Pseudospectral Symbolic Computation For Financial Models

F.O. Bunnin, Y. Ren, Y. Guo and J. Darlington *

April 13, 1999

Abstract

The modelling of financial markets as continuous stochastic processes provides the means to analyse the implications of models and to compute prices for a host of financial instruments. We code as a symbolic computing program the analysis, initiated by Black, Scholes and Merton, of the formation of a partial differential equation whose solution is the value of a derivative security, from the specification of an undelying security's process. The Pseudospectral method is a high order solution method for partial differential equations that approximates the solution by global basis funcations. We apply symbolic transformations and approximating rewrite rules to extract essential information for the Pseudospectral Chebyshev solution. We write these programs in Mathematica. Our C++ template implementing general solver code is parameterised with this information to create instrument and model specific pricing code. The Black-Scholes model and the Cox Ingersoll Ross term structure model are used as examples.

1 Introduction

The theoretical analysis of financial models involves symbolic analysis based on results from stochastic calculus. Yet to derive the logically implied prices of financial instruments numerical methods are often essential. For practitioners in financial markets to evaluate new

*Dept. of Computing/Fujitsu Parallel Computing Research Centre, Imperial College, London, SW7 2BZ, England, email: {fob1, yr, yg, jd}@doc.ic.ac.uk

models and analyse new instruments it is desirable to have a tool that combines the theoretical stochastic analysis with efficient numerical methods that will create pricing programs based on user specifications of model and instrument.

The seminal work of Black, Scholes and Merton in continuous time finance demonstrated a method of analysis whereby modelling asset prices as Ito processes leads to the prices of financial derivatives, such as options, based on the underlying assets, being the solution of a deterministic partial differential equation, for example the Black Scholes equation. We automate this analysis in a symbolic computer program written in Mathematica, so that it can be applied to a wide range of models and instruments.

The Pseudospectral method of solving partial differential equations uses global basis functions to approximate the solution function. When this solution is smooth the Pseudospectral method can obtain high order numerical solutions. We use Chebyshev polynomials as basis functions, and Chebyshev collocation points for space discretisation, in a subsequent symbolic computing program that retrieves the problem specific information from partial differential equations. A coordinate transform function maps the financial space of the equation to an appropriate computation space, and a function that applies the partial differential operator of the space transformed equation to Chebyshev polynomials at Chebyshev collocation points is created. This function and the appropriate boundary and initial conditions then parameterise a template function that contains general solver code in C++.

We show numerical results obtained from pricing a European Call option using the Black-Scholes model, and from pricing a zero-coupon bond using the Cox Ingersoll Ross model of the term structure of interest rates.

In this paper section (2) starts by explaining the Pseudospectral method and discussing its use for financial equations. Next the formation of partial differential equations and the application of the Pseudospectral method to two fundamental equations are described in section (3). An explanantion of the symbolic transformation program and its implementation follows in section (4). We present our numerical results from our programs and our conclusions in section (5).

2 Pseudospectral Chebyshev Method

The finite difference method has been used to solve financial equations. Courtadon [1], Hull and White [6], and Wilmott et al [8] suggest various methods including explicit, fully implicit and Crank-Nicolson schemes. The explicit scheme for the untransformed Black-Scholes equation has convergence and stability conditions, and whilst the fully implicit and Crank-Nicolson schemes are unconditionally convergent and stable, spurious oscillations may yet occur in the solution. The accuracy of the explicit and fully implicit Euler schemes is first order, and Crank-Nicolson's accuracy is second order. The solutions to Black-Scholes for European Call and Put options are smooth. Considering this higher order numerical techniques should lead to higher accuracy.

Pseudospectral (PS) methods play an important role in the numerical solution of differential equations. PS solutions are approximations by global basis functions, whereas finite difference and finite element methods are local approximations. For problems with smooth solutions PS method can obtain higher order numerical solutions as the number of collocation points increases.

