Black-Scholes Option Pricing: PDEs, Probability, and Matlab[1]

Martin V. Day[2]

May 4, 2011

[2]day@math.vt.edu

Contents

Preface

The past two or three decades has seen a bloom of interest in mathematical finance as a topic in applied mathematics. A number of universities now have graduate programs, and many more offer at least some introductory courses in the area. These courses attract graduate students from mathematics, statistics, the sciences and engineering, as well as economics, business and finance.

One central topic of introductory courses is usually the problem of pricing contingent claims or "financial derivatives." The theoretical resolution of this problem is the risk-neutral pricing formula, which expresses the arbitrage-free price of a contingent claim as the discounted mean with respect to a risk-neutral probability measure. The risk-neutral probability is characterized (uniquely for complete markets) by a martingale property of the assets in the market. Thus mathematical treatments of the subject are founded on ideas from stochastic analysis, and those topics occupy most introductions to the subject. See for instance the texts by Karatzas and Shreve [36], Björk [5], Musiela and Rutkowski [45].

The risk-neutral pricing formula provides a theoretical answer to the pricing problem, but there remains the issue of computing actual numerical values of specific contingent claims from it. For binomial trees in discrete time the calculations are elementary. The favorite continuous time formulation is the Black-Scholes model of a single stock evolving according to a "geometric" Brownian motion. For this the pricing problem becomes more substantial. In some instances there are explicit pricing formulae, such as the famous Black-Scholes formula for a European call option. In others there is not, so that numerical or other approximate methods are called for. **This book is concerned specifically with the pricing problem for the Black-Scholes model for a single stock and constant interest rate.** It is not a survey of the many ideas and models of contemporary mathematical finance.

The single-stock Black-Scholes model is simplistic for many reasons, and other more elaborate models are studied in the literature. However the Black-Scholes model provides a relatively uncomplicated but nontrivial setting in which to introduce many ideas and methods which will have generalizations in more sophisticated settings. Existence and uniqueness issues for partial differential equations, change of variable techniques, the role of boundary values and growth conditions, numerical methods for partial differential equations, and the interplay between the stochastic and PDE points of view all are natural topics. **Our goal in this book is to use the option pricing problem in the simple Black-Scholes setting as a laboratory in which to introduce a range of computational and analytic techniques.** We will see that the probabilistic perspective informs both the numerical and analytical treatments of these PDEs, so we keep it closer at hand throughout our treatment than many other introductions to computational techniques. Some other topics not common to other introductory books are more careful attention to boundary and growth conditions as $s \to 0$ and $s \to \infty$, the use of bounds on hitting probabilities to justify approximate boundary conditions, more careful discussion of the linear complementarity problems that come up for American options, and the inclusion of Markov chain methods. Considering the many graduate students from disciplines other than mathematics who take this course, it is particularly important to exhibit the interdependence between conceptual/theoretical understanding and practical calculation. References to source material or more detailed treatments are included, so that students who want to pursue topics further will have a starting place.

Since numerical calculation is a significant part of our discussion, we discuss implementations of our calculations. Matlab provides a particularly convenient and widely available platform for this. Many students are already familiar with it, and those who are not pick it up quickly. We provide several examples of Matlab code, and the problems ask for many additional calculations.

This book has evolved through teaching Math 5726 at Virginia Tech over several years. Math 5726 is the second of a two-semester course. The first covers the probabilistic background, Itô calculus and risk-neutral pricing. That material is largely assumed here, with a brief recap in the first chapter and an appendix. Thus the goal of this book, like the course from which it emerged, is narrowly focused on the problem of calculating or approximating option prices for the single-stock Black-Scholes model.

We wish to thank to Prof. Don Chance, who sat through the course the first time it was taught in Spring 2001 and offered numerous suggestions and insights. We are also grateful to the many students who have taken the course since then, enduring inconsistent notation.

Martin Day; Blacksburg, Virginia, April 2003

Notation

- We will denote random quantities with upper case letters: $S_t,\ W_t,\ X,\ \ldots$ For nonrandom quantities we will generally use lower case letters: $v(s,t),\ u(x,\tau),\ldots$. Exceptions are the exercise time T and $\xi_n,\ \zeta_n$ for the Markov chains in Chapter 6. Hitting times, which *are* random, are typically denoted $\mathcal{T}$.
- Matricies and vectors will be denoted in bold, $\mathbf{M}$ and $\mathbf{v}$, sometimes indicating them by their typical entry written in bracketts:
$$\mathbf{M} = [m_{ij}], \quad \mathbf{v} = [v_n].$$
- Partial derivatives will be denoted with either subscript notation or using operator notation, with ∂_s instead of the classical but more cumbersome $\frac{\partial}{\partial s}$:
$$v_s(s,t) = \partial_s v(s,t), \quad v_{ss}(s,t) = \partial_s^2 v(s,t), \quad v_{st} v(s,t) = \partial_s \partial_t v(s,t).$$
- We will generally use the superscript position for descriptive or identifying information, such as v^{Call}, x^L, $\mathcal{T}^H$, $d^{(i)}$, and so forth. This leaves the subscript position free for partial derivatives or array indices.

Chapter 1

Introduction and Preliminaries

The subject of this book is the calculation of the market price for various derivative financial assets in the Black-Scholes single stock model. This chapter provides a summary of the Black-Scholes model, the no-arbitrage principle, and the resulting risk-neutral valuation formula. We view these as background topics and intend this chapter only as a summary. The reader not familiar with these topics should consult a reference such as Björk [5], Karatzas & Shreve [36], or Musiela & Rutkowski [45] for a more thorough treatment.

1.1 The Black-Scholes Model

The Black-Scholes model postulates that the stock price S_t is described by a geometric Brownian motion:

$$dS_t = \alpha S_t\,dt + \sigma S_t\,dW_t^{\mathrm{o}}, \quad \text{or } S_t = S_0 e^{(\alpha - \frac{1}{2}\sigma^2)t + \sigma W_t^{\mathrm{o}}}. \tag{1.1}$$

Here W_t^{o} is a Brownian motion process[1] defined on a probability space $(\Omega, \mathcal{F}, P^{\mathrm{o}})$. The parameter σ is the *volatility* and α is the *mean rate of growth.* The market also has a risk-free asset or "bond" with constant *interest rate* r:

$$dB_t = rB_t\,dt, \quad \text{or } B_t = B_0 e^{rt}.$$

This model originated with the work of Black and Scholes [6] and Merton [42]. It's shortcomings as a model of the real market have been discussed in the literature, and various modifications have been considered. Our purpose is not to decide on the most accurate model. Without intending to diminish their importance, we are *not* covering models involving multiple stocks, state-dependent or stochastic volitility, incomplete markets, regime switching, jump-diffusions, or Leṽy processes. Nor are we considering model calibration or parameter estimation. Instead our purpose is to use the the simple Black-Scholes model as a "laboratory" in which to learn various techniques for calculating the prices of financial derivatives based on the no-arbitrage principle. The methods we learn here can be adapted to more complex models, though we will not pursue that in this book.

1.2 No-Arbitrage and Risk-Neutral Pricing

Suppose we have a (Euroepen-style) derivative asset whose value at expiry ($t = T$) is given by a random variable X. The problem is to determine the market price V_t of this asset at times $t < T$. The absence of arbitrage in the market implies the following general *risk-neutral pricing formula*:

$$V_t = e^{rt} E[e^{-rT} X \mid \mathcal{F}_t]. \tag{1.2}$$

The critical feature here is that the expectation is *not* calculated under the original probability measure P^{o} but with respect to the the *risk-neutral* probability measure, which is characterized by the property that the discounted stock price,

$$S_t / B_t$$

[1]We have denoted it W_t^{o} here to distinguish it from the W_t of (1.3), which will occur regularly in our analysis.

is a martingale. We will use P for that risk-neutral probability. It will be the underlying probability measure throughout; we will have virtually no need for the original P^{o}.

The hypotheses on which the formula (1.2) is based are as follows.

- The description of S_t and B_t is appropriate, and the values of α, r, and σ are known.
- There are no transaction costs associated with buying and selling shares of stock or bond. Stock and bond may be held in any amounts (not limited to integral nultiples or positive amounts) and can be traded continuously.
- The stock pays no dividends.
- For any time $t \le T$ knowledge of the stock price history $\{S_u : u \le t\}$ constitutes complete knowledge of the state of the market at time t. (I.e. there are no stochastic components of the model other than S_t. The values of all market variables at time t are prescribed by knowledge of the stock price history.) The σ-algebra $\mathcal{F}_t$ is determined (or "generated") by this information.
- X is $\mathcal{F}_T$ measurable and $E[X^2] < \infty$. (Remember, this is with respect to the risk-neutral probability P.)
- There are no arbitrage opportunities in the market. (Mathematically, this follows from the existence of the risk-neutral probability measure.)

An outline of how this leads to (1.2) is as follows. The effect of switching to the risk-neutral probability P is that the original W_t^{o} is no longer a Brownian motion. Instead

$$W_t = W_t^{\text{o}} + (\frac{\alpha - r}{\sigma})t \tag{1.3}$$

is a Brownian motion with respect to P. The equation for the stock price process can be reformulated as

$$dS_t = rS_t\,dt + \sigma S_t\,dW_t. \tag{1.4}$$

The solution can be written explicitly:

$$S_t = S_0 e^{(r-\sigma^2/2)t+\sigma W_t}.$$

Suppose that h_t^B and h_t^S comprise a self-financing portfolio with value process V_t,

$$\begin{aligned} V_t &= h_t^B B_t + h_t^S S_t \\ dV_t &= h_t^B\,dB_t + h_t^S\,dS_t, \end{aligned}$$

and which replicates X at time T:

$$V_T = X.$$

(This is always possible in the Black-Scholes model provided $E[X^2] < \infty$, by the completeness property and the martingale representation theorem.) Then it turns out that the discounted asset price $M_t = V_t/B_t$ is *also* a Q-martingale. This is because its stochastic differential works out to be

$$dM_t = \sigma h_t^S \frac{S_t}{B_t}\,dW_t,$$

with no dt term. By the absence of arbitrage, V_t must agree with the market value at time t of the contingent claim. The risk neutral formula (1.2) is just the statement of this martingale property:

$$M_t = E[M_T \,|\, \mathcal{F}_t], \text{ which we can rewrite as } V_t = B_t E[X/B_t \,|\, \mathcal{F}_t].$$

When the value at expiry is a function of the stock price at expiry,

$$X = \phi(S_T),$$

the market price will be given by a function $v(s,t)$ of the current time and current stock price,

$$V_t = v(S_t, t).$$

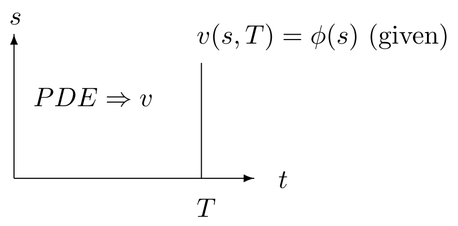
(One argument for this is outlined on page 87.) This $v(s,t)$ is called the *pricing function* for the contingent claim. The formula (1.2) reduces to a partial differential equation for v. We can see this if we use Itô's formula to derive the stochastic differential of $V_t/B_t = B_0^{-1}e^{-rt}v(S_t,t)$: we obtain

$$d[V_t/B_t] = B_t^{-1}\left(\left[v_t(S_t,t) + rsv_s(S_t,t) + \frac{1}{2}s^2\sigma^2 v_{ss}(S_t,t) - rv(S_t,t)\right] dt + v_s(S_t,t)\, dW_t\right).$$

Since $M_t = V_t/B_t$ must be a Q-martingale, the coefficient of the dt term must be 0. Thus we find that $v(s,t)$ satisfies the *Black-Scholes Partial Differential Equation* (see[2] [5, (7.31)]):

$$\boxed{\frac{1}{2}s^2\sigma^2 v_{ss} + rsv_s - rv + v_t = 0, \quad \text{for } t < T,\ 0 < s} \tag{1.5}$$

with the *terminal condition* $v(s,T) = \phi(s)$. This equation is equivalent to the martingale property of $v(S_t,t)/B_t$, and thus constitutes another expression of the risk-neutral formulation. Given $\phi(s)$ we want to solve the PDE (1.5) to find the pricing function $v(s,t)$. The following figure illustrates the pyhsical organization of this task.



1.3 Looking Ahead

The principle problem of this book is to find the pricing function v. For the standard European options discussed above, this becomes the problem of solving the Black-Scholes PDE (1.5) with the given $\phi(s)$ as the terminal data. For some terminal value functions $\phi(s)$ explicit formulas for the solutions can be written down. A simple one is for a forward contract: $\phi(s) = s - K$. In that case it is easy to check that

$$v(s,t) = s - Ke^{-r(T-t)}.$$

The most famous explicit solution corresponds to a call option: $\phi(s) = (s-K)^+$. For this we get the *Black-Scholes formula.*

$$\begin{aligned}
v^{\text{Call};K,T}(s,t) &= s\mathcal{N}(d^{(1)}) - Ke^{-r(T-t)}\mathcal{N}(d^{(2)}), \text{ where}\\
\mathcal{N}(y) &= \int_{-\infty}^{y} \frac{1}{\sqrt{2\pi}} e^{-u^2/2}\, du,\\
d^{(1)} &= \frac{\ln(s/K) + (r + \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}}\\
d^{(2)} &= d^{(1)}(s,t) - \sigma\sqrt{T-t}.
\end{aligned}$$

We will describe this and several other explicit solutions in Chapter 3. But for other choices of $\phi(s)$ no explicit solution is known. Here is one.

[2] Björk [5] gives a different argument; see his Section 7.3. Assuming that $V_t = v(S_t,t)$ is traded on the same market as S_t and B_t, form a self-financing portfolio $\tilde{V}_t$ of V_t and S_t which is risk-free: $d\tilde{V}_t = (\cdots)\, dt + 0\, dW_t$. Then no-arbitrage requires that $d\tilde{V}_t = r\tilde{V}_t\, dt$. This is equivalent to (1.5).

Example. A *chooser* option.

$$\phi(s) = \max(v^{\text{Call};K_1,T_1}(s,T), v^{\text{Put};K_2,T_2}(s,T)).$$

This corresponds to an option for which at time T the holder chooses whether the option will be a call expiring at time $T_1 > T$ with exercise price K_1, or a put expiring at time $T_2 > T$ with exercise price K_2. An explicit formula for v is known only if $K_1 = K_2$ and $T_1 = T_2$; see Hull [25].

When the terminal value function $\phi(s)$ is one for which no explicit formula for $v(s,t)$ is available, we must turn to some sort of approximate method. (Even when an explicit formula is possible, it still may be that an approximate method is easier to implement than the explicit formula.) There are several general types of approximation methods that can be considered.

- In some cases formulae can be derived which reduce the calculation to just one "inexplicit" part. The integral formula (2.18) in Section 2.4 is one example of that; we just need to resort to numerical integration to evaluate the integral. Another instance of this is a formula of Geman and Yor for the value of an Asian options which boils down to inverting a Fourier transform; see Section 9.3.3.

- In some cases there are explicit formulas which are not exact but give good approximations. Such formulae exist for both Asian and American options. An example is the formula of Rogers & Shi; see §9.3.3.

- Monte Carlo methods try to evaluate (1.2) by computing a large number of random simulations and averaging them: $E[X] \approx \frac{1}{N}\sum_1^N X_i$. This approach is used mostly for multidimensional problems, typically with 3 or more stocks.

- A popular approach is to approximate S_t by a discrete time and space model based on a random walk (called "binomial tree models" in the finance literature) and then work backwards in time to compute $E[X]$.

- Apply numerical methods for PDEs to the differential equation which describes v.

Broadie and Detemple [11] provide an overview of computational approaches as of 1996. A more recent book describing a broad range of techniques is Fusari and Roncoroni [20]. The book of Musiela and Rutkowski [45] has an extensive bibliography including both computational and analytical work.

We will emphasize the use of numerical methods for PDEs, particularly finite difference methods. These methods have been studied extensively so a lot of theoretical and practical knowledge is available about how to (or *not* to) use them effectively. Many options have additional features beyond those of the standard European derivative described above. These are sometimes called *exotic* options. The numerical PDE approach can be adapted to approximate the pricing function for virtually all of these. As we discuss different kinds of options we will encounter most of the other approaches mentioned above as well.

In Chapter 2 we will show that the Black-Scholes PDE (1.5) is equivalent to the classical heat equation, and use that connection to present the basic theoretical properties of its solutions. This will include the integral formula mentioned above.

Then in Chapter 3 we will explain how for any piecewise linear $\phi(s)$ an explicit formula for v is possible. Also in Chapter 3 we will talk about barrier options. These are an exotic option for which the value at expiry depends on whether or not the stock price reached some "trigger" level prior to expiry. This makes them "path-dependent" options, like Asians and Lookbacks. We include them in Chapter 3 because they also have explicit pricing formulae, which we will derive by PDE methods.

Chapter 4 begins our look at finite difference methods in the context of the heat equation. Along with the mathematical ideas, we will learn how to implement these calculations using MATLAB. Then in Chapter 5 we will see how these methods work out for the Black-Scholes equation itself.

Finite difference methods are related to bionomial tree methods; the connection is described in Chapter 6. We will find that this provides some insight into the issue of stability for finite difference methods.

Chapter 7 will discuss how our calculations can be generalized to dividend paying stocks, using either discrete or continuous dividend payments.

European options can be exercised only at $t = T$, but American options allow the holder to exercise them at *any* time $t \leq T$. Now the value at expiry is replaced by the value at exercise, given by $\phi(S_{\mathcal{T}}, \mathcal{T})$, where $\mathcal{T}$ is the chosen exercise time and $\phi(s, t)$ is a given payoff function. Naturally the market price of such an option depends on determining the optimal exercise time. A version of the risk-neutral formula (1.2) is not hard to write down; see (8.1). Its evaluation, however, is much harder than in the European case. From the PDE point of view we must deal with a variational inequality, and finite difference methods now involve solving a linear complementarity problem at each stage. These things are the subject of Chapter 8.

Chapter 9 considers Asian options. These have a value at expiry which depends on an average stock price over $0 \leq t \leq T$. There are several different ways to compute the average; we will talk about how we can approach pricing calculations for these. Lookback options, considered in Chapter 10, are similar to Asians except that a historical maximum or minimum stack price is used instead of an average stock price. We are forced to add an extra dimension to our calculations and will want to talk about how to do that efficiently in Matlab.

Finally, Chapter 11 will introduce the use of Monte Carlo methods.

Techniques for implementing calculations in Matlab are discussed throughout. Some additional advice about programming in Matlab is collected in an appendix. A second appendix gives a brief synopsis of facts about martingales and Itô's formula.

Chapter 2

Black-Scholes and the Heat Equation

We will use $\mathcal{B}$ to denote the differential operator

$$\mathcal{B}v = \frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s - rv,$$

so that the the Black-Scholes equation (1.5) is described concisely as

$$\mathcal{B}v(s,t) + \partial_t v(s,t) = 0. \tag{2.1}$$

We are interested in solving this for $0 < s$, $0 \le t \le T$, with a prescribed terminal value function

$$v(s,T) = \phi(s). \tag{2.2}$$

In this chapter we present some basic theory describing this problem. We want to explain what kind of $\phi(s)$ can be considered; how we should understand (2.2) if $\phi(s)$ is discontinuous; whether there will always be a solution; and whether there can be more than one solution for the same $\phi(s)$. We will show that (2.1) is equivalent to the classical heat equation. The questions we just raised have all been answered for the heat equation, so we can simply translate results from the literature on the heat equation into the Black-Scholes context to obtain the answers to our questions. We will see in particular that there is an integral formula for the solution v in terms of $\phi(s)$. We will end the chapter with a collection of explicit solutions and simple transformations for constructing new solutions from old ones.

2.1 Introduction to Partial Differential Equations

To begin we want to place the Black-Scholes equation in the standard taxonomy of partial differential equations. The Black-Scholes PDE is a *second order, linear, homogeneous* equation. It is second order because the highest partial derivative involved is of second order. It is linear because the partial derivatives v_s, v_{ss}, v_t occur simply multiplied by known functions of the independent variables (s and t) and then added.

In general a second order linear PDE for a function $u(x,t)$ takes the form

$$au_{xx} + bu_{xt} + cu_{tt} + du_x + eu_t + fu + g = 0. \tag{2.3}$$

The coefficients are (prescribed) functions of the independent variables: $a = a(x,t), \ldots, g = g(x,t)$. When $g = 0$ the equation is homogeneous. The problem is to find the function(s) $u(x,t)$ which solve (2.3) for (x,t) in some region, usually in conjunction with some other "boundary conditions," i.e. properties or values of the function which are specified on the boundary of the region. For (2.1) the region we are interested is the vertical strip

$$0 \le t \le T, \quad 0 < s$$

and the boundary condtion is the requirement that $v(s,T) = \phi(s)$ for all $s > 0$. There is no boundary condition specified for $s = 0$ (or $t = 0$); later in this chapter we will see why this is so.

Assuming that a, b, and c are not all zero (else the equation would not be second order), equations of the form (2.3) are usually categorized as one of three types: hyperbolic, parabolic, or elliptic[1]. The best known examples of each type are the following.

Wave Equation (hyperbolic): $u_{xx} - u_{tt} = 0$.

Heat Equation (parabolic): $u_{xx} - u_t = 0$.

Laplace Equation (ellptic)[2]: $u_{xx} + u_{yy} = 0$.

Each of the three basic PDEs above has been thoroughly studied. Other equations of the same type (hyperbolic, parabolic, or elliptic) typically share many of the same properties as the appropriate one of these examples. The Black-Scholes equation is parabolic. In fact it is fully equivalent to the heat equation by means of a change of variables which we will work out in the next section. This is very convenient for us, because every question we might ask about the Black-Scholes equation can be translated into an analogous question about the heat equation. We can then turn to the extensive literature concerning the heat equation (or parabolic PDEs in general) to find the answer.

2.2 Conversion of Black-Scholes to the Heat Equation

We are going to develope a change of variables which converts solutions of the Black-Scholes equation (2.1) into solutions of the classical heat equation. We will work this out through a sequence of steps.

Step 1. First we observe that in terms of $y = \log(s)$ the terms $s\frac{\partial}{\partial s}$ and $s^2\frac{\partial^2}{\partial^2 s}$ convert to constant multiples of $\frac{\partial}{\partial y}$ and $\frac{\partial^2}{\partial^2 y}$. To be specific suppose

$$v(s,t) = f(y,t) = f(\log(s),t)$$

for some function $f(y,t)$. Then we find from the chain rule that

$$\begin{aligned} sv_s(s,t) &= s \cdot \frac{1}{s} f_y(\log(s),t) = f_y(y,t) \\ s^2 v_{ss}(s,t) &= s^2 \left[\frac{-1}{s^2} f_y(\log(s),t) + \frac{1}{s^2} f_{yy}(\log(s),t) \right] = f_{yy}(y,t) - f_y(y,t). \end{aligned}$$

The time derivative is unchanged: $v_t(s,t) = \partial_t[f(\log(s),t)] = f_t(y,t)$. If we make these substitutions in (2.1), we find that

$$\frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s - rv + v_t = \frac{1}{2}\sigma^2 f_{yy} + (r - \frac{1}{2}\sigma^2) f_y - rf + f_t,$$

where the left side is evaluated at (s,t) and the right side is evalued at the corresponding value of (y,t). So in terms of the new variable $y = \log(s)$ we find that $v(s,t) = f(y,t)$ solves the Black-Scholes equation (2.1) if and only if f solves the equation

$$\frac{1}{2}\sigma^2 f_{yy} + (r - \frac{1}{2}\sigma^2) f_y + f_t - rf = 0. \tag{2.4}$$

This gives us an equation with constant coefficients, as opposed to the variable coefficients of (2.1). Also observe that $0 < s < \infty$ corresponds to $-\infty < y < \infty$.

[1]The terminology comes from the type of curves in the x,t-plane determinded by equations of the form $ax^2 + bxt + ct^2 + dx + et + f = 0$ where $a, \ldots, f$ are constants.

[2]The wave and heat equations come from physical applications in with the variable t represents time. In the Laplace equation, however, the variables usually correspond to coordinates of a point in the plane, so it is more common to write the Lapace equation for a function $u(x,y)$ rather than $u(x,t)$.

Step 2. We want to make additional variable changes to convert (2.4) into the heat equation. The next step is to eliminate the first order dertivative f_y. This can be accomplished by a change of dependent variable:

$$f(y,t) = e^{c_1 y} g(y,t),$$

for a carefully choosen constant c_1. If we substitute this expression for f into (2.4) we obtain

$$e^{c_1 y}\left[\frac{1}{2}\sigma^2 g_{yy} + (c_1\sigma^2 + r - \frac{1}{2}\sigma^2)g_y + g_t + (c_1 - 1)(r + c_1\frac{1}{2}\sigma^2)g\right] = 0.$$

Making the choice

$$c_1 = \frac{1}{2} - \frac{r}{\sigma^2}$$

eliminates the g_y term, so that equation (2.4) for f is equivalent to the following equation for g:

$$\frac{1}{2}\sigma^2 g_{yy} + g_t + (c_1 - 1)(r + c_1\frac{1}{2}\sigma^2)g = 0. \tag{2.5}$$

Step 3. We next eliminate the undifferentiated term g by another change of dependent variable:

$$g(y,t) = e^{c_2 t} h(y,t).$$

This makes (2.5) equivalent to

$$e^{c_2 t}\left[\frac{1}{2}\sigma^2 h_{yy} + h_t + c_2 h + (c_1 - 1)(r + c_1\frac{1}{2}\sigma^2)h\right] = 0.$$

We make the specific choice

$$c_2 = -(c_1 - 1)(r + c_1\frac{1}{2}\sigma^2) = \frac{(\sigma^2 + 2r)^2}{8\sigma^2}.$$

This makes (2.5) equivalent to

$$\frac{1}{2}\sigma^2 h_{yy} + h_t = 0. \tag{2.6}$$

Step 4. Now we make one last change variables to scale out the $\frac{1}{2}\sigma^2$ and reverse the sign on the time derivative. With $x = \sqrt{2}y/\sigma$, $\tau = T - t$ and

$$h(y,t) = e^{-c_2 T} u(x,\tau) = e^{-c_2 T} u(\sqrt{2}y/\sigma, T - t)$$

we find that (2.6) is equivalent to the heat equation:

$$u_{xx}(x,\tau) - u_\tau(x,\tau) = 0. \tag{2.7}$$

Now put all the steps together to obtain the conversion formula.

$$\begin{aligned}
v(s,t) &= f(y,t) \\
&= e^{c_1 y} g(y,t) \\
&= e^{c_1 y + c_2 t} h(y,t) \\
&= e^{c_1 y + c_2 t} e^{-c_2 T} u(x,\tau) \\
&= e^{(c_1\sigma/\sqrt{2})x - c_2\tau} u(x,\tau).
\end{aligned}$$

Note that the role of $e^{-c_2 T}$ in Step 4 is simply to get all the t occuring in the combination $T - t = \tau$. This τ is the time variable we will use for the heat equation. It is different from the t in the Black-Scholes equation.

To make this change of variables convenient to refer to, define the parameters $\mu = c_1\sigma/\sqrt{2}$ and $\nu = c_2$. (Don't confuse the Greek ν ("nu") with the Latin v.) After simplifying the expressions for μ and ν, we have

$$\boxed{\begin{aligned} &v(s,t) = e^{\mu x - \nu\tau} u(x,\tau), \quad \text{where} \\ &\quad s = e^{\sigma x/\sqrt{2}} \\ &\quad t = T - \tau \\ &\quad \mu = \frac{1}{\sqrt{2}}\left(\frac{\sigma}{2} - \frac{r}{\sigma}\right) \\ &\quad \nu = r + \mu^2. \end{aligned}} \tag{2.8}$$

We will refer to v, s and t as *financial variables* and u, x, and τ as the corresponding *heat variables.* The above formulas establish a one-to-one conversion between functions $v(s,t) : (0,\infty) \times (-\infty, T] \to \mathbb{R}$ and functions $u(x,\tau) : (-\infty,\infty) \times [0,\infty) \to \mathbb{R}$ with the propety that

$$\mathcal{B}v(s,t) + \partial_t v(s,t) = 0 \text{ if and only if } \partial_x^2 u(x,\tau) - \partial_\tau u(x,\tau) = 0.$$

For the heat equation we usually start with values of u for $\tau = 0$ prescribed by some *initial value function*,

$$u(x,0) = \psi(x),$$

and solve for u as τ moves forward in time. For the Black-Scholes equation we start with values given at $t = T$ by the terminal value function, $v(s,T) = \phi(s)$, and solve for v as t moves backwards in time. Observe that $\psi(x)$ and $\phi(s)$ are connected by our change of variables:

$$\phi(s) = v(s,T) = e^{\mu x} u(x,0) = e^{\mu x}\psi(x).$$

2.3 The Heat Equation: Existence and Uniqueness

We next summarize a number of important properties of the heat equation

$$u_{xx}(x,\tau) - u_\tau(x,\tau) = 0; \quad -\infty < x < \infty, \ \tau \geq 0.$$

In Section 2.4 we will translate these into financial variables to obtain analogous properties of the Black-Scholes equation. (A good first reference on PDEs in general, including the heat equation in Chapter 7, is John [32]. For a more extensive treatment of the heat equation specifically, see Widder [60].)

