar{y}', \bar{y}, t)$.

For starting the integration it is necessary to have initial values of the coefficient set $\bar{y} = \bar{y}_0$. For this we need to represent the initial data in terms of the Hermite piecewise quintic coefficients. To that end we compute the local contribution $u := \int p(x) \beta(x) \, dx$. With this available we compute the least-squares solution for the initial coefficients by minimizing

$$\int (p(x) - \beta(x)^T \bar{y}_0)^2 \, dx.$$  

The particular choice that minimizes this least-squares condition is given by solving $M \bar{y}_0 = u$ after accumulation over local contributions over all subintervals. This solution is obtained by using the left and right boundary conditions as equality constraints and the remaining rows as least squares equations. This solution for $\bar{y}_0$ is used as the starting values for the coefficients of the integration.

For the Black-Scholes equation the vector $u$ is typically not integrated exactly. It costs very little to increase the degree of the Gauss-Legendre formula, so the degree used in building the matrices is a parameter of the software. This process does not provide the time derivatives $\bar{y}'_0$. The values for $\bar{y}'_0$ are computed using the regularization method given in Appendix II of [10]. There are cases where the initial values $\bar{y}_0$ should be first provided from basic principles without relying on the least-squares estimation process outlined above. This information can be communicated using function imsl_f_feynman_kac together with optional argument IMSL_FCN_INIT (see [12]).

\[^6\] http://www.netlib.org/lapack/lug/node124.html
\[^7\] http://en.wikipedia.org/wiki/Band_matrix
\[^8\] http://www.netlib.org/lapack/lug/node145.html
Section 2 – Some Black-Scholes Examples

As mentioned, the reasons for using the Hermite quintic basis elements are their small truncation error and the smoothness of the intrinsic partial derivatives at an evaluation time. There is no significant additional expense to obtain the partials. Their shapes are illustrated with a pair of Vanilla Options using the Black-Scholes model of Equation 3. The intrinsic third partial derivative or Greek (\(\text{speed}\)) is not a smooth function. One could re-do the finite element model using Hermite-seven degree splines. These will have three continuous intrinsic partial derivatives, including \textit{speed}. We believe it is not likely that accurate values of \textit{speed} are required by financial analysts.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{example.png}
\caption{Numerical results with intrinsic greeks for European Call}
\end{figure}
Values of Black-Scholes PDE Coefficients

\[ \sigma = 0.2, \ r = 0.1, \ T = 0.5, \ E = 40, \ D = 0 \]

Requested Integration Accuracy

\[ \text{atol} = \text{rtol} = 10^{-6} \]

Number of Intervals

100

Number of Evaluations\(^9\) and Factorizations

281, 24

Table 1. Black-Scholes Equation Solution, European Call. See [7], p. 116

\[ \text{European Call, } -f_v \text{ or } \Theta = \text{Theta} \]

\[ \text{European Call, } f_{\text{nu}} \text{ or Charm} \]

\[ \text{European Call, } f_{\text{nu}} \text{ or Color} \]

Figure 3. More Numerical Results of Intrinsic Greeks with European Call

\(^{9}\) This is the number of evaluations of the residual DAE function \( F(y', \tilde{y}, t) = U\tilde{y}' + V\tilde{y} - w \) and the number of solve steps after a banded matrix factorization. The boundary conditions are given by a \(3 \times 3\) identity matrix at both ends and respective values \( d = [0, 0, 0]^T \) at the left and \( d = [x_{\text{max}} - Ee^{-(T-t)}, 1, 0]^T \) at the right end.
Figure 4. Numerical Results with Intrinsic Greeks for European Put

Table 2. Black-Scholes Equation Solution, European Put. See [1], p. 176

| Values of Black-Scholes PDE Coefficients | $\sigma = 0.4, r = 0.1, T = 0.5, E = 10, D = 0$ |
| Requested Integration Accuracy | $\text{atol} = \text{rtol} = 10^{-6}$ |
| Number of Intervals | 100 |
| Number of Evaluations$^\text{10}$ and Factorizations | 316, 27 |