Suppose a function $u(t,x)$ is approximated by the Nth degree polynomial expressed as

$$u(t,x) \sim u_N(t,x) = \sum_{k=0}^{N} a_k(t) T_k(x)$$

where T_k is a Chebyshev polynomial of the first kind of degree k. Here the Chebyshev polynomial of degree k is defined by

$$T_k(x) = \cos(k \cos^{-1} x).$$

The interpolation points in the interval $(-1,1)$ are chosen to be the extrema

$$x_j = \cos \frac{\pi j}{N} \quad (j = 0, 1, \cdots, N)$$

of the Nth order Chebyshev polynomials $T_N(x)$.

It follows that

$$\begin{aligned} T_k(x_j) &= \cos \frac{\pi j k}{N} \\ T_k'(x_j) &= \frac{k \sin \frac{\pi j k}{N}}{\sin \frac{\pi j}{N}} \end{aligned} \tag{2.1}$$

$$T_k''(x_j) \quad = \quad \frac{k(\cos\frac{\pi j}{N}\frac{\sin\frac{\pi jk}{N}}{\sin\frac{\pi j}{N}} - k\cos\frac{\pi j}{N})}{\sin^2\frac{\pi j}{N}}$$

The coefficients $\{a_k\}_{k=0}^N$ are chosen such that

$$\begin{aligned}
\sum_{k=0}^N T_k(x_j)\frac{da_k}{dt} &= \sum_{k=0}^N LT_k(x_j)a_k(t) \quad for \;\; j = 1, 2, \cdots, N-1 \\
\sum_{k=0}^N a_k(t)T_k(x_0) &= u(t, x_0) \qquad (2.2) \\
\sum_{k=0}^N a_k(t)T_k(x_N) &= u(t, x_N)
\end{aligned}$$

Where $-LT_k(S) = rS\frac{\partial T_k}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 T_k}{\partial S^2} - rT_k$ for the Black-Scholes equation and $-LT_k(r) = A(b-r)\frac{\partial T_k}{\partial r} + \frac{1}{2}C^2 r\frac{\partial^2 T_k}{\partial r^2} - rT_k$ for the Cox Ingersoll Ross equation.

If $u^m(t,x)$ is continuous for all $|x| \leq 1$ for $m = 0, 1, \cdots, n-1$, and $u^N(t,x)$ is integrable, then

$$a_k \ll 1/k^n,$$

as $k \to \infty$. Since $|T_k(x)| \leq 1$, it follows that the remainder after N terms of the Chebyshev series is asymptotically much smaller than $1/N^{n-1}$ as $N \to \infty$.

The problem (2.2) is an initial value ODE system. The initial condition for $\{a_k\}_{k=0}^N$ are determined by $u_N(0,x) = u(0,x)$, that is

$$\sum_{k=0}^N a_k(0)T_k(x_j) = u(0, x_j), \quad for \;\; j = 0, 1, \cdots, N \qquad (2.3)$$

The first derivative of u is approximated by

$$\frac{du_N}{dx} = \sum_{k=0}^N b_k^{(1)}(t)T_k(x)$$

where by the properties of Chebyshev polynomials (see [5])

$$b^{(1)} = E^{(1)}a.$$

We apply the Implicit-Euler method and the Crank-Nicholson scheme to the ODEs (2.2), so that we obtain

$$\begin{aligned}\sum_{k=0}^{N} a_k^{i+1} T_k(x_0) &= u(t^{i+1}, x_0) \\ \sum_{k=0}^{N} [T_k(x_j) - \delta t \theta L T_k(x_j)] a_k^{i+1} &= \sum_{k=0}^{N} [T_k(x_j) + \delta t (1-\theta) L T_k(x_j)] a_k^{i} \\ \sum_{k=0}^{N} a_k^{i+1} T_k(x_N) &= u(t^{i+1}, x_N)\end{aligned} \tag{2.4}$$

for $j = 1, 2, \cdots, N-1$ and $i = 1, 2, \cdots, M$. Where δt is the time stepsize and θ is the parameter of schemes, i.e. $\theta = 1, 1/2, 0$ for Implicit, Crank-Nicolson and Explicit schemes, respectively.