First observe that if we know two solutions $u^{(1)}$ and $u^{(2)}$ of (2.7), then any linear combination of them will also be a solution. To see this consider

$$u(x,\tau) = c_1 u^{(1)}(x,\tau) + c_2 u^{(2)}(x,\tau),$$

where c_1 and c_2 are any constants. Then

$$\begin{aligned} u_{xx} - u_\tau &= (c_1 u^{(1)}_{xx} + c_2 u^{(2)}_{xx}) - (c_1 u^{(1)}_t + c_2 u^{(2)}_t) \\ &= c_1(u^{(1)}_{xx} - u^{(1)}_\tau) + c_2(u^{(2)}_{xx} - u^{(2)}_\tau) \\ &= 0. \end{aligned}$$

This property of combining solutions is called the *principle of superposition.*

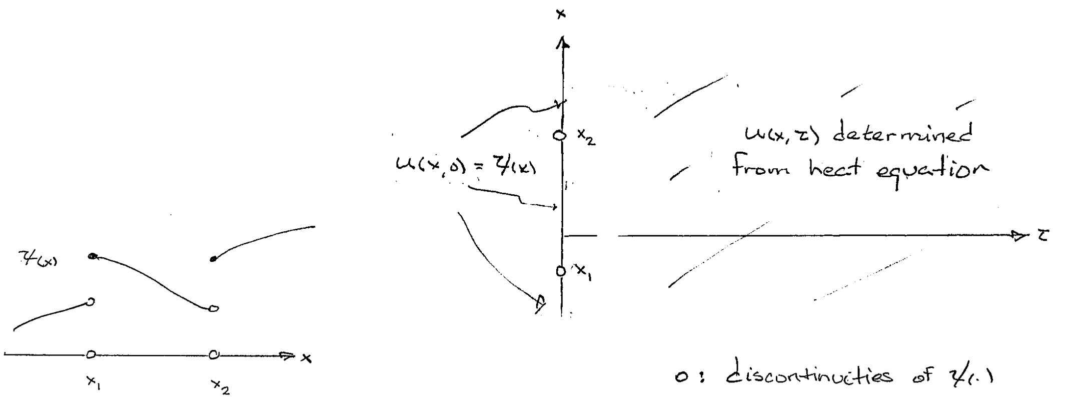
There are many solutions to the heat equation. As we said above, we will usually have an initial function $\psi(x)$ which prescribes initial data for the solution u we want. In other words, we require the solution to match ψ at $\tau = 0$:

$$u(x,0) = \psi(x) \text{ for all } x \in \mathbb{R}. \tag{2.9}$$

When we say $u(x,\tau)$ is a solution of (2.7) that obviously means that the partial derivatives u_{xx} and u_τ exist, as well as the fact that they are equal. The functions $\psi(x)$ which are of interest to us for option pricing purposes are generally *not* differentiable for every x. So we can *not* insist that $u(x,0) = \psi(x)$ be differentiable. In other words we can only expect that (2.7) holds for $\tau > 0$, not for $\tau = 0$. But we *do* want the values of $u(x,0) = \psi(x)$ to be associated with the values of $u(x,\tau)$ for $\tau > 0$. Thus what we will mean by saying that $u(x,\tau)$ is a solution of the *initial value problem* (2.7) and (2.9) is that

- $u(x,\tau)$ is continuous in $x \in \mathbb{R}$ and $\tau \geq 0$,
- $u(x,0) = \psi(x)$ for all $x \in \mathbb{R}$,
- $u(x,\tau)$ is twice continuously differentiable in $x \in \mathbb{R}$ and once differentiable in τ for $\tau > 0$,
- $u(x,\tau)$ satisfies (2.7) for all $x \in \mathbb{R}$ and $\tau > 0$.

Ocasionally we need to consider $\psi(x)$ which is discontinuous at a finite number of x-values: x_1, x_2, ... In that case we need to exclude $(x,\tau) = (x_i, 0)$ from the first two bullets. The following picture indicates the geometric layout of this problem.


$\psi(x)$
x_1
x_2
x
$u(x,0) = \psi(x)$
τ
$u(x,\tau)$ determined from heat equation
o : discontinuities of $\psi(\cdot)$

The heat equation is intimately connected to Brownian motion. We can exploit that find a formula for $u(x,\tau)$ in terms of $\psi(x)$. To see the connection, consider $M_t = u(\sqrt{2}W_t, T-t)$ and apply Itô's formula to see that

$$\begin{aligned} dM_t &= \sqrt{2}u_x \, dW_t + (\frac{1}{2}2u_{xx} - u_\tau)\, dt \\ &= \sqrt{2}u_x \, dW_t, \end{aligned} \tag{2.10}$$

by virtue of the heat equation. So (under some integrability assumptions on u_x) M_t will be a martingale. Using that we find that

$$\begin{aligned} u(\sqrt{2}W_t, T-t) &= M_t \\ &= E[M_T \mid \mathcal{F}_t] \\ &= E[u(\sqrt{2}W_T, 0) \mid \mathcal{F}_t] \\ &= E[\psi(\sqrt{2}W_T) \mid \mathcal{F}_t] \end{aligned} \tag{2.11}$$

Conditioned on $\sqrt{2}W_t = x$, $\sqrt{2}W_T$ is a normal random variable with mean x and variance $2(T-t)$. So, with $\tau = T - t > 0$ we can continue to work out the above as

$$\begin{aligned} u(x,\tau) &= \int_{-\infty}^{\infty} \psi(y) \frac{1}{\sqrt{2\pi \cdot 2(T-t)}} e^{-(y-x)^2/2\cdot 2(T-t)}\, dy \\ &= \int_{-\infty}^{\infty} \frac{1}{\sqrt{4\pi\tau}} e^{-(y-x)^2/4\tau} \psi(y)\, dy \\ &= \int_{-\infty}^{\infty} k(x,\tau;y)\psi(y)\, dy. \end{aligned} \tag{2.12}$$

The function

$$k(x,\tau;y) = \frac{1}{\sqrt{4\pi\tau}} e^{-(y-x)^2/4\tau}$$

occuring here is called the *fundamental solution* of the heat equation, or the *heat kernel.* (Its also the transition density for $\sqrt{2}W_t$.) You can check that for a fixed y, $k(x,\tau;y)$ does solve the heat equation for $\tau > 0$. (See the Problem 2.A at the end of the chapter.) According the the principle of superposition, it follows that for any choce of values y_j and constants $c_j = \psi(y_j)$ the sum

$$\sum_{j=1}^{N} \psi(y_j)k(x,\tau;y_j)$$

will also be a solution. We can view (2.12) as a limiting version of of this construction. The reason (2.12) produces a solution with $u(x,0) = \psi(x)$ is that as $\tau \to 0$ the graph of $y \mapsto k(x,\tau;y)$ "bunches up" at $y = x$, so that when τ is small the integral only uses values of $\psi(y)$ for y very close to x.

The formula (2.12) is called a *convolution* because it is of the form $\int f(x-y)\psi(y)\,dy$. The formula is very nice in that it tells us how to find the solution u for any given ψ. But it is not the the whole story. There are two important issues. One is that for some functions $\psi(x)$ the integral might fail to exist, particularly if $\psi(x)$ grows too fast as $x \to \pm\infty$. Secondly, it turns out that even for the nicest $\psi(x)$, there are always other solutions of the heat equation with $u(x,0) = \psi(x)$; (2.12) gives *only one* of many possible solutions! For instance, if $\psi(x) = 0$ then (2.12) produces $u(x,\tau) = 0$, but there are other solutions with $u(x,0) = 0$ which have $u(x,\tau) \neq 0$ for $\tau > 0$. (See the treatment of the Tychonoff solution in [32].) So how do we know if the solution produced by (2.12) is the one we want, or if we should be looking for one of the others? What sets the solution from (2.12) apart from the others? The following theorem provides an answer to both of these issues.

Theorem. *Suppose $\psi(x)$ is a continuous function of $x \in \mathbb{R}$ with the property that*

$$\frac{\log(1+|\psi(x)|)}{1+|x|} \text{ is a bounded function.} \tag{2.13}$$

Then the integral in (2.12) *does exist for all x and $\tau > 0$, the resulting function function $u(x,\tau)$ solves the heat equation for $\tau > 0$ and is a continuous function of $x \in \mathbb{R}$ and $\tau \geq 0$ with $u(x,0) = \psi(x)$. Moreover u is the only such solution with the property that for each $T < \infty$,*

$$\frac{\log(1+|u(x,\tau)|)}{1+|x|} \text{ is a bounded function of } x \in \mathbb{R},\ 0 \leq \tau \leq T. \tag{2.14}$$

We want to digest what this theorem says. Suppose K is a bound on the expression (2.14). That implies that for all $0 \leq \tau \leq T$ and all x,

$$|u(x,\tau)| \leq e^K e^{K|x|}.$$

On the other hand if there are positive constants k_1 and k_2 so that

$$|u(x,\tau)| \leq k_1 e^{k_2|x|}, \tag{2.15}$$

then

$$\begin{aligned} 1+|u(x,\tau)| &\leq (k_1+1)e^{k_2|x|} \\ &= e^{\log(1+k_1)+k_2|x|} \\ &\leq e^{K(1+|x|)}, \end{aligned}$$

where $K = \max(\log(1+k_1), k_2)$. Thus (2.14) is just a way of saying that (given T) there exist constants k_1 and k_2 for which (2.15) holds. We can describe this by saying that u *grows at most exponentially as* $|x| \to \pm\infty$. What the theorem tells us is that if $\psi(x)$ grows at most exponentially, then the integral formula (2.12) does indeed produce a solution with initial data ψ, and the solution it produces is the *only* solution which grows at most exponentially. All the other solutions *fail* to satisfy the exponential growth condition (2.14).

When $\psi(x)$ is discontinuous at a finite number of points the assertions of the theorem remain true if we are careful about the sense in which $u(x,0) = \psi(x)$. Now u will be continuous at all (x,τ) with $\tau > 0$ and at those $(x,0)$ for which ψ is continuous at x. And we can only say that $u(x,0) = \psi(x)$ at the continuity points of $\psi(x)$. With these adjustments, the theorem above remains true.

You won't find the above theorem in [32], but it can be proven as a consequence of the theorems of his Chapter 7. (We will not go through those details.) If you consult that reference you will see that more can be said about the solution (2.12). For instance $u(x,\tau)$ has continuous derivatives of all orders, provided $\tau > 0$. Actually a version of this theorem is true in which (2.13) and (2.14) are replaced by something like (2.15) but with $k_1 e^{k_2 x^2}$ on the right side. (It is in *that* form that you will find the pieces of this theorem in [32].) The exponential growth condition as we have stated it is a more stringent requirement, but turns out to be well-suited to our application to the Black-Scholes equation in Section 2.4.

2.3.1 The Heat Equation on a Strip

We also need to understand the heat equation when x is limited to an interval. Suppose we want to limit x to an interval $x^{\mathrm{L}} \le x \le x^{\mathrm{H}}$. Then we would want to solve the heat equation on the "strip"

$$u_{xx} - u_t = 0 \text{ for } x^{\mathrm{L}} \le x \le x^{\mathrm{H}}, \quad 0 < \tau,$$

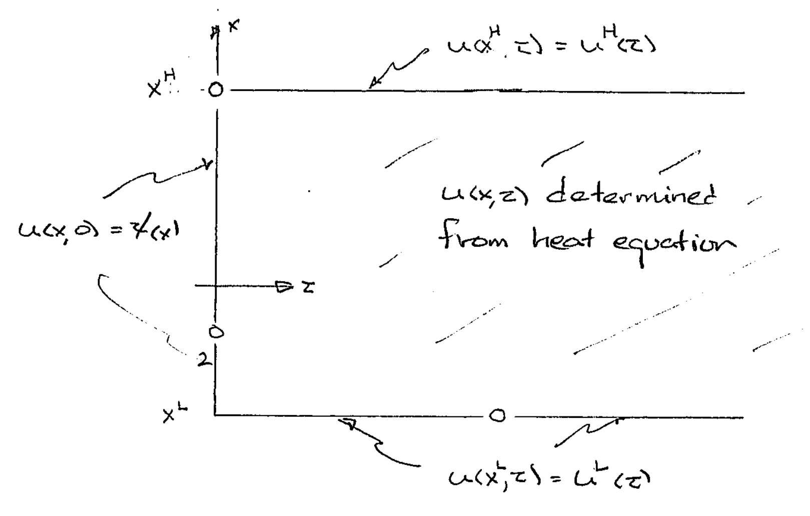
with prescribed initial values

$$u(x,0) = \psi(x) \text{ for } x^{\mathrm{L}} \le x \le x^{\mathrm{H}}.$$

This by itself is not enough information to determine the solution $u(x,\tau)$. In addition we need to know the values of $u(x^{\mathrm{L}},\tau)$ and $u(x^{\mathrm{H}},\tau)$ along the bottom and top of the strip. If we are given functions $u^{\mathrm{L}}(\tau)$ and $u^{\mathrm{H}}(\tau)$ then, in addition to the heat equation and initial values, we also want to ask the the solution satisfy the *boundary conditions*

$$u(x^{\mathrm{L}},\tau) = u^{\mathrm{L}}(\tau) \text{ and } u(x^{\mathrm{H}},\tau) = u^{\mathrm{H}}(\tau), \quad \tau > 0. \tag{2.16}$$

If $\psi(x)$, $u^{\mathrm{L}}(\tau)$ and $u^{\mathrm{H}}(\tau)$ are continuous with agreement at the corners ($\psi(x^L) = u^{\mathrm{L}}(0)$ and $\psi(x^H) = u^{\mathrm{H}}(0)$) then there will be a unique function $u(x,\tau)$ which is continuous in the (closed) strip solving the heat equation and the prescribed intial and boundary conditions. Since x is bounded, there are no growth conditions as $|x| \to \infty$ to worry about; the boundary condtions (2.16) have essentially taken over that role. If there are a finite number of points of discontinuity for u^{L}, u^{H}, and ψ, or mismatch at the corners ($\psi(x^{\mathrm{L}}) \ne u^{\mathrm{L}}(0)$ or $\psi(x^{\mathrm{H}}) \ne u^{\mathrm{H}}(0)$) then u will not be continuous at those disconituity points and will fail to satisfy the inital or boundary conditions at those particular points. However the existence and uniqueness of the solution still hold.



A formula like (2.12) which gives the solution u explicitly in terms of ψ, u^{L}, and u^{H} is possible, using either Fourier series (the method of "separation of variables") or theta-functions (see [32]). We will not write it down, because it would take us too far afield and we will not need it for any purpose below.

2.3.2 The Heat Equation on a Semi-Strip

The heat equation can also be considered on a semi-infinite strip like

$$-\infty < x \leq x^{\mathrm{H}}, \quad 0 < \tau.$$

In that case a boundary condition is needed at $x = x^{\mathrm{H}}$

$$u(x^{\mathrm{H}}, \tau) = u^{\mathrm{H}}(\tau) \text{ for all } \tau > 0,$$

and a growth condition (2.13) and (2.14) as $x \to -\infty$. (Or we could use the strip with $x^{\mathrm{L}} \leq x$ with a boundary conditon at $x = x^{\mathrm{L}}$ and a growth condition for $x \to +\infty$.) The theory of this kind of problem is analogous to that described above.

Example. Suppose we want to solve the above problem with $u^{\mathrm{H}}(\tau) = 1$ and $\psi(x) = 0$. We can tease a formula for the solution out of (2.12) by extending the defintion of $\psi(x)$ to $x > x^H$:

$$\psi(x) = \begin{cases} 2 & \text{if } x > x^H \\ 0 & \text{if } x \leq x^H. \end{cases}$$

Let $u(x, \tau)$ be the solution produced by (2.12).

$$u(x, \tau) = \int_{x^H}^{\infty} 2k(x, \tau; y)\, dy = 2\mathcal{N}\left(\frac{x - x^H}{\sqrt{2\tau}}\right).$$

It solves the heat equation for all $x \in \mathbb{R}$, $\tau > 0$. So for $x < x^H$ we have $u(x, 0) = \psi(x) = 0$. For $x = x^H$ and $\tau > 0$ we find that

$$u(x^H, \tau) = \int k(x^H, \tau; y)\psi(y)\, dy = 2\int_{x^H}^{\infty} \frac{1}{\sqrt{4\pi\tau}} e^{-(y-x^H)^2/4\tau}\, dy = 1.$$

Thus this is the solution we sought. (There is a mismatch at the corner, so strictly speaking neither the boundary or initial conditions hold at $x = x^H$, $\tau = 0$.) This trick of carefully selecting the initial data on the *opposite side* of the boundary in order to achieve desired values *on* the boundary is something we will use again in Chpater 3 in order to price barrier options.

The solution of this example has a useful interpretation for Brownian motion. Pick a value b, take $x^H = \sqrt{2}b$ and let $u(x, \tau)$ be the solution above. We know that

$$M_t = u(\sqrt{2}W_t, T - t)$$

is a martingale. Let $\mathcal{T}^b$ be the first time the Brownian motion W_t reaches the value b. Then $M_{t\wedge\mathcal{T}^b}$ is also a martingale, so we can write

$$\begin{aligned} u(0, T) &= M_0 \\ &= E[M_{T\wedge\mathcal{T}^b}] \\ &= E[u(\sqrt{2}b, T - \mathcal{T}^b); \mathcal{T}^b \leq T] + E[u(\sqrt{2}W_T, 0); T < \mathcal{T}^b] \\ &= E[1; \mathcal{T}^b \leq T] + E[0; T < \mathcal{T}^b] \\ &= P(\mathcal{T}^b \leq T). \end{aligned}$$

(In spite of the mismatch between boundary and initial conditions at the corner this formula is true, because $P_x(\mathcal{T}^b = T) = 0$.) Thus $u(0, T)$ provides a formula for the distribution function of the first passage time $\mathcal{T}^b$ of Brownian motion:

$$P(\mathcal{T}^b \leq T) = u(0, T) = 2\mathcal{N}\left(\frac{-\sqrt{2}b}{\sqrt{2T}}\right) = 2\mathcal{N}\left(-b/\sqrt{T}\right)$$

By differentiating with respect to $T > 0$ we calculate the density of $\mathcal{T}^b$ to be $\frac{b}{\sqrt{2\pi}} t^{-3/2} e^{-b^2/2t}$, for $t > 0$.

2.4 The Black-Scholes Equation: Existence and Uniqueness

We can now use our change of variables (2.8) to translate the results for the heat equation into analogous results for the Black-Scholes equation. Especially important is what the growth condition (2.14), or equivalently (2.15), becomes in terms of financial variables. Because $s = e^{\sigma x/\sqrt{2}}$, we see that $x \in \mathbb{R}$ corresponds to $0 < s$ and we have

$$e^{|x|} = \begin{cases} s^{\sqrt{2}/\sigma} & \text{for } s \geq 1 \\ s^{-\sqrt{2}/\sigma} & \text{for } 0 < s < 1 \end{cases} = \max(s, 1/s)^{\sqrt{2}/\sigma}.$$

Thus (2.15) means that

$$|v(s,t)| \leq s^{\sqrt{2}\mu/\sigma} e^{-\nu(T-t)} k_1 \max(s, 1/s)^{k_2\sqrt{2}/\sigma} = e^{-\nu(T-t)} k_1 \begin{cases} s^{\frac{\sqrt{2}}{\sigma}(\mu+k_2)} & \text{for } s \geq 1 \\ (1/s)^{\frac{\sqrt{2}}{\sigma}(k_2-\mu)} & \text{for } 0 < s < 1. \end{cases}$$

It follows that (2.15) in heat variables (for $0 \leq \tau \leq T$) is equivalent to saying that in financial variables there exist constants m_i so that

$$|v(s,t)| \leq \begin{cases} m_1 s^{m_2} & \text{when } s \geq 1 \\ m_1 (1/s)^{m_2} & \text{when } 0 < s < 1 \end{cases} \tag{2.17}$$

for $0 \leq t \leq T$. We describe this by saying that v *grows at most polynomially* in s as $s \to \infty$, and in $1/s$ as $s \to 0$. Here is what our theorem about the heat equation says when translated to the Black-Scholes equation.

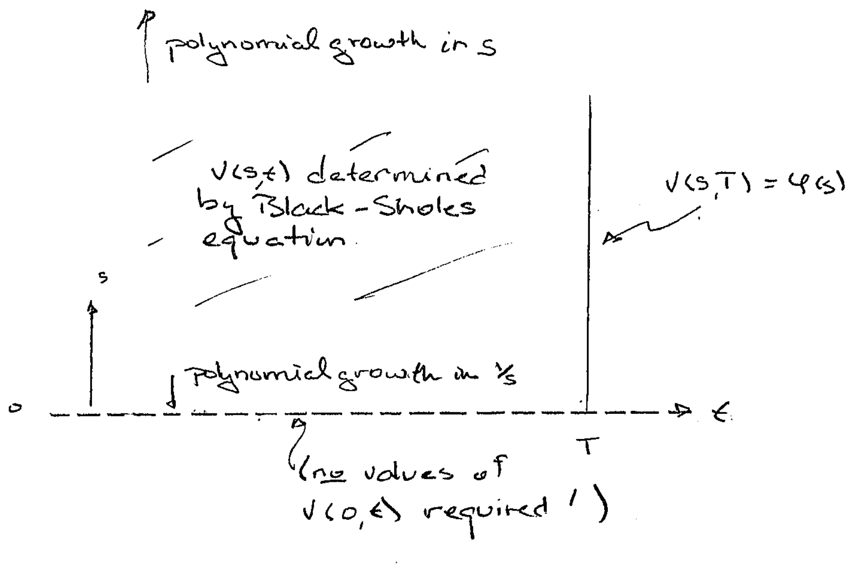
Theorem. *Suppose that $\phi(s)$ is a continuous function defined for $s > 0$ which grows at most polynomially in s as $s \to \infty$ and in $1/s$ as $s \to 0$. Then there is a unique function $v(s,t)$ defined and continuous for $0 < s$ and $0 \leq t \leq T$ which solves the Black-Scholes equation* (1.5) *for $0 < s$ and $t < T$, satisfies $v(s,T) = \phi(s)$, and which grows at most polynomially in s as $s \to \infty$ and in $1/s$ as $s \to 0$. This solution is given by the integral formula*

$$v(s,t) = e^{-r(T-t)} \int_0^\infty q(s,t;z,T)\phi(z)\,dz, \tag{2.18}$$

where

$$q(s,t;z,T) = g(T-t, \frac{z}{s})\frac{1}{s}, \tag{2.19}$$
$$g(\tau, x) = \frac{1}{\sigma\sqrt{2\pi\tau}} e^{-\mu^2(T-t)} x^{-\left(1+\frac{\sqrt{2}\mu}{\sigma}+\frac{1}{2\tau\sigma^2}\log(x)\right)},$$

μ and ν being the parameters defined in (2.8).


polynomial growth in s
v(s,t) determined by Black-Sholes equation.
v(s,T) = φ(s)
s
polynomial growth in 1/s
0
t
T
(no values of v(0,t) required!)

The integral formula is simply the result of taking

$$\psi(x) = u(x,0) = e^{-\mu x} v(e^{\sigma x/\sqrt{2}}, T) = e^{-\mu x}\phi(e^{\sigma x/\sqrt{2}}),$$

using it in (2.12), and then changing back to financial variables ($y = \sqrt{2}\log(z)/\sigma$, $\psi(y) = e^{-\mu y}\phi(z)$).

$$\begin{aligned} v(s,t) &= e^{-\nu(T-t)+\mu x}\int_{-\infty}^{\infty} k(x,\tau;y)\psi(y)\,dy \\ &= e^{-\nu(T-t)}\int_{-\infty}^{\infty} k(x,\tau;y)e^{\mu x}e^{-\mu y}\phi(z)\,dy \\ &= e^{-r(T-t)}\int_0^{\infty} e^{-\mu^2(T-t)}k(x,\tau;y)e^{\mu(x-y)}\phi(z)\frac{\sqrt{2}}{\sigma}\frac{1}{z}\,dz \\ &= e^{-r(T-t)}\int_0^{\infty} q(s,t;z,T)\phi(z)\,dz. \end{aligned}$$

We obtain the formula (2.19) for $q(s,t;z,T)$ by substituting variables and simplifying the expression

$$q(s,t;z,T) = e^{-\mu^2(T-t)}\frac{\sqrt{2}}{\sigma}e^{\mu(x-y)}k(x,\tau;y)\frac{1}{z}.$$

We could arrive at this formula directly from the risk-neutral expression

$$\begin{aligned} v(x,t) &= e^{-r(T-1)}E[\phi(S_T)\mid S_t = s] \\ &= e^{-r(T-t)}\int_0^{\infty} q(s,t;z,T)\phi(z)\,dz, \end{aligned}$$

where q is the transition density for the stock price process, under the risk neutral probability. We can solve the stochastic equation for S_t explicitly:

$$\begin{aligned} S_T &= S_0 e^{(r-\sigma^2/2)t+\sigma W_t} \\ &= S_t e^{(r-\sigma^2/2)(T-t)+\sigma(W_T-W_t)}. \end{aligned}$$

Given that $S_t = s$, $q(s,t;z,T)$ is the density (as a function of z) of above random variable. This works out to the same expression for q as in the theorem.

The integral formula (2.18) is only of theoretical interest. Although it could be implemented numerically, it would probably be easier to convert $\phi(s)$ to $\psi(x)$; evaluate (2.12), and then convert back to financial variables. (See Problem 2.I at the end of the chapter.) What is significant about the theorem is that, unlike the heat equation on a semi-strip, the Black-Scholes equation *does not require boundary values* for $v(0,t)$ to be specified in order to determine a unique solution. We don't really consider $s = 0$ to even be a legitimate value of the variable s; it corresponds to $x = -\infty$ in heat variables. Given $\phi(s)$, all that we require about $v(s,t)$ for $s \approx 0$ is that $v(s,t)$ is bounded by some power of $1/s$ as $s \downarrow 0$. This does *not* mean that $\lim_{s\downarrow 0} v(s,t)$ does not exist or is undetermined. It only means that we don't need values of $v(0,t)$ to be prescribed in order to determine v. In fact, as we will see in Section 3.3, if $\phi(0) = \lim_{s\downarrow 0} \phi(s)$ exists then $\lim_{s\downarrow 0} v(s,t) = e^{-r(T-t)}\phi(0)$.

2.4.1 Polynomial Growth and Risk-Neutral Pricing

The solutions $v(s,t)$ of (2.1) and (2.2) that we are interested in are those which come from the risk neutral pricing formula

$$v(S_t, t) = E[e^{-r(T-t)}\phi(S_T) \,|\, \mathcal{F}_t]. \tag{2.20}$$

We want to be sure that these *are* the solutions to which the above theorem applies, in other words that they *do* satisfy the polynomial growth conditons of the theorem. What makes this true is that all the terminal value functions $\phi(s)$ that we are interested in satisfy a linear bound:

$$|\phi(s)| \le Cs + C, \text{ for some nonnegative constant } C \text{ and all } s > 0. \tag{2.21}$$

Consider some of our typical choices for $\phi(s)$.

- For a forward, $\phi(s) = s - K$, so $|\phi(s)| \le s + K$. So (2.21) holds with $C = K$.
- For a call or put, $\phi(s) = (s-K)^{\pm}$, so again $|\phi(s)| \le s + K$.
- Even for something more complicated, like a chooser ($T < T_i$):

  $$\phi(s) = \max(v^{\text{Call};K_1,T_1}(s,T), v^{\text{Put};K_2,T_2}(s,T)),$$

  because of the bounds

  $$0 \le v^{\text{Call};K_1,T_1}(s,T) = s\mathcal{N}(d_1) - K_1 e^{-r(T_1-T)}\mathcal{N}(d_2) \le s,$$

  and

  $$0 \le v^{\text{Put};K_2,T_2}(s,T) = -s[1-\mathcal{N}(d_1)] + K_2 e^{-r(T_2-T)}[1-\mathcal{N}(d_2)] \le K_2,$$

  we have $|\phi(s)| \le s + K_2$. Thus (2.21) holds.

If we use the linear bound (2.21) in the risk-neutral formula, we obtain

$$|v(S_t,t)| \le e^{-r(T-t)}E^Q[|\phi(S_T)| \,|\, \mathcal{F}_t] \le e^{-r(T-t)}E^Q[CS_T + C|\, \mathcal{F}_t] = CS_t + e^{-r(T-t)}C.$$

Therefore for all $t < T$, v satisfies the same linear bound as ϕ:

$$|v(s,t)| \le Cs + C$$

This clearly satisfies the polynomial growth condition both as $s \to \infty$ and $s \to 0$. Thus the pricing functions v produced by the risk-neutral formula using $\phi(s)$ satisfying the linear growth condition (2.21) *do* correspond to solutions of the Black-Scholes equation described by our theorem above.

2.5 Solution Transformations

Suppose $u(x, \tau)$ solves the heat equation. It is easy to check that both

$$\tilde{u}(x,\tau) = u(x+c,\tau) \text{ and } \tilde{u}(x,\tau) = u(-x,\tau)$$

also solve the heat equation. If we start with a solution $v(s,t)$ of the Black-Scholes equation, convert it to a solution of the heat equation using (2.8), apply one of the above, and then convert back to financial variables, we discover that the following transformations convert any solution v of the Black-Scholes equation to a new solution $\tilde{v}$:

$$\tilde{v}(s,t) = v(cs,t) \quad \text{any constant } c > 0 \tag{2.22}$$

$$\tilde{v}(s,t) = s^{\kappa} v(s^{-1},t) \quad \text{where } \kappa = 1 - \frac{2r}{\sigma^2} = \sqrt{2}\sigma\mu. \tag{2.23}$$

These are verified by direct calculation; see the problems at the end of the chapter.

2.6 Some Particular Solutions

We close this chapter by exhibiting several solutions to the Black-Scholes equation. These will be components for building up the solutions of interest as pricing functions (which we will do in the next chapter). Most of them involve the standard normal distribution function, so we record a few properties of it first.

2.6.1 The Normal Distribution Function: $\mathcal{N}(x)$

The *standard normal distribution function* is the cumulative distribution function of a standard normal random variable Y:

$$\mathcal{N}(x) = P(Y \le x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}y^2}\, dy.$$

Some elementary properties are

$$0 < \mathcal{N}(x) < 1, \quad 1 - \mathcal{N}(x) = \mathcal{N}(-x).$$

The following bound on the tail of $\mathcal{N}$ will be helpful as well:

$$\mathcal{N}(x) \le \frac{1}{|x|}\frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \text{ for } x < 0. \tag{2.24}$$

Most software packages (such as Matlab) provide the standard *error function*,

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt,$$

and the *complimentary error function*,

$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2}\, dt.$$

(These are normalized so that $\operatorname{erf}(+\infty) = 1$.) There are several ways to express $\mathcal{N}(x)$ in terms of the error functions; the first of these is the most succinct:

$$\begin{aligned}\mathcal{N}(x) &= \frac{1}{2}\operatorname{erfc}(-x/\sqrt{2}) \\ &= \frac{1}{2}(1 - \operatorname{erf}(-x/\sqrt{2})) \\ &= \frac{1}{2}(1 + \operatorname{erf}(+x/\sqrt{2})).\end{aligned}$$

The following simple formulas, for a standard normal random variable Y, are behind most of the nontrivial explicit solutions to the Black-Scholes Equation. For any constants c and b,

$$E[e^{cY};\ Y < b] = e^{c^2/2}\mathcal{N}(b-c), \tag{2.25}$$

$$E[e^{cY};\ b \le Y] = e^{c^2/2}\mathcal{N}(c-b). \tag{2.26}$$

These are easy to derive, by completing the square in the exponent and making a change of variables in the appropriate integrals.

