Unconditionally stable and second-order accurate explicit Finite Difference Schemes using Domain Transformation Part I. One-Factor Equity Problems

Daniel J. Duffy, dduffy@datasim.nl

September 7, 2009

Abstract

We introduce a class of stable and second-order accurate finite difference schemes that resolve a number of problems when approximating the solution of option pricing models in computational finance using the finite difference method (FDM). In particular, we show how to avoid having to apply truncation methods to the domain of integration as well as the resulting ad-hoc experimentation in choosing suitable numerical boundary conditions that we must apply on the boundary of the truncated domain. Instead, we transform the PDE option model (which is originally defined on a semi-infinite interval) to a PDE that is defined on a bounded interval. We then apply the elegant Fichera theory to help us determine which boundary conditions to apply to the transformed PDE. We show that the new system is well-posed by proving energy inequalities in the space of square-integrable functions. Having done that, we adapt the Alternating Direction Explicit (ADE) (a method that dates from the first half of the last century) method to compute an approximation to the solution of the transformed PDE. We prove that the explicit scheme is unconditionally stable and second-order accurate. The scheme is very easy to model, to program and to parallelise and the generalization to n-factor problems is easily motivated.

We examine equity problems and we compare the ADE approach (in terms of accuracy and speedup) with more traditional finite difference schemes and with the Monte Carlo method. Finally, we stress-test the scheme by varying critical parameters over a spectrum of values and the results are noted.

The methods in this article - in particular the combination of pure and applied techniques - are relatively new in the computational finance literature in our opinion, in particular the realization that a PDE on a semi-infinite interval can be transformed to one on a bounded interval and that the new PDE can be approximated by schemes other than Crank-Nicolson and its workarounds. Finally, we shall report on n-factor models in future articles.

Keywords: ADE method, domain transformation, Fichera method and boundary condition, Feller condition, conditionally consistent schemes, option pricing, exponentially fitted schemes, C++, OpenMP and parallel processing.

1 Background and Challenges

In this article we discuss a number of techniques and methods for solving partial differential equations (PDEs) in derivatives pricing and we reduce the scope by concentrating on one-factor model problems relating to equities, although our approach is applicable to many-factor pricing applications and more general partial differential equations in finance, science and engineering. In general, the PDEs in this article will have two independent variables, the first of which represents time in the interval $[0, T]$ where T is the expiry time and the second variable represents some quantity such as stock, an index, interest rate or other 'underlying'. This latter variable is defined on the positive semi-infinite interval.

There are three main novel contributions to the numerical solutions of PDEs in finance in this article; first, we transform the original PDE on a semi-infinite interval to a PDE on the unit interval by a transformation of the underlying variable. Second, we show how to define boundary conditions for this transformed PDE by appealing to the Fichera theory (Fichera 1956). This theory allows us to put our analysis on a firm footing and it helps us avoid ad-hoc methods to determine which boundary conditions to use. Finally, we approximate the transformed PDE by the so-called *Alternating Direction Explicit* (ADE) (Saul'yev 1964) method which is a finite difference method (FDM) dating from the first half of the twentieth century. This method is explicit, unconditionally stable, second-order accurate and it can be parallelized. This means that we can achieve good performance on multi-core CPUs using C++ in combination with multi-threading libraries such as OpenMP or Boost threads, for example. We give numerical examples to show how the new methods work in practice.

In the rest of this section we give an overview of some standard problems, challenges and solutions when pricing derivatives using PDE techniques and how current methods attempt to solve them. In succeeding sections we show how to resolve these difficulties using the new approach.

1.1 PDE Models for Derivatives' Pricing

In this section we focus on generic one-factor PDEs of the form:

$$\frac{\partial u}{\partial t} = a(x,t)\frac{\partial^2 u}{\partial x^2} + b(x,t)\frac{\partial u}{\partial x} + c(x,t)u + f(x,t), \quad 0 < x < \infty, \quad 0 < t < T \tag{1}$$

In this case the variable t represents (increasing) time while the variable x represents a spatial dimension. We shall discuss specializations of (1) in later sections where the variable x has a special meaning (for example, it could represent an interest rate or a stock price) and the coefficients $a(x,t)$, $b(x,t)$ and $c(x,t)$ have specific forms. In order to specify (1) we first need to augment it with an initial condition:

$$u(x,0) = f(x), \quad 0 < x < \infty \tag{2}$$

and we must specify boundary conditions when $x = 0$ and $x = \infty$. We defer discussion of this last issue until section 2 where we introduce the Fichera theory and domain transformations that together allow us to describe and determine how to define appropriate boundary conditions that when taken with equations (1) and (2) lead to a well-posed initial boundary value problem (IBVP). In particular, we wish to know if the IBVP has a unique solution in some space of functions and if this solution depends continuously on the input representing the boundary and initial conditions as well as the known coefficients in equation (1). In particular, proving that the IBVP satisfies the maximum principle (Il'in 1962, Ladyenskaja 1988) is extremely useful because it ensures that positive input leads to a solution of the IBVP whose values are always positive and hence we cannot get non-physical solutions or solutions that have no financial meaning, for example a negative option price. It is obvious that finite difference schemes that approximate the IBVP should also produce positive values but unfortunately not all schemes have this property. There is however a family of schemes - called *monotone FDM* - which do satisfy the discrete equivalent of the maximum principle which ensures positive values for the discrete solution. We have produced monotone schemes for the above IVBP in Duffy 1980 using *M-matrix theory* (Varga 1962).

1.2 Convection-dominated Problems and the Influence of the PDE Coefficients

The PDE represented by equation (1) is an example of a time-dependent inhomogeneous convection-diffusion-reaction equation. In particular, the known functions appearing in that equation have a particular meaning:

- $a(x,t)$; the diffusion coefficient
- $b(x,t)$; the convection coefficient
- $c(x,t)$; the reaction coefficient
- $f(x,t)$; the inhomogeneous forcing term

The qualitative properties of the IBVP are in large part determined by the diffusion and convection terms and the relationships between them:

- They determine whether boundary conditions need to be specified and if so, on which parts of the boundary they should be imposed; the Fichera theory answers this question. The question of defining the correct boundary conditions for PDEs in computational finance is not well understood and there are many opinions on how they should be applied. Hopefully the presentation of the Fichera theory should resolve some of these misunderstandings.
- Many problems in computational fluid dynamics, hydrodynamics and groundwater flow can be modelled using PDEs that are similar to equation (1) (see for example, Roache 1998, Anderson 1995, Laney 1998). In particular, it exhibits *boundary layer behavior* when the diffusion coefficient is small or when the convection term is large (or both). In these cases we see that traditional finite difference schemes lead to oscillating solutions and thus exhibit signs of *static instability* (Roache 1998). One solution to this problem is to employ *exponentially fitted finite difference schemes* (Duffy 1980) that are able to model these boundary layer problems by avoiding *spurious* (non-physical) solutions. In finance, we encounter boundary layer behaviour when volatility is large and interest rates are low. We discuss their numerical solution and implementation using C++ in Duffy 2006 and Duffy 2004, respectively.
- Equation (1) is a general convection-diffusion-reaction equation and it describes a wide range of phenomena in mathematical physics. It is worth noting that it has a number of special cases when one or more of the coefficients in the equations are identically zero, for example:
  - *Diffusion equation*: this is the generalization of the heat equation and is one of the most important PDEs in mathematical physics

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2} \tag{3}$$

- *Convection equation*: this is a first order hyperbolic wave equation

$$\frac{\partial u}{\partial t} = b\frac{\partial u}{\partial x} \tag{4}$$

- *Diffusion reaction*

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2} + cu \tag{5}$$

- *Convection reaction*

$$\frac{\partial u}{\partial t} = b\frac{\partial u}{\partial x} + cu \tag{6}$$

- *Ordinary differential equation*

$$\frac{du}{dt} = cu \tag{7}$$

These equations arise in computational finance in various forms and it is important to know what their qualitative properties are and how to formulate them as well-posed initial boundary value problems. Furthermore, in most cases these PDEs do not have analytic or readily computable solutions and we must resort to numerical methods in order to find an approximate solution to them. In particular, we shall see that the Alternating Direction Explicit (ADE) method can be used to solve them.

1.3 A Critique of Finite Difference Schemes

The finite difference method is used in computational finance for a range of equity, commodity and interest rate problems. In general, finite difference schemes are suitable for one, two and three factor problems and in this article we concentrate on one factor models, as already stated.

In this section we give a short summary of how FDM is used in computational finance in general, which methods are being used and what the consequences of using these methods are. We start with some general comments:

- Many schemes are based on three mesh points in the x variable and centred divided differences are used to approximate the derivatives in equation (1). This approach can lead to spurious oscillations especially when the diffusion term is small and/or the drift term is large.