3 Application of Pseudospectral Method to Financial Equations

In this section we have chosen two important financial models to sketch the analysis that leads to partial differential equations and show concretely their numerical solution by the Pseudospectral Chebyshev method. The first model is the constant proportional drift and diffusion stock price process that leads to the famous Black-Scholes equation. The second model is the Cox Ingersoll Ross model of the term structure of interest rates. This model demonstrates a mean reverting drift term and a square root diffusion term. The prices of zero coupon bonds are fundamental to pricing any interest rate instrument, such as swaps, swaptions, and caps. More detail on the analysis used can be found in Duffie [2].

Example 1. Given a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t), \mathcal{P})$ we assume the price of a non-dividend paying stock S is an $\mathcal{F}_t$ adapted process that satisfies:

$$dS = \mu S dt + \sigma S dW,$$

Where $(W_t, \mathcal{F}_t; 0 \leq t \leq \infty)$ is a standard Brownian Motion, μ and σ are constants. Applying Ito's lemma to derive an expression for a derivative u written on S, constructing a dynamic trading strategy that replicates the expiry time payoff of the derivative, applying the no arbitrage condition, and using sample path properties of Brownian Motion, the Black-Scholes equation is formed:

$$\frac{\partial u}{\partial t} + rS\frac{\partial u}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - ru = 0. \tag{3.5}$$

$$for\ \ 0 \leq t \leq T,\ \ 0 < S < \infty.$$

The infinite space domain is truncated to $[a, b]$ and then for application of the Chebyshev Pseudospectral method the truncated space is transformed to the computational domain $[-1, 1]$. Let $S = \frac{b-a}{2}x + \frac{b+a}{2}$ and let $\tau = T - t$ to make the equation a forward one. Then $J = \frac{\partial S}{\partial x} = \frac{b-a}{2}$. Where $x \in [-1, 1]$ and $S \in [a, b]$ and T is expiry time.

Therefore the Black-Scholes equation becomes

$$\frac{\partial U}{\partial \tau} = \frac{1}{2}\sigma^2 S^2/J^2 \frac{\partial^2 U}{\partial x^2} + (r - q)S/J \frac{\partial U}{\partial x} - rU := LU \tag{3.6}$$

where $U(\tau, x) := u(t, S)$. The payoff of a European Call is $(S - K)^+$, where K is the strike price. The expiry time condition for (3.5) is thus $u(T, S) = (S - K)^+$. When $S \to 0, u(t, S) \to 0$, and when $S \to \infty, u(t, S) \to S - Ke^{-r(T-t)}$. The time dimension is discretised by a mesh $\pi : 0 = \tau_0 < \tau_1 < \ldots < \tau_M = T$ with equidistant time steps $\tau_{i+1} - \tau_i = \delta\tau$ for $i = 0$ to $M - 1$. For each time step the solution is approximated by a linear combination of the first N+1 Chebyshev polynomials:

$$U(\tau_i, x) \sim U_N(\tau_i, x) = \sum_{j=0}^{N} a_j(\tau_i)T_j(x)$$

The coefficients $a_j(\tau_i)$ are found by setting the approximation U_N equal to the solution U at the collocation points $x_j, j = 1$ to $N - 1$, and using the boundary conditions for x_0 and x_N. This is achieved by replacing U by U_N in (3.6), and performing the required differentiation. This results in a set of N+1 equations, N-1 ordinary differential equations and the two boundary conditions, in N+1 unknowns: $a_j(\tau_i)$ $j = 0$ to $m, i = 1$ to n. The solutions of the coefficients at τ_{i-1} are the initial conditions for the set of ordinary differential equations at τ_i. At τ_0 the coefficients are found directly from the initial condition. Crank-Nicolson is used to solve the ordinary differential equations at each time step. At T the solution in the form of a function of x is found from the calculated $a_j(T)$.
Table 3.1 has results for when K=10, r=0.1, σ=0.4, T=1, N=100 and dt=0.01.

Example 2. Assuming that the term structure is driven by the short term interest rate r, which under an appropriate measure satisfies:

$$dr = A(b - r)dt + C\sqrt{r}dW$$

with A, b, and C constants, the zero coupon bond function f(t,T,r) for the price of a T-maturity bond at time t, satisfies the partial differential equation:

$$\frac{\partial f}{\partial t} + A(b - r)\frac{\partial f}{\partial r} + \frac{1}{2}C^2 r \frac{\partial^2 f}{\partial r^2} - rf = 0 \tag{3.7}$$

Table 3.1: European Call Option.