2.6.2 The Basic Solutions $v^{(0)}$–$v^{(5)}$

For reference we number the solutions $v^{(i)}$ and list them along with their terminal values. The simplest ones are

$$v^{(0)}(s,t) = e^{-r(T-t)}, \qquad v^{(0)}(s,T) = 1$$

$$v^{(1)}(s,t) = s, \qquad v^{(1)}(s,T) = s.$$

For the others we will let $d^{(i)}$ denote the expressions

$$d^{(1)}(s,t) = \frac{\ln(s) + (r + \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{(T-t)}}, \quad d^{(2)}(s,t) = \frac{\ln(s) + (r - \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{(T-t)}}.$$

(Note that $d^{(1)}(s,t) = d^{(2)}(s,t) + \sigma\sqrt{T-t}$.)

$$v^{(2)}(s,t) = e^{-r(T-t)}\mathcal{N}(-d^{(2)}(s,t)), \qquad v^{(2)}(s,T) = 1_{(0,1)}(s)$$

$$v^{(3)}(s,t) = e^{-r(T-t)}\mathcal{N}(d^{(2)}(s,t)), \qquad v^{(3)}(s,T) = 1_{[1,\infty)}(s)$$

$$v^{(4)}(s,t) = s\mathcal{N}(-d^{(1)}(s,t)), \qquad v^{(4)}(s,T) = s1_{(0,1)}(s)$$

$$v^{(5)}(s,t) = s\mathcal{N}(d^{(1)}(s,t)), \qquad v^{(5)}(s,T) = s1_{[1,\infty)}(s).$$

Observe that $v^{(2)} - v^{(5)}$ all have a discontinuity in their terminal values at $s = 1$. Thus the terminal conditions only hold for $s \neq 1$. These various solutions are not unrelated to each other. For instance $v^{(4)} = v^{(1)} - v^{(5)}$ and $v^{(2)} = v^{(0)} - v^{(3)}$. Notice that $d^{(i)}$ depend on r, σ and T as parameters, but we have not included that dependence explicitly in the notation.

All of these formulas are all derived in the same way from (2.25) and (2.26). To illustrate, we work out $v^{(5)}$. Observe that we can write

$$S_T = S_t e^{\left((r-\sigma^2/2)(T-t)+\sigma\sqrt{T-t}Y\right)},$$

where Y is a standard normal random variable independent of S_t (and the rest of $\mathcal{F}_t$). If we let $s = S_t$ then, taking advantage of the independence of Y from $\mathcal{F}_t$, we can write (2.20) as $v(S_t, t)$ where

$$v(s,t) = se^{-r(T-t)}E[\phi(se^{\left((r-\sigma^2/2)(T-t)+\sigma\sqrt{T-t}Y\right)})].$$

For $v^{(5)}$ we want to use $\phi(s) = s1_{[1,\infty)}(s)$. Now

$$se^{\left((r-\sigma^2/2)(T-t)+\sigma\sqrt{T-t}Y\right)} \ge 1$$

is equivalent to

$$Y \ge \frac{-\log(s) - (r-\sigma^2/2)(T-t)}{\sigma\sqrt{T-t}} = -d^{(2)}.$$

We can write the expression for $v^{(5)}$ in the form

$$v^{(5)} = se^{-r(T-t)} \cdot e^{(r-\sigma^2/2)(T-t)}\ E[e^{cY}; b \le Y],$$

using $b = -d^{(2)}$ and $c = \sigma\sqrt{T-t}$. Now using (2.26) we obtain

$$v^{(5)} = se^{-r(T-t)} \cdot e^{(r-\sigma^2/2)(T-t)} \cdot e^{(\sigma\sqrt{T-t})^2/2} \cdot \mathcal{N}(\sigma\sqrt{T-t} + d^{(2)}) = s\mathcal{N}(d^{(1)}).$$

In case there is any doubt, one can check directly that $v^{(5)}(s,t)$ solves the Black-Scholes equation, although the calculation is somewhat tedious.

2.7 Problems

Problem 2.A
Verify that each of the following is a solution of the heat equation.

1. $u(x, \tau) = x^3 + 6x\tau$.
2. $u(x, \tau) = x^4 + 12x^2\tau + 12\tau^2$.
3. $u(x, \tau) = e^{cx+c^2\tau}$, for any constant c.
4. $u(x, \tau) = e^{-c^2\tau} \sin(cx)$, for any constant c.
5. $u(x, \tau) = e^{c^2\tau} \sinh(cx)$, for any constant c.
6. $u(x, \tau) = k(x, \tau; y) = (4\pi\tau)^{-1/2} e^{-(x-y)^2/4\tau}$ for any y, provided $\tau > 0$.
7. $u(x, \tau) = \mathcal{N}(x/\sqrt{2\tau})$.

. . . . $\boxed{\texttt{A}}$

Problem 2.B
Suppose $v(s, t)$ solves the Black-Scholes equation in financial variables. Show that the change of variables

$$\begin{aligned} s &= e^{(\frac{\sigma^2}{2}+r)\tau+\frac{\sigma}{\sqrt{2}}x} \\ t &= T - \tau \\ v(s, t) &= e^{r\tau} u(x, \tau) \end{aligned}$$

produces a solution $u(x, \tau)$ of the heat equation. (The reason we don't use this instead of (2.8) is that the connection between s and x depends on t as well.)
. . . . $\boxed{\texttt{B}}$

Problem 2.C
Verify by direct calculation that $v^{(0)}$, $v^{(1)}$, $v^{(3)}$, and $v^{(5)}$ are solutions of the Balck-Scholes equation.
. . . . $\boxed{\texttt{CheckV}}$

Problem 2.D
Find a relationship between constants c and γ for which $v(s, t) = e^{ct}s^\gamma$ solves the Black-Sholes equation.
. . . . $\boxed{\texttt{Other}}$

Problem 2.E
For each of our solutions $v^{(i)}$ of the Black-Scholes equation, let $u^{(i)}$ be the corresponding of the heat equation, according to our change of variables (2.8). Find the initial functions $u^{(i)}(x, 0)$ for each of them.
. . . . $\boxed{\texttt{heati}}$

Problem 2.F
Show that both (2.22) and (2.23) are correct. What does (2.23) correspond to in heat variables?
. . . . $\boxed{\texttt{invtrnsf}}$

Problem 2.G
Here is a funtion m-file that does the conversion from heat variables to financial variables.

fh.m

```
function [v,s,t]=fh(u,x,tau,T)
%[v,s,t]=fh(u,x,tau,T) converts from heat variables to financial variables,
%using equations (2.7) in the text.  v, s, u, x may all be vectors (of
%compatible sizes).  It is presumed that sigma and r are already defined as
%global variables.
global sigma r
mu=(sigma/2 - r/sigma)/sqrt(2);
nu=r+mu^2;
t=T-tau;
s=exp(sigma*x/sqrt(2));
v=exp(mu*x-nu*tau).*u;
end
```

Write a counterpart, `hf.m`, that converts from financial variables to heat variables.
. . . . $\boxed{\texttt{hf}}$

Problem 2.H
Write a MATLAB m-file `vi.m` which evaluates $v^{(i)}(s, T-t)$ for $i = 0, \dots 5$. The syntax should be `vi(s,tau,i)` (with `tau` corresponding to $T - t$, and `i` being one of the values $0, \dots, 5$). You might look at `BSCall.m` in the Appendix as a example to start from.
. . . . $\boxed{\texttt{vi}}$

Problem 2.I
Write a MATLAB function (m-file) `BSconv(s,T,sL,sH,N)` that computes an approximation to the solution $v(s, 0)$ of the Black-Scholes equation for any initial function $v(s, T) = \phi(s)$ by converting the problem to heat variables and then using a trapezoidal rule approximation to the integral in the formula (2.12):

$$\begin{aligned} u(x,\tau) &= \int_{-\infty}^{\infty} k(x,\tau;y)\psi(y)\,dy \\ &\approx \int_{x^L}^{x^H} k(x,\tau;y)\psi(y)\,dy \\ &\approx \text{ trapezoidal rule calculation.} \end{aligned}$$

Suppose we have an m-file `vT.m` so that, for a vector `s`$= (s_0, \dots, s_N)$ of positive s-values, `vT(s)` will return the vector of $(\phi(s_0), \dots, \phi(s_N))$ of corresponding to the desired initial values. Here is how `BSconv(s,T,sL,sH,N)` should work:

1. Using `hf.m` (from Problem 2.G) compute the values of the heat variables x^{L}, x, and x^{H} that correspond to financial variables s^{L}, s, s^{H}.
2. Compute a vector `y`$= (y_0, \dots y_N)$ of $N + 1$ evenly spaced values between $y_0 = x^{\mathrm{L}}$ and $y_N = x^{\mathrm{L}}$. (The `linspace` command makes this easy.)
3. Convert the vector `y` from the previous step to the corresponding vector `sv` of s_i in financial variables. (`fh.m` will compute them for you.)
4. Use `vT(sv)` to produce the vector `phi` of $v(s_i, T) = \phi(s_i)$ values which you can then convert to a vector `psi` of corresponding $u(y_i, 0) = \psi(y_i)$ values (using `hf.m`). Multiplying each of these by $e^{-(y_i-x)^2/4\tau}/\sqrt{4\pi\tau}$ will give you a vector (`fv`, say) of values of the integrand: $k(x, \tau, y_i)\psi(y_i)$ for the integral you want to compute in (2.12).
5. The command `u=trapz(y,fv)` will compute the approximate value of the integral, based on the trapezoidal rule for the data you have generated.
6. Finally, use `fh.m` to convert the value `u` $\approx u(x, \tau)$ back to the corresponding approximate value for $v(s, t)$ in original variables and return this as the value of `BS_conv`.

By using `hf` and `fh` your `BSconv` does not need to know the values of r and σ; only `hf` and `fh` need them. To run `BSconv` for different ϕ all you would need to do is change `vT.m` appropriately. An m-file `vT.m` for a call option is posted, as well as an m-file `BSCall.m` that evaluates the Black-Scholes formula directly. Test your program for a few choices of σ, r, K, s and T to see if it produces the same values as `BSCall.m`. Think about and/or experiment with your choices for s^{L}, s^{H}, and N to see what is needed to get decent results. (Including a `plot(y,fv)` command may be helpful for this.)

This problem is intended as an exercise to get you used to working with MATLAB . You should review the appendix on MATLAB. Starting with pencil and paper, go over what your m-file needs to do, step by step, and how you can write the MATLAB instructions needed. (Especially if you are new to MATLAB, don't try to write the whole thing at once but test/experiment with each line of your code, one at a time, to be sure it does what you intend it to do before you go on to the next.) Try to make your m-file as efficient as possible. You should *not* need any for-loops; everything can be done with vector operations. It doesn't need more than a dozen lines! Turn in a listing of your program and the results of testing it against `BSCall`.

. . . . $\boxed{\texttt{C}}$

Chapter 3

Explicit Pricing Functions: European and Barrier Options

The explicit solution and transformation formulas for the Black-Scholes equation of the previous chapter can now be used to present explicit pricing functions for many European derivatives. Explicit formula are also possible for typical barrier options. These formulas are based on the *superposition principle.* We mentioned this in Section 2.3 in the context of the heat equation. It holds for the Black-Scholes equation as well: if we have solutions $v^{(i)}$ then a linear combination

$$v(s,t) = \sum c_i v^{(i)}(s,t), \tag{3.1}$$

where c_i are any constants, will also be a solution. This is because $\mathcal{B}v = \sum c_i \mathcal{B}v^{(i)}$ and $\partial_t v = \sum c_i \partial_t v^{(i)}$, so that

$$(\mathcal{B} + \partial_t)v = \sum c_i(\mathcal{B} + \partial_t)v^{(i)} = 0.$$

(Recall that $\mathcal{B}$ is our notation for the Black-Scholes differential operator (2.1).) Suppose we are interested in a particular terminal value function $\phi(s)$. If we can find c_i so that

$$\phi(s) = \sum c_i v^{(i)}(s,T),$$

then (3.1) will give us the desired pricing function.

3.1 Piecewise Linear Terminal Value

We can apply the above idea to get the most familiar pricing formulas.

Forwards. Here we want $\phi(s) = s - K = v^{(1)}(s,T) - Kv^{(0)}(s,T)$. So we get

$$\begin{aligned} v^{\text{Forward}}(s,t) &= v^{(1)}(s,t) - Kv^{(0)}(s,t) \\ &= s - Ke^{-r(T-t)}. \end{aligned}$$

Calls. Now we want $\phi(s) = (s-K)^+ = s1_{[K,\infty)}(s) - K1_{[K,\infty)}(s)$. We can write

$$1_{[K,\infty)}(s) = 1_{[1,\infty)}(s/K) = v^{(3)}(s/K,T),$$

and

$$s1_{[K,\infty)}(s) = K(s/K)1_{[1,\infty)}(s/K) = Kv^{(5)}(s/K,T).$$

We know from (2.22) that $v^{(i)}(s/K,t)$ are solutions, so we find

$$\begin{aligned} v^{\text{Call}}(s,t) &= Kv^{(5)}(s/K,t) - Kv^{(3)}(s/K,t) \\ &= K(s/K)\mathcal{N}(d^{(1)}(s/K,t)) - e^{-r(T-t)}K\mathcal{N}(d^{(2)}(s/K,t)) \\ &= s\mathcal{N}(d^{(1)}(s/K,t)) - e^{-r(T-t)}K\mathcal{N}(d^{(2)}(s/K,t)), \end{aligned} \tag{3.2}$$

the famous Black-Scholes formula, already mentioned in Section 1.3.

Put-Call Parity. Since $s - K = (s-K)^+ - (s-K)^-$, superpositon implies that

$$v^{\text{Call}} - v^{\text{Put}} = v^{\text{Forward}}.$$

Puts. We can either use put-call parity (above) or just work it out directly. The desired terminal value function is

$$\phi(s) = (s-K)^- = K1_{(0,K)}(s) - s1_{(0,K)}(s) = K1_{(0,1)}(s/K) - K(s/K)1_{(0,1)}(s/K).$$

So we find

$$\begin{aligned} v^{\text{Put}}(s,t) &= Kv^{(2)}(s/K,t) - Kv^{(4)}(s/K,t) \\ &= Ke^{-r(T-t)}\mathcal{N}(-d^{(2)}(s/K,t)) - K(s/K)\mathcal{N}(-d^{(1)}(s/K,t)) \\ &= Ke^{-r(T-t)}\mathcal{N}(-d^{(2)}(s/K,t)) - s\mathcal{N}(-d^{(1)}(s/K,t)). \end{aligned}$$

Digitals. These have the terminal value function $\phi(s) = 1_{[K,\infty))}(s) = 1_{[1,\infty)}(s/K) = v^{(3)}(s/K,T)$. So we have

$$\begin{aligned} v^{\text{Digital}}(s,t) &= v^{(3)}(s/K,t) \\ &= e^{-r(T-t)}\mathcal{N}(-d^{(2)}(s/K,t)). \end{aligned}$$

It should be clear from these examples that if $\phi(s)$ is any piecewise-linear function, i.e. a function whose graph consists of a finite collection of line segments $\phi = m_i s + b_i$ for $K_{i-1} \le s < K_i$, then it will be possible to produce an explicit pricing function v by assembling the appropriate combination of our $v^{(i)}$ as above. If $\phi(s)$ is continuous and piecewise-linear, then it is always possible to write

$$\phi(s) = c_0 + \sum c_i(s-K_i)^+$$

for some choice of constants c_i (some negative perhaps). This means the pricing function v can be written as a bond (c_0) and a sum of calls with different exercise prices K_i. If desired, the calls can be rewritten (using put-call parity) in terms of forwards and puts. Thus any continuous piecewise-linear $\phi(s)$ can be assembled using the standard bond, calls, puts, and forwards that are available in the market. In the finance literature such a combination of standard options to duplicate a desired $\phi(s)$ is usually called a *package*.

3.2 Barrier Options

Barrier options are different from the standard European variety in that the evolution of S_t prior to time T can trigger a change in the status of the option. Suppose we have a prescribed terminal value function $\phi(s)$, and a barrier level $\theta > 0$. We can consider either a *knock-out* or *knock-in* version of the option. The knock-out version expires early (with value 0) at the first moment $t = \mathcal{T}^\theta$ that $S_t = \theta$. So its final value is

$$V_T^{\text{k-out}} = \begin{cases} 0 & \text{if } \mathcal{T}^\theta \le T \\ \phi(S_T) & \text{if } T < \mathcal{T}^\theta. \end{cases}$$

If the initial stock price is below the barrier, $S_0 < \theta$, this is called an *up & out* option. It's a *down & out* if $\theta < S_0$. A more general version provides a *rebate* payment R in the event that $\mathcal{T}^\theta < T$:

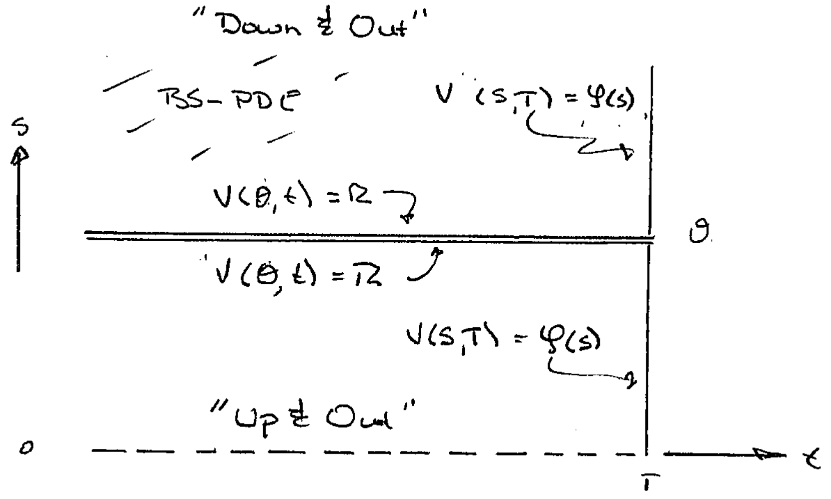
$$V_T^{\text{k-out}} = \begin{cases} Re^{r(T-\mathcal{T}^\theta)} & \text{if } \mathcal{T}^\theta \le T \\ \phi(S_T) & \text{if } T < \mathcal{T}^\theta. \end{cases}$$

(The rebate R is paid at time $\mathcal{T}^\theta$ and so appreciates in value until the final time T.) The knock-in version (with rebate R) has final value

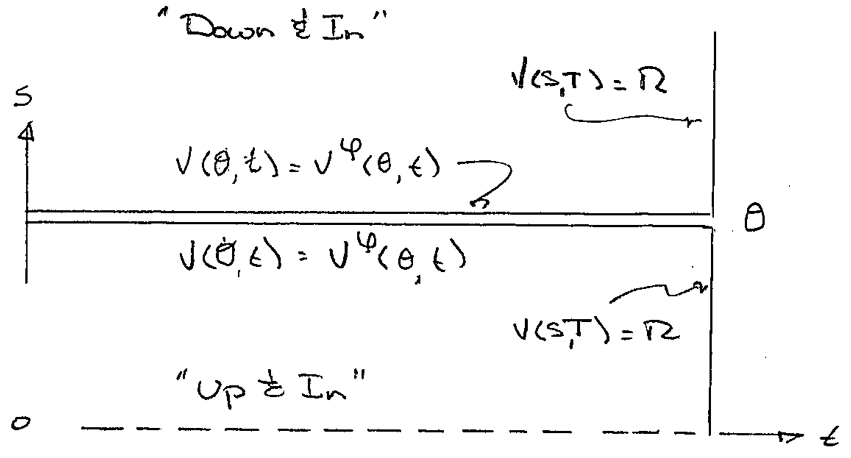
$$V_T^{\text{k-in}} = \begin{cases} \phi(S_T) & \text{if } \mathcal{T}^\theta \le T \\ Re^{r(T-\mathcal{T}^\theta)} & \text{if } T < \mathcal{T}^\theta. \end{cases}$$

We have the *down & in* case when $\theta < S_0$ and *up & in* when $S_0 < \theta$. Note that when S_t touches θ the value of the option becomes the value of the European version using terminal value ϕ, which we will write as v^ϕ: $v(\theta, t) = v^\phi(\theta, t)$ for $t < T$.

Knock-Outs



Knock-Ins


"Down & In"
$V(S,T) = R$
S
$V(\theta,t) = V^\varphi(\theta,t)$
θ
$V(\theta,t) = V^\varphi(\theta,t)$
$V(S,T) = R$
"Up & In"
0
t

Explicit formulae for the pricing function $v(s,t)$ are possible (in principle) for any of these, provided $\phi(s)$ is piecewise-lienar. (See [25, page 462] for the standard versions.) If we were interested in a more general $\phi(s)$, these would be easy to approximate by the numerical methods of the later chapters.

We will describe a general procedure for finding the barrier pricing functions, assuming we can produce the value function v^{ϕ} for a European option with a piece-wise linear terminal value. The following observation help us break down this task.

- For the case of zero rebate ($R = 0$) the sum of the knock-out and knock-in versions must agree with the standard European option with terminal value $\phi(s)$: $V_T^{\text{k-out}} + V_T^{\text{k-in}} = \phi(S_T)$. Therefore we have
$$v^{\text{k-out}}(s,t) + v^{\text{k-in}}(s,t) = v^{\phi}(s,t).$$
So if we have an explicit pricing formula $v^{\phi}(s,t)$, and if we can produce a formula for $v^{\text{k-out}}(s,t)$, then we will be able to produce an explicit formula for $v^{\text{k-in}}(s,t)$ as well.
- The value of a positive rebate $R > 0$ can be computed separately in both the knock-out and knock-in cases. (See Problems 3.E and 3.F at the end of the chapter.)

As a consequence, the task of finding a pricing formula reduces to the case of a knock-out with zero rebate. For this we now outline a general approach, based on producing the appropriate solution of the Black-Scholes equation, and illustrate it for case of a down & out call: $\phi(s) = (s-K)^+$.

We seek a solution v of the Black-Scholes equation for $\theta \le s$ and $t \le T$ with boundary and terminal boundary conditions
$$v(\theta,t) = 0, \quad v(s,T) = \phi(s).$$
We will construct it in several steps.

1. Given a piecewise linear $\phi(s)$, define the *truncated* terminal value function
$$\phi^{\theta}(s) = 1_{[\theta,\infty)}(s)\phi(s) = \begin{cases} \phi(s) & \text{if } \theta < s \\ 0 & \text{if } s \le \theta. \end{cases} \tag{3.3}$$
2. This ϕ^{θ} is also piecewise linear, though not in general continuous. Take v^{θ} to be the solution of the Black-Scholes equation in its *full* domain $s > 0$, $t \le T$, with the truncated terminal value function:
$$v^{\theta}(s,T) = \phi^{\theta}(s), \text{ for all } s > 0.$$
3. Next transform v^{θ} into a second solution using (2.22) and (2.23):
$$\tilde{v}^{\theta}(s,t) = \theta^{\kappa}(s/\theta^2)^{\kappa} v^{\theta}(\theta^2/s,t) = (s/\theta)^{\kappa} v^{\theta}(\theta^2/s,t).$$
4. Now consider
$$\begin{aligned} v(s,t) &= v^{\theta}(s,t) - \tilde{v}^{\theta}(s,t) \\ &= \boxed{v^{\theta}(s,t) - (s/\theta)^{\kappa} v^{\theta}(\theta^2/s,t)}. \end{aligned} \tag{3.4}$$
This too is a solution of the Black-Scholes equation. Observe that for $s > \theta$ we have that $\theta^2/s < \theta$, so $\tilde{v}^{\theta}(s,T) = (s/\theta)^{\kappa}\phi^{\theta}(\theta^2/s) = 0$. Consequently
$$v(s,T) = v^{\theta}(s,T) = \phi^{\theta}(s) = \phi(s) \text{ for } s > \theta.$$
Next, for $s = \theta$ we have $\theta^2/s = \theta$ and $(s/\theta)^{\kappa} = 1$. Therefore
$$v(\theta,t) = v^{\theta}(\theta,t) - \tilde{v}^{\theta}(\theta,t) = v^{\theta}(\theta,t) - v^{\theta}(\theta,t) = 0.$$
This confirms that $v(s,t)$ is the solution we seek!

This construction works in general provided we can find v^θ. For a call, $\phi(s) = (s-K)^+$ this is particularly simple if $\theta \le K$, because $\phi^\theta = \phi$. So the v^θ above is simply the pricing function for a Euorpean call with exercise price K, as given by the Black-Scholes formula (3.2):

$$v^\theta(s,t) = v^{\text{Call}}(s,t).$$

Using this in (3.4) gives the pricing function for the down and out call in the case $\theta \le K$.

When $K < \theta$ we have

$$\begin{aligned}\phi^\theta(s) &= \begin{cases} 0 & \text{for } s < \theta \\ s-K & \text{for } \theta \le s. \end{cases} \\ &= (s-K)1_{[\theta,\infty)}(s) \\ &= \theta v^{(5)}(s/\theta, T) - K v^{(3)}(s/\theta, T).\end{aligned}$$

So we have

$$v^\theta(s,t) = \theta v^{(5)}(s/\theta,t) - K v^{(3)}(s/\theta,t),$$

and use this in (3.4) for the down and out call in the case of $K < \theta$. The resulting formula is cumbersome to write because it will have four terms, but easy to evaluate with a Matlab m-file.

For up and out options, we simply replace (3.3) by

$$\phi^\theta(s) = 1_{(0,\theta]}(s)\phi(s) = \begin{cases} 0 & \text{if } \theta < s \\ \phi(s) & \text{if } s \le \theta. \end{cases} \tag{3.5}$$

and proceed as above. Equation (3.4) still gives the correct formula, but now we are only interested in $s < \theta$.

3.3 Approximations for Large and Small Price

The explicit solutions of the Black-Scholes equation allow us to deduce some results which will be useful to us in the considerations of later chapters. In Section 10.4 we will see how our barrier pricing calculations are important for lookback options. Here we will develop some results on approximate values of $v(s,t)$ for s very small and for s large. Such approximate formulas will be important for many of our finite difference calculations.

We know that if $\phi(s) = c_1 s + c_2$ *for all* $s > 0$ then $v(s,t) = c_1 s + c_2 e^{-r(T-t)}$. If $\phi(s) = c_1 s + c_2$ is only true for s close to 0, but not for all s, then we might expect $v(s,t) \approx c_1 s + c_2 e^{-r(T-t)}$ to be a good approximation for small s. This means we could use this formula for the v^{L} of Chapter 5 . Our goal in this section is to use our explicit solutions to prove a theorem to this effect. The theorem also addresses the case of large s: if $\phi(s) = c_1 s + c_2$ hold for all large s then $v(s,t) \approx c_1 s + c_2 e^{-r(T-t)}$ should be a good approximation for large s. Here is the theorem.

Theorem. *Suppose* $|\phi(s)| \le C(s+1)$ *for some constant* C*, and let* $v(s,t)$ *be the solution of the Black-Scholes equation with* $v(s,T) = \phi(s)$*. Let*

$$v^{\text{e}}(s,t) = v(s,T) - \left(c_1 s + c_2 e^{-r(T-t)}\right).$$

a) If $\phi(s) = c_1 s + c_2$ *for all* $s < K$*, then for all* $s < K$ *and all* $0 \le t < T$ *we have*

$$|v^{\text{e}}(s,t)| \le C^* \mathcal{N}(d^*(s)),$$

where $C^* = C(1+K) + |c_1|K + |c_2|$ *and* $d^*(s) = d^{(1)}(s/K, 0)$*. Moreover,* $|v^{\text{e}}(s,t)| = \text{o}(s^\gamma)$ *as* $s \to 0$ *for every* $\gamma > 0$*.*

b) if $\phi(s) = c_1 s + c_2$ *for all* $s > K$*, then for all* $K < s$ *and all* $0 \le t < T$ *we have*

$$|v^{\text{e}}(s,t)| \le C^* \mathcal{N}(-d^*(s))$$

with the same C^* as in part a), and

$$d^*(s) = \begin{cases} \frac{\log(s/K)}{\sigma\sqrt{T}} & \text{if } r - \frac{1}{2}\sigma^2 \geq 0 \\ \frac{\log(s/K)+(r-\frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}} & \text{if } r - \frac{1}{2}\sigma^2 < 0. \end{cases}$$

Moreover, $|v^{\mathrm{e}}(s,t)| = \mathrm{o}(s^{-\gamma})$ as $s \to \infty$ for every $\gamma > 0$.

We said in Section 2.4 that if $\phi(0) = \lim_{s\to 0}\phi(s)$ existed, then $\lim_{s\to 0} v(s,t) = e^{-r(T-t)}\phi(0)$. This is the content of the following corollary.

Corollary. *Suppose $|\phi(s)| \leq C(s+1)$ for some constant C. If $\phi(0) = \lim_{s\to 0}\phi(s)$ exists, then for every $t < T$*

$$\lim_{s\to 0} v(s,t) = e^{-r(T-t)}\phi(0).$$

Let's consider the theorem in the first case, $\phi(s) = c_1 s + c_2$ for $s < K$. The error

$$v^e(s,t) = v(s,t) - [c_1 s + c_2 e^{-r(T-t)}]$$

is just the pricing function for terminal value

$$\phi^e(s) = \phi(s) - (c_1 s + c_2) = \begin{cases} \phi(s) - (c_1 s + c_2) & \text{for } K \leq s \\ 0 & \text{for } s < K. \end{cases}$$

For $K \leq s$ we have $1 \leq s/K$ so we can write

$$\begin{aligned} |\phi^e(s)| &\leq |\phi(s)| + |c_1 s + c_2| \\ &\leq C(s+1) + |c_1|s + |c_2| \\ &\leq C(s + \frac{s}{K}) + |c_1|s + |c_2|\frac{s}{K} \\ &= \frac{C^*}{K}s, \quad \text{where } C^* \text{ is the value in the theorem.} \end{aligned}$$

So if $v^b(s,t)$ is the solution with terminal value

$$v^b(s,T) = \begin{cases} \frac{C^*}{K}s & \text{for } K \leq s \\ 0 & \text{for } s < K, \end{cases}$$

then $|v^e| \leq v^b$. But $v^b(s,T) = \frac{C^*}{K}Kv^{(5)}(s/K,T)$, so we have

$$|v^e(s,t)| \leq v^b(s,t) = C^* v^{(5)}(s/K,t) = C^*\frac{s}{K}\mathcal{N}(d^{(1)}(s/K,t)).$$

For $s < K$ we have $\frac{s}{K} < 1$. Since $\log(s/K) < 0$ and $r + \frac{1}{2}\sigma^2 > 0$ it follows that

$$d^{(1)}(s/K,t) = \frac{\ln(s/K) + (r+\frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{(T-t)}} \leq \frac{\ln(s/K) + (r+\frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}} \doteq d^*(s).$$

Since $\mathcal{N}$ is monotone increasing, the bound

$$|v^e(s,t)| \leq C^*\mathcal{N}(d^*(s)), \text{ for } s < K$$

follows.