- The PDEs in computational finance tend to be defined in a semi-infinite interval in space (typically the underlying financial variable has non-negative values). Of course, the finite difference method must be defined on a finite interval and to this end most methods truncate the semi-infinite interval. How this is done and what the ensuing numerical boundary conditions will be has been a source of debate and uncertainty. We resolve this problem, not by truncation but by transformation of the above semi-infinite interval to the unit interval.

- The Crank Nicolson scheme seems to be one of the most popular methods for time discretisation but it produces wiggles in both the option price and its sensitivities near the points of discontinuity (see Duffy 2004A for a discussion).

- Implicit schemes are stable but we must solve a matrix system at each time level to arrive at a solution. It would be nice if we could find stable schemes which are explicit and hence they should be efficient in time and resource usage.

- Most popular finite difference schemes are difficult to parallelise which is unfortunate because modern laptop and desktop computers have multiple core processors. We would like to exploit this hardware parallelism in the software algorithms that implement finite difference schemes.

Our objective in this article is to resolve these problems by introducing a new way of looking at PDEs and their numerical solution.

1.4 Numerical Boundary Conditions

The formulation and consequent numerical approximation of boundary conditions is one of the least understood aspects of the application of PDE methods to problems in computational finance. In general, the original PDE is defined on a semi-infinite interval but when approximating it using finite differences we need to replace this interval by a bounded interval (A, B) and having done that we define appropriate numerical boundary conditions on the boundaries of the interval. Articles and books abound with techniques, tricks and rules-of-thumb that allow us to find the interval and corresponding boundary conditions. Some authors apply continuous boundary conditions at the new *far field boundary* B (for example, demanding that the first derivative or second derivative is zero there, the latter condition being called the *linearity or convexity condition* (Windcliff 2003) but after having truncated the interval we are by definition solving a numerical model and we should be defining numerical boundary conditions and not continuous ones. Incidentally, the boundary point A above is called the *near field*.

In this section we distance ourselves from domain truncation methods (see Kangro 2000, Tavella 2000) because we have not been able to find a general theory for these problems. Our reservations about current methods are threefold: first, the mathematical techniques to find the truncated boundary tend to be very complex, for example using the Laplace transform (Wong 2008), second the methods tend to be tightly coupled to specific PDEs and finally the methods tend to have little relationship to PDE theory. Instead, we transform the PDE on the semi-infinite interval to a PDE on the unit interval [0, 1]. Furthermore, it will become clear which boundary conditions to define (if at all) at the transformed near and far fields. In particular, we shall be able to apply the Fichera theory to both the near and far fields.
The idea behind domain transformation is to define a new independent space variable that is defined in the unit interval (0, 1). In this article we take the transformation:

$$y = \frac{x}{x+\alpha} \tag{8}$$

where α is a free scale factor that can be chosen by the user. We have found this to be useful for our applications and we remark that there are other ones, for example:

$$
\begin{aligned}
& y = \tanh \alpha x \\
& y = \tfrac{1}{1+\alpha x} \\
& y = e^{-\alpha x} \\
& y = 1 + e^{-\alpha x}
\end{aligned} \tag{9}
$$

These and other formulae are used in fluid dynamics and computational finance applications. For example, the second formula in (9) has been used in two-factor mortgage modelling in which the independent variables are house price and interest rate (Sharp 2006). The third transformation is suitable for the CEV (Constant Elasticity of Variance) model.

Having decided on the transformation (8) we now rewrite the PDE (1) in the new coordinates (y, t). To this end, we need to do some simple mathematical manipulation. First, the inverse transformation is given by:

$$
x = \frac{\alpha y}{1 - y} \tag{10}
$$

and furthermore:

$$
\begin{aligned}
& \tfrac{dy}{dx} = \tfrac{(1-y)^2}{\alpha}, \quad \tfrac{d^2 y}{dx^2} = -\tfrac{2}{\alpha}(1-y)^3 \\
& \tfrac{\partial u}{\partial x} = \alpha^{-1}(1-y)^2 \ \tfrac{\partial u}{\partial y} \\
& \tfrac{\partial^2 u}{\partial x^2} = \alpha^{-2}(1-y)^2 \ \tfrac{\partial}{\partial y} \left\{ (1-y)^2 \tfrac{\partial u}{\partial y} \right\}
\end{aligned} \tag{11}
$$

In general, we model convection-diffusion-reaction PDEs of the form:

$$
\frac{\partial u}{\partial t} = a(x,t)\frac{\partial^2 u}{\partial x^2} \ + \ b(x,t)\frac{\partial u}{\partial x} \ + \ c(x,t)u \ + \ f(x,t) \quad 0 < x < \infty \tag{12}
$$

When applying the bespoke transformation we can transform the PDE (12) into one involving the variable y with new (modified) drift and diffusion terms:

$$
\frac{\partial u}{\partial t} = A(y,t)\frac{\partial^2 u}{\partial y^2} \ + \ B(y,t)\frac{\partial u}{\partial y} \ + C \ (y,t)u \ + \ F(y,t) \quad 0 < y < 1 \tag{13}
$$

In computational finance the drift and diffusion terms tend to be low-order monomials in x and when transformed will be products of low-order polynomials, for example in the case when $\alpha = 1$:

$$
\begin{aligned}
& x^\beta \tfrac{\partial^2 u}{\partial x^2} = \left(\tfrac{y}{1-y}\right)^\beta (1-y)^2 \ \tfrac{\partial}{\partial y} \left\{ (1-y)^2 \ \tfrac{\partial u}{\partial y} \right\} \\
& = y^\beta (1-y)^{2-\beta} \tfrac{\partial}{\partial y} \left\{ (1-y)^2 \ \tfrac{\partial u}{\partial y} \right\} \\
& x^\beta \tfrac{\partial u}{\partial x} = \left(\tfrac{y}{1-y}\right)^\beta (1-y)^2 \ \tfrac{\partial u}{\partial y} = y^\beta \ (1-y)^{2-\beta} \ \tfrac{\partial u}{\partial y}, \text{ for } \beta \in \mathbb{R}
\end{aligned} \tag{14}
$$

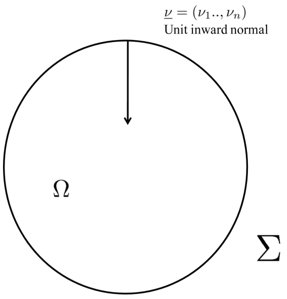

Figure 1: Region Ω and boundary Σ

Then the canonical equation:

$$x^{\beta_1}\,\frac{\partial^2 u}{\partial x^2} + x^{\beta_2}\,\frac{\partial u}{\partial x} \tag{15}$$

results in a new PDE with the modified diffusion and drift terms that we write in the *non-conservative form* as:

$$\begin{aligned} A(y,t) &= y^{\beta_1}\,(1-y)^{4-\beta_1} \\ B(y,t) &= -2y^{\beta_1}\,(1-y)^{3-\beta_1} + y^{\beta_2}\,(1-y)^{2-\beta_2} \end{aligned} \tag{16}$$

In general, the parameters β_1 and β_2 will take on specific values depending on the type of PDE that we are modelling. This will become clear in later sections.

The only question that remains is to decide how to define the boundary conditions associated with the PDE (13). We have the necessary mathematical theory that we now discuss.

2 The Fichera Theory

The Fichera function is applicable to a wide range of elliptic, parabolic and hyperbolic PDEs and it is particularly useful for PDEs whose coefficients are zero on certain boundaries of a bounded domain Ω in n-dimensional space (for more information, see Fichera 1956 and Oleinik 1973). We depict this domain, its boundary Σ and *inward* unit normal $\underline{\nu}$ in Figure 1. For the moment, let us examine the elliptic equation defined by:

$$Lu \equiv \sum_{i,j=1}^{n} a_{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} \;+\; \sum_{i=1}^{n} b_i \frac{\partial u}{\partial x_i} + cu = f \text{ in } \Omega \tag{17}$$

where

$$\begin{aligned} &\sum_{i,j=1}^{n} a_{ij}\,\xi_i\,\xi_j \geq 0 \text{ in } \Omega \cup \Sigma\; \forall\, \xi = (\xi_1, .., \xi_n) \;\in\; \mathbb{R}^n \\ &\text{where } \Sigma \text{ is the boundary of } \Omega \end{aligned} \tag{18}$$

Please note that the characteristic form is strictly positive in most cases but we are interested in when it is identically zero, and in particular in finding the subsets of the boundary Σ where it is identically zero. To this end, we partition Σ into two sub-boundaries:

$$\Sigma_3 = \left\{ x \ \in \Sigma : \sum_{i,j=1}^{n} a_{ij} \, \nu_i \nu_j > 0 \right\} \tag{19}$$

$\Sigma - \Sigma_3$, where the characteristic form is zero.