Stock Price	Option Price (PS)	Option Price (True)	Relative Error
3	0.0020	0.0018	-0.1059
4	0.0187	0.0180	-0.0370
5	0.0817	0.0808	-0.0116
6	0.2307	0.2302	-0.0021
7	0.4954	0.4960	0.0012
8	0.8878	0.8897	0.0020
9	1.4036	1.4065	0.0020
10	2.0284	2.0318	0.0017
11	2.7436	2.7474	0.0014
12	3.5305	3.5347	0.0012
13	4.3724	4.3774	0.0011
14	5.2556	5.2619	0.0012
15	6.1689	6.1776	0.0014
16	7.1039	7.1163	0.0017
17	8.0539	8.0717	0.0022
18	9.0141	9.0393	0.0027
19	9.9810	10.0158	0.0035
20	10.9516	10.9987	0.0043

with terminal condition $f(T,T,r) = 1$. As $r \to \infty$, $f(t,T,r) \to 0$, and as $r \to 0$, $f(t,T,r) \to 1$. We apply a transformation from truncated physical space to computational space similarly to the previous example: $r \in [a,b]$ to $y \in [-1,1]$ and let $J = \frac{\partial S}{\partial y} = \frac{b-a}{2}$. Equation (3.7) becomes

$$\begin{aligned}\frac{\partial F}{\partial \tau} &= A(b - (\frac{b-a}{2}y + \frac{b+a}{2}))/J\frac{\partial F}{\partial y} + \frac{1}{2}C^2(\frac{b-a}{2}y + \frac{b+a}{2})\frac{\partial^2 F}{\partial y^2} \\ &- (\frac{b-a}{2}y + \frac{b+a}{2})F \end{aligned} \tag{3.8}$$

In this case the intitial condition becomes $F(r,t) = 1$. The computation then proceeds similarly to example 1. Results for a one year zero-coupon bond price when A=0.5, b=0.05, C=0.8 with N=100 and dt=0.01 are shown in table 3.2.

4 Symbolic Transformation Program

The economic and mathematical analysis applied to financial models, detailed in the previous two sections, is systematic and therefore can be automised by a computer program. The result of the analysis is the pricing of the specified instrument given a model of the relevant fincancial market, i.e. of the underlying asset or variable. In first example the instrument

Table 3.2: Bond Price for CIR model

Short term rate	Computed Price	Analytic Price	Absolute Error
0	1.0000	1.0035	0.0035
0.49%	0.9954	0.9999	0.0045
1.97%	0.9830	0.9892	0.0061
4.44%	0.9640	0.9716	0.0075
7.89%	0.9389	0.9475	0.0086
12.31%	0.9083	0.9175	0.0092
17.71%	0.8726	0.8821	0.0095
24.08%	0.8327	0.8416	0.0089
31.42%	0.7892	0.7984	0.0092
39.71%	0.7429	0.7517	0.0088
48.94%	0.6946	0.7028	0.0082

was a European Call option on a non-dividend paying stock modelled by geometric Brownian Motion, and in our second example the instrument was a zero coupon non-defaultable bond assuming a term structure of interest rates driven by a short rate modelled by a mean-reverting square root process.

As many such models admit no closed form solution for the prices of certain financial instruments, the solution is found numerically using the high order Pseudospectral technique. The systematic analysis is valid for a wide range of models and instruments: generally it is valid for no-arbitrage models of complete markets, i.e. markets in which every contingent claim can be replicated by a dynamic position in the underlying assets (completeness) and in which there do not exist riskless trading strategies with expected profit greater than the risk free rate (no-arbitrage), and for instruments that whose no-arbitrage value can be expressed as functions of the current values of the underlying assets and time.

Therefore the program that implements this analysis takes as input the specifications of the model and instrument in the form of SDEs and the terminal payoff function of the instrument. It performs the step by step symbolic transformation of these inputs to result in a computer program that returns the price of the instrument given the relevant parameter values. The symbolic transformation program is thus a map from models and instruments to pricing programs, and the pricing programs are functions of underlying parameters, such as the current value and volatility of the underlying asset, that return the number that is the theoretical price of the instrument.