To finish part a) we need to show that $\mathcal{N}(d^*(s)) = \mathrm{o}(s^\gamma)$ as $s \to 0$. As $s \to 0$ we have $d^*(s) \to -\infty$. The bound (2.24) implies

$$\mathcal{N}(d^{(1)}) \leq \frac{1}{\sqrt{2\pi}|d^*|}e^{-d^{*2}/2}.$$

Since $\frac{1}{\sqrt{2\pi}|d^*|} \to 0$, we just need to show that, as $s \to 0$, $e^{-d^{*2}/2}$ is bounded by a constant time s^γ. If we define ρ by

$$\rho = e^{(r+\frac{1}{2}\sigma^2)T},$$

then we have $d^*(s) = \frac{1}{\sigma\sqrt{T}} \log(\frac{\rho}{K}s)$, and so

$$-(d^*)^2 = -\left(\frac{1}{\sigma\sqrt{T}} \log(\frac{\rho}{K}s)\right)^2.$$

We can write

$$\begin{aligned} e^{-(d^*)^2/2} &= \left(e^{\log(\frac{\rho}{K}s)}\right)^{-\frac{1}{\sigma^2 T}\log(\frac{\rho}{K}s)} \\ &= \left(\frac{\rho}{K}s\right)^{-\frac{1}{\sigma^2 T}\log(\frac{\rho}{K}s)} \end{aligned}$$

For any $\gamma > 0$ we will have

$$\gamma < -\frac{1}{\sigma^2 T}\log(\frac{\rho}{K}s)$$

for s sufficiently close to 0, and therefore

$$e^{-(d^{(1)})^2/2} \le \left(\frac{\rho}{K}s\right)^\gamma$$

for all s sufficiently small. This confirms that $\mathcal{N}(d^*(s)) = \mathrm{o}(s^\gamma)$ as $s \to 0$.

Part b) of the theorem is proved in a similar manner. Now v^e is the solution with terminal value

$$v^e(s,T) = \begin{cases} 0 & \text{for } s > K \\ \phi(s) - (c_1 s + c_2) & \text{for } s \le K. \end{cases}.$$

For $s \le K$ we have $|\phi(s) - (c_1 s + c_2)| \le C^*$ (with no factor of s). So $|v^e| \le v^b$ where

$$v^b(s,T) = \begin{cases} 0 & \text{for } s > K \\ C^* & \text{for } s \le K. \end{cases} = C^* v^{(2)}(s/K,T).$$

So using the formula for $v^{(2)}$ we have

$$|v^e(s,t)| \le C^* e^{-r(T-t)}\mathcal{N}(-d^{(2)}(s/K,t)) \le C^*\mathcal{N}(-d^{(2)}(s/K,t)).$$

For $K < s$, we bound $d^{(2)}$ below independently of $0 \le t < T$, but it depends on the sign of $r - \frac{1}{2}\sigma^2$:

$$d^{(2)}(s/K,t) = \frac{\log(s/K) + (r - \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}} \ge d^*(s) \doteq \begin{cases} \frac{\log(s/K)}{\sigma\sqrt{T}} & \text{if } r - \frac{1}{2}\sigma^2 \ge 0 \\ \frac{\log(s/K) + (r-\frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}} & \text{if } r - \frac{1}{2}\sigma^2 < 0. \end{cases}$$

The monotonicity of $\mathcal{N}$ gives us the bound

$$|v^{\mathrm{e}}(s,t)| \le C^*\mathcal{N}(-d^*(s)), \text{ for } K < s$$

and we can argue that $\mathcal{N}(-d^*(s)) = \mathrm{o}(s^{-\gamma})$ for any $\gamma > 0$ in a manner similar to the above.

Lastly, consider the corollary. If $\phi(s)$ were linear on some interval $(0,K)$, then we would have $\phi(s) = c_1 s + \phi(0)$ for $s < K$. We could just use $c_2 = \phi(0)$ in the first part of the theorem. However, as stated the corollary does not assume ϕ is linear near the origin, only that the limit as $s \to 0$ exists. Consider

$$v^e(s,t) = v(s,t) - \phi(0)e^{-r(T-t)}.$$

We want to show that $\lim_{s\to 0} v^e(s,t) = 0$. Since $\phi(s) \to \phi(0)$, given any $\epsilon > 0$ there exists $\delta > 0$ so that $|\phi(s) - \phi(0)| < \epsilon$ for $0 < s < \delta$. It follows that $|v^e| \le v^b$ where v^b is the solution with

$$v^b(s,T) = \begin{cases} |\phi(s) - \phi(0)| & \text{for } \delta \le s \\ \epsilon & \text{for } s < \delta. \end{cases}$$

The theorem (part a) with $K = \delta,\ c_1 = 0,\ c_2 = \epsilon$) applies to v^b: for any $\gamma > 0$,

$$v^b(s,t) = \epsilon e^{-r(T-t)} + \mathrm{o}(s^\gamma) \text{ as } s \to 0.$$

Therefore

$$\lim_{s\to 0} |v(s,t) - \phi(0)e^{-r(T-t)}| \le \lim_{s\to 0} v^b(s,t) \le \epsilon e^{-r(T-t)}.$$

Since $\epsilon > 0$ is arbitrary, this limit in fact is 0, proving the corollary.

3.4 Problems

Problem 3.A
A *capped call* is an option with a terminal value

$$\phi(s) = (\min(s,L) - K)^+ = \begin{cases} 0 & \text{if } s < K \\ s - K & \text{if } K \le s \le L \\ L - K & \text{if } L < s. \end{cases}$$

Find an explicit pricing function for this option. (Assume $K < L$.)
.... $\boxed{\texttt{CapCall}}$

Problem 3.B
Pick parameters and a value $t < T$, and plot the values of both $v^{\mathrm{Put}}(s,t)$ and $(s-K)^-$ as functions of s. Observe that $v^{\mathrm{Put}}(s,t) < (s-K)^-$ for small values of s. This shows that it would be advantageous to exercise the put prior to the exercise time T. For European options, this is not allowed, but for American options (Chapter 8) it will be.
.... $\boxed{\texttt{AmEarly}}$

Problem 3.C
Write a function m-file `DnOutPut.m` so that `DnOutPut(s,tau,theta,K)` willl return the value of a down & out put for $S_t = s$ and $\tau = T - t$. It should correctly handle both cases $K \le \theta$ and $\theta < K$. (You will need to do some pencil and paper work before you are ready to write the code for the latter.) You can take advantage of the function `vi.m` that you wrote for Problem 2.H. You may find MATLAB's subfunction feature useful. If, in the same file `DnOutPut.m` you follow the function definition for `DnOutPut` with the definition of a second function `vTheta`, then `DnOutPut` will be able to just use (3.4) to do its evaluation. (But `vTheta` will be accessible only to `DnOutPut`; it won't be accessible from the command line or functions defined in other files.) So if you just get `vTheta` right, then your `DnOutPut` should execute correctly.
.... $\boxed{\texttt{DOP}}$

Problem 3.D
Show (simply by checking the terminal and boundary conditions) that (3.4) will also produce the formula for the value of an up & out option with terminal value $\phi(s)$ if v^θ solves the standard Black-Scholes P.D.E. with $v^\theta(s,T) = \phi^\theta(s)$ as defined in (3.5). Use this to produce the value formulas for an up & out put option in each of the two cases $K \le \theta$ and $K > \theta$.
.... $\boxed{\texttt{R}}$

Problem 3.E
Suppose $v^{\text{k-in}}(s,t)$ is the value of a knock-in option for some terminal value function $\phi(s)$. If we now add to the contract the provision that if the option expires before the price hits the knock-in barrier ($T < \mathcal{T}^\theta$) then the bearer of the contract receives a rebate payment of R (paid at time $\mathcal{T}^\theta$). This revision to the contract should increase its value by an amount $v^{\text{R}}(s,t)$ where v^{R} solves the Black-Scholes equation (on the appropriate side of the barrier $s = \theta$) with

$$v^{\text{R}}(\theta,t) = 0, \quad \text{and } v^{\text{R}}(s,T) = R.$$

Explain why this should be so. Find formulas for v^{R} for both the down & in and up & in cases. (Your can leave your formulas expressed in terms of the $v^{(i)}$ of Chapter 2.)

.... $\boxed{\texttt{S}}$

Problem 3.F
Now we want to do the same thing as preceeding problem for the knock-out case, adding the rebate of R in the event that the knock-out barrier *is* hit prior to $t = T$. Explain why the the contract value should now increase by an amount v^{R} which solves the Black-Scholes equation with

$$v^{\text{R}}(\theta,t) = R, \quad \text{and } v^{\text{R}}(s,T) = 0.$$

This time, to find a formula first write

$$v^{\text{R}}(s,t) = \frac{R}{\theta}s + w(s,t).$$

Note that $v(s,t) = \frac{R}{\theta}s$ is itself a solution of the Black-Scholes equation. What boundary and terminal conditions should $w(s,t)$ satisfy? Find formulas for v^{R} for both the down & out and up & out cases.

.... $\boxed{\texttt{T}}$

Problem 3.G
Consider the compound option which is a put (to be exercised at $t = T_1$ with an exercise price of K_1) on a call (with expiry time $T_2 > T_1$ and exercise price K_2). Thus at time T_1 the owner of this contract can, at their discretion, sell for \$$K_1$ one unit of the call option (which will expire $T_2 - T_1$ units of time after T_1). To determine the value of this option at a time t prior to T_1 we would need to find the appropriate solution $v(s,t)$ of the Black-Scholes equation. What would the appropriate terminal value function $v(s,T_1) = \phi(s)$ be? In order to solve the PDE on a strip $0 = s^{\text{L}} < s < s^{\text{H}}$, for s^{H} sufficiently large, what approximate formulas for $v^{\text{L}}(t)$ and $v^{\text{H}}(t)$ should we use?

.... $\boxed{\texttt{E}}$

Chapter 4

Introduction to Finite Difference Methods: The Heat Equation

This chapter introduces the use of finite difference methods to approximate solutions to partial differential equations. We introduce the technique by considering the heat equation $u_{xx} - u_\tau = 0$ with a prescribed initial function $u(x,0) = \psi(x)$. In the next chapter we will develope the technique for the Black-Scholes equation. The basic idea is to place a "grid" of *nodes* of the half-plane ($\tau > 0$), evenly spaced at intervals of $\Delta\tau$ in the τ-direction and Δx in the x-direction. Our notation will be

$$x_n = x_0 + n\Delta x, \quad \tau_m = m\Delta\tau.$$

The plan is to develop a scheme for computing values u_n^m which approximate the values of u at the nodes:

$$u_n^m \approx u(x_n, \tau_m).$$

To compute the u_n^m we will replace the partial derivatives in the heat equation by approximations to them expressed in terms of the u_n^m. This will lead to equations for the u_n^m, which we hope to solve. The details of *how* we do this will have a big influence on the effectiveness of the resulting procedure.

First consider u_τ.

$$u_\tau(x,\tau) = \lim_{\Delta\tau\to 0^+} \frac{u(x,\tau+\Delta\tau) - u(x,\tau)}{\Delta\tau},$$

which suggests the *forward difference approximation*

$$u_\tau(x_n,\tau_m) \approx \frac{u_n^{m+1} - u_n^m}{\Delta\tau}. \tag{4.1}$$

But other approximations are also possible. From

$$u_\tau(x,\tau) = \lim_{\Delta\tau\to 0^+} \frac{u(x,\tau) - u(x,\tau-\Delta\tau)}{\Delta\tau}$$

we get the *backward difference approximation*

$$u_\tau(x_n,\tau_m) \approx \frac{u_n^m - u_n^{m-1}}{\Delta\tau}, \tag{4.2}$$

or from

$$u_\tau(x,\tau) = \lim_{\Delta\tau\to 0} \frac{u(x,\tau+\Delta\tau) - u(x,\tau-\Delta\tau)}{2\Delta\tau}$$

we get the *central difference approximation*

$$u_\tau(x_n,\tau_m) \approx \frac{u_n^{m+1} - u_n^{m-1}}{2\Delta\tau}. \tag{4.3}$$

Since these are only approximations, when we replace u_τ by one of these approximations we are introducing some error into the resulting equation. Typically the size of this error is described in terms of its *order* as a power of $\Delta\tau$. Suppose we consider the second order Tayor polynomial with remainder

$$u(x,\tau+h) = u(x,t) + hu_\tau(x,\tau) + \frac{1}{2}h^2 u_{\tau\tau}(x,\tau) + \frac{1}{6}h^3 u_{\tau\tau\tau}(x,\tau+\bar{h}).$$

(The last term is the remainder formula; $\bar{h}$ is some value between 0 and h.) Using $h=\Delta\tau$ we find that

$$\begin{aligned} u_\tau(x,\tau) &= \frac{u(x,\tau+\Delta\tau)-u(x,\tau)}{\Delta\tau} - \Delta\tau\left[\frac{1}{2}u_{\tau\tau}(x,\tau) + \frac{1}{6}\Delta\tau u_{\tau\tau\tau}(x,\tau+\bar{h})\right] \\ &= \frac{u(x,\tau+\Delta\tau)-u(x,\tau)}{\Delta\tau} + \mathcal{O}(\Delta\tau). \end{aligned}$$

The notation $\mathcal{O}(\Delta\tau)$ is used to refer to an expression of the form

$$\Delta\tau \cdot (\text{ something which remains bounded as } \Delta\tau \to 0).$$

We don't care about the "something;" the point is that the forward difference approximation is *first order* in $\Delta\tau$. If we repeat this with $h=-\Delta\tau$ we find that the backward difference approximation is also first order:

$$u_\tau(x,\tau) = \frac{u(x,\tau)-u(x,\tau-\Delta\tau)}{\Delta\tau} + \mathcal{O}(\Delta\tau).$$

For the central difference approximation, write the Taylor approximations for both $h=\pm\Delta\tau$

$$\begin{aligned} u(x,\tau+\Delta\tau) &= u(x,t) + \Delta\tau u_\tau(x,\tau) + \frac{1}{2}\Delta\tau^2 u_{\tau\tau}(x,\tau) + \frac{1}{6}\Delta\tau^3 u_{\tau\tau\tau}(x,\tau+\bar{h}). \\ u(x,\tau-\Delta\tau) &= u(x,t) - \Delta\tau u_\tau(x,\tau) + \frac{1}{2}\Delta\tau^2 u_{\tau\tau}(x,\tau) - \frac{1}{6}\Delta\tau^3 u_{\tau\tau\tau}(x,\tau+\hat{h}). \end{aligned}$$

and subtract, to find

$$u(x,\tau+\Delta\tau) - u(x\tau-\Delta\tau) = 2\Delta\tau u_\tau(x,\tau) + 0\Delta\tau^2 + \Delta\tau^3\left[\cdots\right].$$

This gives us

$$\frac{u(x,\tau+\Delta\tau)-u(x,\tau-\Delta\tau)}{2\Delta\tau} = u_\tau(x,\tau) + \mathcal{O}(\Delta\tau^2).$$

In other words the central difference approximation is *second order*, which means that it will be considerably more accurate for small $\Delta\tau$. On the other hand to use it to approximate $u_\tau(x_n,\tau_m)$ involves both u_n^{m+1} and u_n^{m-1}, which complicates the resulting algebraic equations. Higher order approximations for $u_\tau(x_n,\tau_m)$ can be given which involve more than two different m-values.

Similar approximations can be used for u_x. For u_{xx} we typically use the *second central difference approximation*

$$u_{xx}(x,\tau) = \lim_{h\to 0}\frac{u(x+h,\tau) - 2u(x,\tau) + u(x-h,\tau)}{h^2}. \tag{4.4}$$

If we use the third order Taylor polynomial

$$u(x\pm h,\tau) = u(x,\tau) \pm hu_x(x,\tau) + \frac{1}{2}(\pm h)^2 u_{xx}(x,\tau) + \frac{1}{6}(\pm h)^3 u_{xxx}(x,\tau) + \frac{1}{24}(\pm h)^4 u_{xxxx}(x+\bar{h},\tau)$$

with $h=\Delta\tau$, we find that

$$u(x,\tau+\Delta\tau) - 2u(x,\tau) + u(x,\tau-\Delta\tau) = \Delta\tau^2 u_{xx}(x,\tau) + \frac{1}{12}\Delta\tau^4 u_{xxxx}(\cdots).$$

We see that the *symmetric central difference approximation*

$$\frac{u_{n+1}^m - 2u_n^m + u_{n-1}^m}{\Delta x^2} = u_{xx}(x_n,\tau_m) + \mathcal{O}(\Delta x^2)$$

is second order. There are other finite difference approximations to $u_{xx}(x_n,\tau_m)$, but this will serve us well enough for now.

4.1 The Explicit Finite Difference Method

To proceed, let's use the second central difference approximation (4.4) for u_{xx} and the forward difference approximation (4.1) for u_τ. The heat equation evaluated at the node $(x,\tau)=(x_n,\tau_m)$ says that

$$\frac{u_n^{m+1}-u_n^m}{\Delta\tau}+\mathcal{O}(\Delta\tau)=u_\tau(x_n,\tau_m)=u_{xx}(x_n,\tau_m)=\frac{u_{n+1}^m-2u_n^m+u_{n-1}^m}{\Delta x^2}+\mathcal{O}(\Delta x^2).$$

Now we simply ignore the $\mathcal{O}(\Delta\tau)$ and $\mathcal{O}(\Delta x^2)$ terms, and consider the equation

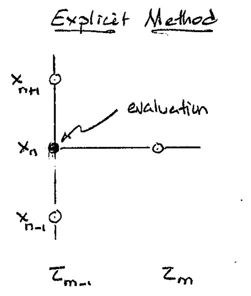
$$\frac{u_n^{m+1}-u_n^m}{\Delta\tau}=\frac{u_{n+1}^m-2u_n^m+u_{n-1}^m}{\Delta x^2}.$$

This involves the values of u at four nodes in our grid. Rearranging, it becomes

$$u_n^{m+1}=\alpha u_{n+1}^m+(1-2\alpha)u_n^m+\alpha u_{n-1}^m, \tag{4.5}$$

where α refers to the value

$$\alpha=\frac{\Delta\tau}{\Delta x^2}.$$



Equation (4.5) is called the *explicit finite difference method* for the heat equation. It tells us how to calculate the values of u at the τ_{m+1} nodes in terms of the values at the τ_m nodes. If we use our initial function to get us started

$$u_n^0=\psi(x_n),$$

we can then use (4.5) to "march" our calculations through τ_1, τ_2, ... until we get to the values of u_n^m for whatever τ_m we are interested in. *But remember* that (4.5) is only an approximation to the actual heat equation. Every time we use (4.5) we are introducing additional errors of magnitude $\mathcal{O}(\Delta\tau)+\mathcal{O}(\Delta x^2)$. So even though $u_n^0=u(x_n,0)$ is exact, all the subsequent values are only approximations:

$$u_n^m\approx u(x_n,\tau_m).$$

We should recognize that the cumulative effect of these errors is a concern. It is conceivable that many applications of (4.5) might lead to a poor approximation in the end.

In spite of that concern, the simplicity of (4.5) is so appealing that it seems worth trying before we explore the error issue any further. We want to get MATLAB to do the calculations for us. There are a couple issues to deal with before we can give it a test run. One is that we can't really deal with all infinitely many values of n; we must limit ourselves to n in some finite range $0\le n\le N+1$, which corresponds to an interval of x-values:

$$x^{\mathrm{L}}=x_0\le x\le x_{N+1}=x^{\mathrm{H}}.$$

Now notice that (4.5) will tell us values for u_n^{m+1} for $1\le n\le N$, but not for $n=0,N+1$. So we need to do something different to produce the values

$$u_0^{m+1}\approx u(x^{\mathrm{L}},\tau_{m+1}),\quad u_{N+1}^{m+1}\approx u(x^{\mathrm{H}},\tau_{m+1}).$$

If we know or can somehow approximate $u(x^{\mathrm{L}},\tau)$ and $u(x^{\mathrm{H}},\tau)$ then we will be in good shape. Since our limitation to $0\le n\le N+1$ means we are only considering the heat equation in the strip $x^{\mathrm{L}}\le x\le x^{\mathrm{H}}$, $0\le\tau$, it is no suprize that we find the need for boundary values, as we saw in Section 2.3.1.

4.1.1 Approximate Boundary Conditions in Heat Variables

We saw in Section 3.3 how we could approximate pricing functions $v(s,t)$ for small or large s:

$$v(s,t) = v^{\text{approx}}(s,t) + \mathcal{O}(s^\gamma) \quad \text{or} \quad + \mathcal{O}(1/s^\gamma).$$

We should pause to think about how accurate such approximations will be when converted to heat variables. The concern is the factor of $e^{-\mu x}$ in the conversion formula from (2.8):

$$u(x,\tau) = e^{-\mu x + \nu\tau} v(s,t).$$

The sign of μ depends on r and σ, but that factor will magnify errors for for either positive or negative x. Is it going to ruin our accuracy of $\mathcal{O}(s^\gamma)$ as $s \to 0$, or of $\mathcal{O}(1/s^\gamma)$ as $s \to \infty$, when v^{approx} is converted to an approximation for u? Observe that

$$e^{-\mu x} s^\gamma = e^{-\mu x} e^{\gamma\sigma x/\sqrt{2}}.$$

we can make $c = \gamma\frac{\sigma}{\sqrt{2}} - \mu$ as large (positive) as we like by our choice of γ, so that

$$e^{-\mu x} s^\gamma = e^{cx},$$

which $\to 0$ exponentially fast as $s \to 0$ and $x \to -\infty$. For $s \to +\infty$ we have

$$e^{-\mu x} s^{-\gamma} = e^{-(\gamma\frac{\sigma}{\sqrt{2}} + \mu)x},$$

The exponential constant will be negative for large γ, so this too decays exponentially as $x \to \infty$. Thus the approximate formulas from Section 3.3, when converted to heat variables, will differ from the true values by an error that vanishes exponentially as $x \to \pm\infty$. So provided $[x^{\text{L}}, x^{\text{H}}]$ is large enough, we can use the formulas from Section 3.3 in conjunction with (2.8) to obtain formulas for $u^{\text{L}}(\tau)$ and $u^{\text{H}}(\tau)$ to use as approximate boundary values:

$$u_0^m = u^{\text{L}}(\tau_m), \quad u_{N+1}^m = u^{\text{H}}(\tau_m).$$

4.2 Implementing the Explicit Method

Given formulas for ψ, u^{L} and u^{H} we can write a MATLAB script `efd.m` (listed below) to carry out the finite difference calculation. Here is how it works.

Values for $x^{\text{L}}, x^{\text{H}}, N, M$ are to be specified (in MATLAB variables `xL,xH,N,M`), as well as a final time T (`T` in MATLAB). We also will need function m-files to compute the following.

- `u0(x)`$= \psi(x)$, the initial data,
- `uL(xL,tau)`$= u^{\text{L}}(\tau)$ and `uH(xH,tau)`$= u^{\text{H}}(\tau)$, the boundary conditions.

(We want `uL` to take `xL` as an argument so that if we decide to change the value we use for x^{L} we don't have to rewrite the m-files. Likewise for `uH` and `xH`.) We determine the values

$$\begin{aligned}
&\alpha = \Delta\tau/\Delta x^2, \\
&\Delta x = \frac{x^{\text{H}} - x^{\text{L}}}{N+1} \text{ and } \Delta\tau = \frac{T}{M}, \\
&x_0 = x^{\text{L}}, \quad x_1 = x^{\text{L}} + \Delta x, \quad \ldots, \quad x_N = x^{\text{L}} + N\Delta x, \quad x_{N+1} = x^{\text{H}}. \\
&\tau_0 = 0, \quad \ldots \quad \tau_M = \tau.
\end{aligned}$$

In the script, `dx` $= \Delta x$, `dt` $= \Delta\tau$, and `alpha` $= \alpha$. The `linspace` command produces an array `x` which contains

$$\mathtt{x} = [x_0, x_1, \ldots, x_{N+1}].$$

The full array values of τ_i are not calculated in advance because only the one value of τ_i at at time (`tau` in the code) is needed in the calculation.

The MATLAB variable `u` will be an N-dimensional array which will hold the values

$$\mathtt{u} = [u_1^m, u_2^m, \ldots, u_N^m],$$

for the successive values $m = 0, 1, 2, \ldots M$ as the calculation proceeds. The command `u=u0(x(2:N+1))` gives it the values for $m = 0$: `u(i)`$= \psi(x_i)$. (Note that this assumes that `u0` is written so that it will accept a vector of input values and return the appropriate vector of ψ values.) Observe that we generated the full vector of x-values, including `x(1)`$= x_0 = x^{\mathrm{L}}$ and `x(N+2)`$= x_{N+1} = x^{\mathrm{H}}$. But we left off the first and last `x` values when generating the initial vector `u`. So after that command `u` will contain the values u_n^0 for $n = 1, \ldots N$. `tau` will hold the value of τ_m corresponding to the current contents of `u`, so we initialize `tau` $= 0$.

Having initialized all the variables, we want to iterate (M times) the explicit finite difference formula,

$$u_n^{m+1} = \alpha u_{n-1}^m + (1-2\alpha)u_n^m + \alpha u_{n+1}^m, \quad \text{for } 1 \le n \le N.$$

We can write this as a single vector operation, using u^{L} and u^{H} to give the values for u_0^m and u_{N+1}^m.

$$\begin{bmatrix} u_1^{m+1} \\ u_2^{m+1} \\ \vdots \\ u_{N-1}^{m+1} \\ u_N^{m+1} \end{bmatrix} = \alpha \begin{bmatrix} u^{\mathrm{L}}(x^{\mathrm{L}}, \tau_m) \\ u_1^m \\ \vdots \\ u_{N-2}^m \\ u_{N-1}^m \end{bmatrix} + (1-2\alpha) \begin{bmatrix} u_1^m \\ u_2^m \\ \vdots \\ u_{N-1}^m \\ u_N^m \end{bmatrix} + \alpha \begin{bmatrix} u_2^m \\ u_3^m \\ \vdots \\ u_N^m \\ u^{\mathrm{H}}(x^{\mathrm{H}}, s) \end{bmatrix}.$$

In thie script, this is the fourth line from the bottom. The occurences of `u` on the right are the values for the "old" `tau` $= \tau_m$. The `u` on the left will then have the values for the "new" `tau` $= \tau_{m+1}$, which we reset after updating `u`. This explains the update loop. When the `for`-loop is finished `u` contains the values corresponding to `tau`$= \tau_M = T$. The last line of the script puts the two boundary values back onto `u`, so that its entries correspond to the full list of x_n in `x`.

efd.m

```
% Script for explicit finite difference method.
% Initial values are to be provided by u0(x).
% Boundary values are to be provided by uL(xL,tau), uH(xH,tau).
%
% Before running the script, m-files for u0, uL, and uH need to
% be provided and values need to be given to the following variables:
% xL = lower boundary
% xH = upper boundary
% N = number of spatial intervals (less 1)
% M = number of time steps
% T = final time
%
dx=(xH-xL)/(N+1);
dt=T/M;
alpha=dt/(dx)^2;
x=linspace(xL,xH,N+2);
u=u0(x(2:N+1));
tau=0;
%
% Update Loop:
for m=1:M
     u=alpha*[u(2:N),uH(xH,tau)]+(1-2*alpha)*u+alpha*[uL(xL,tau),u(1:N-1)];
     tau=m*dt;
end
%
u=[uL(xL,T),u,uH(xH,T)];
```

4.3 Testing the Explicit Method

We want to test the explicit method, as implemented in our script `efd.m`, on a problem for which we know what the values of the true solution $u(x,\tau)$ actually are, so that we can compare our computed results to the values they are supposed to be approximating. We will use the values for a call option, but in their heat variables form. The initial values are $v(s,0) = (s-K)^+$, and we use

$$v(s,t) \approx 0 \text{ for } s \approx 0, \quad v(s,t) \approx s - Ke^{-r(T-t)} \text{ for } s \approx \infty.$$

These give formulas for $\phi(s)$, $v^{\mathrm{L}}(s,t)$ and $v^{\mathrm{H}}(s,t)$ (in financial variables) and are converted to heat variables to give the values returned by `u0.m`, `uL.m`, and `uH.m`. For instance here are the m-files we use for `uH.m` and `vH.m`.

uH.m

```
function u=uH(x,tau)
%uH(x,tau) gives the upper boundary values for finite difference
%calculations in heat variables.  The values are obtained simply by
%evaluating vH in financial variables and converting to heat variables.
[junk,s]=fh(0,x,tau,tau);
v=vH(s,0,tau);
u=hf(v,s,0,tau);
end
```

vH.m

```
function v = vH(s,t,T)
%vL(s,t,T) returns the approximate upper boundary value at (s,t)
%for terminal time T.
global r K
v=s-K*exp(-r*(T-t)); %for a call
end
```

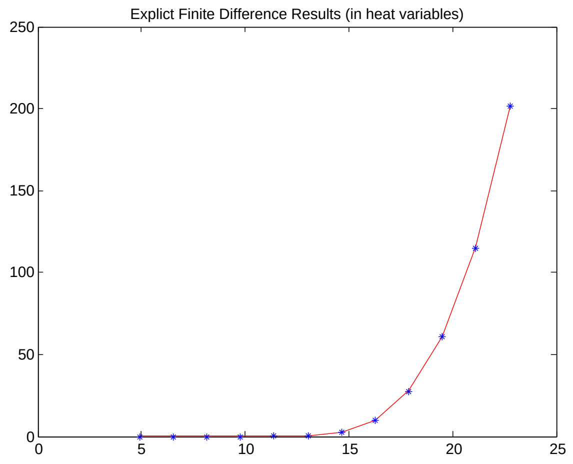
We use values of the stock price s in a specified range $s^{\mathrm{L}} \leq s \leq s^{\mathrm{H}}$, and then do the explicit finite difference calculation on the corresponding range of x-values: $x^{\mathrm{L}} \leq x \leq x^{\mathrm{H}}$ using the script `efd.m` above. The resulting values u_n^M should approximate the true values $u(x_n, T)$. We get the true values from `BS_Call.m` and conversion to heat variables. Here is the script which carries out the evaluation and presents the results.

rune.m

```
%Script for test calculations using efd to calculate call values.
%Initial and boundary values provided by u0(*), uL(*) and uh(*).
%xL and xH are converted form sL and sH.
[junk,x]=hf(0,[sL,sH],0,0); % Convert [sL,sH] to xL, xH.
xL=x(1);
xH=x(2);
% Run explicit method
efd
% Compute true values: BSCall and convert to heat variables
[junk,s]=fh(0,x,0,0);
true=hf(BSCall(s,T,K),s,0,T);
```

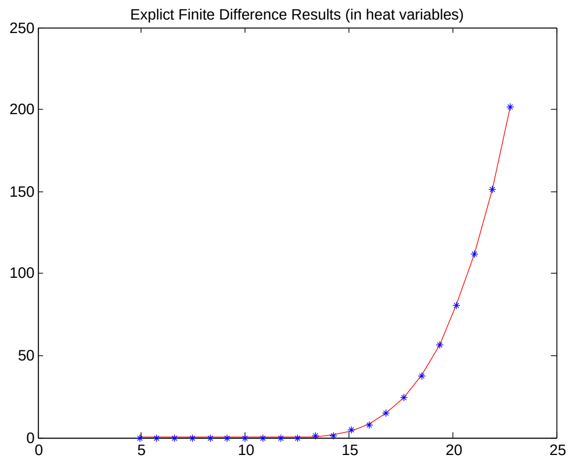
```
% Display results
plot(x,true,'r')
hold on
plot(x,u,'*')
% Print summary
title('Explict Finite Difference Results (in heat variables)')
hold off
fprintf('\n[xL,xH]=[%g , %g]\n',xL,xH)
fprintf('dx=%g, using N+1=%g x-intervals.\n',dx,N+1)
fprintf('T=%g, dt=%g, using M=%g time steps.\n',T,dt,M)
fprintf('alpha=%g\n',alpha)
fprintf('Minimal M for stability=%g\n',ceil(2*T/(dx^2)))
fprintf('Maximum error=%g\n\r',max(abs(u-true)))
```

We now present the results from this calculation using the parameter values $K = 10$, $r = .05$, $\sigma = .2$, $T = 2$, with $s^{\mathrm{L}} = 2$ and $s^{\mathrm{H}} = 25$. The computed values are plotted (with asterisks) along the with the true values (the solid lines) for comparison using various values of N and M.