On the boundary where the characteristic form is zero (the *characteristic boundary*) we define the so-called *Fichera function*:

$$b \equiv \sum_{i=1}^{n} \left(b_i - \sum_{k=1}^{n} \frac{\partial a_{ik}}{\partial x_k} \right) \nu_i \tag{20}$$

where ν_i is the ith component of the inward normal $\underline{\nu}$ on Σ.

Having computed the Fichera function, we then determine its sign on all parts of the characteristic boundary; there are three mutually exclusive options:

$$\begin{aligned} &\Sigma_0 : b = 0 \\ &\Sigma_1 : b > 0 \\ &\Sigma_2 : b < 0 \end{aligned} \tag{21}$$

In other words, the boundary consists of the following sub-boundaries:
$\Sigma \equiv \Sigma_0 \cup \Sigma_1 \cup \Sigma_2 \cup \Sigma_3$

We demand that no boundary conditions (BC) are allowed when the Fichera function is zero or positive (in other words, Σ_0 and Σ_1) and then the PDE (17) degenerates to a lower-order PDE on these boundaries. When $b < 0$ (that is, on Σ_2) we define a boundary condition.

We pose parabolic PDEs in the form:

$$\tfrac{\partial u}{\partial t} = Lu + f \text{ or } -\tfrac{\partial u}{\partial t} + Lu = -f \tag{22}$$

where the elliptic operator L has already been defined in equation (17). Then the same conclusions concerning characteristic and non-characteristic boundaries hold as in the elliptic case. In other words, we focus on the elliptic part of the PDE to calculate the Fichera function.

Let us take an example. In this case we examine the PDE that prices a zero-coupon bond under a Cox-Ingersoll-Ross (CIR) interest-rate model:

$$\frac{\partial B}{\partial t} + \frac{1}{2}\sigma^2 r \frac{\partial^2 B}{\partial r^2} + (a - cr)\frac{\partial B}{\partial r} - rB = 0 \tag{23}$$

Please note that we are using backward time in this case, hence the sign difference when compared with equation (22). Using the definition in equation (20) we see that the Fichera function is given by:

$$\begin{array}{l} b = ((a - cr) - \sigma^2/2)\nu \text{ where } \nu \text{ is the inward unit normal at } x = 0 \ (\nu = 1) \\ \text{and at } x = 1 \ (\nu = -1) \end{array} \tag{24}$$

We are particularly interested in the case $r = 0$ (because this is the only characteristic boundary) and we see that each choice in (21) can be valid depending on the relative sizes of the parameters a and σ:

$$\begin{array}{ll} \Sigma_2 : b < 0 \rightarrow \sigma > \sqrt{2a} & \text{(BC needed)} \\ \Sigma_0 : b = 0 \rightarrow \sigma = \sqrt{2a} & \text{(No BC needed)} \\ \Sigma_1 : b > 0 \rightarrow \sigma < \sqrt{2a} & \text{(No BC needed)} \end{array} \tag{25}$$

In the last two cases we see that no boundary condition is allowed and then the PDE (23) evolves into the hyperbolic PDE:

$$\frac{\partial B}{\partial t} + a\frac{\partial B}{\partial r} = 0 \tag{26}$$

on $r = 0$. These results are consistent with the conclusions in Tavella 2000, pages 126-128). In general, we need to solve (26) either analytically or numerically. From a financial perspective, the third condition in (25) states that the interest rate cannot become negative. The inequality in this case is called the *Feller condition* for the CIR process. We have also investigated it for the process for the Heston square-root model and we have reproduced the well-known Feller condition:

$$\kappa\theta \geq \frac{1}{2}\sigma^2 \tag{27}$$

A full discussion of this topic is outside the scope of this current article. For a discussion on solutions of the Heston PDE using finite difference methods, see the thesis by Sheppard 2007 (also available from the site *www.datasimfinancial.com*).

3 Numerical Approximation of Transformed PDEs

In this section we introduce the ADE method and we apply it to several one-factor derivatives models. We assume that the PDE we wish to approximate has already been transformed from one on a semi-infinite interval to one on the unit interval. Consequently, the transformed PDE's boundary conditions are of the Dirichlet type, in other words the values of the unknown solution are given at $x = 0$ and at $x = 1$ or can easily be found at those points.

3.1 An Introduction to the Alternating Direction (ADE) Method

We motivate the ADE method by first applying it to the heat equation, which is equation (3) with $a(x, t) = 1$. To this end, we use standard notation to denote finite difference approximations to this PDE (see Duffy 2006). We assume that equation (3) is augmented by Dirichlet boundary conditions at $x = 0$ and at $x = 1$ and by an initial condition at $t = 0$. Basically, ADE is an explicit and unconditionally stable scheme of second order accuracy. To motivate ADE, let us first consider the *implicit Euler scheme* on a discrete mesh:

$$\frac{u_j^{n+1}-u_j^n}{k}=\frac{1}{h^2}\left(u_{j+1}^{n+1}-2u_j^{n+1}+u_{j-1}^{n+1}\right)\quad 1\leq j\leq J-1,\quad 0\leq n<NT \tag{28}$$

In this case the index j corresponds to a mesh point in the x variable and the index n corresponds to the mesh point in the t variable. The mesh sizes h and k are defined by $h=\frac{1}{J},\quad k=\frac{1}{NT}$, respectively where J is the number of subdivisions in the x direction and NT is the number of subdivisions in the t direction. Scheme (28) is first-order accurate in time and second-order accurate in space and the solution at time level $n+1$ is found by solving a tridiagonal matrix system. On the other hand, the *explicit Euler scheme*

$$\frac{u_j^{n+1}-u_j^n}{k}=\frac{1}{h^2}\left(u_{j+1}^{n}-2u_j^{n}+u_{j-1}^{n}\right)\quad 1\leq j\leq J,\quad 0\leq n\leq NT \tag{29}$$

is also first-order accurate and the solution at time level $n+1$ can be found without having to solve a matrix system. However, this scheme is *conditionally stable* only and this leads to the unrealistic constraint $k\leq h^2/2$ in order to keep it stable.

In order to motivate ADE, we consider equation (28) again because ADE is a variation on it. The method consists of two *sweeps* (which can be executed in parallel); the first sweep computes the solution at time level $n+1$ at indices $j-1$ and j while the second sweep computes the solution at time level $n+1$ at indices $j+1$ and j:

$$\frac{U_j^{n+1}-U_j^n}{k}=\frac{1}{h^2}\left(U_{j+1}^{n}-U_j^{n}-U_j^{n+1}+U_{j-1}^{n+1}\right)\quad 1\leq j\leq J-1,\quad n\geq 0 \tag{30}$$

$$\frac{V_j^{n}-V_j^n}{k}=\frac{1}{h^2}\left(V_{j+1}^{n+1}-V_j^{n+1}-V_j^{n}+V_{j-1}^{n}\right),\quad J-1\geq j\geq 1,\quad n\geq 0 \tag{31}$$

Finally, we average the solutions from each sweep to give the final approximate solution:

$$u_j^n=\frac{1}{2}\left(U_j^n+V_j^n\right),\quad 0\leq j\leq J,\quad n\geq 0 \tag{32}$$

If we examine these schemes carefully, we see that the solutions U and V at time level $n+1$ can be explicitly computed without having to solve a matrix system. We shall carry out a numerical analysis of the ADE method in a later section and at this stage we give the main conclusions. First, schemes (30) and (31) are *conditionally consistent* with equation (3) and have $O\left(\frac{k}{h}\right)$ local truncation error. By adding these schemes to produce the solution of equation (32) we get a stable explicit scheme with $O\left(h^2+k^2\right)$ local truncation error and hence it is second order accurate; see for example, Larkin 1964 where the two dimensional diffusion equation is discussed while the original textbook on the method is Saul'yev 1964.