4.1 Symbolic Transformation Implementation

We implement the symbolic transformation program in Mathematica in order to exploit its inbuilt symbolic calculus and its rewrite system. Deterministic calculus is performed automatically in Mathematica, and we have written functions to perform stochastic calculus, for example the application of Ito's Lemma which is the stochastic extension of the chain rule. Rewrite rules of the form $Rule[a, b]$ or $a \to b$ find instances of a and replace them with b. Rules are applied to expressions using the syntax $expression/.rule$.

The first step is for the user to specify the financial model of the underlying asset or variable in the form of a SDE. This requires the name chosen for the process and the functional form of the drift and diffusion coefficients. Following our first example from the previous section the constant proportional drift and diffusion stock price model is specified by the user.

$$dS = SDE[S, \mu S, \sigma S]$$

The function SDE takes arguments of the process name, its drift and its diffusion. It returns a SDE:

$$dS = \mu S dt + \sigma S dW.$$

We choose to represent SDEs in the common symbolic 'differential' form, which really should be understood as shorthand for an integral equation. Thus

$$dS = a(S, t)dt + b(S, t)dW,$$

is shorthand for

$$S_t = S_0 + \int_0^t a(S, v)dv + \int_0^t b(S, v)dW_v,$$

where the second integral is an Ito integral, with respect to a standard Brownian motion, W. As the 'differential' form can still be manipulated in a meaningful way, and as the integral notation can become cumbersome, we prefer to use the 'differential' form.

The next step is to derive an expression for the process followed by the instrument specified by its payoff function. At this stage we assume that it can be expressed as a function of the current values of the underlying asset, S_t, and time, t. This means we can us Ito's Lemma to derive another SDE, this time the one that is satisfied by the derivative instrument. The underlying asset's SDE is passed as the second argument to a function that applies Ito's lemma to result in the SDE whose solution is named by the first argument. Therefore choosing to name the derivative u, and using the fact that dS has been assigned by the SDE for the underlying stock S we have:

$$du = ItoLemma[u[S,t], dS]$$

$$(\frac{\partial u[S,t]}{\partial t} + \mu S\frac{\partial u[S,t]}{\partial S} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 u[S,t]}{\partial S^2})dt + \sigma S\frac{\partial u[S,t]}{\partial S}dW.$$

Similarly to the first SDE this has a drift term and a diffusion term, though the forms are more complex.

Now that we have expressions for both the underlying asset and the derivative asset we can follow the economic analysis pioneered by Black and Scholes to form a deterministic PDE whose solution given appropriate boundary conditions is the pricing function u. We form an instantaneously risk free portfolio of -1 derivative securities and $\frac{\partial u}{\partial S}$ shares, and use the no-arbitrage condition to equate the return on this portfolio to the risk free rate, r. Thus we derive, in this case, the Black-Scholes PDE. Note that this analysis can be applied to more general Ito processes involving more complicated functional forms for the drift and diffusion coefficients, and that the formation of the PDE relies on market completeness and no-arbitrage, which we assume from the start. The function FormPDE is written in Mathematica and uses the above analysis, incorporating the Ito's Lemma function within its body. It returns the linear second order parabolic equation whose solution is the pricing function.

$$blackscholespde = FormPDE[u[S,t], dS]$$

$$\frac{\partial u[S,t]}{\partial t} + rS\frac{\partial u[S,t]}{\partial S} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 u[S,t]}{\partial S^2} - ru[S,t] == 0. \tag{4.9}$$

The final condition is provided by the payoff at expiry time, T, of the derivative security. In this example the payoff is for a European Call option, thus we assign:

$$payoff[S\_, T\_] := Max[S-K, 0] \tag{4.10}$$

where K is the strike price. The upper and lower boundary conditions are derived from this final condition and assigned similarly:

$$\begin{aligned} upperboundary[S\_, t\_] &:= S - KExp[-r(T-t)] \\ lowerboundary[S\_, t\_] &:= 0 \end{aligned}$$

The transformations up to this stage therefore formulate the problem in a tractable form, i.e. in a well posed PDE with boundary conditions. The next transformations approximate the problem by one that can be solved numerically by the pseudospectral method. The first step here is to map the financial space of the problem to the computational space by applying a coordinate transform to the PDE. Let $u[x]$ be the function that solves the PDE with x being the original coordinates, and $U[X] := u[x]$ be the function that solves the PDE in transformed coordinates X, with the dimensionality of x and X being equal.