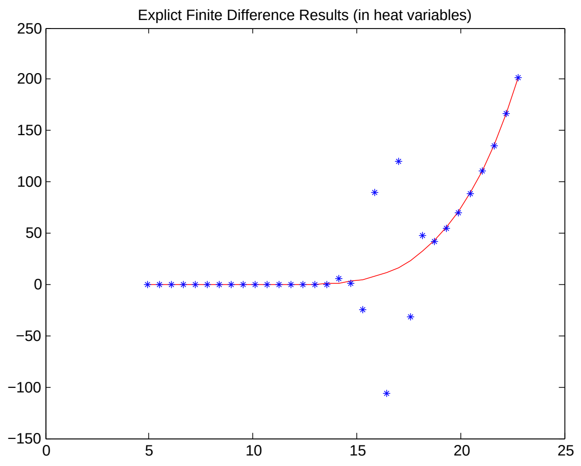
```
[xL,xH]=[4.90129 , 22.7609]
dx=1.6236, using N+1=11 x-intervals.
T=2, dt=0.5, using M=4 time steps.
alpha=0.189676
Minimal M for stability=2
Maximum error=0.338661
```

We have used $N = 10$ and $M = 4$. The results here are not bad, considering we only used 4 time steps. To get better accuracy we can try increasing N.



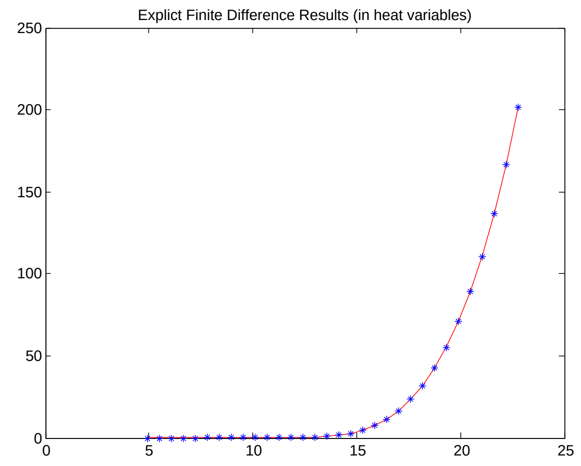
```
[xL,xH]=[4.90129 , 22.7609]
dx=0.850457, using N+1=21 x-intervals.
T=2, dt=0.5, using M=4 time steps.
alpha=0.691298
Minimal M for stability=6
Maximum error=1.02033
```

Increasing to $N = 20$ did not improve the accuray s we expected. Rather the error has increased! Let's try increasing N some more.



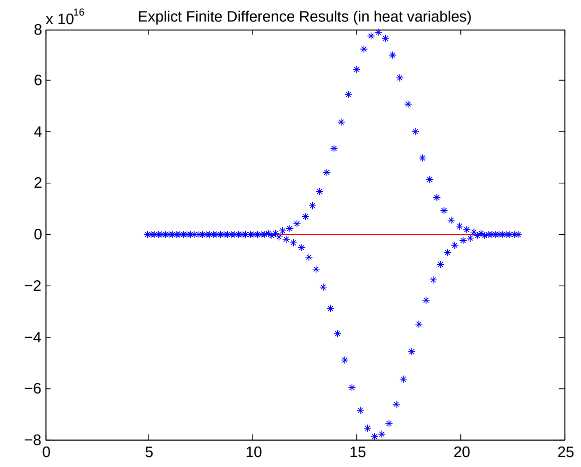
```
[xL,xH]=[4.90129 , 22.7609]
dx=0.576116, using N+1=31 x-intervals.
T=2, dt=0.5, using M=4 time steps.
alpha=1.50643
Minimal M for stability=13
Maximum error=117.886
```

With $N = 30$ the error is even worse worse, hopelessly bad in fact. Let's try increasing M instead.



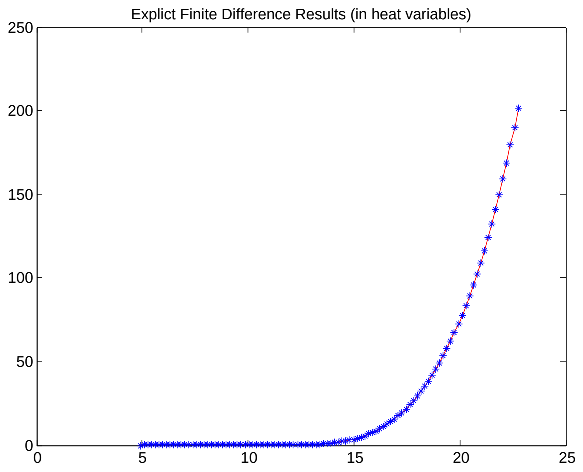
```
[xL,xH]=[4.90129 , 22.7609]
dx=0.576116, using N+1=31 x-intervals.
T=2, dt=0.133333, using M=15 time steps.
alpha=0.401716
Minimal M for stability=13
Maximum error=0.0506617
```

This made a big improvement, the error is now much smaller than previously. To improve the error lets increase both M and N some more.



```
[xL,xH]=[4.90129 , 22.7609]
dx=0.176828, using N+1=101 x-intervals.
T=2, dt=0.02, using M=100 time steps.
alpha=0.639631
Minimal M for stability=128
Maximum error=7.88723e+16
```

With $N = M = 100$ the error is so large that it has completely swamped the calculation, making the results useless. Increasing M helped before. Let's try that again.



```
[xL,xH]=[4.90129 , 22.7609]
dx=0.176828, using N+1=101 x-intervals.
T=2, dt=0.01, using M=200 time steps.
alpha=0.319815
Minimal M for stability=128
Maximum error=0.00291943
```

With $N = 100$ and $M = 200$ we have the best results yet. The error is quite small.

Reviewing the above, we observe that the results seem good when $\alpha \le \frac{1}{2}$, but worthless otherwise. There is obviously an issue here that we need to understand.

4.4 The Issue of Stability

To understand what is going on, observe that our update formula (4.5) can be written in the following matrix form:

$$\begin{bmatrix} u_1^{m+1} \\ u_2^{m+1} \\ \vdots \\ u_{N-1}^{m+1} \\ u_N^{m+1} \end{bmatrix} = \mathbf{A} \begin{bmatrix} u_1^m \\ u_2^m \\ \vdots \\ u_{N-1}^m \\ u_N^m \end{bmatrix} + \alpha \begin{bmatrix} u^{\mathrm{L}}(\tau_m) \\ 0 \\ \vdots \\ 0 \\ u^{\mathrm{H}}(\tau_m) \end{bmatrix},$$

where $\mathbf{A}$ is the tridiagonal matrix

$$\mathbf{A}(\alpha) = \begin{bmatrix} (1-2\alpha) & \alpha & 0 & \cdots & 0 \\ \alpha & (1-2\alpha) & \alpha & 0 & \vdots \\ 0 & \alpha & \ddots & \ddots & 0 \\ \vdots & \ddots & \alpha & (1-2\alpha) & \alpha \\ 0 & \cdots & 0 & \alpha & (1-2\alpha) \end{bmatrix}$$

If we use $\mathbf{u}^m = [u_\cdot^m]$, $\mathbf{b}^{\mathrm{L}} = \alpha \mathbf{e}_1$, and $\mathbf{b}^{\mathrm{H}} = \alpha \mathbf{e}_N$ we can write it even more succinctly as

$$\mathbf{u}^{m+1} = \mathbf{A}\mathbf{u}^m + u^{\mathrm{L}}(\tau_m)\mathbf{b}^{\mathrm{L}} + u^{\mathrm{H}}(\tau_m)\mathbf{b}^{\mathrm{H}}. \tag{4.6}$$

To simplify our discussion, consider the special case of $u^{\mathrm{L}}(\tau) = u^{\mathrm{H}}(\tau) = 0$. Then we have $\mathbf{u}^{m+1} = \mathbf{A}\mathbf{u}^m$. The result of our calculation is simply raising $\mathbf{A}$ to successively higher powers:

$$\mathbf{u}^m = \mathbf{A}^m \mathbf{u}^0.$$

It turns out that the problem that we observed in our test calculations is that in some cases it is possible for successive powers of $\mathbf{A}$ to magnify certain vectors. This easiest to see in terms of the *eigenvalues* of $\mathbf{A}$. Recall that λ is an eigenvalue of $\mathbf{A}$ if there is a nonzero vector $\mathbf{v}$ for which

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{v}.$$

This implies that $\mathbf{A}^m\mathbf{v} = \lambda^m\mathbf{v}$. If $|\lambda| > 1$, then λ^m will grow without bound as m increases. Although $\mathbf{u}^0$ may not be the eigenvector $\mathbf{v}$, the effects of roundoff error, and the adddition of the boundary terms $\mathbf{b}^{\mathrm{L}}$ and $\mathbf{b}^{\mathrm{H}}$ are likely to introduce small multiples of $\mathbf{v}$ into $\mathbf{u}^m$ as the calculation proceeds. Once $\mathbf{u}^m$ includes a small multiple of $\mathbf{v}$, subsequent multiplications by $\mathbf{A}$ will magnify it until it dominates the intended values for $u^m_\cdot$. That is what we saw happening in the "bad" calculations from the previous section. In order for this phenomenon not to occur, we need to know that $|\lambda| < 1$ for *all* the eigenvalues of $\mathbf{A}$.

For our particular matrix $\mathbf{A}(\alpha)$ we can write down exactly what the eigenvalues are. (See the Problem 4.A at the end of the chapter.)

$$\lambda_k = 1 - 2\alpha\left[1 - \cos\left(\frac{k\pi}{N+1}\right)\right]; \quad k = 1, \ldots, N.$$

The crucial question is under what circumstances all of these satisfy $|\lambda_k| \le 1$. This is what determines whether our explicit finite difference calculation succeeds or not. Since $\alpha \ge 0$ and $1 - \cos(k\pi/(N+1)) > 0$ it is clear that $\lambda_k < 1$. But we also want $-1 < \lambda_k$. This is equivalent to

$$\alpha < \frac{1}{1 - \cos(k\pi/(N+1))}, \text{ all } k = 1, \ldots, N.$$

The right side is smallest when the $\cos(\cdot)$ is closest to -1, namely for $k = N$. So $|\lambda_k| < 1$ for all k is equivalent to

$$\alpha < \frac{1}{1 - \cos(\frac{N}{N+1}\pi)}.$$

The right side is slightly larger than $1/2$, but very nearly equal to $1/2$ for large N. So if $\alpha \le 1/2$ then all $|\lambda_k| < 1$. But if $\alpha > 1/2$, then for sufficiently large N there will be one or more eigenvalue with $|\lambda_k| > 1$, which means our numerical calculations will "blow" up for increasing m. This is the issue of *numerical stability.* What this means for our explicit finite difference method is that we need to be sure

$$\Delta\tau \le \frac{1}{2}\Delta x^2$$

in order to get useful results. The number M of time steps needs to be relatively large compared to the number N of spacial steps.

The actual definition of stability for a numerical method is more involved. The choice of Δt determines how large the power m in $\mathbf{A}^m$ will be. We could allow $\mathbf{A}$ to magnify some vectors slightly depending on how large m will be. For a thorough treatment see Richtmeyer & Morton [47]. A sufficient condition is that there is some constant K so that

$$\|\mathbf{A}^m\mathbf{x}\| \le K\|\mathbf{x}\| \text{ for all } m = 1, 2, 3, \cdots \text{ and all } x \in \mathbb{R}^N, \tag{4.7}$$

This will be true if

1. all eigenvalues λ of $\mathbf{A}$ have $|\lambda| \le 1$, and
2. any $|\lambda| = 1$ are simple.

We have been able to verify this directly for the $\mathbf{A}$ of the explicit method. In more complicated situations, like Chapter 5, we won't be able to calculate λ explicitly. However, in Chapter 6 we will find a convenient sufficient condition.

For the actual heat equation, we see from (2.12) that if the initial function $\psi(x)$ is bounded, then $u(x,\tau)$ also reamins bounded for $\tau > 0$. For the heat equation on a strip with boundary conditions (2.16), the same is true: if $\psi(x)$, $u^a(\tau)$, and $u^b(\tau)$ are bounded functions, then the solution $u(x,\tau)$ will be bounded. The phenomenon of "blowing up" as $\tau = \tau_m$ increases that we observed in the "bad" calculations of Section 4.3 is *not* a problem of the heat equation itself, but is a property of the numerical method we are using to approximate the heat equation. Now we know how to avoid it: make sure that $\Delta\tau \le \frac{1}{2}\Delta x^2$. But there is a better way!

4.5 The Implicit and Crank-Nicholson Methods

It turns out that by changing the choice of finite difference approximations we can produce numerical methods that have better stability properties. Let's view

$$u_\tau(x_n, \tau_{m+1}) = \frac{u_n^{m+1} - u_n^m}{\Delta\tau} + \mathcal{O}(\Delta\tau)$$

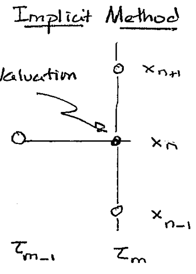
as the backward difference approximation (4.2) at (x_n, τ_{m+1}), instead of the forward difference formula at (x_n, τ_m). Now the heat equation at $(x, \tau) = (x_n \tau_m)$ says that

$$\frac{u_n^{m+1} - u_n^m}{\Delta\tau} + \mathcal{O}(\Delta\tau) = u_\tau(x_n, \tau_{m+1}) = u_{xx}(x_n, \tau_{m+1}) = \frac{u_{n+1}^{m+1} - 2u_n^{m+1} + u_{n-1}^{m+1}}{\Delta x^2} + \mathcal{O}(\Delta x^2)$$

We drop the $\mathcal{O}(\cdot)$ terms as before, and rearrange. With the same $\alpha = \Delta\tau/\Delta x^2$ as before, this gives the formula

$$-\alpha u_{n-1}^{m+1} + (1 + 2\alpha)u_n^{m+1} - \alpha u_{n+1}^{m+1} = u_n^m. \tag{4.8}$$

We use u^{L} and u^{H} to provide values for $n = 0, N+1$ as before. Instead of giving a formula for u_n^{m+1} in terms of the u_n^m values, (4.8) gives us a set of equations which must be solved to find the u_n^{m+1} values. We call (4.8) the *implicit method* for the heat equation because it only gives u_n^{m+1} implicitly.



If we write it in matrix form we get

$$\mathbf{M}\mathbf{u}^{m+1} = \mathbf{u}^m + u^{\mathrm{L}}(\tau_{m+1})\mathbf{b}^{\mathrm{L}} + u^{\mathrm{H}}(\tau_{m+1})\mathbf{b}^{\mathrm{H}}. \tag{4.9}$$

Here $\mathbf{b}^{\mathrm{H}}$ and $\mathbf{b}^{\mathrm{H}}$ are the same as previously (see the line above (4.6)), and $\mathbf{M} = \mathbf{A}(-\alpha)$ using the same formula for $\mathbf{A}$ as before, just with $-\alpha$ in place of α. Provided $\mathbf{M}$ is invertible, we have

$$\mathbf{u}^{m+1} = \mathbf{M}^{-1}\left[\mathbf{u}^m + u^{\mathrm{L}}(\tau_{m+1})\mathbf{b}^{\mathrm{L}} + u^{\mathrm{H}}(\tau_{m+1})\mathbf{b}^{\mathrm{H}}\right].$$

In fact, we know that $\mathbf{M}$ *is* invertible because we know its eigenvalues are $\lambda_k = 1 + 2\alpha(1 - \cos(k\pi/(N+1))$ which are all > 1. (If $\mathbf{M}$ were not invertible, then 0 would be an eigenvalue.)

In the next section we will talk about how to implement the $\mathbf{M}^{-1}$ efficiently in MATLAB . But first let's consider the stability issue. We want to know when all the eigenvalues of $\mathbf{M}^{-1}$ have $|\lambda_k| \le 1$. But λ_k is an eigenvalue of $\mathbf{M}^{-1}$ if and only if λ_k^{-1} is an eigenvalue of $\mathbf{M}$. And since $\mathbf{M} = \mathbf{A}(-\alpha)$ we have a formula for those. So the eigenvalues of $\mathbf{M}^{-1}$ are

$$\lambda_k = \left[1 + 2\alpha(1 - \cos(k\pi/(N+1))\right]^{-1}, \quad k = 1, \cdots, N.$$

Since $\alpha(1 - \cos(k\pi/(N+1)) > 0$ we see that $0 < \lambda_k < 1$ for all k *with no restriction on* α! This is usually described by saying that the implicit method is *unconditionally stable.* This is our reward for the extra difficulty of dealing with $\mathbf{M}^{-1}$.

Before we rush to implement the implicit method, we can actually do even better for only a little more computational effort. To explain this lets first establish some notation that will help us work out the development of finite difference methods more systematically. Let $\mathbf{C}$ be the $N \times N$ matrix

$$\mathbf{C} = \begin{bmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & \vdots \\ 0 & 1 & \ddots & \ddots & 0 \\ \vdots & \ddots & 1 & -2 & 1 \\ 0 & \cdots & 0 & 1 & -2 \end{bmatrix}. \tag{4.10}$$

The vector of our second order central difference approximations to $u_{xx}(x_n, \tau_m)$ $(n = 1, \ldots, N)$ can be written

$$\begin{bmatrix} \vdots \\ u_{xx}(x_n, \tau_m) \\ \vdots \end{bmatrix} \approx \begin{bmatrix} \vdots \\ \frac{u^m_{n+1} - 2u^m_n + u^m_{n-1}}{\Delta x^2} \\ \vdots \end{bmatrix} = \frac{1}{\Delta x^2}\mathbf{C}\mathbf{u}^m + \frac{1}{\Delta x^2} u^{\mathrm{L}}(\tau_m)\mathbf{e}_1 + \frac{1}{\Delta x^2} u^{\mathrm{H}}(\tau_m)\mathbf{e}_N \tag{4.11}$$

This is our discrete approximation to $\partial_x^2 u$ at $(x_1, \ldots, x_N)$ and τ_m. The forward difference approximation for u_τ at these same points is

$$\begin{bmatrix} \vdots \\ u_\tau(x_n, \tau_m) \\ \vdots \end{bmatrix} \approx \frac{1}{\Delta\tau}(\mathbf{u}^{m+1} - \mathbf{u}^m).$$

With this notation the explicit method is derived by using the above approximations in

$$\begin{bmatrix} \vdots \\ u_\tau(x_n, \tau_m) \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ u_{xx}(x_n, \tau_m) \\ \vdots \end{bmatrix}$$

to obtain

$$\frac{1}{\Delta\tau}(\mathbf{u}^{m+1} - \mathbf{u}^m) = \frac{1}{\Delta x^2}\mathbf{C}\mathbf{u}^m + \frac{1}{\Delta x^2} u^{\mathrm{L}}(\tau_m)\mathbf{e}_1 + \frac{1}{\Delta x^2} u^{\mathrm{H}}(\tau_m)\mathbf{e}_N.$$

Solving for $\mathbf{u}^{m+1}$ we get

$$\begin{aligned} \mathbf{u}^{m+1} &= \mathbf{u}^m + \alpha\left(\mathbf{C}\mathbf{u}^m + u^{\mathrm{L}}(\tau_m)\mathbf{e}_1 + u^{\mathrm{H}}(\tau_m)\mathbf{e}_N\right) \\ &= (\mathbf{I} + \alpha\mathbf{C})\mathbf{u}^m + u^{\mathrm{L}}(\tau_m)\mathbf{b}^{\mathrm{L}} + u^{\mathrm{H}}(\tau_m)\mathbf{b}^{\mathrm{H}}. \end{aligned}$$

This is (4.6), just written in different notation. For the implicit method, we use the backward difference approximation instead:

$$\begin{bmatrix} \vdots \\ u_\tau(x_n, \tau_{m+1}) \\ \vdots \end{bmatrix} \approx \frac{1}{\Delta\tau}(\mathbf{u}^{m+1} - \mathbf{u}^m).$$

Now we approximate the right side of

$$\begin{bmatrix} \vdots \\ u_\tau(x_n, \tau_{m+1}) \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ u_{xx}(x_n, \tau_{m+1}) \\ \vdots \end{bmatrix}$$

using (4.11) with $m + 1$. We obtain

$$(\mathbf{I} - \alpha\mathbf{C})\mathbf{u}^{m+1} = \mathbf{u}^m + u^{\mathrm{L}}(\tau_{m+1})\mathbf{b}^{\mathrm{L}} + u^{\mathrm{H}}(\tau_{m+1})\mathbf{b}^{\mathrm{H}}.$$

This is our implicit method (4.9).

Now we can introduce a clever improvement on the implicit method. Instead of evaluating the heat equation at the points (x_n, τ_{m+1}) we are going to evaluate it at $(x_n, \tau_{m+1/2})$, where

$$\tau_{m+1/2} = \frac{1}{2}(\tau_m + \tau_{m+1}).$$

We can view

$$\frac{u_n^{m+1} - u_n^m}{\Delta\tau} = u_\tau(x_n, \tau_{m+1/2}) + \mathcal{O}(\Delta\tau^2)$$

as the central difference approximation (4.3) at $(x_n, \tau_{m+1/2})$ with an increment size of $\frac{1}{2}\Delta\tau$. This gives us the higher order accuracy of $\mathcal{O}(\Delta\tau^2)$. To approximate $u_{xx}(x_n, \tau_{m+1/2})$ we will use the average of the values at the left and right gridpoints,

$$u_{xx}(x_n, \tau_{m+1/2}) \approx \frac{1}{2}\left[u_{xx}(x_n, \tau_m) + u_{xx}(x_n, \tau_{m+1})\right],$$

and then approximate each term on the right using the second central difference approximation (4.4). It turns out that this is again an $\mathcal{O}(\Delta x^2)$ approximation. So overall this pair of finite difference approximations is of order $\mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta\tau^2)$, which is better than the $\mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta\tau)$ for both the explicit and implicit methods. So we approximate the heat equation

$$\begin{bmatrix} \vdots \\ u_\tau(x_n, \tau_{m+1/2}) \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ u_{xx}(x_n, \tau_{m+1/2}) \\ \vdots \end{bmatrix}$$

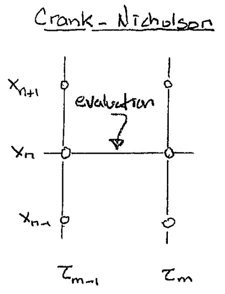
by

$$\frac{1}{\Delta\tau}(\mathbf{u}^{m+1} - \mathbf{u}^m) = \frac{1}{2}\left[\frac{1}{\Delta x^2}\mathbf{C}\mathbf{u}^{m+1} + \frac{1}{\Delta x^2}u^{\mathrm{L}}(\tau_{m+1})\mathbf{e}_1 + \frac{1}{\Delta x^2}u^{\mathrm{H}}(\tau_{m+1})\mathbf{e}_N\right]$$
$$+ \frac{1}{2}\left[\frac{1}{\Delta x^2}\mathbf{C}\mathbf{u}^{m} + \frac{1}{\Delta x^2}u^{\mathrm{L}}(\tau_{m})\mathbf{e}_1 + \frac{1}{\Delta x^2}u^{\mathrm{H}}(\tau_{m})\mathbf{e}_N\right].$$

When reorganized this becomes

$$(\mathbf{I} - \frac{\alpha}{2}\mathbf{C})\mathbf{u}^{m+1} = (\mathbf{I} + \frac{\alpha}{2}\mathbf{C})\mathbf{u}^m + \frac{u^{\mathrm{L}}(\tau_m) + u^{\mathrm{L}}(\tau_{m+1})}{2}\mathbf{b}^{\mathrm{L}} + \frac{u^{\mathrm{H}}(\tau_m) + u^{\mathrm{H}}(\tau_{m+1})}{2}\mathbf{b}^{\mathrm{H}}. \quad (4.12)$$

This is the *Crank-Nicholson* method for the heat equation. You might observe that it it is exactly the result of averaging the explicit and implicit formulas above. But it has the advantage of being a higher order approximation to the actual heat equation.



What about its stability? For that we consider the case of $u^{\mathrm{L}} = u^{\mathrm{H}} = 0$:

$$(\mathbf{I} - \frac{\alpha}{2}\mathbf{C})u^{m+1}_{\cdot} = (\mathbf{I} + \frac{\alpha}{2}\mathbf{C})u^m_{\cdot}$$

which is the same as

$$u_{\cdot}^{m+1} = (\mathbf{I} - \frac{\alpha}{2}\mathbf{C})^{-1}(\mathbf{I} + \frac{\alpha}{2}\mathbf{C})u_{\cdot}^{m}.$$

We want to know about the eigenvalues. Observe that in Problem A below the eigenvectors $\mathbf{v}_k$ do not depend on α. We have

$$\begin{aligned}
\mathbf{A}(\alpha/2)\mathbf{v}_k &= [1 - \alpha(1 - \cos(k\pi/(N+1))]\mathbf{v}_k \\
\mathbf{A}(-\alpha/2)\mathbf{v}_k &= [1 + \alpha(1 - \cos(k\pi/(N+1))]\mathbf{v}_k \\
\mathbf{A}(-\alpha/2)^{-1}\mathbf{v}_k &= [1 + \alpha(1 - \cos(k\pi/(N+1))]^{-1}\mathbf{v}_k.
\end{aligned}$$

We find that $\mathbf{v}_k$ is an eigenvector of

$$(\mathbf{I} - \frac{\alpha}{2}\mathbf{C})^{-1}(\mathbf{I} + \frac{\alpha}{2}\mathbf{C}) = \mathbf{A}(-\alpha/2)^{-1}\mathbf{A}(\alpha/2)$$

with eigenvalue

$$\lambda_k = \frac{1 - \alpha(1 - \cos(k\pi/(N+1))}{1 + \alpha(1 - \cos(k\pi/(N+1))}.$$

Since the denominator is > 1, we find that

$$\lambda_k \leq \frac{1}{1 + \alpha(1 - \cos(k\pi/(N+1))} < 1.$$

And the inequality $-1 < \lambda_k$ is equivalent to

$$-[1 + \alpha(1 - \cos(k\pi/(N+1))] \leq 1 - \alpha(1 - \cos(k\pi/(N+1)),$$

which simplifies to $-1 < 1$ which is always true. So we find that $|\lambda_k| < 1$ is always true, meaning that the Crank-Nicholson method is unconditionally stable!

Observe that all three methods we have discussed use the same approximation (4.11) to to $\partial_x^2 u$. The difference between them is based on how we view the discretiaztion of $\partial_\tau u$. There are yet other possibilities. For instance Tavella & Randall [57, page 85] describe a multi-time step method.

4.6 LU Factorization and Matlab Implementation

Both the implicit and Crank-Nicholson methods are of the form

$$\mathbf{M}\mathbf{u}^{m+1} = \mathbf{N}\mathbf{u}^{m} + \mathbf{b}^{m}.$$

We need to solve this equation for $\mathbf{u}^{m+1}$. The right side involes only quantities we know or can calculate from u^{L} and u^{H}. The matricies $\mathbf{M}$ and $\mathbf{N}$ are likewise known. Mathematically the solution is

$$\mathbf{u}^{m+1} = \mathbf{M}^{-1}(\mathbf{N}\mathbf{u}^{m} + \mathbf{b}^{m}).$$

What is the best way to implement this for computer evaluation? The naive thing to do would be to calculate $\mathbf{M}^{-1}$ once at the beginning of our program, store it and then reuse it for the matrix multiplication to calculate each new $u_{\cdot}^{m+1}$. But this is not the best way. We expect to do this calculation a large number of times ($m = 1, 2, \ldots, M$). Writing our program to do it efficiently can make a big difference in how quickly the program runs, especially for large N and M. Effecient solution of large linear systems is very important in many scientific computing applications, there has been much study of how best to carry it out. MATLAB was designed to do this kind of calculation, so we ought to take advantage of its abilities. The matricies $\mathbf{M}$ that we are interested in are tridiagonal, meaning that the only nonzero entries are either on or immediately above or below the diagonal:

$$\mathbf{M} = \begin{bmatrix} d_1 & f_1 & 0 & \cdots & 0 \\ e_2 & d_2 & f_2 & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & e_{N-1} & d_{N-1} & f_{N-1} \\ 0 & \cdots & 0 & e_N & d_N \end{bmatrix}$$

There are more efficient ways to solve $\mathbf{Mu} = \mathbf{b}$ than calculating $\mathbf{M}^{-1}$ and then multiplying $\mathbf{b}$ by it. Matlab 's "backslash" command `M\b` will compute the solution using an effieient form of Gaussian elimination called the *LU method.* Here is how it works.