The computational forms of the scheme is:

$$U_j^{n+1}\left(1+\lambda\right)=U_j^n\left(1-\lambda\right)+\lambda\left(U_{j+1}^{n}+U_{j-1}^{n+1}\right),\quad \lambda=\frac{k}{h^2} \tag{33}$$

and

$$V_j^{n+1}\left(1+\lambda\right)=V_j^n\left(1-\lambda\right)+\lambda\left(V_{j+1}^{n+1}+V_{j-1}^{n}\right) \tag{34}$$

3.2 Some Examples of ADE applied to Differential Equations

In the previous section we motivated the ADE method by examining the one-dimensional heat equation as a model case. The method can be applied to other equations such as (3) to (7). In particular, we need to show how the method is applied to the pure convection equation (4) and the convection-diffusion-reaction equation (1). In the latter case we merge - as it were - the ADE method from the convection and diffusion equations. Furthermore, many of the PDEs in computational finance are special cases of a convection-diffusion PDE and the ADE method will be applied to these cases as well. Finally, in some applications the boundary conditions are represented by PDEs; for example, for the CIR model (Cox 1985) that computes the price of a zero-coupon bond we see that the boundary condition when the interest rate is zero is a PDE of the form (4). To this end, we now discuss the application of the ADE method to equations (4). Without loss of generality we assume that the coefficient b is negative. Let us first examine the *unconditionally unstable* explicit scheme defined by:

$$\begin{aligned} &\tfrac{u_j^{n+1}-u_j^n}{k} = \tfrac{b}{2h}\left(u_{j+1}^n - u_{j-1}^n\right), \quad 1 \leq j \leq J-1, \quad n \geq 0 \\ &u_0^n \text{ given}, n \geq 0 \end{aligned} \tag{35}$$

We can now motivate the ADE method by using a slight modification of (35). In this case we still have two sweeps and we take the average of each solution as in the case of the diffusion equation:

$$\begin{aligned} \tfrac{U_j^{n+1}-U_j^n}{k} &= \tfrac{b}{2h}\left(U_{j+1}^n - U_{j-1}^{n+1}\right), \quad 1 \leq j \leq J-1 \\ \tfrac{V_j^{n+1}-V_j^n}{k} &= \tfrac{b}{2h}\left(V_{j+1}^{n+1} - V_{j-1}^n\right), \quad J-1 \geq j \geq 1 \\ u_j^n &= \tfrac{1}{2}\left(U_j^n + V_j^n\right) \end{aligned} \tag{36}$$

We can prove that that scheme is stable and non-dispersive using discrete Fourier analysis (see Duffy 2006). In general, we expect to find oscillations near points of discontinuity. But in general our interest is in convection-diffusion equations and hence this issue does not concern us in this article.

A final remark on scheme (36); it resembles a number of schemes that are used in computational fluid dynamics, in particular MacCormack's method (Anderson 1995). This latter method uses one-sided divided differences to approximate the x derivative term and the advantage is that it does not need to create numerical downstream boundary conditions. In our case we have already transformed the PDE to one on the unit interval resulting in the fact that we essentially have a time-dependent two-point boundary value problem with Dirichlet boundary conditions. The technique is to define a new independent variable y as in section 1.4 and we use it instead of the original variable x in equation (4), resulting in the transformed PDE:

$$\begin{aligned} &\tfrac{\partial u}{\partial t} = H(y)\tfrac{\partial u}{\partial y}, \; 0 \leq y \leq 1, \quad t > 0 \\ &\text{where } y = \tfrac{x}{x+1} \text{ and } H(y) = b(1-y)^2 \end{aligned} \tag{37}$$

We can solve this problem by using the Crank-Nicolson method in time and centred difference in space or by the ADE method (36). In each case we have an unconditionally stable second-order scheme and the boundary condition at $y = 1$ (which corresponds to infinite x) follows from (37) because:

$$\tfrac{\partial u}{\partial t} = 0, \quad \text{where } y = 1 \tag{38}$$

We can easily integrate this equation to compute the solution at $y = 1$. Now, traditional finite difference schemes (that is, those that are defined on a semi-infinite interval) that use three-point centred difference schemes need to take special precautions at the *downstream boundary* because we approximate the derivative in x at that point. To show what we mean, let us take the untransformed equation (4) again. For the sake of this example, it is sufficient to discretise in space to give the *semi-discrete scheme*:

$$\begin{aligned} &\tfrac{du_j}{dt} = \tfrac{b}{2h}\left(u_{j+1} - u_{j-1}\right), \quad 1 \leq j \leq J-1 \\ &u_0(t) \text{ given} \end{aligned} \tag{39}$$

This is a three point scheme and the essential difficulty appears at $j = J$. We need to find the value of the unknown solution at this point but a centered difference scheme is not suitable because we will need to incorporate mesh points outside the domain of integration. To this end, we can model the PDE by a two-point *upwind* approximation:

$$\frac{du_J}{dt} = \frac{b}{h}\left(u_J - u_{J-1}\right) \tag{40}$$

One of the effects of this additional (but necessary) approximation is that it leads to *spurious oscillations*, in other words non-physical reflections that affect the accuracy of the solution at interior mesh points. Furthermore, the truncation error of the difference formula (40) at the boundary is only first order accurate in h. A better but more complex approximation is the three-point formula:

$$\frac{du_J}{dt} = \frac{b}{2h}\left(3u_J - 4u_{J-1} + u_{J-2}\right) \tag{41}$$

which has a second order truncation error in h (Vichnevetsky 1982). Finally, the amplitude of the reflected solution of (41) converges to zero as $O\left(h^3\right)$ whereas it is $O\left(h^2\right)$ for the solution (40).

First-order PDEs of the form (4) appear in a number of applications in computational finance, for example Asian option pricing (Duffy 2006), interest rate models (Kohl-Landgraf 2007) and most schemes are based on equations (39) and its generalizations. In general, many authors and practitioners have spent many hours getting these schemes to work properly and the causes are based on less-than optimal solutions for the problem at hand, unfamiliarity with first-order hyperbolic PDEs and possibly, an implicit assumption that ADI and Crank Nicolson methods are suitable for all problems in computational finance. We avoid these problems if we use the ADE method (for example) in combination with domain transformation. An open question in this regard is to prove the stability of ADE schemes and of numerical boundary conditions for equation (37) using the discrete Laplace transform (also known as the *z-transform*). For more details, see Thomas 1999.

3.3 ADE for Equity Pricing Problems

The theory of finite difference methods for the one-factor Black Scholes PDE is well developed. One of the most popular choices is to discretise the space (underlying stock S) using centred differences and the Crank Nicolson (time averaging) for the time derivative (Tavella 2000). The discrete equations leads to a tridiagonal matrix system that is solved at each time level up to the expiry time. However, we avoid the Crank Nicolson scheme in general because it can produce spurious oscillations in the computed option price and its sensitivities. This is caused by discontinuous payoff functions (which are the same as the initial condition in the corresponding initial boundary value problem). For a discussion of this problem and how to resolve it, see Duffy 2004A. A better solution is to apply the implicit Euler method with Richardson extrapolation, thus avoiding these spurious oscillations and achieving second-order accuracy in time (Lawson 1978). An application of this method to the two-factor Heston model using splitting methods can be found in Sheppard 2007. In his thesis Sheppard gives an example of how Crank Nicolson fails to compute an accurate approximation to the option price and he shows that the Lawson extrapolation technique gives good results.

The untransformed Black Scholes PDE for a call or put option is given by:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} + rS\frac{\partial u}{\partial S} - ru \tag{42}$$

After applying the transformation (8) to PDE (42) we get a PDE of the form (13) (with $\alpha = 1$) with

$$\begin{aligned} &A(y,t) = \tfrac{1}{2}\sigma^2 y^2 \left(1-y\right)^2 \\ &B\left(y,t\right) = ry\left(1-y\right) - \sigma^2 y^2 \left(1-y\right), \quad y = \tfrac{S}{S+1} \\ &C(y,t) = -r, F(y,t) = 0 \end{aligned} \tag{43}$$

We see that the diffusion term is zero on the boundaries $y = 0$ and $y = 1$ and hence we can apply the Fichera theory. In this case the Fichera function is given by:

$$b_F \equiv \left(ry\left(1-y\right) - \sigma^2 y\left(1-y\right)^2\right)\nu \tag{44}$$

when ν is the inward normal direction on the boundary. The term b_F is zero on the boundaries. Here no boundary conditions are needed (or allowed) and hence the PDE degenerates to an ordinary differential equation (ODE) that can easily be integrated to produce a solution (which becomes in essence a Dirichlet boundary condition):

$$\tfrac{\partial u}{\partial t} + ru = 0 \text{ when } y = 0, \quad y = 1 \tag{45}$$

Referring to Figure 2, the initial conditions for (45) at $y = 0$ and $y = 1$ are determined at the points A and B, respectively and are in fact the values of the payoff function at those points. The ADE method for the transformed Black Scholes PDE is based on the schemes (30), (31) and (32). We give the schemes in the current context for completeness. The 'upward' scheme is given by:

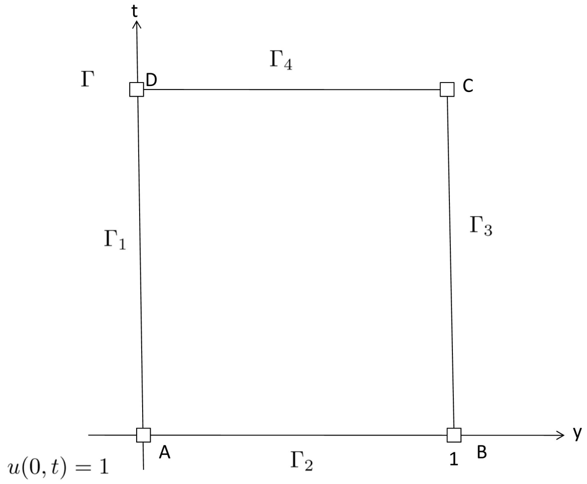

Figure 2: Region of integration

$$U_j^{n+1}\left(1+\alpha_j+rk\right)=U_j^n\left(1-\alpha_j\right)\;+\;U_{j-1}^{n+1}\left(\alpha_j-\beta_j\right)\;+\;U_{j+1}^n\left(\alpha_j+\beta_j\right),\quad 1\le j\le J-1 \tag{46}$$

And the 'downward' scheme is given by:

$$V_j^{n+1}\left(1+\alpha_j+rk\right)=V_j^n\left(1-\alpha_j\right)\;+\;V_{j-1}^{n+1}\left(\alpha_j-\beta_j\right)\;+\;U_{j+1}^{n+1}\left(\alpha_j+\beta_j\right),\quad 1\le j\le J-1 \tag{47}$$

where

$$\begin{array}{l}\alpha_j=\frac{kA(y_j,t)}{h^2}\quad 1\le j\le J-1,\quad \beta_j=\frac{kB(y_j,t)}{h^2}\\ \text{and } \{y_j\}_{j=0}^J \text{ is the set of mesh points in the } y \text{ direction.}\end{array} \tag{48}$$

The schemes (46) and (47) can be solved in parallel since there are no data dependencies between them and their solutions satisfy Dirichlet boundary conditions. Finally, we average these two solutions to produce the second-order approximation to the option price, that is:

$$u_j^n=\frac{1}{2}\left(U_j^n+V_j^n\right),\quad 0\le j\le J,\quad n\ge 0 \tag{49}$$

Summarising, we have now completed our discussion of the ADE method for pricing one-factor European call and put options. Some of the advantages are:

1) The scheme is explicit and unconditionally stable for any values of the mesh sizes h and k. It performs well because no matrix system needs to be solved at each time level.

2) The scheme can be parallelized in C++ using OpenMP (Chapman 2008, Chandra 2001) or some other threading library.

3) It can be used with the exponentially fitted difference schemes (Duffy 1980, Duffy 2006) for problems with small diffusion or large convection terms.

4) It can be used to compute option sensitivities without causing spurious oscillations as we experienced with the Crank Nicolson method.

5) It can be used with non-smooth payoff functions.

6) It is generalisable to n-factor PDE and schemes; this will be the topic of future articles.

7) The ADE method is applicable to a wider range of problems than traditional implicit schemes for nonlinear and high-order problems such as Hamilton-Jacobi, diffusion and lubrication and TV denoising (Leung 2005).

3.4 Numerical Analysis of the ADE Method

In this section we give an analysis of the ADE method. To this end, let us discretise equation (1) in the x direction only. This is called the semi-discretisation process or the *Method of Lines* (MOL) (Duffy 2006). This process leads to a system of ordinary differential equation for the dependent vector T (use variable name T for this):

$$\frac{\partial T}{\partial t} = AT \equiv (L + D + U)\,T, \quad t \geq 0 \tag{50}$$

where A is the matrix that originates from semi-discretisation that we decompose into lower triangular L, diagonal D and upper triangular U sub-matrices. In the analysis we assume that the boundary values of T are given and in fact equation (50) is a discretisation of a PDE with Dirichlet boundary conditions, essentially.

The next step is to discretise equation (50) in time. This is a well-known process for traditional finite difference methods but the main difference in the present case is that we construct two schemes by sweeping from the corners of the space domain. In our case we have two sweeps:

$$\frac{T_1^{n+1} - T_1^n}{k} = LT_1^{n+1} + \frac{D}{2}T_1^{n+1} + \frac{D}{2}T_1^n + UT_1^n \equiv BT_1^{n+1} + CT_1^n, \quad n \geq 0 \tag{51}$$

and

$$\frac{T_2^{n+1} - T_2^n}{k} = LT_2^n + \frac{D}{2}T_2^{n+1} + \frac{D}{2}T_2^n + UT_2^{n+1} \equiv BT_2^n + CT_2^{n+1} \quad n \geq 0 \tag{52}$$

Finally, the approximate solution is the average of the solutions of equations (51) and (52):

$$T^n = \frac{1}{2}\left(T_1^n + T_2^n\right), \quad n \geq 0 \tag{53}$$

These systems are easier to solve than traditional implicit schemes because the value of T at time level $n+1$ can be computed if we rewrite the systems as:

$$
\begin{aligned}
&(I-kB)\,T_1^{n+1} = (I+kC)T_1^n \\
&(I-kC)T_2^{n+1} = (I+kD)T_2^n \\
&\text{where } B = L + D/2 \text{ and } C = U + D/2
\end{aligned}
\tag{54}
$$

This system is easy to solve because we are working with lower triangular and upper triangular matrices whose inverses can be found by using simple forward and backward substitution. Here we see immediately how generally applicable ADE is. Using (54) we can rewrite the solution (53) in the form:

$$
T^{n+1} = \frac{1}{2}[(I-kB)^{-1}(I+kC) + (I-kC)^{-1}(I+kB)]T^n, \quad n \geq 0 \tag{55}
$$

We report on some results concerning the scheme (55) (Leung 2005). Other approaches can be found in Sauly'ev 1964, Barakat 1966 and Larkin 1964.

Theorem 1: The ADE scheme (55) is second order accurate in time if all the diagonal elements of the matrix A in equation (50) are non-positive.

A useful exercise is to compute the individual *local truncation errors* for the schemes (51) and (52) and we see that they typically are of the order $O(k^2 + h^2 + \left(\frac{k}{h}\right)^2)$ while the averaging step (53) ensures that some terms cancel each other out and we are finally left with a truncation error $O(k^2 + h^2)$.

It is instructional to actually carry out this process for a number of specific cases (for example, the convection-diffusion equation) to convince yourself that the 'offending' terms do actually cancel each other out.

4 Numerical Results

Since ADE is a relatively new method and approach to solving PDEs in computational finance we compare our results with some other methods that we have used in the past. To be precise, we compare the method with the exact solution (Haug 2007), the implicit Euler method (Duffy 2006, Duffy 2004) and the Monte Carlo method (Duffy 2009). In general, it is possible to achieve any order of accuracy (barring round-off errors) if we make the time steps small enough. But there are also timing constraints. For example, we compare the results of the ADE method with the Monte Carlo method with as many as a million draws or simulations and it is obvious that the response time of the latter method is of the order of seconds and not of milliseconds.

We have tested the method for 'benign' values of the Black Scholes parameters as well as using more extreme values. We have used the non-conservative form of the transformed PDE (13) with coefficients defined in (43). The examples we take are:

- I: ADE for a European put option for a range of values of the mesh sizes k and h.
- II: Stress testing and extreme parameter values (for example, zero interest rate and large volatility).
- III: Convection-dominated problems (small diffusion and/or large drift) and exponentially fitted schemes.

- IV: Constant Elasticity of Variance (CEV) model with positive and negative elasticity factor.

- V: Option sensitivities (the Greeks).

- VI: Early exercise feature (American options).

Example I: in this case we take an initial example to get a feeling for how good the accuracy of ADE is. We price a European put option with $K = 65$, $S = 60$, $T = 0.25$, $r = 0.08$, $\sigma = 0.30$. The exact put price is $P = 5.84584$ ($C = 2.13293$). Table 1 gives the approximate values using ADE for a range of mesh sizes. Even with conservative values of NY and NT we still attain reasonable accuracy. Of course, increasing the number of steps improves the accuracy. Finally, taking $NY = 1000$ and $NT = 2000$ gives $P = 5.84681$.

NY/NT	100	200	500	1000
50	5.750994	5.746798	5.744265	5.743418
200	5.847411	5.843841	5.84152	5.840716
500	5.848931	5.848239	5.846728	5.846041

Table 1: Accuracy of ADE, put option

In Table 2 we show the results for the call option price with the same parameters as above. A point to take into consideration is that the payoff function for a call option is monotonically increasing as a function of S which means that the function given by the transformation (8) has a singularity at $y = 1$. We had to modify the payoff function somewhat in order to avoid overflow. On the other hand, another approach is to price a put option and then use put-call parity to compute the call price.