$^\text{10}$This is the number of evaluations of the residual DAE function $F(\ddot{y}, \dot{y}, t) = U\ddot{y} + V\dot{y} - w$ and the number of solve steps after a banded matrix factorization. The boundary conditions are given by respective values $d = [-1,0]^T$ at the left end and $d = [0,0,0]^T$ at the right end.
The evaluation of the value or intrinsic Greeks does not give all the function information commonly used by financial analysts. For example there are variational equations in the Black-Scholes model that may be required. Two common ones are \( \frac{\partial f}{\partial r}, \frac{\partial f}{\partial \sigma} \). These are known as Rho and Vega. They can be integrated as additional variational equations, e.g. [7], pp. 126-132. For constant interest rates and volatility factors an elementary approach is to use divided differences but with care to choose the optimal value of the difference.

The following example shows some numerical results for a plain Vanilla European Put (Source code). The data for this put in the Black-Scholes model are listed in Table 3.
Parameter | Value
---|---
Volatility $\sigma$ | 0.4
Interest rate $r$ | 0.1
Time to Maturity | 3 months, 6 months
Strike Price $E$ | 10.0
Dividend $D$ | 0

Table 3. Parameters for Plain Vanilla European Put

For plain Vanilla European Puts, the Black-Scholes Equation has an analytical solution. Setting

$$a := \frac{\log(S / E) + (r + \sigma^2 / 2) \Delta T}{\sigma \sqrt{\Delta T}},$$

where $\Delta T := T - t$ denotes the time to expiry, the value of the put at stock price $S$ and time point $t$ is given by

$$f(S,t) = -S \Phi(-a) + E e^{-r \Delta T} \Phi(-a + \sigma \sqrt{\Delta T}).$$

Here, $\Phi(\cdot)$ is the standard normal distribution function.

The numerical values are compared with these analytical solutions.

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Formula</th>
<th>Delta</th>
<th>Gamma</th>
<th>Charm</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>9.75310</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.9753</td>
<td>-0.0000</td>
</tr>
<tr>
<td>2.0</td>
<td>7.75310</td>
<td>-1.0000</td>
<td>-0.0000</td>
<td>-0.9753</td>
<td>-0.0000</td>
</tr>
<tr>
<td>4.0</td>
<td>5.75310</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.9753</td>
<td>-0.0003</td>
</tr>
<tr>
<td>6.0</td>
<td>3.75689</td>
<td>0.0221</td>
<td>0.1677</td>
<td>-0.7194</td>
<td>-0.0651</td>
</tr>
<tr>
<td>8.0</td>
<td>1.90243</td>
<td>0.12792</td>
<td>0.34769</td>
<td>-0.04484</td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td>0.66939</td>
<td>0.04484</td>
<td>0.00539</td>
<td>0.00539</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Put value</th>
<th>Delta</th>
<th>Gamma</th>
<th>Charm</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>9.75310</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.9753</td>
<td>-0.0000</td>
</tr>
<tr>
<td>2.0</td>
<td>7.75310</td>
<td>-1.0000</td>
<td>-0.0000</td>
<td>-0.9753</td>
<td>-0.0000</td>
</tr>
<tr>
<td>4.0</td>
<td>5.75310</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.9753</td>
<td>-0.0003</td>
</tr>
<tr>
<td>6.0</td>
<td>3.75689</td>
<td>0.0221</td>
<td>0.1677</td>
<td>-0.7194</td>
<td>-0.0651</td>
</tr>
<tr>
<td>8.0</td>
<td>1.90243</td>
<td>0.12792</td>
<td>0.34769</td>
<td>-0.04484</td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td>0.66939</td>
<td>0.04484</td>
<td>0.00539</td>
<td>0.00539</td>
<td></td>
</tr>
</tbody>
</table>