Step 1. First factor $\mathbf{M} = \mathbf{LU}$ where $\mathbf{L}$ is (unit) lower triangular and $\mathbf{U}$ is upper triangluar. This is easy to do with the command `[L,U]=lu(M)`, which goes quite fast if $\mathbf{M}$ is tridiagonal. When $\mathbf{M}$ is tridiagonal the resulting $\mathbf{L}$ and $\mathbf{U}$ will have the form

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ * & 1 & 0 & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & * & 1 & 0 \\ 0 & \cdots & 0 & * & 1 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} * & * & 0 & \cdots & 0 \\ 0 & * & * & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 0 & * & * \\ 0 & \cdots & 0 & 0 & * \end{bmatrix}$$

Step 2. Write $\mathbf{Mu} = \mathbf{b}$ as $\mathbf{L}(\mathbf{Uu}) = \mathbf{b}$ and solve it in two stages:

$$\mathbf{Lv} = \mathbf{b}$$
$$\mathbf{Uu} = \mathbf{v}.$$

Matlab 's backslash command looks for triangular structure and takes advantage of it, so this calculation will execute *very* fast with the single line `u=U\(L\b)`.

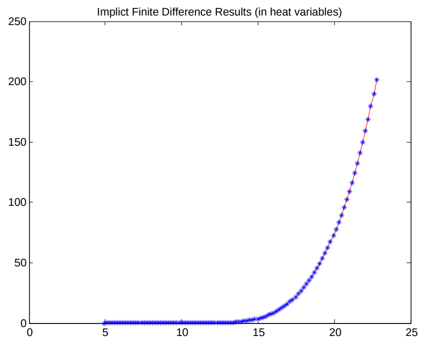
In general `u=M\b` does both of these steps. But we don't want to use this directly in our code because it would waste effort by re-calculating $\mathbf{L}$ and $\mathbf{U}$ in Step 1 every time. So the efficient thing is to do Step 1 once, saving the resulting $\mathbf{L}$ and $\mathbf{U}$, and then just repeat Step 2 with a new $\mathbf{b}^{m+1}$ for each new $\mathbf{u}^{m+1}$.

One more thing — most of the matricies we are working with are *sparse*, which means most of the entries are 0. A sparse matrix can be stored more efficiently by just storing the non-zero entries. Matlab has a special format for storing sparse matrices, and special versions of many of its commands which will take advantage of the sparse structure to speed up the calculation. Here is a Matlab script which implements the implicit method using these efficiency techniques, and some results using it for $N = M = 100$, with the other parameters the same as for our previous tests.

imlu.m

```
% Script for the fully implicit method, using LU factorization.
% Initial values are to be provided by u0(x).
% Boundary values are to be provided by uL(x,tau), uH(x,tau).
%
% Before runing the script, m-files for u0, uL, uH need to
% be provided and values given to the following variables:
% xL = lower boundary
% xH = upper boundary
% N = number of spatial intervals (less 1)
% M = number of time steps
% T= final time
%
dx=(xH-xL)/(N+1);
dt=T/M;
alpha=dt/(dx^2);
e=ones(N,1);   %Note: this needs to be a column
A=spdiags([-alpha*e,(1+2*alpha)*e,-alpha*e],[-1,0,1],N,N);
[L,U]=lu(A);
bL=sparse(1,1,alpha,N,1);  %[alpha;zeros(N-1,1)];
bH=sparse(N,1,alpha,N,1);  %[zeros(N-1,1);alpha];
```

```
x=linspace(xL,xH,N+2);
u=u0(x(2:N+1))';
tau=0;
%
%Update Loop:
for m=1:M
     tau=m*dt;
     rhs=u+uL(xL,tau)*bL+uH(xH,tau)*bH;
     u=U\(L\rhs);
end
u=[uL(xL,T),u',uH(xH,T)];
```



```
[xL,xH]=[4.90129 , 22.7609]
dx=0.176828, using N+1=101 x-intervals.
T=2, dt=0.02, using M=100 time steps.
alpha=0.639631
Maximum error=0.0162208
```

We see that this worked well, in spite of $\alpha > .5$.

4.7 Problems

Problem 4.A
Let A be the $N \times N$ matrix

$$\mathbf{A} = \begin{bmatrix} 1-2\alpha & \alpha & 0 & \dots & 0 \\ \alpha & 1-2\alpha & \alpha & \ddots & \vdots \\ 0 & \alpha & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & & \alpha \\ 0 & \dots & 0 & \alpha & 1-2\alpha \end{bmatrix}$$

and $\mathbf{v}_k$ the vector

$$\mathbf{v}_k = \begin{bmatrix} \sin(\frac{k\pi}{N+1} \cdot 1) \\ \sin(\frac{k\pi}{N+1} \cdot 2) \\ \vdots \\ \sin(\frac{k\pi}{N+1} \cdot N) \end{bmatrix}.$$

Show by direct calculation that

$$\mathbf{A}\mathbf{v}_k = \lambda \mathbf{v}_k$$

where

$$\lambda_k = 1 - 2\alpha(1 - \cos(\frac{k\pi}{N+1})).$$

This shows that λ_k is indeed an eigenvalue of $\mathbf{A}$ for each $k = 1, \ldots, N$, as claimed in class. Since each such choice of k produces a different value of λ_k, this exhibits N distinct eigenvalues, and hence there are no others. Hint: You will need a trig identity for the calculation.

. . . . $\boxed{\texttt{D}}$

Problem 4.B

Create a Matlab sctipt to carry out the Crank-Nicholson method using the LU factorization technique for the heat equation on a strip $x^{\mathrm{L}} < x < x^{\mathrm{H}}$, $0 < \tau$. First test it on the solution of Problem 2.A part 2 to insure that it is working correctly. Then use it to compute the solution of the heat equation on the strip $0 < x < 1$ for $0 < \tau < 2$ with the following initial and boundary values:

$$u(x, 0) = 2x, \quad u(0, \tau) = 0, \quad u(1, \tau) = 2 - \tau.$$

Plot the graph of the resulting $u(x, 2)$. Find the value of x as accurately as you can which maximizes $u(x, 2)$.

. . . . $\boxed{\texttt{F}}$

Chapter 5

Finite Difference Methods in Financial Variables

In Chapter 4 we learned about finite difference calculations for the heat equation. If we are interested in a particular solution v of the Black-Scholes equation, we can convert the problem to an equivalent one for the heat equation, use a finite difference calculation for the heat equation to approximate u, then convert the results back to financial variables. This is not difficult to do; we did it many times in Chapter 4. But there are several reasons to consider doing the finite difference calculation directly in financial variables. For one, if x_n are the evenly spaced grid points for the heat equation then the corresponding grid points in financial variaables, $s_n = e^{\sigma x_n/\sqrt{2}}$, will be very closely spaced for small s and quite widely spaced for large s. We might prefer to have the s_n values themselves evenly spaced, in which case just doing the whole finite difference calculation in financial variables seems simpler. Another reason is that for some alternative models of the stock price, the resulting PDEs do not readily convert to the heat equation. (See for instance the Cox model in [26].) In that case we have no alternative but to work in the original variables. However, working in financial variables also raises some new issues, as we will see in this chapter.

5.1 Finite Difference Formulas

We will use a grid consisting of $s_n = s^{\mathrm{L}} + n\Delta s$, $(n = 0, \ldots, N+1)$ and $t_m = m\Delta t$ $(m = 0, \ldots, M)$. The value $s^{\mathrm{L}} = 0$ is the simplest choice for the boundary; we will need to an pick upper boundary $s_{N+1} = s^{\mathrm{H}}$. The time of expiry will be $T = t_M$. Given a terminal value function $\phi(s)$ we seek approximations

$$v_n^m \approx v(x_n, t_m)$$

to the Black-Scholes equation

$$\partial_t v(s,t) + \mathcal{B}v(s,t) = 0, s^{\mathrm{L}} < s < s^{\mathrm{H}}, t < T$$

satisfying the terminal conditon

$$v(s,T) = \phi(s).$$

Note that instead of starting at $m = 0$ and proceeding up to $m = M$ as we did for the heat equation, in financial variables we will start at $m = M$ and work backwards until we reach m for the desired $t = t_m$.

The first thing we must do is develop a finite difference approximation to $\mathcal{B}v(s,t)$ analogous to that of (4.11). We can do this systematically, following our precedent from the last chapter. With the same matrix $\mathbf{C}$ as (4.10), and the same notational conventions,

$$\begin{bmatrix} \vdots \\ v_{ss}(s_n, t_m) \\ \vdots \end{bmatrix} \approx \frac{1}{\Delta s^2}\mathbf{C}\mathbf{v}^m + \frac{1}{\Delta s^2} v^{\mathrm{L}}(t_m)\mathbf{e}_1 + \frac{1}{\Delta s^2} v^{\mathrm{H}}(t_m)\mathbf{e}_N.$$

Let $\mathbf{S}$ be the diagonal matrix with entries s_i:

$$\mathbf{S} = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & 0 & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & s_N \end{bmatrix}.$$

We can write the vector of $[s_.^2 v_{ss}(s_., t_m)]$ simply as $\mathbf{S}^2[v_{ss}(s_., t_m)]$, so we get

$$\begin{aligned} \begin{bmatrix} \vdots \\ \frac{\sigma^2}{2} s_n^2 v_{ss}(s_n, t_m) \\ \vdots \end{bmatrix} &= \frac{\sigma^2}{2}\mathbf{S}^2 \begin{bmatrix} \vdots \\ v_{ss}(s_n, t_m) \\ \vdots \end{bmatrix} \\ &\approx \frac{\sigma^2}{2\Delta s^2}\mathbf{S}^2\mathbf{C}v(s_., t_m) + \frac{\sigma^2}{2\Delta s^2} v^{\mathrm{L}}(t_m)\mathbf{S}^2\mathbf{e}_1 + \frac{\sigma^2}{2\Delta s^2} v^{\mathrm{H}}(t_m)\mathbf{S}^2\mathbf{e}_N. \end{aligned}$$

For the v_s term, we need to decide which of the several possible difference approximations to use. Since the approximation we are using for v_{ss} is of second order accuracy, we ought use an approximation to v_s which is also (at least) second order. That suggests the central difference approximation (4.3) (applied to the s-variable). Using our boundary formulas for s_0 and s_{N+1}, this leads us to

$$\begin{bmatrix} \vdots \\ v_s(s_., t_m) \\ \vdots \end{bmatrix} \approx \frac{1}{2\Delta s}\mathbf{F}\mathbf{v}^m - \frac{1}{2\Delta s} v^{\mathrm{L}}(t_m)\mathbf{e}_1 + \frac{1}{2\Delta s} v^{\mathrm{H}}(t_m)\mathbf{e}_N,$$

where

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ -1 & 0 & 1 & 0 & \vdots \\ 0 & -1 & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 0 & 1 \\ 0 & \cdots & 0 & -1 & 0 \end{bmatrix}.$$

This gives

$$\begin{bmatrix} \vdots \\ r s_n v_s(s_n, t_m) \\ \vdots \end{bmatrix} \approx \frac{r}{2\Delta s}\mathbf{S}\mathbf{F}\mathbf{v}^m - \frac{r}{2\Delta s} v^{\mathrm{L}}(t_m)\mathbf{S}\mathbf{e}_1 + \frac{r}{2\Delta s} v^{\mathrm{H}}(t_m)\mathbf{S}\mathbf{e}_N.$$

Also,

$$\begin{bmatrix} \vdots \\ r v(s_n, t_m) \\ \vdots \end{bmatrix} = r\mathbf{I}\mathbf{v}^m.$$

It will be convenient to use

$$\alpha = \frac{\Delta t}{\Delta s^2}, \text{ and } \beta = \frac{\Delta t}{\Delta s}.$$

Putting the pieces together, we get the $\mathcal{O}(\Delta s^2)$ approximation

$$\begin{aligned} \begin{bmatrix} \vdots \\ \mathcal{B}v(s_n, t_m) \\ \vdots \end{bmatrix} \approx \frac{1}{\Delta t}\Bigg(&\left[\frac{\sigma^2}{2}\alpha\mathbf{S}^2\mathbf{C} + \frac{r}{2}\beta\mathbf{S}\mathbf{F} - r\Delta t\mathbf{I}\right]\mathbf{v}^m \\ &+ v^{\mathrm{L}}(t_m)\left[\frac{\sigma^2}{2}\alpha\mathbf{S}^2 - \frac{r}{2}\beta\mathbf{S}\right]\mathbf{e}_1 + v^{\mathrm{H}}(t_m)\left[\frac{\sigma^2}{2}\alpha\mathbf{S}^2 + \frac{r}{2}\beta\mathbf{S}\right]\mathbf{e}_N\Bigg) \end{aligned}$$

This looks messy, but it will be the basic ingredient of the explicit, implicit and Crank-Nicholson methods, just like (4.11) was in the last chapter. Of most importance is the matrix

$$\mathbf{B} = \frac{\sigma^2}{2}\alpha\mathbf{S}^2\mathbf{C} + \frac{r}{2}\beta\mathbf{S}\mathbf{F} - r\Delta t\mathbf{I}.$$

It will be helpful to write it differently. Let $\mathbf{L}$ and $\mathbf{U}$ be the matricies with 1s on the sub- and superdiagonals.

$$\mathbf{L} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \ddots & \\ \vdots & \ddots & 0 & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & 0 \end{bmatrix}.$$

We can write

$$\mathbf{C} = \mathbf{L} - 2\mathbf{I} + \mathbf{U}, \text{ and } \mathbf{F} = \mathbf{U} - \mathbf{L},$$

and substitute these in the expression for $\mathbf{B}$ to obtain

$$\begin{aligned} \mathbf{B} &= \frac{1}{2}\sigma^2\alpha\mathbf{S}^2\mathbf{C} + \frac{1}{2}r\beta\mathbf{S}\mathbf{F} - r\Delta t\mathbf{I} \\ &= \frac{1}{2}\sigma^2\alpha\mathbf{S}^2(\mathbf{L} - 2\mathbf{I} + \mathbf{U}) + \frac{1}{2}r\beta\mathbf{S}(\mathbf{U} - \mathbf{L}) - r\Delta t\mathbf{I} \\ &= \frac{1}{2}(\sigma^2\alpha\mathbf{S}^2 - r\beta\mathbf{S})\mathbf{L} - (\sigma^2\alpha\mathbf{S}^2 + r\Delta t\mathbf{I}) + \frac{1}{2}(\sigma^2\alpha\mathbf{S}^2 + r\beta\mathbf{S})\mathbf{U}. \end{aligned}$$

From this we can read off the subdiagonal, diagonal and superdiagonal entries of $\mathbf{B}$. Define

$$a_n = \frac{1}{2}(\sigma^2\alpha s_n^2 - r\beta s_n), \; b_n = \sigma^2\alpha s_n^2 + r\Delta t, \; c_n = \frac{1}{2}(\sigma^2\alpha s_n^2 + r\beta s_n). \tag{5.1}$$

Our approximation is

$$\begin{bmatrix} \vdots \\ \mathcal{B}v(s_n, t_m) \\ \vdots \end{bmatrix} \approx \frac{1}{\Delta t}\left[\mathbf{B}\mathbf{v}^m + v^{\mathrm{L}}(t_m)\mathbf{b}^{\mathrm{L}} + v^{\mathrm{H}}(t_m)\mathbf{b}^{\mathrm{H}}\right], \tag{5.2}$$

where

$$\mathbf{b}^{\mathrm{L}} = a_1\mathbf{e}_1, \quad \mathbf{b}^{\mathrm{H}} = c_N\mathbf{e}_N, \tag{5.3}$$

and $\mathbf{B}$ is the tridiagonal matrix

$$\mathbf{B} = \begin{bmatrix} -b_1 & c_1 & 0 & \cdots & 0 \\ a_2 & -b_2 & c_2 & 0 & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & 0 & a_{N-1} & -b_{N-1} & c_{N-1} \\ 0 & \cdots & 0 & a_N & -b_N \end{bmatrix}. \tag{5.4}$$

We are ready now to write down our three basic finite difference formulations. The *explict method* uses

$$\partial_t v(s_n, t_m) \approx \frac{v(s_n, t_m) - v(s_n, t_{m-1})}{\Delta t}$$

in $v_t(s_n, t_m) + \mathcal{B}v(s_n, t_m) = 0$. Using (5.2) we obtain

$$\mathbf{v}^{m-1} = (\mathbf{I} + \mathbf{B})\mathbf{v}^m + v^{\mathrm{L}}(t_m)\mathbf{b}^{\mathrm{L}} + v^{\mathrm{H}}(t_m)\mathbf{b}^{\mathrm{H}}. \tag{5.5}$$

The *implicit method* uses

$$\partial_t v(s_n, t_{m-1}) \approx \frac{v(s_n, t_m) - v(s_n, t_{m-1})}{\Delta t}$$

in $v_t(s_n, t_{m-1}) + \mathcal{B}v(s_n, t_{m-1}) = 0$ to obtain

$$(\mathbf{I} - \mathbf{B})\mathbf{v}^{m-1} = \mathbf{v}^m + v^{\mathrm{L}}(t_{m-1})\mathbf{b}^{\mathrm{L}} + v^{\mathrm{H}}(t_{m-1})\mathbf{b}^{\mathrm{H}}. \tag{5.6}$$

For the *Crank-Nicholson method* we use

$$v_t(s_n, t_{m-1/2}) \approx \frac{v(s_n, t_m) - v(s_n, t_{m-1})}{\Delta t}$$

and

$$\mathcal{B}v(s_n, t_{m-1/2}) \approx \frac{1}{2}\left[\mathcal{B}v(s_n, t_m) + \mathcal{B}v(s_n, t_{m-1})\right],$$

with each $\mathcal{B}$-term on the right approximated by (5.2). When these are used in $\partial_t v(s_n, t_{m-1/2}) + \mathcal{B}v(s_n, t_{m-1/2}) = 0$ we obtain

$$(\mathbf{I} - \frac{1}{2}\mathbf{B})\mathbf{v}^{m-1} = (\mathbf{I} + \frac{1}{2}\mathbf{B})\mathbf{v}^m + \frac{v^{\mathrm{L}}(t_{m-1}) + v^{\mathrm{L}}(t_m)}{2}\mathbf{b}^{\mathrm{L}} + \frac{v^{\mathrm{H}}(t_{m-1}) + v^{\mathrm{H}}(t_m)}{2}\mathbf{b}^{\mathrm{H}}. \tag{5.7}$$

Thus once we generate the tridiagonal matrix $\mathbf{B}$ the implementation of any of these methods will be largely the same as for the heat equation, using the LU factorization for the implicit and Crank-Nicholson methods.

The following script will generate $\mathbf{B}$, the boundary vectors $\mathbf{b}^{\mathrm{L}}$ and $\mathbf{b}^{\mathrm{H}}$ and other relevant quantities.

genB.m

```
% Script to generate matricies for the discrete approximation B to the
% Black-Schole operator, in financial variables.  The following variables
% need to be defined:
%     boundary values 0<=sL<sH need to be specified,
%     the number N of subintervals,
%     time increment dt
%     interest rate r,
%     volatility sigma.
% The script will generate the following:
%     spatial increment ds,
%     vector of N price values s=[sL+ds, ... , sH-ds] (row),
%     vectors a, b, c of coefficients (columns)
%     the matrix B.
ds=(sH-sL)/(N+1);
s=linspace(sL+ds,sH-ds,N);
alpha=dt/ds^2;
beta=dt/ds;
a=(sigma^2*alpha*s.^2-r*beta*s)'/2;
b=(sigma^2*alpha*s.^2+r*dt*ones(size(s)))';
c=(sigma^2*alpha*s.^2+r*beta*s)'/2;
B=spdiags([[a(2:N);0],-b,[0;c(1:N-1)]],[-1,0,1],N,N);
bL=sparse(1,1,a(1),N,1);
bH=sparse(N,1,c(N),N,1);
```

Here is a script to implement the Crank-Nicholson method.

cnf.m

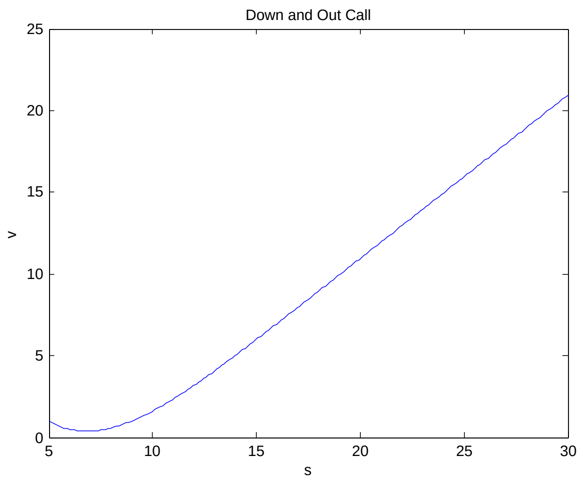
```
% Script for Crank-Nicolson finite difference method in financial
% variables using explicit boundary values.  The variables sL, sH, N, M, r,
% sigma, and T should be set at the command line before running the script.
% (r, sigma, and K should be global).  Separate m-files vT, vL, and vH give
% initial and boundary values.
```

```
dt=T/M;
genB                               %Generate matricies and coefficients
Mat=speye(N)-B/2;
[L,U]=lu(Mat);
Mat=speye(N)+B/2;
v=vT(s)';                          %Initialize
t=T;
for m=M-1:-1:0                     %Time-step loop
rhs=Mat*v+(vL(sL,t,T)*bL+vH(sH,t,T)*bH)/2;
t=m*dt;
rhs=rhs+(vL(sL,t,T)*bL+vH(sH,t,T)*bH)/2;
v=U\(L\rhs);
end
s=[sL,s,sH];                       %Include boundaries and
v=[vL(sL,t,T),v',vH(sH,t,T)];      %Convert results to rows
```

Example. To illustrate, consider the down & out call option. The pricing function solves the Black-Scholes equation in $s > \theta$ with $v(\theta, t) = R$ and terminal value $v(s,T) = (s-K)^+$. We will work in an interval $[s^{\mathrm{L}}, s^{\mathrm{H}}]$ with $s^{\mathrm{L}} = \theta$ and s^{H} some "artifical" upper boundary (it should be significantly larger than K and than the ranges of s-values for which we are interested in $v(s,t)$). The boundary values (using a rebate R) should be

$$v^{\mathrm{L}}(t) = R, \quad v^{\mathrm{H}}(t) = s^{\mathrm{H}} - e^{-t(T-t)}K.$$

This gives all the ingredients to do the numerical calculation. We simply incorporate these specifications into `vT.m`, `vL.m`, and `vH.m`, specify the parameters at the command line and run the script `cnf.m`. Here is a plot of the results using $r = .05$, $\sigma = .2$, $T = 2$, $s^{\mathrm{L}} = \theta = 5$, $R = 1$, $K = 10$, $s^{\mathrm{H}} = 30$, $N = 200$, and $M = 100$.



(From Section 3.2 we have an exact formula for v to which we can compare our numerical values.)

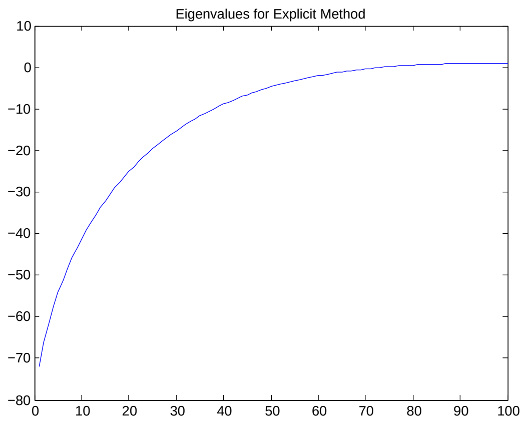
5.2 Stability and Performance Issues

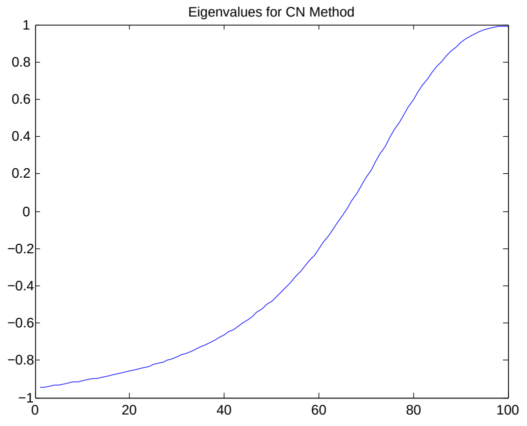
What about the stability of these various methods? For explicit methods (with or without boundary conditons) we expect some restriction on Δs and Δt to be necessary. According to Wilmott [61] both the implicit and Crank-Nicholson methods, using boundary values, are stable with no stepsize restrictions. Although we cannot present an explicit calculation of eigenvalues to justify these claims, it is not hard to test them experimentally using MATLAB. In the problems at the end of the chapter you will do the same thing for the modified versions using (5.9).

As an example we can check the eigenvalues numerically for the explicit, implicit and Crank-Nicholson methods based on (5.2). First observe that if γ_k are the eigenvalues of $\mathbf{B}$, then the eigenvalues for our three basic methods are as follows:

1. Explicit method: $\lambda_k = 1 + \gamma_k \in (-1, 1)$ iff $-2 < \gamma_k < 0$.
2. Implicit method: $\lambda_k = (1 - \gamma_k)^{-1} \in (-1, 1)$ iff $\gamma_k < 0$ or $2 < \gamma_k$.
3. Crank-Nicholson method: $\lambda_k = \frac{1+\gamma_k/2}{1-\gamma_k/2} \in (-1, 1)$ iff $\gamma_k < 0$.

We take $N = 100$ and $\Delta s = .1$, with $[s^{\mathrm{L}}, s^{\mathrm{H}}] = 0, 10.1]$ (so $s_N = 10$), and the parameter values $r = .05$, $\sigma = .2$. Using $\Delta t = .1$ we calculate $\mathbf{B}$ using the script `genB.m` above, and then the command `gamma=sort(eig(full(B)))` to produce a sorted list of all the eigenvalues. We find that the eienvalues γ_k are all real, with the largest being $\approx -.0064$. Now we can plot the list of $\lambda_k = 1 + \gamma_k$ for the explicit method and $\lambda_k = \frac{1+\gamma_k/2}{1-\gamma_k/2}$ for Crank Nicholson and examine the results. Here is what we find.





We see that $\mathbf{B}$ has several large negative eigenvalues, the largest being ≈ -73. Consequently the explicit method is unstable. The Crank-Nicholson and implicit methods are stable. In fact this follows from $\gamma_k < 0$.

We might ask how small Δt needs to be (all other paramters being the same) in order for the explicit method to be stable. By calculating as above for $\Delta t = .005$ ($\alpha = .5$) we find that it is still unstable, but for $\Delta t = .001$ it is stable. This is consistent with what we found for the heat equation: the explicit method will be stable if Δt is small enough, relative to Δs. But no such restriction is necessary for the implicit and Crank-Nicholson methods.

Our eigenvalue calculation reveals one drawback that some authors have pointed out for the Crank-Nicholson method. The large negative γ_k correspond to eigenvalues $\lambda_k \approx -1$. While these still have $|\lambda_k| < 1$, so that the method is stable, the contributions to $\mathbf{M}^m$ from these "modes" will alternate in sign like $(-1)^m$, before the $|\lambda_k|^m \to 0$ takes over (which happens rather slowly since $|\lambda_k| \approx 1$). This means we can expect the error in a Crank-Nicholson calculation to oscillate with m. For the implicit method, from $\gamma_k < 0$ it follows that all $0 < \lambda_k$, so it will not have this oscillatory behavior. Tavella & Randal [57] consider "multi-level" methods that use the v-values from more than one apreviously computed t_m to compute the next time, and show that this can remove the oscilatory phenomon, while maintaining the second order accuracy and stability with no stepsize constraints.

5.3 Some Alternatives

Mathematically we know that a solution of the Black-Scholes equation is uniquely determined by the terminal value function $\phi(s)$. The values of $v(0, t)$ don't need to be specifed in advance but are in fact are determined after the solution is formed: $v(0,t) = \lim_{s\to 0} v(s,t)$. But our finite difference methods, applied on $[s^{\mathrm{L}}, s^{\mathrm{H}}]$ *do* require that we provide values for $v(s^{\mathrm{L}},t)$ fand $v(s^{\mathrm{H}},t)$ in the form of boundary value functions v^{L} and v^{H}. In the case of $s^{\mathrm{L}} = 0$ we are fortunate to know the exact formula from the corollary in Section 3.3:

$$v(0,t) = e^{-r(T-t)}\phi(0).$$

In fact this is what we would deduce by formally plugging $s = 0$ into the Black-Scholes equation:

$$v_t(0,t) - rv(0,t) = 0, \text{ with } v(0,T) = \phi(0). \tag{5.8}$$

For $s^{\mathrm{H}}(t)$ we have relied on the approximate formula

$$v(s,t) \approx c_1 s + c_2 e^{-r(T-t)} = v^{\mathrm{H}}(t),$$

valid for sufficiently large s provided $\phi(s) = c_1 s + c_2$ for s larger than some K.

There are alternatives to relying on approximate formulas for $v^{\mathrm{L}}(t)$ and $v^{\mathrm{H}}(t)$. The idea is to find some sort of discrete equation to help us solve for v_0^m and v_{N+1}^m. For $s^{\mathrm{L}} = 0$ we can just use a discrete approximation to (5.8), using the same discrete approximation to $v_t(0,t_m)$ as for the other s_n values.

- $\dfrac{v_0^m - v_0^{m-1}}{\Delta t} = rv_0^m$ for the explicit method: $v_0^{m-1} = (1 - r\Delta t)v_0^m$.
- $\dfrac{v_0^m - v_0^{m-1}}{\Delta t} = rv_0^{m-1}$ for the implicit method: $(1 + r\Delta t)v_0^{m-1} = v_0^m$.
- $\dfrac{v_0^m - v_0^{m-1}}{\Delta t} = r\dfrac{1}{2}(v_0^m + v_0^{m-1})$ for the Crank-Nicholson method: $(1 + r\Delta t/2)v_0^{m-1} = (1 - r\Delta t/2)v_0^m$.

Instead of the boundary condition $v_0^m = v^{\mathrm{L}}(0)$ we can use the appropriate choice of the above equations. The effect of this is to include a new $n = 0$ component to $\mathbf{v}^m$ and add a new top row to the matricies $\mathbf{I} \pm \mathbf{B}$, eliminating the $\mathbf{b}^{\mathrm{L}}$ terms from the descriptions of the methods on page 50. As a result we would no longer need $v^{\mathrm{L}}(t)$ at all. Since our formula for $v^{\mathrm{L}}(t)$ when $s^{\mathrm{L}} = 0$ is exact, this does not seem very beneficial, because we have replaced our exact formula for $v^{\mathrm{L}}(t)$ by a discrete approximation to the equation (5.8). The only benefit is that the finite difference formula is cleaner and a little simpler to implement.