NY/NT	100	200	500
100	2.129	2.118	2.112449
200	2.147244	2.13737	2.131265
500	2.148627	2.141648	2.136358

Table 2: Accuracy of ADE, call option

With NY = 500 and NT = 1000 the call price becomes C = 2.136517. A modification of the code at the boundary $y = 1$ results in a better approximation. In this case we define the payoff function in the closed-open interval [0, 1). Due to the fact that the boundary condition at $y = 1$ is the discounting equation (an ordinary differential equation (ODE)) we can solve it at $t = 0$ by defining the initial value of this ODE as the value of the payoff function, not at $y = 1$ but at some point very close to it (to the left). This now results in good approximate values for both calls and puts. The results improve on the values in Table 2.

Example II: In this case we take $r = 0$ (zero interest rate), $K = S = 100$ (ATM) and a range of values for σ and T. In the case $T = 1$, we take NY = 200 and NT = 300 while in the case $T = 5$ (5 years) we increase the number of mesh points to around NY = [300,500], NT = [500,1000] to see how accuracy improves. We compare the exact and Monte Carlo solutions (using the code from Duffy 2009) on the one hand with the implicit Euler method (using the codes in Duffy 2006) and the ADE method on the other hand. In one sense we are comparing implicit Euler with ADE because the former method can be seen as the vanguard of traditional finite difference schemes and ADE is the young pretender as it were.

	T = 1		T = 5	
	$\sigma = 0.2$	$\sigma = 1.0$	$\sigma = 0.2$	$\sigma = 1.0$
Exact P	7.96632	38.2893	17.6933	73.6463
IEuler	7.965621	38.27265	17.69072	73.63129
Monte Carlo	7.97947	38.322	17.7204	73.5974
ADE	7.981916	38.27344	17.69147	73.70187

Table 3: Stress-testing ADE

Example III: Traditional finite difference schemes shows signs of spatial instability (see Roache 1998) when the drift (also known as the convection term) dominates the diffusion term. In order to resolve this problem we have developed *exponentially fitted schemes* (Duffy 1980) and we subsequently applied them to option pricing problems in Duffy 2004 and Duffy 2006. We take an example in which we compare the exact solution of a convection-dominated problem with the ADE solution with and without fitting. The values are $r = 0.20$, $\sigma = 0.1$, $S = K = 100$, $T = 0.5$.

	put (.238825)		call (9.75508)	
	No Fit	Fit	No Fit	Fit
200 × 200	.21181	.236347	9.814975	9.808313
500 × 500	.235024	.238892	9.773767	9.7748
1000 × 1000	.23809	.239055	9.772502	9.765205

Table 4: Convection-dominated problems

Example IV: This is the test of the CEV model:

$$-\frac{\partial u}{\partial t} + \frac{1}{2}\sigma^2 S^{2\beta}\frac{\partial^2 u}{\partial S^2} + r\frac{\partial u}{\partial S} - ru = 0 \tag{56}$$

where β is the so-called *elasticity factor.* Of course, we transform (56) using the transformation (8). We tested the ADE method using a range of values for the elasticity factor, some of which are shown in Table 5. The other parameters where $r = 0.05$, $T = 0.5$, $K = 110$, $S = 100$, $\sigma = 0.2$ $(NY = 500, NT = 500)$

β	Exact	ADE
0	9.95517	9.955171
-1	9.73787	9.737663
-2	9.53622	9.535646
2/3	10.1099	10.11005

Table 5: Testing the CEV Model, (put option)

Example V: We approximate option sensitivities and we can report that the values obtained are as accurate as those from the implicit Euler scheme. Some results are presented in Table 6 based on $r = 0.05$, $T = 0.5$, $S = K = 100$, $\sigma = 0.2$. The cases ADEI and ADEII refers to mesh sizes

NY = 200, NT = 200 and NY = 400, NT = 400, respectively. The case ADEIII corresponds to NY = 1000, NT = 1000.

	Exact	ADE I	ADE II	ADE III
P	4.418995	4.414167	4.424243	4.420607
δ	-0.40227	-0.43423	-0.41796	-0.40857
γ	0.027359	0.029524	0.028439	0.027812

Table 6: Calculating Sensitivities

Example VI: As last set of numerical examples, we examine the ability of the ADE method to price put options with early exercise feature (American options). We have compared the method with a number of other numerical methods from various sources. The basic algorithm for American options is based on that for European options. Having computed the two 'sweep' solutions we then test if their values are greater than the *intrinsic value* and the modified value will be the larger of itself and this intrinsic value.

As first example, we take a put option with $K = 10$, $S = 9$, $NY = 400, NT = 400$. Tables 7 and 8 show comparisons with the binomial method in the case when the number of steps $M = 256$. The results are in good agreement with the binomial method.

T	Binomial (M = 256)	ADE
0.25	1.1260	1.126501
0.50	1.2553	1.2555837
0.75	1.3541	1.354917
1.0	1.4354	1.434963

Table 7: Early exercise, $\sigma = 0.3$, $r = 0.06$

T	Binomial (M = 256)	ADE
0.25	1.3805	1.381034
0.50	1.6178	1.616883
1.0	1.9094	1.904257

Table 8: Early exercise, $\sigma = 0.5$, $r = 0.12$

The next example is based on comparisons with a randomly chosen online American option calculator. In Table 9 we have used $NY = 200$, $T = 400$ while if we set $NY = 500$, $NT = 1000$ our values improve with respect to those from the calculator as shown in Table 10. The parameters are $K = 100$, $r = 0.1$ and $T = 0.1$.

Online Calculator				ADE		
S	90	100	110	90	100	110
$\sigma = 0.15$	10.0	1.5116	0.0232	10.0	1.474916	0.027138
$\sigma = 0.35$	10.4748	3.9697	1.0823	10.4609	3.980881	1.073052

Table 9: Comparison with online calculator

ADE			
S	90	100	110
$\sigma = 0.15$	10.0	1.512486	0.023854
$\sigma = 0.35$	10.46223	3.99212	1.069732

Table 10: ADE improving accuracy

Finally, we compare our results with implicit finite difference schemes (see Neu-Ner 2005) for the CEV model in the case of the square-root process. For this example, we take $S = 100$, $\beta = 0.5$, $r = 0.1$, $\sigma = .20$. We present the results for various values of the strike price and expiry time in Table 11. In the table, the 'upper' values are from Neu-Ner 2005 while the 'lower' values come from ADE. We witness good agreement between the two approaches ($NY = 200$, $NT = 400$).

K/T	1/2	1
90	1.06 1.069352	1.81 1.824111
100	3.89 3.893676	4.76 4.765286
110	10.20 10.20316	10.55 10.55242

Table 11: Comparison of FDM methods

Table 12 shows put option prices based on the implicit Euler method (which, by the way is robust and accurate) with $NT = 200$, $NY = 400$, $S = 100$, $r = 0.1$, $\sigma = 0.2, \beta = 0.5$ for a range of values of the strike price and expiry time. The processing time for non-parallel code is similar to that of the ADE method but the implicit Euler method is more difficult to program and it is not as easy to parallelise it as ADE.

K/T	1/2	1
90	1.05874	1.813477
100	3.884078	4.755361
110	10.19907	10.54583

Table 12: Implicit Euler, early exercise

Concluding, the ADE is a fast and accurate numerical method for pricing American options. It is easier to model and program than penalty or front-fixing methods (see Nielsen 2002) and it is much faster. The computing time (depending on what accuracy we wish to attain) is between 50 and 130 microseconds on a 1.8 GHz commodity laptop computer using C++ and OpenMP.

5 Coding and Computational Issues

We now give a discussion of how we designed and implemented the software system. We have developed the algorithms in both C++ and C#. We concentrate on the C++ implementation

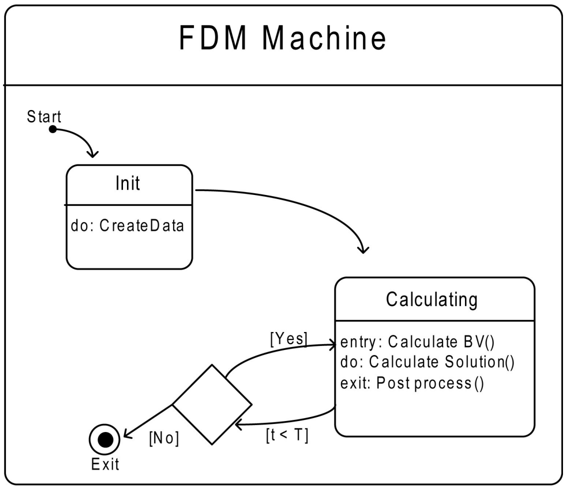

Figure 3: UML State Machine of FDM Algorithm

in this section and the design is described in Duffy 2006. Our main interest here is to show how we designed and implemented the ADE method as a C++ class in the software framework as described in Duffy 2006 and which is documented using UML (Unified Modeling Language). We describe the algorithm at a high level by using the UML statechart diagram in Figure 3. We have two main states, the first of which is responsible for creating the arrays and parameters that are needed by the algorithm and the second state called '*Calculating*' that contains the code to actually calculate the solution at time level $n + 1$ based on the known information at time level n. This process continues until we reach the expiry time, in which case we are finished and the program exits, after having saved the option price data in Excel, for example. The main code blocks in '*Calculating*' have to do with:

- *entry to state*: update the boundary conditions at level $n + 1$.
- *do while in state*: perform the two sweeps of the ADE method.
- *exit from state*: update current values and increment the time step.