European Option Premium for Vanilla Put, 3 and 6 Months Prior to Expiry
Number of equally spaced spline knots: 101
Number of unknowns: 303
Strike= 10.00, sigma= 0.40, Interest Rate= 0.10
Environment: SPARC(64-bit), Solaris 10
Compiler: C-compiler from Sun Studio 12
Integration time (in CPU secs): 0.850879

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Formula</th>
<th>PDE</th>
<th>Formula</th>
<th>PDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>9.75310</td>
<td>9.75310</td>
<td>9.51229</td>
<td>9.51229</td>
</tr>
<tr>
<td>2.0</td>
<td>7.75310</td>
<td>7.75310</td>
<td>7.51229</td>
<td>7.51229</td>
</tr>
<tr>
<td>4.0</td>
<td>5.75310</td>
<td>5.75310</td>
<td>5.51229</td>
<td>5.51229</td>
</tr>
<tr>
<td>6.0</td>
<td>3.75689</td>
<td>3.75689</td>
<td>3.55829</td>
<td>3.55829</td>
</tr>
<tr>
<td>8.0</td>
<td>1.90243</td>
<td>1.90243</td>
<td>1.91810</td>
<td>1.91810</td>
</tr>
<tr>
<td>10.0</td>
<td>0.66939</td>
<td>0.66939</td>
<td>0.87033</td>
<td>0.87033</td>
</tr>
<tr>
<td>12.0</td>
<td>0.16751</td>
<td>0.16751</td>
<td>0.34769</td>
<td>0.34769</td>
</tr>
<tr>
<td>14.0</td>
<td>0.03262</td>
<td>0.03262</td>
<td>0.12792</td>
<td>0.12792</td>
</tr>
<tr>
<td>16.0</td>
<td>0.00539</td>
<td>0.00539</td>
<td>0.04484</td>
<td>0.04484</td>
</tr>
</tbody>
</table>

Greeks at 3 months

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Put value</th>
<th>Delta</th>
<th>Gamma</th>
<th>Charm</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>9.75310</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.9753</td>
<td>-0.0000</td>
</tr>
<tr>
<td>2.0</td>
<td>7.75310</td>
<td>-1.0000</td>
<td>-0.0000</td>
<td>-0.9753</td>
<td>-0.0000</td>
</tr>
<tr>
<td>4.0</td>
<td>5.75310</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.9753</td>
<td>-0.0003</td>
</tr>
<tr>
<td>6.0</td>
<td>3.75689</td>
<td>0.0221</td>
<td>0.1677</td>
<td>-0.7194</td>
<td>-0.0651</td>
</tr>
<tr>
<td>8.0</td>
<td>1.90243</td>
<td>0.12792</td>
<td>0.34769</td>
<td>-0.04484</td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td>0.66939</td>
<td>0.04484</td>
<td>0.00539</td>
<td>0.00539</td>
<td></td>
</tr>
</tbody>
</table>
Section 3 – A Non-Linear Forcing Term for American Options

An approximate solution of the American Option (exercise at any time) is obtained by a forcing or source term, [7], (2.40)-(2.41):

\[ f_i + \mu(x,t)f_x + \frac{\sigma^2(x,t)}{2}f_{xx} - \kappa(x,t)f = \phi(f,x,t), \quad f(x,T) = p(x), \quad f = \frac{\partial f}{\partial t}, \text{etc.} \]

\[ \phi = 0, \quad \forall \ f > \max(K - x, 0) \]

\[ \phi < 0, \quad \forall \ f = \max(K - x, 0) \]

We use the particular forcing function \( \phi = -rK(\varepsilon/(f + \varepsilon - (K - x)))^\mu, \quad \mu \geq 1, \quad 0 < \varepsilon << 1 \) for evaluating the American Put option, as suggested in [8]. Our experiments lead to the values \( \mu = 2 \) and \( \varepsilon = 10^{-3} \) as workable parameters. The value \( \mu = 1 \) was used in [8].