A similar device is often used to eliminate the boundary condtion at s^{H} (and at s^{L} if it is positive but close to 0). The idea is that for large (or small) s our approximate formula $v(s,t) \approx c_1 s + c_2 e^{-t(T-t)}$ suggests that $v_{ss}(s_N,t) \approx 0$. In our second order approximation this becomes

$$\frac{v(s_{N+1},t) - 2v(s_N,t) + v(s_{N-1},t)}{\Delta s^2} \approx 0, \text{ or } v_{N+1}^m \approx 2v_N^m - v_{N-1}^m. \tag{5.9}$$

We can use this formula, instead of $v^{\mathrm{H}}(t_m)$, to produce the value of v_{N+1}^m needed in the finite difference approximation for $\mathcal{B}v(s_N,t_m)$. This eliminates the $\mathbf{b}^{\mathrm{H}}$ terms on page 50 and the need for $v^{\mathrm{H}}(t)$, while modifying the Nth row of $\mathbf{B}$. The tridiagonal structure of $\mathbf{B}$ is preserved, so we can still use our LU factorization method to implement the implicit and Crank-Nicholson versions of the method. You will work out the details in Problem 5.B at the end of the chapter. This approach is quite popular in the literature. A good discussion of it is in Tavella & Randall [57]. They make the point (page 121) that in some complicated situations it can be hard to produce decent approximate formulas for v^{H}, so these boundary value-free methods can be particulary useful. We will encounter that for Asian options.

We have seen that the boundary at $s = 0$ is actually rather nice, because we can produce an exact formula for $v(0,t)$. This works because the coefficients of v_s and v_{ss} in the Black-Scholes equation have a factor of s in them, so that they become 0 at $s = 0$, leaving us with the simplified equation (5.8). There is another change of variables in which the price variable $0 < s < \infty$ is replaced by a variable $0 < \xi < 1$ so that the coefficients of $\partial/\partial\xi$ and $\partial^2/\partial\xi^2$ in the resulting equation vanish at *both* $\xi = 0, 1$. This eliminates the need for an artificial upper boundary s^{H}, in the same way that $s^{\mathrm{L}} = 0$ is not artificial but the natural lower limit of s-values. Now something like (5.8) is possible at both boundaries $\xi = 0, 1$, so there is no need for artificial or approximate boundary values at all. This approach is developed in the text by Zhu, Wu, and Chern [63].

The Black-Scholes equation has a first order term, rsv_s while the heat equation does not. The presence of a first order term (called a "convection" term) along with a diffusion term (v_{ss}) is known to degrade the performance of finite difference methods. For this reason the heat equation formulation is sometimes considered preferable to the financial variables formulation. Another approach which improves the performance is to use "upwind differencing." This means using a carefully selected asymmetrical finite difference approximation for v_s. Most common is

$$v_s(s,t) \approx \frac{-3v(s,t) + 4v(s+\Delta s,t) - v(s+2\Delta s,t)}{2\Delta s}. \tag{5.10}$$

This should be used when the coefficient of v_s is positive (as it is for us: $rs > 0$). If the coefficient were negative we would replace Δs by $-\Delta s$ in (5.10) and use that instead. In future chapters we will encounter situations in which the coefficient changes sign. Then we use (5.10) for some s_n and the $-\Delta s$ version for others. This generally improves the performance of the finite difference calculation, but the resulting matricies $\mathbf{I} \pm \mathbf{B}/2$ are no longer tridiagonal; they have an additional nonzero superdiagonal two steps away from the main diagonal. You can find some discussion of this in [63] (see page 300) as well as [52], and [26] in the context of American options.

When the terminal value function $\psi(s)$ is nondifferentiable, as it typically is for us, that degrades the accuracy of the computed approximations. One way to remediate this is to use some reference function $w(s,t)$ and use the finite difference approach to compute the difference $v(x,t) - w(x,t)$ rather than v itself. If w is choosen so that $v - w$ is smooth, including $t = T$ then the computation of $v - w$ will be more accurate, and we just need to add w to the result to get v. This idea is developed in [63], where it is called the "singularity separating" approach.

Chapter 4 of [57] describes a number of refinements that can achieve an order of magnitude reduction of errors, such as cell-averaging of terminal values, and using cubic spline iterpolation when needed for discontinuous updates, such as across dividend payment times. In general, to develope high accuracy methods involves fine tuning the methods to the features of the praticular problem at hand, using a variety of special techniques such as those described above. A careful treatment of these things is beyond the scope of our discussion. We have tried simply to indicate what some of them are and provide a reference or two for interested readers to pursue.

5.4 Financial vs. Heat Variables

We have considered approaches to numerical approximation of pricing functions both in the original financial variables as well as in the equivalent heat equation formulation. In this section we want to indicate some of the pros and cons of these two approaches.

The use of heat variables has these advantages.

- The finite difference equations take a simple symmetric form, and stability can be verified explicitly.
- The absence of a first order "convection" term $\partial_x u$ in the equation helps the performance of the numerical methods. No special up-wind differencing schemes are needed.

The use of financial variables has these advantages.

- We ultimately want the results in financial variables. The regular grid spacing in financial variables will be more useful for interpolating values for nongrid points.
- We can use the natural boundary at $s = 0$, for which we usually have an *exact* formula for $v(0,t) = v^{\mathrm{L}}(t)$. (In heat variables, both boundary functions $u^{\mathrm{L}}(\tau)$ and $u^{\mathrm{H}}(\tau)$ will be approximations only.)
- When we consider path-dependent options (Asians and lookbacks) we will obtain modifications of the Black-Scholes equations in the financial variable setting. These will not convert to the heat equation as readily as the standard Black-Scholes equation.

5.5 Problems

Problem 5.A
In this problem you are to work out the modification of the Crank-Nicholson method based on using (5.8) instead of an explicit approximate formula for v^{H}, and its analogue base on $v_{ss}(s_1, t) = 0$ for v^{L}. You should end up with a revised version of (5.7) using a modified version of $\mathbf{B}$ and no $\mathbf{b}^{\mathrm{L}}$ or $\mathbf{b}^{\mathrm{H}}$ terms. Write revisions of `genB.m` and `cnf.m` and test it on a call option to see how it works.
. . . . `ABdry`

Problem 5.B
Explore the stability of the version of our methods using the $\mathbf{B}$ from Problem 5.A. Take $s^{\mathrm{L}} = 0$, $s^{\mathrm{H}} = 20$, $r = .05$, $\sigma = .3$, and $N = 100$. Choose Δt, generate $\mathbf{B}$ and calculate the list of its eigenvalues (`eig(full(B))`). Following what we did on page 52 you can convert this to the list of eigenvalues for the appropriate $\mathbf{M}$ and then find the largest (`max(abs(...))`). Experiment with different values of Δt. How small does Δt need to be for the explicit method to be stable? What do you find regarding the stability of the implicit and Crank-Nicolson methods?
. . . . `H`

Problem 5.C
Compare the accuracy of the Crank-Nicolson method for a standard European call option, using explicit boundary value approximations $v^{\mathrm{L}}(t) = 0$, $v^{\mathrm{H}}(t) = s^{\mathrm{H}} - Ke^{-r(T-t)}$, to the alternate version of Crank-Nicolson from the preceeding problem. I.e. implement each method in Matlab and compare the computed results with the exact values. Try s^{H} moderately close to K as well as significantly larger than K. Which method produces more accurate results? (You choose values of the parameters to use.)
. . . . `I`

Problem 5.D
Implement the Crank-Nicolson method in financial variables using the SOR approach in place of the LU method. Run it first for a standard European call option (parameters of your choice). Find the typical value of the relaxation parameter ω used by the algorithm, and the typical number of iterations used in the SOR loop. Next set it up for a "cash-or-nothing" option – this is a (European) option whose terminal value is given by

$$\phi(s) = \begin{cases} C & \text{if } s \geq K \\ 0 & \text{if } s < K, \end{cases}$$

where C and K are positive constants. (You can choose parameter values to work with, and will need to work out the approximate boundary value expressions.) Are the typical values of ω and the iteration count any different in this case?
. . . . `J`

Problem 5.E
Use a finite difference calculation to compute and plot $v(s, 0)$ where v is the pricing function of the put on call from Problem 3.G. Use $K_1 = 15$, $K_2 = 10$, $T_1 = 1$, $T_2 = 2$, $r = .03$, $\sigma = .2$. You decide on the rest of the particulars. Try to choose a range $[s^{\mathrm{L}}, s^{\mathrm{H}}]$ which is appropriate considering the features of the problem.
. . . . `Compound`

Chapter 6

Markov Chain Methods

We have been finding the pricing function implied by the risk-neutral formula

$$v(s,t) = e^{-r(T-t)} E[\phi(S_T) \,|\, S_t = s]$$

by characterizing $v(s,t)$ as a solution of the Black-Scholes equation and then constructing approximations to the solution by means of finite differences. The finite difference approach reduces consideration to just a finite set of possible values for the variables: $s = s_n$, $t = t_m$. Instead of "discretizing" the PDE, we could consider discretizing the process S_t itself, approximating it by a process that assumes only a finite set of possible values, at a finite set of possible times. Since S_t is a Markov it seems natural to use a Markov chain. We will use the same set of times t_m and price values s_n as in our finite difference chapter. The idea is to find such a chain ξ_m, $m = 0, 1, \ldots, M$ taking values in the set $\{0, 1, \ldots, N+1\}$ so that

$$S_{t_m} \approx s_{\xi_m}.$$

A Markov chain is determined by its transition probabilities:

$$p_{k,\ell} = P(\xi_{m+1} = \ell \,|\, \xi_m = k).$$

If we can pick the transition probabilities to that $S_{t_m} \approx s_{\xi_m}$ is a good approximation (in some sense), then we can hope to approximate

$$v(s_n, t_m) = e^{-r(T-t_m)} E[\phi(S_T) \,|\, S_{t_m} = s_n] \approx e^{-r\Delta t(M-m)} E[\phi(s_{\xi_M}) \,|\, \xi_m = n] = v_n^m.$$

Calculating the right side will be easy, very like our explicit finite difference method calculations.

This chapter considers the problem of approximating option prices from this point of view. We will see that (depending on Δs and Δt) our finite difference calculations can be viewed in these terms, even the implicit and Crank-Nicholson Methods. In particuar this implies some sufficient condtions for stability and provides another approach to finding good finite difference approximations.

6.1 Binomial Trees and Black-Scholes

Random walks (often called bionomial trees in the finance literature) are often considered as first models to explain the idea of arbitrage-free pricing, and the role of risk neutral probabilities, without reference to the Balck-Scholes model. In other words if a random walk is assumed as the description of the random evolution of the stock price, then no-arbitrage considerations lead to risk-neutral transition probabilities which are used to calculate option prices at each node in the tree, working from $t = T$ backward to the initial time $t = 0$. But if the tree is designed carefully it can be viewed as an approximation to the continuous time Black-Scholes model itself (under the risk-neutral probability). Some discussion of this can be found in Higham [23] or Tavella [56] for instance.

6.2 Finite Difference Methods and Markov Chains

We are going to explore the interpretation of our finite difference methods as Markov chain calculations. This point of view lends additional insight to those calculations.

We want to start with the explicit method for the heat equation (2.7). We have already observed a stochastic interpretation of the heat equation, but it involves a reversal of the time axis: the $t = T - \tau$ in (2.10). It will be more natural for the present discussion if we write the heat equation in terms of x, t, rather than x, τ.

$$u_{xx} + u_t = 0, \text{ with } u(x,T) = \psi(x).$$

Now our finite difference calculations start at $t_M = T$ and proceed through decreasing time steps $t_m \to t_{m-1}$, in the same way as for financial variables. The explicit finite difference method for the heat equation takes the form

$$u_n^{m-1} = \alpha u_{n+1}^m + (1-2\alpha)u_n^m + \alpha . u_{n-1}^m. \tag{6.1}$$

We want to view the coefficients α, $1-2\alpha$, and α as transition probabilities for a Markov chain. We can do this provided $\alpha \leq \frac{1}{2}$ so that the probabilities are nonnegative. They *do* add up to 1 as probabilites should.

Let ξ_m be the random walk that moves through the integers in the following way. If $\xi_{m-1} = n$ then the transition $\xi_{m-1} \to \xi_m$ will occur according to the following transition probabilities.

- $n \to n-1$ occurs with probability $p_{n,n-1} = \alpha$.
- $n \to n$ occurs with probability $p_{n,n} = 1 - 2\alpha$.
- $n \to n+1$ occurs with probability $p_{n,n+1} = \alpha$.

All other transitions have probability 0. In terms of this random walk our finite difference formula (6.1) can be expressed as

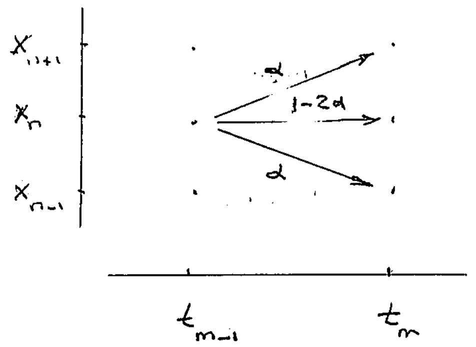
$$u_n^{m-1} = E[u_{\xi_m}^m \,|\, \xi_{m-1} = n].$$

Using the tower law,

$$\begin{aligned} u_n^{m-1} &= E[u_{\xi_m}^m \,|\, \xi_{m-1} = n] \\ &= E[u_{\xi_{m+1}}^{m+1} \,|\, \xi_{m-1} = n] \\ &\;\;\vdots \\ &= E[u_{\xi_M}^M \,|\, \xi_{m-1} = n]. \end{aligned}$$

We can convert the random walk ξ_m to a Markov chain on our finite difference grid by

$$X_{t_m} = x_{\xi_m}.$$



Since $u_k^M = \psi(x_k)$, we can write the result of the explicit method calculation as

$$u_n^{m-1} = E[\psi(X_T) \,|\, X_{t_{m-1}} = n].$$

We observe how much this resembles (2.11):

$$u(x,t) = E[\psi(\sqrt{2}W_t) \,|\, \sqrt{2}W_t = x]; \tag{6.2}$$

we simply have X_t in place of $\sqrt{2}W_t$.

Thus we can view the explicit method finite difference calculation as the result of making the approximation $X_t \approx \sqrt{2}W_t$ in the conditional expectation (6.2) which produces the exact value of $u(x,t)$. So the accuracy of the explicit method should be related to the accuracy of this approximation – the better X_t approximates $\sqrt{2}W_t$ the better (6.1) will produce approximations to the heat equation.

As evidence that X_{t_m} is a reasonable approximation to $\sqrt{2}W_{t_m}$ we offer the following.

$$E[X_{t_m} - X_{t_{m-1}}] = E[x_{\xi_m} - x_{\xi_{m-1}}] = \Delta x(1\cdot\alpha + 0\cdot(1-2\alpha) - 1\cdot\alpha) = 0$$

which agrees with

$$E[\sqrt{2}(W_{t+\Delta t} - W_t)] = 0.$$

Checking the second moments of the increments,

$$E[(X_{t_m} - X_{t_{m-1}})^2] = E[(x_{\xi_m} - x_{\xi_{m-1}})^2] = \Delta x^2(1\cdot\alpha + 0\cdot(1-2\alpha) + 1\cdot\alpha) = 2\alpha\Delta x^2 = 2\Delta t$$

which agrees with

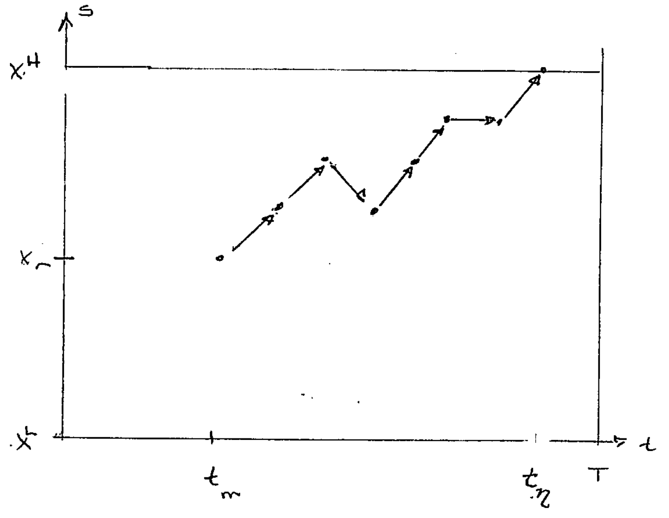
$$2E[(W_{t+\Delta t} - W_t)^2] = 2\Delta t.$$

To use prescribed boundary conditions at $x_0 = x^{\mathrm{L}}$ and $x_{N+1} = x^{\mathrm{H}}$ we stop the chain as soon as it reaches either boundary and evaluate u there without letting ξ_m continue to evolve. Let η be the value of $m < M$ at which ξ_m hits 0 or $N+1$ for the first time (a discrete stopping time), or else M if ξ_m does not reach the boundaries. Then take

$$U^\eta = \begin{cases} u^{\mathrm{H}}(x^{\mathrm{H}}, t_\eta) & \text{if } \eta < M \text{ and } \xi_\eta = N+1 \\ \psi(\xi_N) & \text{if } \eta = M \\ u^{\mathrm{L}}(x^{\mathrm{L}}, t_\eta) & \text{if } \eta < M \text{ and } \xi_\eta = 0. \end{cases}$$

Then our explicit finite difference calculation can be interpreted as

$$u_n^{m-1} = E[U^\eta \,|\, \xi_{m-1} = n].$$



Implicit methods can also be given a Markov chain interpretation. Consider the implicit method for the heat equation.

$$-\alpha u_{n-1}^{m-1} + (1+2\alpha)u_{n+1}^{m-1} - \alpha u_n^{m-1} = u_n^m.$$

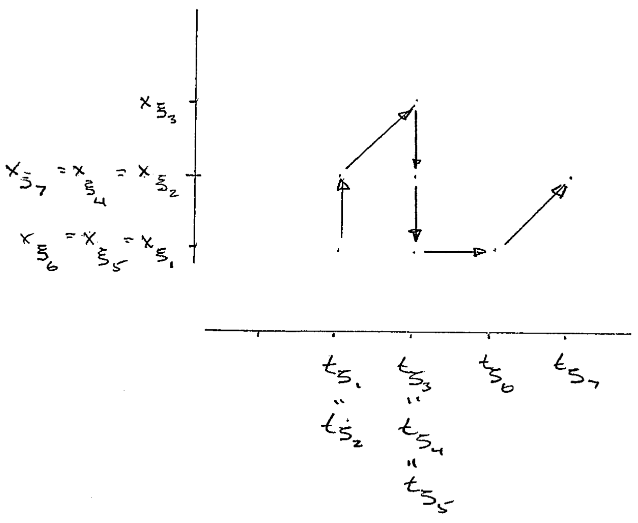
This can be rearranged as

$$u_n^{m-1} = \frac{\alpha}{1+2\alpha}u_{n-1}^{m-1} + \frac{1}{1+2\alpha}u_n^m + \frac{\alpha}{1+2\alpha}u_{n+1}^{m-1}.$$

This too can be interpreted as the single-step mean for a Markov chain on our grid, with the new feature that the chain can move "vertically" to a different state value while staying at the same time value. We will call this a *space-time* Markov chain, and will denote it by (ξ_k, ζ_k). Now k is simply the counter for the chain (with no fixed connection to actual time); ξ_k gives the state index and ζ_k gives the time index of the chain. There are two types of transitions:

- Same-time transitions: $(\xi_k, \zeta_k) = (n, m) \to (n \pm 1, m) = (\xi_{k+1}, \zeta_{k+1})$ occuring with probability $\frac{\alpha}{1+2\alpha}$;
- Next-time transitions: $(\xi_k, \zeta_k) = (n, m) \to (n, m+1) = (\xi_{k+1}, \zeta_{k+1})$ occuring with probability $\frac{1}{1+2\alpha}$.

Observe that these are a valid set of probabilities for any $\alpha > 0$; no gridsize restrictions are needed to interpret the implicit method probabilistically.


x_{ξ_3}
$x_{\xi_7} = x_{\xi_4} = x_{\xi_2}$
$x_{\xi_6} = x_{\xi_5} = x_{\xi_1}$
$t_{\xi_1} = t_{\xi_2}$
$t_{\xi_3} = t_{\xi_4} = t_{\xi_5}$
t_{ξ_6}
t_{ξ_7}

We can interpret our calculations by letting our space-time chain start at $(\xi_k, \zeta_k) = (n, m)$ and letting it run until the first time η that it reaches the border of our grid-strip.

$$u_n^m = E[U^\eta \,|\, (\xi_k, \zeta_k) = (n, m)].$$

6.3 Financial Variables

Now consider the explicit method in financial variables:

$$v_n^{m-1} = a_n v_{n-1}^m + (1 - b_n)v_n^m + c_n v_{n+1}^m.$$

To interpret the coefficients as transition probabilities we would need $a_n \geq 0$, $1 - b_n \geq 0$ and $c_n \geq 0$, using the coefficient formulas from (5.1). We see that $c_n \geq 0$ is always true but the other two entail restrictions on the stepsizes: $a_n \geq 0$ is equivalent to

$$\frac{r}{\sigma^2} \leq \frac{s_n}{\Delta s} \tag{6.3}$$

for every n. If we insist on $s^{\mathrm{L}} = s_0 = 0$ this is equivalent to

$$\sigma^2 \geq r.$$

But more generally this can be achieved by fixing $s_1 > 0$ and then taking Δs sufficiently small.

The inequality $1 - b_n \geq 0$ is equivalent to

$$\frac{s_n}{\Delta s} \leq \frac{\sqrt{\Delta t^{-1} - r}}{\sigma}$$

We could insure this for all n by making Δt sufficiently small.

The implicit method is

$$-a_n v_{n-1}^{m-1} + (1 + b_n) v_n^{m-1} - c_n v_{n+1}^{m-1} = v_n^m,$$

which rearranges as

$$v_n^{m-1} = \frac{1}{(1 + b_n)} [a_n v_{n-1}^{m-1} + v_n^m + c_n v_{n+1}^{m-1}].$$

This can be interpreted probabilistically whenever $a_n \geq 0$, which is (6.3).

The Crank-Nicholson method is

$$-\frac{1}{2} a_n v_{n-1}^{m-1} + (1 + \frac{1}{2} b_n) v_n^{m-1} - \frac{1}{2} c_n v_{n+1}^{m-1} = \frac{1}{2} a_n v_{n-1}^m + (1 - \frac{1}{2} b_n) v_n^m + \frac{1}{2} c_n v_{n+1}^m.$$

We rearrange this as

$$v_n^{m-1} = \frac{1}{1 + \frac{1}{2} b_n} \left[\frac{1}{2} a_n v_{n-1}^{m-1} + \frac{1}{2} c_n v_{n+1}^{m-1} + \frac{1}{2} a_n v_{n-1}^m + (1 - \frac{1}{2} b_n) v_n^m + \frac{1}{2} c_n v_{n+1}^m \right].$$

This can be interpreted as a space-time Markov chain provided $a_n \geq 0$ and $1 - \frac{1}{2} b_n \geq 0$. The first of these is (6.3) again and the second is equivalent to

$$\frac{s_n}{\Delta s} \leq \frac{\sqrt{2/\Delta t - r}}{\sigma}.$$

In each case we see that a step size constraint is needed to interpret the methods in terms of Markov chains. The implicit method is least restrictive since only (6.3) is needed. For both the explicit and Crank-Nicholson methods there is a second constraint. One way to avoid these step size restrictions is to use a different finite difference approximation to v_s. Instead of $v_s(s_n, t_m) \approx \frac{v_{n+1}^m - v_{n-1}^m}{2\Delta s}$, suppose we use

$$v_s(s_n, t_m) \approx \frac{v_{n+1}^m - v_n^m}{\Delta s}.$$

This is an example of "up-wind" differencing, as mentioned earlier. As we said previously, it is known to have better computational properties in various circumstances. However this is only a $\mathcal{O}(\Delta s)$ approximation, not second order as was the cetral difference formula. So we are giving up some accuracy in making this change. But it has the benefit of giving us a probabilistic interpretation. It leads to revised coefficient formulas

$$\begin{aligned}
a_n' &= \frac{1}{2} \sigma^2 s_n^2 \alpha, \\
b_n' &= \sigma^2 s_n^2 \alpha + r s_n \beta + r \Delta t, \\
c_n' &= \frac{1}{2} \sigma^2 s_n^2 \alpha + r s_n \beta.
\end{aligned}$$

Now $a_n' \geq 0$ and $c_n' \geq 0$. For explicit and Crank-Nicholson methods we would still need an upper bound on b_n'. But the implicit method takes the form

$$v_n^{m-1} = \frac{a_n'}{1 + b_n'} v_{n-1}^{m-1} + \frac{1}{1 + b_n'} v_n^m + \frac{c_n'}{1 + b_n'} v_{n+1}^{m-1},$$

in which all the coefficients are positive, so no step size restriction is needed.

There is a second issue in the interpretation of the above methods in terms of Markov chains: the coefficients don't sum to 1. Let's consider the standard implicit method in financial variables.

$$\frac{a_n}{1+b_n} + \frac{1}{1+b_n} + \frac{c_n}{1+b_n} = \frac{1+b_n - r\Delta t}{1+b_n} < 1.$$

So the terms on the left do not comprise a full set of probabilities. We can remedy this by using a discount factor $0 < \rho < 1$, chosen so that

$$\frac{a_n}{1+b_n} + \rho^{-1}\frac{1}{1+b_n} + \frac{c_n}{1+b_n} = 1. \tag{6.4}$$

Solving this for for ρ we find $\rho = (1 + r\Delta t)^{-1}$. We can now use the three terms in (6.4) as the transition probabilities for a space-time Markov chain *with no gridsize restrictions.* A single step of the implicit method formula takes the form

$$v_n^{m-1} = E[\rho^{(\zeta_{k+1}-\zeta_k)} v_{\xi_{k+1}}^{\zeta_{k+1}} \,|\, (\xi_k, \zeta_k) = (n, m-1)].$$

The idea is that the terms of the expection for which the time component ζ moves forward, $\zeta_k = m-1 \to \zeta_{k+1} = m$, the discount factor ρ is applied. The result of the full implicit method takes the form

$$v_n^m = E[\rho^{(\zeta_\eta - \zeta_k)} v_{\xi_\eta}^{\zeta_\eta} \,|\, (\xi_k, \zeta_k) = (n, m-1)].$$

Here η would be the first time the space-time chain reaches the boundary of our strip.

We recognize that this is an approximation to our risk-neutral formula based on

$$\rho \approx e^{-r\Delta t},$$

together with $S_{t_{\zeta_k}} \approx s_{\xi_k}$. In particular we see that the extra factors of ρ turn out to be just what is needed to account for the discounting in the risk-neutral formula. For the explicit or Crank-Nicholson methods the appropriate formula for ρ works out a little differently, but the same idea still applies. In general, if the error in these approximations converges to 0 as $\Delta t \to 0$ and $\Delta s \to 0$, then the results of the finite difference calculation will converge to the value v given by the risk-neutral formula. The idea of developing Markov chain approximations based on such moment approximations has been explored by Duan, Gauthier and Simonato [15]. Kushner and Dupuis [38] provide the analysis needed to prove the convergence of the finite difference calculations to the true $v(s,t)$ by means of these ideas.

6.4 Implicataions for Stability

We have noted that the Markov chain interpretation provides a way to theoretically prove that the results of our finite difference calculations do indeed to converge to the value defined by the risk-neutral formula, and in a "direct" manner that does not depend on the connection with partial differential equations. There is another, more practical consequence of a Markov chain interpretation: stability. Whenever a finite difference method can be interpred as a Markov chain approximation, the method is stable. In this section we will explain this assertion. A method can be stable even if the grid sizes are such that a Markov chain interpretation is not possible, so this is not necessary for stability, but it does provide a sufficient conditon.

To explain this, we describe the same-time and next-time transition probabilities fseparately. Let p_{ij} be the probability of the next-time transition

$$\xi_k = i,\ \zeta_k = m-1 \quad \to \quad \xi_{k+1} = j,\ \zeta_{k+1} = m,$$

and q_{ij} be the probability of the same-time transition

$$\xi_k = i,\ \zeta_k = m-1 \quad \to \quad \xi_{k+1} = j,\ \zeta_{k+1} = m-1.$$

Let $\mathbf{P} = [p_{ij}]$ and $\mathbf{Q} = [q_{ij}]$ be the matricies of these probabilities, and $\mathbf{R} = \mathrm{diag}(\rho_i)$ be the diagonal matrix of the discount factor(s), as in (6.4). In these terms the method can be written in matrix-vector form

$$\mathbf{v}^{m-1} = \mathbf{Q}\mathbf{v}^{m-1} + \mathbf{R}\mathbf{P}\mathbf{v}^m + \mathbf{b},$$

where $\mathbf{b}$ is the contribution of the boundary conditions. To investigate stability we usually just consider $\mathbf{b} = \mathbf{0}$. What we want to show is that given $\mathbf{v}^m$, the solution $\mathbf{v}^{m-1}$ of the above obeys $\|\mathbf{v}^{m-1}\| \le \|\mathbf{v}^m\|$; see (4.7). If we solve for $\mathbf{v}^{m-1}$ we find

$$\mathbf{v}^{m-1} = (\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R}\mathbf{P}\mathbf{v}^m.$$

Here is a brief outline of how stability follows from this formula. First observe that there is a value $0 < r < 1$ so that $\mathbf{Q}\mathbf{1} \le r\mathbf{1}$ (coordinatewise). This is because for each i there is a positive probability p_{ij} of making a time-forward jump, so the sum of the same-time jump probabilities $\sum_j q_{ij}$ is less than 1. It follows that $0 \le \mathbf{Q}^n\mathbf{1} \le r^n\mathbf{1}$. Using this we see that the infinite series $\sum_0^\infty \mathbf{Q}^n$ converges. It is easy to check that this series gives the inverse: $(\mathbf{I} - \mathbf{Q})^{-1} = \sum_0^\infty \mathbf{Q}^n$. This series representation shows us that all entries of $(\mathbf{I} - \mathbf{Q})^{-1}$ are nonnegative. Therefore all entries of $(\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R}\mathbf{P}$ are nonnegative.