These three steps must be calculated in sequence but it is possible to parallelise the ADE legs and the boundary update procedures, as we shall see later in this section. The approach taken was to create the code for the ADE method, integrate it into our software framework and test the results with known solutions as described in section 4. Having done that, we now proceed to modify the code so that it makes use of a multithreading (MT) library, in this case OpenMP (Chapman 2008, Chandra 2001). The two major issues here are:

- Question 1: do the single-threaded and multi-threaded solutions give the same result?
- Question 2: can we improve the speedup of the MT solution?

By speedup, we mean the processing time T(p) on a CPU with p processors compared with the processing time T(1) on a CPU with a single processor. We have tested the ADE method on a standard duo-core laptop. To this end, the code that can be parallelized corresponds to the two sweeps of the scheme and we employ OpenMP's *parallel sections* to realise this. A parallel section construct allows different threads to carry out different kinds of work, in this case the ADE sweeps. The code is given by:

```
void ADE::calculate()
{ // Tells how to calculate sol. at n+1, Explicit ADE schemes

     omp_set_num_threads(2); // Number of threads = 2


     #pragma omp parallel for
     for (long j = U.MinIndex(); j <= U.MaxIndex(); ++j)
     { // Coefficients calculated in parallel

        alpha[j] = k* fitting_factor(xarr[j], tnow)/ h2;
        beta[j] = (0.5* k * (ibvp->convection(xarr[j], tnow))) / h;
        gamma[j]=1.0/(1.0+alpha[j]-ibvp->zeroterm(xarr[j],tnow)* k );
     }

     #pragma omp parallel sections
     {
        #pragma omp section
        {
          // Upward sweep
          for (long j=U.MinIndex()+1; j<= U.MaxIndex()-1; ++j)
          {
               U[j] = (alpha[j] - beta[j])*U[j-1]
                   + (1.0 - alpha[j])*UOld[j]
                   + (alpha[j] + beta[j])*UOld[j+1];
               U[j] *= gamma[j];
          }
     }

     #pragma omp section
     {
        // Downward sweep
        for (long j=V.MaxIndex()-1; j >= V.MinIndex()+1; --j)
        {

          V[j] = (alpha[j] - beta[j])*VOld[j-1]
                  + (1.0 - alpha[j])*VOld[j]
                  + (alpha[j] + beta[j])*V[j+1];
          V[j] *= gamma[j];
        }
     }
} // End of parallel sections


     #pragma omp parallel for
     for (long j = vecNew.MinIndex(); j <= vecNew.MaxIndex(); ++j)
     {
        vecNew[j] = 0.5 * (U[j] + V[j]);

     }
```

```
    UOld = vecNew;
    VOld = vecNew;
}
```

We have tested this code on a number of cases and we have seen that it is on average about 1.6 times faster on a duo-core machine than on a single core machine. In fact, sometimes the problem is not big or demanding enough to warrant a parallel solution. For example, using loop-level parallelism may not be efficient if the *loop's iteration count* is below a certain critical value because of overhead costs in creating, managing and deleting threads.

6 Appendix: Well-posedness of the Transformed PDE Problem

One of the major techniques in this article was to transform a time-dependent PDE on a semi-infinite interval to one that is defined on the unit interval. The original PDE may or may not have associated boundary conditions as we have seen in section 2 where we discussed the Fichera theory. In this appendix we wish to show that the solution of the transformed PDE depends continuously on its input, namely the boundary conditions (when they are defined), initial condition and inhomogeneous term. To this end, we employ well-known techniques to allow us to estimate the norm of the solution of the corresponding initial value problem (see Kreiss 2004 for a discussion of this principle). As a first equation, we consider the first-order initial boundary value problem:

$$\begin{aligned}
&\text{(a) } \tfrac{\partial u}{\partial t} = H(y)\tfrac{\partial u}{\partial y}, \quad 0 < y \leq 1, \quad t > 0, \quad H(y) \equiv b(1-y)^2 \\
&\text{(b) } u(y,0) = f(y), \quad 0 \leq y \leq 1 \\
&\text{(c) } u(0,t) = g(t), \quad t > 0
\end{aligned} \tag{57}$$

We wish to find an integral estimate for the solution of (57) in terms of the functions $f(y)$ and $g(t)$. We summarize the main steps - which are valid for a wide range of problems - and we then show how these steps are realized:

a) Multiply both sides of (6.1)(a) by u; call this equation B.

b) Integrate equation B between 0 and 1 in space and use integration by parts to produce equation C; in this case some terms on the boundary are zero and other terms relating to boundary conditions may appear in the equation.

c) Integrate equation C in time between 0 and t to produce an integral equality and inequality called D.

d) Use Gronwall's lemma on inequality D to produce the a priori estimate for the solution of the initial boundary value problem (57).

The precise steps are:

$$\begin{aligned}
&\tfrac{\partial u}{\partial t}(y,t).u(y,t) = H(y)\tfrac{\partial u}{\partial y}(y,t).u(y,t)\\
&\tfrac{1}{2}\tfrac{\partial}{\partial t}u^2(y,t) = \tfrac{1}{2}H(y)\tfrac{\partial}{\partial y}u^2(y,t)\\
&\tfrac{1}{2}\textstyle\int_0^1 \tfrac{\partial}{\partial t}u^2(y,t)dy = \tfrac{1}{2}\textstyle\int_0^1 H(y)\tfrac{\partial}{\partial y}u^2(y,t)dy\\
&\text{Then}\\
&\tfrac{d}{dt}\|u\|^2(t) = \textstyle\int_0^1 H(y)\tfrac{\partial}{\partial y}u^2(y,t)dy\\
&= [H(y))u^2(y,t)]_{y=0}^{y=1} \ - \ \textstyle\int_0^1 \tfrac{\partial H(y)}{\partial y}u^2(y,t)dy\\
&\leq -bg^2(t) \ + \ M\|u\|^2(t), \text{ where } \|u\|^2(t) \equiv \textstyle\int_0^1 u^2(y,t)dy
\end{aligned} \tag{58}$$

Integrating in time now leads to the inequality:

$$\|u\|^2(\xi) \leq \|u\|^2(0) + \int_0^\xi |g(t)|^2 dt + M\int_0^\xi \|u\|^2(t)dt, \quad 0 < \xi \leq T \tag{59}$$

Finally, we use Gronwall's inequality to give the desired estimate for the solution of (57):

$$\begin{aligned}
&\|u\|^2(\xi) \leq (\|u\|^2(0) + \textstyle\int_0^\xi |g(t)|^2 dt)\exp(M\xi) \text{ for } 0 < \xi \leq T\\
&\text{or } \|u\|^2(\xi) \leq (\|f\|^2 + \textstyle\int_0^\xi |g(t)|^2 dt)\exp(M\xi), \text{ where } M > 0 \text{ is a constant}
\end{aligned}$$

We now discuss a priori estimates in the presence of diffusion and combined convection-diffusion terms. To this end, we examine the transformed PDE for the one-factor equity option problem again:

$$\begin{aligned}
&\tfrac{\partial u}{\partial t} = \tfrac{1}{2}\sigma^2 y^2 \tfrac{\partial}{\partial y}\left\{(1-y)^2\tfrac{\partial u}{\partial y}\right\} + ry(1-y)\tfrac{\partial u}{\partial y} - ru\\
&= A(y)\tfrac{\partial^2 u}{\partial y^2} + B(y)\tfrac{\partial u}{\partial y} - ru\\
&\text{where}\\
&A(y) = \tfrac{1}{2}\sigma^2 y^2(1-y)^2\\
&B(y) = ry(1-y) - \sigma^2 y^2(1-y)
\end{aligned} \tag{60}$$