It is necessary to evaluate the contribution of partial derivatives of the Jacobian matrix of the integral term \( \tilde{p} = \int \phi \tilde{\beta} \). Focus on a typical interval, say the first one \([x_1, x_2]\). Using the Gauss-Legendre formula yields a local contribution

\[ h_i \sum_k w_k \beta_j(\tilde{x}_k) \phi(f, \tilde{x}_k, t) = -h_i rK \sum_k w_k \beta_j(\tilde{x}_k)(\varepsilon/(f(\tilde{x}_k, t) + \varepsilon - (K - \tilde{x}_k)))^\mu \]

The values \( w_k, \tilde{x}_k \) are the weights and translated abscissas for the G-L formula. For this interval

\[ f(x,t) = y_i h_0(z) + y_i h_0(1-z) + h y_2 b_1(z) - h y_2 b_1(1-z) + h^2 y_3 b_2(z) + h^2 y_3 b_2(1-z) \]
From this we differentiate with respect to the \( y_j \) in this single interval. The contributions from all the intervals are accumulated as an outer step.

\[
\frac{\partial}{\partial y_i} \sum_k w_k \beta_j (\tilde{x}_k) \phi(f, \tilde{x}_k, t) = -h_i \mu \sum_k w_k \beta_j (\tilde{x}_k) \beta_i (\tilde{x}_k) \phi(f, \tilde{x}_k, t) / (f(\tilde{x}_k, t) + c - (K - \tilde{x}_k))
\]

This non-linear forcing term is useful for obtaining modest accuracy of the value of the American put function and related Greeks or partials of the function. It has the primary attribute of leading to a DAE that is solved with reasonable efficiency. Alternative methods appear to be relatively expensive or do not yield the Greeks without resorting to divided differences. On the other hand the amount of accuracy that can be obtained with this forcing term may be limited. This is difficult to measure without an alternative source or tables for comparison. There is 2-4 place agreement with tables found in [1], p. 176, and [9], p. 127. It is possible that (more expensive) adaptive quadrature for \( \tilde{p} = \int \phi \beta \) would be helpful to get accurate results in the solution. That was not investigated.

These graphics show the option value and two of the Greeks for the American Put example presented by [9], Table 1, p. 127.

![Figure 6. Value of American Put Option Over One Year](image-url)
Figure 7. The Value of Delta on the American Put Over One Year

Figure 8. The Value of Theta on the American Put
The following table lists some numerical values of the put and its greeks shown graphically in Figures 6 – 8 (Source code).

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Put Value</th>
<th>3 months</th>
<th>6 months</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0</td>
<td>30.0000</td>
<td>30.0000</td>
<td></td>
</tr>
<tr>
<td>20.0</td>
<td>20.0000</td>
<td>20.0000</td>
<td></td>
</tr>
<tr>
<td>30.0</td>
<td>10.0231</td>
<td>10.2532</td>
<td></td>
</tr>
<tr>
<td>40.0</td>
<td>2.9232</td>
<td>3.9784</td>
<td></td>
</tr>
<tr>
<td>50.0</td>
<td>0.5112</td>
<td>1.2802</td>
<td></td>
</tr>
<tr>
<td>60.0</td>
<td>0.0627</td>
<td>0.3681</td>
<td></td>
</tr>
<tr>
<td>70.0</td>
<td>0.0062</td>
<td>0.0999</td>
<td></td>
</tr>
<tr>
<td>80.0</td>
<td>0.0006</td>
<td>0.0264</td>
<td></td>
</tr>
<tr>
<td>90.0</td>
<td>0.0000</td>
<td>0.0065</td>
<td></td>
</tr>
</tbody>
</table>