Our function $coordinatetransform$ takes as arguments the PDE in original coordinates, the name of the solution function, u, a list of the old coordinates, x, a list of the maps from new coordinates to old, and a list of the names of the new coordinates, X. It works by replacing the partial derivatives of the solution function with respect to the old coordinates by equivalent expressions involving partial derivatives with respect to the new coordinates. The helper function $gradtransform$ calculates the equivalent expressions. It returns a pure function mapping a function to a list of its partial derivatives (w.r.t. x) calculated as the symbolic matrix-vector product of the inverse of the function's Jacobi matrix (w.r.t. X), and the vector of partial derivatives with respect to X. Thus when nested $gradtransform$ returns higher order partial derivatives. The definition of the $coordinatetransform$ function is:

```
coordtransform  [expr_, soln_, oldCoords_, transf_, newCoords_] :=
    Module      [{oldseq, oldfirstp, oldsecondp, firstp, secondp, neweqtn},
                oldseq = Sequence@@oldCoords;
                oldfirstp = Grad[f_[oldseq], oldCoords];
                oldsecondp = Flatten[Grad[oldfirstp, oldCoords]];
                firstp = gradtransform[transf, newCoords][f@@newCoords];
                secondp = Flatten[Nest[gradtransform[transf, newCoords], f@@newCoords, 2]
                neweqtn = FullSimplify[expr/.Join[MapThread[Rule, {oldfirstp, firstp}],
                MapThread[Rule, {oldsecondp, secondp}],
                MapThread[Rule, {oldCoords, transf}]]];
                neweqtn/.soln[Sequence@@transf]-> soln[Sequence@@newCoords]
        ]
```

(See Wolfram ([9]) for explanations of inbuilt Mathematica list processing functions used above.)

The Chebyshev Pseudospectral method requires a space domain of $[-1,1]$ so we truncate the financial domain to $[a,b]$ and map this onto $[-1,1]$, thereby forming a new PDE:

$$transformedpde = coordinatetransform[blackscholespde, u, \{S,t\}, \{(a+b-ax+bx)/2, -\tau\}, \{x,\tau\}]$$

$$\frac{\partial U[x,t]}{\partial \tau} = \frac{(a+b-ax+bx)^2\sigma^2}{2(a-b)^2}\frac{\partial^2 U[x,t]}{\partial x^2} + \frac{r(a+b-ax+bx)}{-a+b}\frac{\partial U[x,t]}{\partial x} - rU[x,t]. \quad (4.11)$$

In our numerical solution the function $u[x,t]$ is approximated by a polynomial which is the sum of terms of the form $a_k(t)T_k(x)$, where $a_k(t)$ is a coefficient and T_k is the Chebyshev polynomial of the first kind of degree k. In the section detailing the Pseudospectral method it was shown that the coefficients $\{a_k\}_{k=0}^N$ are calculated by solving the initial value ODE system (2.2).

Whichever method is chosen to solve such a system, the problem specific information is contained in the partial differential operator L applied to the Chebyshev polynomilas T_k at the Chebyshev collocation points $\{x_j\}_{j=0}^N$; that is we can find expressions for $LT_k(x_j)$ from the specific problem in the form of a function taking arguments of k,j and n and this can be passed to general ODE solution code. Such a function can be derived from a PDE, for example (4.11), by replacing instances of the solution function in the PDE by the Chebyshev polynomial of kth degree evaluated at the jth collocation point, replacing partial derivatives with respect to the space coordinates of the solution function by the derivatives of the Chebyshev polynomial of kth degree evaluated at the jth collocation point (see equations (2.1)), and removing the partial derivative with respect to the time variable. The function $FormLTk$ consists of a list of these rewrite rules:

$$\begin{aligned} FormLTk\ [\ V\_, k\_, j\_, n\_] := \{&Rule[V[x\_,t\_], Cos[\pi kj/n],\\ &Rule[\frac{\partial U[x\_,t\_]}{\partial x\_}, \frac{Sin[\pi kj/n]k}{Sin[\pi j/n]}]\\ &Rule[\frac{\partial U[x\_,t\_]^2}{\partial x\_^2}, \frac{k(Cos[\pi j/n]\frac{Sin[\pi kj/n]}{Sin[\pi j/n]} - kCos[\pi kj/n]}{Sin[\pi j/n]^2}],\\ &Rule[\frac{\partial U[x\_,t\_]}{\partial t\_}, 0]\} \end{aligned} \quad (4.12)$$

Thus we apply the rules above to (4.11) and create a new function LTk that given arguments of k, the degree of the Chebyshev function, j, the collocation point, and n, the number of collocation points minus one, returns the result of evaluating the problem specific partial differential operator L on the Chebyshev polynomial of degree k at the collocation point x_j.

$$LTk[k\_, j\_, n\_] := transformedpde/.FormLTk[V, k, j, n].$$

The function LTk is then passed to general ODE system solver code. The implicit time stepping algorithm implied by the equations (2.4) forms the basis for an algorithmic template written in C++ which is parameterised by the C++ versions of the functions LTk, $payoff$, *upperboundary* and *lowerboundary*. This can be achieved either through the use of the solver code taking arguments of pointers to functions, or by the solver code being a function template parameterised by a class or classes that have the necessary functions as member functions. This latter approach is the one we prefer. The function template is further parameterised by the number of basis functions to use in the approximation (which also gives the number of collocation points), and by θ, the parameter of numerical schemes. Implicit scheme solutions of the ODE system require LU decomposition. A library call within the function template is used so that parallel or sequential machine optimised versions can be used.

5 Conclusions.

Systematic economic and mathematical analysis forms the basis for our symbolic computing program that derives pricing programs from user specified financial models. Results from stochastic calculus and continuous time finance theory applied to financial instruments that are modelled as stochastic processes enable the formation of a partial differential equation, which is then made suitable for numerical solution by the Pseudospectral Chebyshev method. Problem specific information is extracted form the equation and combined with general solver code to create the model and instrument specific pricing program. Pricing codes for European Call options under the Black Scholes model, and for zero-coupon bonds under the Cox Ingersoll Ross model of interest rates have been produced and accurate results achieved.

Acknowledgements:
This project is funded by the ESRC award RO22250129. F.O. Bunnin is funded by the EPSRC award 97306304.

References

[1] G. Courtadon, *A More Accurate Finite Difference Approximation for the Valuation of Options,* Journal of Financial and Quantitative Analysis, Vol.17 December 1982

[2] D. Duffie, *Dynamic Asset Pricing Theory,* Princeton University Press, 1996.

[3] B. Fornberg, *A practical guide to pseudospectral methods,* Cambridge University Press, 1996.

[4] B. Fornberg and D. Sloan, *A review of PS methods for solving PDEs,* Acta Numerica, pp. 203-267, 1994.

[5] D. Gottlieb and S. A. Orszag, *Spectral Methods for Partial Differential Equations,* SIAM, Philadelphia, 1977.

[6] J. Hull and A. White, *Valuing Derivative Securites Using the Explicit Finite Difference Method,* Journal of Financial and Quantitative Analysis, Vol. 25 March 1990.

[7] C. Randall, E. Kant and A. Chhabra, *Using Program Synthesis to Price Derivatives,* Journal of Computational Finance, Vol.1, No.2, pp 97-129, 1997/98.

[8] P. Wilmott, G. Dewynne and S. Howison *Option Pricing: Mathematical Models and Computation,* Oxford Financial Press, 1993.

[9] Wolfram S., "The Mathematica Book", Cambridge University Press 1996

[10] R. Zvan, P.A. Forsyth and K.R. Vetzal, *Robust Numerical Methods for PDE models of Asian Options,* Journal of Computational Finance, Vol.1, No.2, pp 39-78, 1997/98.