We can apply the same process to PDE (60) that we used for the PDE system (57). We summarise the main steps in equations (61) and (62); we then integrate in time and use Gronwall's lemma again:

$$\begin{aligned}
&\int_0^1 y^2 \frac{\partial}{\partial y}\left\{(1-y)^2\frac{\partial u}{\partial y}\right\} u dy = \int_0^1 y^2 u \frac{\partial}{\partial y}\left\{(1-y)^2\frac{\partial u}{\partial y}\right\} dy \\
&= [y^2 u(1-y)^2 \tfrac{\partial u}{\partial y}]_0^1 \ - \ \textstyle\int_0^1 \tfrac{\partial}{\partial y}(y^2 u)(1-y)^2 \tfrac{\partial u}{\partial y} dy \\
&= -\textstyle\int_0^1 2y(1-y)^2 u \tfrac{\partial u}{\partial y} \ - \ \int_0^1 y^2(1-y)^2\left(\tfrac{\partial u}{\partial y}\right)^2 dy \\
&\leq -\textstyle\int_0^1 2y(1-y)^2 \tfrac{1}{2}\tfrac{\partial u^2}{\partial y} dy \\
&= -[y(1-y)^2.u^2]_0^1 \ + \ \textstyle\int_0^1 \tfrac{\partial}{\partial y}(y(1-y)^2)u^2 dy \leq M \int_0^1 u^2 dy
\end{aligned} \tag{61}$$

$$\int_0^1 ry(1-y)u\frac{\partial u}{\partial y}dy = [\frac{1}{2}ry(1-y)u^2]_0^1 \ - \ \frac{1}{2}\int_0^1 r\frac{\partial}{\partial y}(y(1-y))u^2 dy \leq M_2 \int_0^1 u^2 dy \tag{62}$$

We then get an estimate similar to inequality (59).
As an exercise, it is possible to prove the well-posedness of the CIR PDE:

$$\frac{\partial B}{\partial t} = \frac{1}{2}\sigma^2 r \frac{\partial^2 B}{\partial r^2} + (a - br)\frac{\partial B}{\partial r} - rB = 0 \tag{63}$$

with payoff (initial condition) function:

$$B(r, 0) = 1 \tag{64}$$

If we go through the steps again, we will be able to prove well-posedness assuming the Feller condition (27).

7 Discussions, Extensions and Conclusions

We have introduced a number of methods and techniques to create unconditionally stable, explicit second-order finite difference schemes for the solution of one-factor PDEs that describe put and call equity options. We introduce the technique for mapping a PDE on a semi-infinite interval to one on the unit interval and we showed how to use the Fichera theory to help determine where to define boundary conditions for the corresponding initial boundary value problem. We then applied the Alternating Direction Explicit (ADE) method to solving this problem and we took five scenarios to test the accuracy and robustness of the method.

The techniques in this article can be applied to a wide range of one-factor problems, including standard European options, barrier options with both fixed and time-dependent barriers. In follow-up articles we shall discuss the application of this method to interest-rate, stochastic volatility and n-factor models.

8 Bibliography

Anderson, J. D. 1995 *Computational Fluid Dynamics* McGraw-Hill 1995

Barakat, H.Z. and Clark, J.A. 1966 *On the solution of diffusion equation by numerical methods* J. Heat Transfer 88:421-427

Chapman, B., Jost, G. and van der Pas 2008 *Using OpenMP* MIT Press Cambridge

Chandra, R., Dagnum, L., Kohr, D., Maydan, D., McDonald, J. and Menon, R. 2001 *Parallel Programming in OpenMP* Morgan Kaufmann San Francisco

Cox, J.C., Ingersoll, J.E. and Ross, S.A. 1985 *An Intertemporal General Equilibrium Model of Asset Prices, Econometrica* 53, pp. 363-384

Duffy, D. J. 1980 *Uniformly Convergent Difference Schemes for Problems with a Small Parameter in the Leading Derivative* PhD thesis Trinity College Dublin

Duffy, D. J. 2004 *Financial Instrument Pricing using C++* John Wiley and Sons Chichester UK

Duffy, D. J. 2004A *A critique of the Crank-Nicolson scheme, strengths and weaknesses for financial engineering* Wilmott Magazine July 2004

Duffy, D. J. 2006 *Finite difference methods in financial engineering* John Wiley and Sons Chichester UK

Duffy, D.J. and Kienitz, J. 2009 *Monte Carlo Frameworks, Building Customisable High Performance C++ Applications* John Wiley and Sons Chichester

Fichera, G. 1956 *Sullle equazioni differenziali lineari ellittico-paraboliche del secondo ordine*, Atti Accad. Naz. Lincei. Mem. CI. Sci. Fis. Mat. Nat. Sez. I (8), 5 (1956), 1-30. MR 19, 658; 1432

Fichera, G. 1960 *On a Unified Theory of Boundary Value Problems for Elliptic-Parabolic Equations of Second Order in Boundary Value Problems* (ed. R.E. Langer) University of Wisconsin Press

Haug, E. 2007 *The Complete Guide to Option Pricing Formulas* McGraw-Hill.

I'lin, A.M, Kalashnikov, A.S. and Oleinik, O.A. 1962 *Linear Equations of the Second Order of Parabolic Type*, (translation) Russian Mathematical Surveys

Kangro, R. and Nicolaides, R. 2000 *Far Field Boundary Conditions for Black Scholes Equations* SIAM J. Numer. Anal. Vol. 38, No. 4, 1357-1368

Kau, J.B., Keenan, D.C., Muller, W.J. and Epperson, J.F. 1993 *Option Theory and Floating-Rate Securities with a Comparison of Adjustable- and Fixed-Rate Mortgages.* Journal of Business. 66 595-618.

Kohl-Landgraf, P. 2007 PDE *Valuation of Interest Rate Derivatives* Books on Demand Gmbh Nordstedt

Kreiss, Heinz-Otto, Lorenz, J. 2004 *Initial-Boundary Value Problems and the Navier-Stokes* An

Ladyenskaja, O. A., Solonnikov, V. A. and Ural'ceva, N. N. 1988 *Linear and Quasi-linear Equations of Parabolic Type* American Mathematical Society

Laney, C. B. 1998 *Computational Gasdynamics* Cambridge University Press

Larkin, B.M. 1964 *Some stable explicit difference approximations to the diffusion equation* Math. Comp. 18, 196-202.

Lawson, J.D. and Morris, J.L. 1978 *The extrapolation of first order methods for parabolic partial differential equations* SIAM J. Numer. Anal. 15, 6 December

Leung, S. and Osher, S. 2005 *Alternating Direction Explicit (ADE) Scheme for Time-Dependent Evolution Equations* Preprint UCLA June 9, 2005

Nielsen, B.F., Skavhaug, O. and Tveito, A. 2002 *Penalty and front-fixing methods for the numerical solution of American option problems* Journal of Computational Finance Volume 5/Number4, Summer

Neu-Ner, T. 2005 *An Effective Binomial Tree Algorithm for the CEV Model*, Thesis University of the Witwatersrand, Johannesburg, South Africa

Oleinik, O.A. and Radkevic, E.V. 1973 *Second order Equations with nonnegative characteristic Form* translated from Russian by Paul C. Fife, American Mathematical Society

Roache, P. J. 1998 *Fundamentals of Fluid Dynamics* Hermosa Publishers Albuquerque

Samarski, A.A. 1964 *An Economical Algorithm for the Numerical Solution of Systems of Differential and Algebraic Equations* Zh. vych. mat., 4, No, 3, 580-585.

Saul'yev, V. K. 1964 *Integration of Equations of Parabolic Type by the Method of Nets* Pergamon Press

Sharp, N.J. 2006 *Advances in Mortgage Valuation: An Option-Theoretic Approach*, PhD Thesis Manchester University UK

Sheppard, R. 2007 *Pricing Equity Derivatives under Stochastic Volatility: A Partial Differential Equation Approach* Master of Science Thesis University of the Witwatersrand, Johannesburg, South Africa

Tavella, D. and Randall, C. 2000 *Pricing Financial Instruments, The Finite Difference Method* John Wiley and Sons New York

Thomas, J.W. 1999 *Numerical Partial Differential Equations Volume II; Conservation Laws and Elliptic Equations* Springer

Varga, R.S. 1962 *Matrix Iterative Analysis* Prentice-Hall Inc. Englewood Cliffs New Jersey

Vichnevetsky, R., and Bowles, J.B. 1982 *Fourier Analysis of Numerical Approximations of Hyperbolic Equations* SIAM Philadelphia

Windcliff, H., Forsyth, P.A. and Vetzal, K.R. 2003 *Analysis of the Stability of the Linear Boundary Conditions for the Black-Scholes Equation* Journal of Computational Finance, Vol. 8, 65-92

Wong H. Y., Jing, Z. 2008 *An Artificial Boundary Method for American Option Pricing under the CEV Model* SIAM J. Numerical Analysis V.46, #4 May 2008