Greeks at 3 months

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Put value</th>
<th>Delta</th>
<th>Gamma</th>
<th>Theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0</td>
<td>30.0000</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>20.0</td>
<td>20.0000</td>
<td>-1.0000</td>
<td>-0.0000</td>
<td>-0.0000</td>
</tr>
<tr>
<td>30.0</td>
<td>10.0231</td>
<td>-0.9586</td>
<td>0.0402</td>
<td>0.5501</td>
</tr>
<tr>
<td>40.0</td>
<td>2.9233</td>
<td>-0.4410</td>
<td>0.0511</td>
<td>5.3133</td>
</tr>
<tr>
<td>50.0</td>
<td>0.5112</td>
<td>-0.0997</td>
<td>0.0176</td>
<td>3.1959</td>
</tr>
<tr>
<td>60.0</td>
<td>0.0627</td>
<td>-0.0139</td>
<td>0.0030</td>
<td>0.8014</td>
</tr>
<tr>
<td>70.0</td>
<td>0.0063</td>
<td>-0.0015</td>
<td>0.0003</td>
<td>0.1288</td>
</tr>
<tr>
<td>80.0</td>
<td>0.0006</td>
<td>-0.0001</td>
<td>0.0000</td>
<td>0.0163</td>
</tr>
<tr>
<td>90.0</td>
<td>0.0000</td>
<td>-0.0000</td>
<td>0.0000</td>
<td>0.0018</td>
</tr>
</tbody>
</table>

Greeks at 6 months

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Put value</th>
<th>Delta</th>
<th>Gamma</th>
<th>Theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0</td>
<td>30.0000</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>-0.0000</td>
</tr>
<tr>
<td>20.0</td>
<td>20.0000</td>
<td>-1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>30.0</td>
<td>10.2532</td>
<td>-0.8505</td>
<td>0.0451</td>
<td>1.1019</td>
</tr>
<tr>
<td>40.0</td>
<td>3.9785</td>
<td>-0.4191</td>
<td>0.0367</td>
<td>3.4490</td>
</tr>
<tr>
<td>50.0</td>
<td>1.2803</td>
<td>-0.1538</td>
<td>0.0171</td>
<td>2.8736</td>
</tr>
<tr>
<td>60.0</td>
<td>0.3681</td>
<td>-0.0472</td>
<td>0.0058</td>
<td>1.4925</td>
</tr>
<tr>
<td>70.0</td>
<td>0.0999</td>
<td>-0.0132</td>
<td>0.0017</td>
<td>0.6123</td>
</tr>
<tr>
<td>80.0</td>
<td>0.0264</td>
<td>-0.0035</td>
<td>0.0005</td>
<td>0.2201</td>
</tr>
<tr>
<td>90.0</td>
<td>0.0066</td>
<td>-0.0010</td>
<td>0.0001</td>
<td>0.0678</td>
</tr>
</tbody>
</table>

Table 5. Numerical results for American option with forcing term
About the Author

Dr. Richard Hanson, Ph.D. is a Senior Scientist at Rogue Wave Software. His major project responsibilities include technical supervision, project management, and involvement in the full cycle of software development. Early IMSL Library projects included notable improvements in the IMSL real, symmetric, dense eigenvalue-eigenvector code, from which (given the right environment) an order of magnitude speed-up is obtained compared with LAPACK.

For the past twenty years, Dr. Hanson has also served on the IFIP WG 2.5 Committee on Mathematical Software. He was appointed for significant contributions to the publication of mathematics software and continues to participate in projects, which monitor and improve mathematical software tools, languages and working environments.

Some of his most recent significant work has been to design and implement MPI-enhanced subroutines and functions for scientific applications on distributed network computers. The IMSL Distributed Network FORTRAN Library (DNFL) was developed on arrays of workstations, and is currently available for the IBM SP-2, Hitachi SR2201, and Cray T3E.

References

12. IMSL C Numerical Library Version 7.0 (see Chapter 5: Differential Equations).