Next, since $(\mathbf{Q} + \mathbf{P})\mathbf{1} = \mathbf{1}$ and all $\rho_i \le 1$ we have

$$\mathbf{Q}\mathbf{1} + \mathbf{R}\mathbf{P}\mathbf{1} \le \mathbf{1}, \text{ which we can rewrite as } \mathbf{R}\mathbf{P}\mathbf{1} \le (\mathbf{I} - \mathbf{Q})\mathbf{1}.$$

Since all entries of $(\mathbf{I} - \mathbf{Q})^{-1}$ are nonnegative it follows that

$$(\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R}\mathbf{P}\mathbf{1} \le \mathbf{1}.$$

So all entries of $(\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R}\mathbf{P}\mathbf{1}$ are nonnegative and each row sums to at most 1. In other words if $[h_{ij}] = \mathbf{H} = (\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R}\mathbf{P}$, then $0 \le h_{ij}$ and $\sum_j h_{ij} \le 1$. Now $\mathbf{v}^{m-1} = \mathbf{H}\mathbf{v}^m$. Pick any entry v_n^{m-1}. Then (using the sup-norm $\|\mathbf{u}\| = \max(|u_n|)$) we have

$$|v_n^{m-1}| = \left|\sum_j h_{nj} v_j^m\right| \le \sum_j h_{nj}|v_j^m| \le \sum_j h_{nj}\|\mathbf{v}^m\| \le \|\mathbf{v}^m\|.$$

This implies $\|\mathbf{v}^{m-1}\| \le \|\mathbf{v}^m\|$, proving stability.

Problem 6.A
Check that for the Markov chain approximation associated with the implicit method for the heat equation, the mean increment is $o(\Delta t)$ and the second moment should is $2\Delta t + o(\Delta t)$.

. . . . $\boxed{\texttt{Hcst}}$

Problem 6.B

a) Consider the Crank-Nicholson method for the heat equation. We know that, like the implicit method, the Crank-Nicholson method is stable regardless of the stepsizes Δx and Δt. Are limitations on the stepsizes necessary in order to interpret the method in terms of a Markov chain? Describe the jumps and their probabilities for the Markov chain.

b) Now consider the Crank-Nicholson method for the Black-Scholes equation. Arrange the method so that it can be interpreted in terms of a state-time Markov chain (ξ_n, ζ_n). (Assume that any step size and parameter restrictions needed to validate a Markov chain interpretation are satisfied.) Observe that the quantities that appear in the positions of the probabilities do not sum to 1, so like our discussion of both the explicit and implicit methods in financial variables, a discount factor ρ needs to be included in order to formulate a chain interpretation. This leads to a formula

$$v_n^m = E_{m,n}[\rho^{\zeta_\eta/\Delta t} v(\xi_\eta, T - \zeta_\eta)].$$

Find ρ. The expression for ρ should be such that

$$\rho^{1/\Delta t} \to e^{-r}, \quad \text{as } \Delta t \to 0.$$

Confirm this.

. . . . $\boxed{\texttt{BB}}$

Chapter 7

Dividends

In this chapter we generalize what we have done to a dividend-paying stock.

7.1 Continuous Dividends

Let's go back to our original description (1.1) of the stock, using the real-world probability measure and the associated Brownian motion:

$$dS_t = \alpha S_t\, dt + \sigma S_t\, dW_t^{\circ}.$$

First assume that the stock pays dividends at a continuous rate δS_t where $\delta > 0$ is a constant[1]. What we mean by this is that if D_t denotes the cumulative amount paid in dividends (on one share) between time 0 and time t, then $D_t = \int_0^t \delta S_u\, du$, or in brief,

$$dD_t = \delta S_t\, dt.$$

This is nice mathematically, but it is more realistic to consider dividends which are paid at discrete times. We will discuss that in Section 7.2. The technique we develop for the case of continuous payment will be valuable there as well.

The pricing of financial derivatives based on such a stock requires a modified version of the Black-Scholes equation (1.5). To make that point, suppose we (naively) tried to price derivatives in the same way as before, simply ignoring the dividends. Consider for instance a simple forward contract for one share of stock in exchange for K in cash at time T. Our usual pricing function (for a non-dividend-paying stock) is

$$v(s,t) = s - Ke^{-r(T-t)}.$$

If the market really offered these contracts for that price, but the stock paid dividends, there would be an arbitrage opportunity in the market. At time $t = 0$ suppose we do the following:

- Sell one contract, receiving $S_0 - Ke^{-rT}$.
- Buy one share of stock, spending S_0.
- Borrow Ke^{-rT} by selling bonds to finance the difference.

We have spent a total of \$0 on this transaction, but now have a portfolio consisting of the stock, our obligation as the writer of the contract, and our obligation to pay the bond holder at time T. Consider the value of this portfolio at time T. It consists of

- Our obgligation to the buyer of the contract, which is worth $-(S_T - K)$.
- One share of stock, worth S_T.

[1]More generally we could assume that $\delta = \delta(S_t)$ is a function of the stock price, but we will be content with the simpler case of a constant δ.

- The dividends $D_T > 0$ on the stock we have owned.
- Our obligation to pay back the bonds we sold, $-K$.

Totalling these up, we have earned $D_T > 0$ from an initial expenditure of \$0. This is an arbitrage opportuity, and tells us that the price of $v(S_0, 0)$ the market paid for the forward contract was too high.

The source of the problem here is that the notion of a self-financing portfolio on which the derivation of the Black-Scholes equation was based is no longer appropriate, because it ignores the revenue produced by the dividend payments. The correct notion of self-financing for a portfolio $V_t^h = h_t^B B_t + h_t^S S_t$ using our dividend-paying stock is

$$dV_t^h = h_t^B \, dB_t + h_t^S (dS_t + \delta S_t \, dt). \tag{7.1}$$

Based on this we could revise the derivation of Section 1.2 to find the correct revision of the Black-Scholes equation to use with such a dividend-paying stock. But there is an alternate approach which will account for the case of discrete dividends as well. The idea is to use the dividend stream to continually purchase additional shares of stock, creating a *reinvested stock portfolio* $\tilde{S}_t = h_t S_t$, where $h_0 = 1$ and increases as dividends are used to purchase new shares. The dividend revenue of this portfolio will increase at the rate

$$h_t \frac{dD_t}{dt} = h_t \delta S_t$$

allowing new shares to be purchased at the rate

$$h_t' = \frac{h_t \delta S_t}{S_t} = \delta h_t.$$

Since $h_0 = 1$ we deduce that $h_t = e^{\delta t}$. Thus

$$\tilde{S}_t = h_t S_t, \text{ where } h_t = e^{\delta t}.$$

Observe that

$$\begin{aligned} d\tilde{S}_t &= h_t \, dS_t + S_t \, dh_t \text{ (because } dS_t \, dh_t = 0 \text{)} \\ &= h_t \, dS_t + S_t \delta h_t \, dt \\ &= h_t \left(dS_t + \delta S_t \, dt \right), \end{aligned}$$

agreeing with he description (7.1) of self-financing for the portfolio $h_t S_t$. We now view $\tilde{S}_t$ as another *non-dividend paying* asset in the market, with stochastic differential

$$d\tilde{S}_t = (\alpha + \delta) \tilde{S}_t \, dt + \sigma \tilde{S}_t \, dW_t^{\text{o}}.$$

If we price everything as functions of $\tilde{S}_t$ and t, then our former pricing calculations for a non-dividend paying stock apply. In the market consisting of B_t and $\tilde{S}_t$ as the two primary assets, the risk-neutral probability measure would make $\tilde{S}_t / B_t$ a martingale, and $\tilde{S}_t$ would be described by

$$d\tilde{S}_t = r \tilde{S}_t \, dt + \sigma \tilde{S}_t \, dW_t.$$

The market price of a (European) contingent claim with $V_T = X$ is given by the risk-neutral formula

$$V_t = B_t E[X / B_t \,|\, \mathcal{F}_t].$$

If the value of a (European) contingent claim is expressed as a function $\tilde{v}(\tilde{S}_t, t)$, then $\tilde{v}(\tilde{s}, t)$ should satisfy the usual Black-Scholes equation in the variables $\tilde{s}$ and t:

$$\frac{1}{2} \sigma^2 \tilde{s}^2 \tilde{v}_{\tilde{s}\tilde{s}} + r \tilde{s} \tilde{v}_{\tilde{s}} - r \tilde{v} + \tilde{v}_t = 0. \tag{7.2}$$

We want to express all this in terms of the actual stock price S_t. Since $S_t = \tilde{S}_t / h_t$ and $dh_t^{-1} = -\delta h_t^{-1} \, dt$, we find that with respect to the risk neutral probability the stock price obeys

$$\begin{aligned} dS_t &= \frac{1}{h_t} (r \tilde{S}_t \, dt + \sigma \tilde{S}_t \, dW_t) - \tilde{S}_t \delta \frac{1}{h_t} \, dt \\ &= (r - \delta) S_t \, dt + \sigma S_t \, dW_t. \end{aligned}$$

And with $(\tilde{s}, t)$ and (s, t) related by $\tilde{s} = e^{\delta t} s$, if $v(s,t) = \tilde{v}(\tilde{s}, t)$, then

$$\tilde{v}(\tilde{s}, t) = v(e^{-\delta t}\tilde{s}, t).$$

Using $\tilde{v}_{\tilde{s}} = e^{-\delta t} v_s$, $\tilde{v}_{\tilde{s}\tilde{s}} = e^{-2\delta t} v_{ss}$, and $\tilde{v}_t = v_t - \delta e^{-\delta t} v_s$, we find that (7.2) is equivalent to

$$\frac{1}{2}\sigma^2 s^2 v_{ss}(s,t) + (r-\delta) s v_s(s,t) + v_t(s,t) - r v(s,t) = 0. \tag{7.3}$$

This is the generalization of the Black-Schoes equation which should be used for all pricing calculations for stocks paying continuous dividends. Generalizing the notation of (2.1), we define the differential operator

$$\mathcal{B}^{\mathrm{D}} v = \frac{1}{2}\sigma^2 s^2 v_{ss} + (r-\delta) s v_s - r v,$$

so that we can write (7.3) in the more concise form

$$\partial_t v(s,t) + \mathcal{B}^{\mathrm{D}} v(s,t) = 0.$$

The pricing of European and barrier options proceeds just as before, but using $\mathcal{B}^{\mathrm{D}}$ instead of $\mathcal{B}$.

7.2 Discrete Dividends

For an individual stock dividends are typically paid at discrete moments in time, not continuously. We can handle these too, with some adjustments. Suppose there is a sequence of times $T_1 < T_2 < \cdots$ at which dividends are paid and that the amount of the payment at time T_i is

$$\delta S_{T_i-}.$$

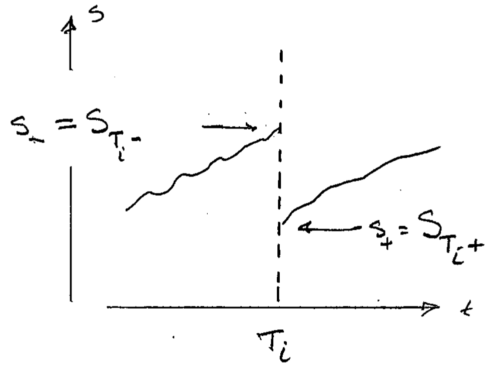
(The T_i- denotes the limit from the left.) Again $\delta > 0$ is a constant, but now we assume that $\delta < 1$ as well. A $\delta > 1$ would mean the stock pays a dividend greater than its market value (which would create an arbitrage opportuity). We will see below that $\delta = 1$ would mean the stock liquidates itself, paying out its value in cash and becoming worthless ($S_{T_1+} = 0$), at the first dividend payment time T_1. We will not bother ourselves with that extreme case, and so assume $0 < \delta < 1$.

In between the T_i the stock should evolve according to our usual description, $dS_t = \alpha S_t \, dt + \sigma S_t \, \delta W_t^{\mathrm{o}}$. But at a dividend payment time the price of the stock makes a discontinuous jump down. The value of the stock S_{T_i-} just prior the dividend payment must be the same as the value of the stock S_{T_i+} just after the dividend payment plus the dividend amount:

$$S_{T_i-} = S_{T_i+} + \delta S_{T_i-}.$$

(If this were not so there would exist an arbitrage opportunity.) As a consequence we find that

$$S_{T_i+} = (1-\delta) S_{T_i-}.$$



We can work out pricing calculations in the same way as for continuous dividends by thinking in terms of the value $\tilde{S}_t = h_t S_t$ of the pure stock portfolio that results from reinvesting the dividends. We start with $h_0 = 1$. At T_i we should have

$$\begin{aligned} h_{T_i+} &= h_{T_i-} + \frac{h_{T_i-}\delta S_{T_i-}}{S_{T_i+}} \\ &= \left(1 + \frac{\delta}{1-\delta}\right) h_{T_i-} \\ &= (1-\delta)^{-1} h_{T_i-}. \end{aligned}$$

In general we see that

$$h_t = (1-\delta)^{-n(t)},$$

where $n(t)$ is the number of $T_i \leq t$.

For $T_{i-1} < t < T_i$ between dividend times h_t is constant, so $\tilde{S}_t$ satisfies our usual stochastic equation $d\tilde{S}_t = \alpha \tilde{S}_t\,dt + \sigma \tilde{S}_t\,DW_t$. At a dividend time we have

$$\tilde{S}_{T_i+} = h_{T_i+} S_{T_i+} = (1-\delta)^{-1} h_{T_i-}(1-\delta) S_{T_i-} = h_{T_i-} S_{T_i-} = \tilde{S}_{T_i-}.$$

In other words $\tilde{S}_t$ is continuous across the T_i and therefore our usual stochastic equation

$$d\tilde{S}_t = \alpha \tilde{S}_t\,dt + \sigma \tilde{S}_t\,\delta W_t^{o}$$

in fact holds for all $0 \leq t$. So in terms of $\tilde{S}_t$ and t asset prices will follow

$$V_t = \tilde{v}(\tilde{S}_t, t),$$

where $\tilde{v}(\tilde{s}, t)$ satisfies our usual Black-Scholes equation in the variables $\tilde{s}, t$.

We would like to express this as a function $v(s,t)$ of the actual stock price S_t.

$$v(s,t) = \tilde{v}(\tilde{s}, t) \quad \text{where} \quad \tilde{s} = h_t s.$$

For t between the T_i we know that h_t is constant, so everything converts simply:

$$\tilde{s}\partial_{\tilde{s}}\tilde{v} = \frac{\tilde{s}}{h_t}\partial_s v = s\partial_s v, \quad \tilde{s}^2\partial_{\tilde{s}}^2 v = s^2 \partial_s^2 v, \quad \partial_t \tilde{v} = \partial_t v,$$

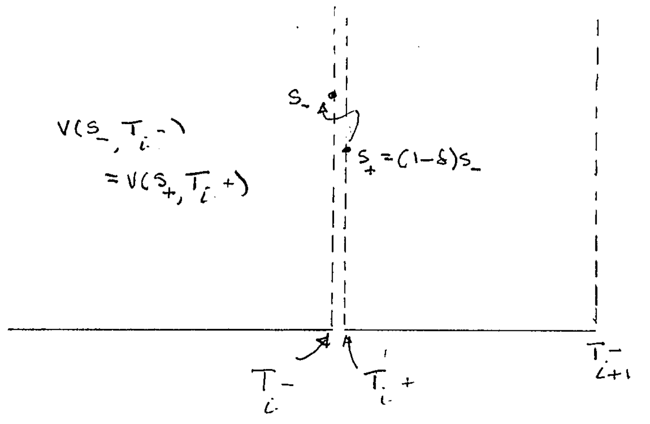
and we find that $v(s,t)$ satisfies our standard Black-Scholes equation. At $t = T_i$ there is a discontinuity in the relationship between $\tilde{s}$ and s which becomes a discontinuity in v itself. Because $h_{T_i+} = (1-\delta)^{-1} h_{T_i-}$, the values s_- and s_+ corresponding to $\tilde{s}$ just before and after T_i are related as follows.

$$s_+ = \frac{\tilde{s}}{h_{T_i+}} = (1-\delta)\frac{\tilde{s}}{h_{T_i-}} = (1-\delta)s_-.$$

So

$$\begin{aligned} v(s_-, T_i-) = \tilde{v}(\tilde{s}, T_i) &= v(s_+, T_i+) \\ &= v((1-\delta)s_-, T_i+). \end{aligned}$$

The value of the option the instant prior to the dividend payment for stock price s_- equals the value just after the payment for stock price $s_+ = (1-\delta)s_-$. This is a discontinuity in the function v, but the owner of the option will not see a jump in the value of their option, because the stock price will make exactly the same jump across T_i: $S_{T_i+} = (1-\delta)S_{T_i-}$.



To implement this numerically, here is how the calculation should be organized.

- Start with $v(s,T) = \phi(s)$ and $t = T$.
- Use some method for solving the Balck-Scholes equation to work incrementally down through t values, $t \to t - \Delta t \to t - 2\Delta t \cdots$ until we get to the next dividend payment time $t = T_i$. The values of v are now the values for $v(s, T_i+)$
- Update the values across T_i using the jump condition above: $v(s, T_i-) = v((1-\delta)s, T_i+)$.
- Repeat the second step to get to the next highest payment time, $t = T_{i-1}$.
- Update the values across T_{i-1} using the jump condition.
- ...(continue the above) ...
- Stop when reaching $t = 0$.

There are some some technical issues to consider in doing this.

- It would be most convenient if all the T_i are among our time-nodes t_m: $T_i = t_{m_i}$. If that is not practical, we could make some special small time steps $\Delta' t = t_m - T_i$ to hit the desired payment times, or use different Δt on the different $[T_{i-1}, T_i]$.
- It would be most convenient if for each price-node $s_- = s_m$ the corresponding $s_+ = (1-\delta)s_n$ is also a price-node. Unfortunately that will be impossible in financial variables; if $(1-\delta)s_n = (1-\delta)n\Delta s$ is also an integer multiple of Δs for every $n = 1, 2, \dots N$, then $1 - \delta$ would have to be an integer. But $0 < 1 - \delta < 1$. However this *is* possible in heat variables, because if $x = \log(\sqrt{2}s_-/\sigma)$ then $x = \log(\sqrt{2}(1-\delta)s_-/\sigma) = \log(1-d) + x$. So if we pick Δx so that $-\log(1-\delta)/\Delta x$ is an integer we will have a "node-to-node" update across T_i. If we want to stay in a financial variable formulation, then we should interpolate between the values of $v(s_n, T_i+)$ to obtain values for $v((1-\delta)s_n, T_i-)$. Tavella & Randal [57] make the point that it is important to use something better than mere linear interpolation, such as cubic splines (and that's easy to do in Matlab).

7.3 Problems

Problem 7.A
Use Itô's formula to verify that if $dS_t = (r-\delta)S_t\,dt + \sigma S_t\,dW_t$ and $v(s,t)$ satisfies (7.3), then $v(S_t,t)/B_t$ is a martingale.
.... `D-verify`

Problem 7.B
Find an explicit pricing formula for a digital option on a stock paying continuous dividends at rate $\delta > 0$ with threshold K. (I.e. solve (7.3) for terminal function $\phi(s) = 1_{[K,\infty)}(s)$.)

. . . . `digit-D`

Problem 7.C
We expressed our finite difference methods for the original Black-Scholes PDE in terms of formulas for coefficients a_n, b_n, c_n. Find modified versions of the formulas for a_n, b_n, c_n that should be used be to apply our methods to the equation (7.3) in the presence of continuous dividends.

. . . . `O`

Problem 7.D
As described in class, the equation describing the valuation function $v(s,t)$ for a derivative based on a stock which pays dividends continuously at the rate δ is (7.3). Suppose that $v(s,t)$ solves this equation.

1. Show that $\bar{v}(s,t) = e^{-\delta t}v(s,t)$ will solve the original Black-Scholes PDE (1.5) if the interest rate is replaced by $\bar{r} = r - \delta$. (This provides a second way to obtain explicit solutions of (7.3) using the explicit solutions of Section 2.6.)
2. Find an explicit formula for the solution of (7.3) if $v(s,T) = (s-K)^+$, a European call for a stock paying dividends. Show (pick some parameter values and produce a graph) that for $\delta > 0$ it *can* happen that $v(s,t) < (s-K)^+$. In other words early exercise *can* be advantageous for calls on dividend paying stocks!
3. Find the explicit formula for the solution of (7.3) if $v(s,T) = c_1 s + c_2$, where c_1, c_2 are any constants. (This will be the formula we should use to predict approximate boundary values for $v(s,t)$ as $s \to \infty$ (or $s \to 0$) when we know $v(s,T) \approx c_1 s + c_2$ as $s \to \infty$ (or $s \to 0$).)

. . . . `M`

Problem 7.E
Consider a stock which makes a single dividend payment of amount δS_{T_1-} at time $t = T_1 > 0$, and a (European) call on this stock whose value at expiry $t = T > T_1$ is $(S_T - K)^+$. At a time $t < T_1$ what explicit formula in terms of S_t and the various parameters gives the market value of the call option? (Think carefully about how you would compute the value in terms of the price of the dividend-reinvested portfolio value $\tilde{S}_t$. In particular what is the value at expiry in terms of $\tilde{S}_T$ versus S_T?) We said above that the option price is continuous across the dividend payment date:

$$v(S_{T_1-}, T_1-) = v(S_{T_1+}, T_1+).$$

Show however that the value of $\partial_s v$ is *not* continuous across the payment date:

$$v_s(S_{T_1-}, T_1-) \neq v_s(S_{T_1+}, T_1+).$$

(Use the formula for v that you derived. Recall the s-derivative of the Black-Scholes formula:

$$\partial_s[s\mathcal{N}(d^{(1)}) - Ke^{-r(T-t)}\mathcal{N}(d^{(2)})] = \mathcal{N}(d^{(1)}).$$

This was non-trivial because $d^{(i)} = d^{(i)}(s/K, t)$ depend on s. You should use this fact (without re-deriving it) if it is helpful.)

. . . . `N`

Chapter 8

American Options

We now turn our attention to American options. The holder of an American option can choose to exercise it at any time up to and including T. They don't need to wait until the time of expiry, as they would for a European option. This has a *big* influence on how we characterize and calculate the value of the option.

8.1 The American Put

Let's start with an American put, whose pricing function will be denoted $v^{\text{A-Put}}(s,t)$. Exercise can occur on or before the expiry date T. If we exercise the option at time t its value (at t) will be $(S_t - K)^-$. In problem 3.B of Chapter 3 we observed that the price of the European put can satisfy $v^{\text{E-Put}}(s,t) < (s-K)^-$ for small s. This means when S_t is small it would be better to exercise the option now, rather than waiting until T. If I own an American option, I can (though it would be foolish) choose to treat it like a Euorpean option and simply wait until T to exercise it. So the inequality

$$v^{\text{A-Put}}(s,t) \geq v^{\text{E-Put}}(s,t)$$

must hold. Since I can choose to exercise it immediately, we also have the inequality

$$v^{\text{A-Put}}(s,t) \geq (s-K)^-.$$

Our goal is to find ways to compute or approximate the values of $v^{\text{A-Put}}$.

The situation is different for American calls on non-dividend-paying stocks. It follows from $(s-K)^+ \geq (s-K)$ that a European call is more valuable than a forward:

$$v^{\text{E-Call}}(s,t) \geq v^{\text{Forward}}(s,t) = s - e^{-r(T-t)}K > s - K.$$

Since $v^{\text{E-Call}}(s,t) \geq 0$ as well, we see that

$$v^{\text{E-Call}}(s,t) \geq \max(s-K, 0) = (s-K)^+.$$

In other words there is never an advantage to exercising a call early; European and American calls have the same value. This is why we focus on puts to study American options. (In Problem 7.D of Chapter 7 we saw that for dividend paying stocks early exercise *is* advantageous.)

So how do we determine the value of an American put? Any strategy that we might devise for deciding when to exercise it corresponds to the choice of an exercise time $\mathcal{T} \leq T$. For example, we might decide to exercise it as soon a the stock price falls below some level θ. Then $\mathcal{T}$ would be the first moment that $S_t \leq \theta$. In general $\mathcal{T}$ is a time-valued random variable; its value depends on how the stock price evolves. But whether $\mathcal{T} \leq t$ or not can only depend on the stock price history up to time t. In other words $\mathcal{T}$ must be a stopping time. If we acquire the option at time t, then our strategy must be $t \leq \mathcal{T} \leq T$, and its market value must be at least as much as our exercise strategy will produce from it:

$$v^{\text{A-Put}}(S_t,t) \geq E[e^{-r(\mathcal{T}-t)}(S_{\mathcal{T}} - K)^- \,|\, \mathcal{F}_t].$$

The market price ought to be the highest possible value obtainable from the option, considered over all such exercise strategies, otherwise there would be an arbitrage opportunity. This leads us to the *American risk-neutral formula*:

$$v^{\text{A-Put}}(S_t,t) = \sup_{\mathcal{T}} E[e^{-r(\mathcal{T}-t)}(S_{\mathcal{T}}-K)^- \mid \mathcal{F}_t], \tag{8.1}$$

the supremum being over all stopping times $t \le \mathcal{T} \le T$. This is called an *optimal stopping problem*; the problem is to find the stopping time $\mathcal{T}^*$ corresponding to the optimal exercise strategy. This point of view can be exploited in a Markov chain approach; see Section 8.2.5 below. But it is not very helpful for the PDE approach. For that we will need a different point of view.

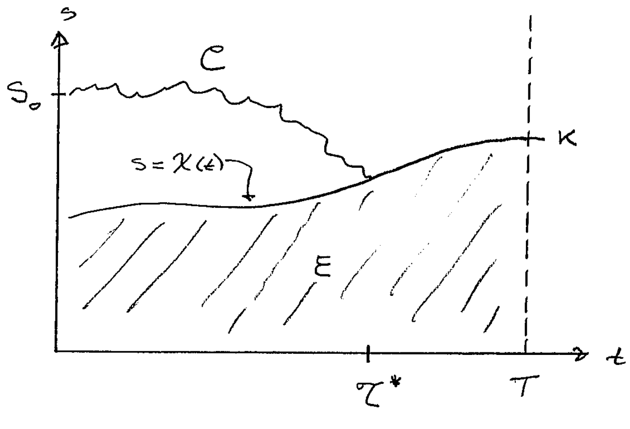
We anticipate that the set of possible (s,t) values can be classified into one of two types: those for which is would be best to exercise the option immediately, and those for which it would be best to wait at least a little longer. We will let the *exercise region* (or *stopping region*) $\mathcal{E}$ consist of those (s,t) for which

$$v^{\text{A-Put}}(s,t) = (s-K)^-.$$

These are the locations at which immediate exercise is optimal. The *continuation region* $\mathcal{C}$ consists of those (s,t) for which

$$v^{\text{A-Put}}(s,t) > (s-K)^-,$$

These are the locations at which immediate exercise is *not* optimal.



Since $v^{\text{A-Put}} \ge v^{\text{E-Put}} > 0$ but $(K-s)^+ = 0$ when $s \ge K$, we see right away that $\mathcal{C}$ includes all (s,t) with $s \ge K$. Moreover, for $t < T$ we have $v^{\text{A-Put}}(K,t) > 0$ while $(K-K)^- = 0$ so that $(K,t) \in \mathcal{C}$. So $\mathcal{E}$ is some subset of $(0,K) \times [0,T]$. We expect these two regions to be separated by some curve $s = \chi(t)$, called the *exercise boundary*. The best strategy for the owner of the option is to exercise at the first time $\mathcal{T}^*$ that $(S_t,t) \in \mathcal{E}$. (We will explain below why it would be a mistake to wait beyond that time.)

The exercise boundary is the new feature in the pricing of American options. If we knew the location of the exercise boundary $s = \chi(t)$ in advance, the pricing problem would consist of solving the Black-Scholes equation in the region $\chi(t) < s$ with the boundary condition

$$v^{\text{A-Put}}(s,t) = (s-K)^- \text{ for } s = \chi(t),$$

and the usual terminal condition: $v^{\text{A-Put}}(s,T) = (s-K)^-$. But we *don't* know $\chi(t)$; it has to be determined as part of solving the problem. From this perspective we have a *free boundary problem*; the boundary is "free" in the sense that it is not specified at the outset, but must be determined so as to satisfy all the requirements of the problem (which we have yet to formulate).

We need to find how to characterize $v^{\text{A-Put}}$ in each of the two regions, using the Black-Scholes equation, and whatever special properties the excercise boundary $s = \chi(t)$ must have. We can put this into our general risk-neutral context by reasoning as follows. Start at $t = 0$ with $S_0 = s$. Let $\mathcal{T}^*$ be the first time (S_t,t)

touches $\mathcal{E}$. At that moment we exercise the option, which gives us $(K - S_{\mathcal{T}^*})^+$ and invest this in bonds until we reach the expiration date T. At time T we will have

$$e^{r(T-\mathcal{T}^*)}(S_{\mathcal{T}^*} - K)^-.$$

If we reach $t = T$ without contacting $\mathcal{E}$, then we just cash in the option for its terminal value $v^{\text{A-Put}}(s, T) = (K - S_T)^+$. In this way we convert the option into a European-style option with final value given by

$$X = \begin{cases} e^{r(T-\mathcal{T}^*)}(S_{\mathcal{T}^*} - K)^- & \text{if } \mathcal{T}^* \le T \\ (S_T - K)^- & \text{if } t < \mathcal{T}^* \end{cases}$$

Of course to do this requires that we now what $\mathcal{E}$ is. Also note that this X is path-dependent, not of the simple form $\phi(S_T)$. Even so, the value of the option should be given by the risk-neutral formula

$$V_t = e^{rt}E[e^{-rT}X \mid \mathcal{F}_t],$$

and $e^{-rt}V_t$ should be a martingale:

$$d(e^{-rt}V_t) = 0\,dt + (\cdots)\,dW_t.$$

Prior to $\mathcal{T}^*$, while (S_t, t) is in $\mathcal{C}$, we have $V_t = v^{\text{A-Put}}(S_t, t)$, so Itô's formula says

$$d(e^{-rt}v^{\text{A-Put}}) = e^{-rt}[\partial_t v^{\text{A-Put}} + \mathcal{B}v^{\text{A-Put}}]\,dt + e^{-rt}\sigma S_t v_s^{\text{A-Put}}\,dW_t.$$

So inside the continuation region $\mathcal{C}$ the pricing function $v^{\text{A-Put}}$ should satisfy our usual Black-Scholes equation:

$$\partial_t v^{\text{A-Put}} + \mathcal{B}v^{\text{A-Put}} = 0.$$

Inside $\mathcal{E}$ we know that $s < K$, so that $v^{\text{A-Put}} = K - s$. Observe that

$$\partial_t v^{\text{A-Put}} + \mathcal{B}v^{\text{A-Put}} = rs + r(s - K) = -rK < 0. \tag{8.2}$$

In general, if a function v satisfies $\partial_t v + \mathcal{B}v < 0$ everywhere, then Itô's formula would tell us that for $t_1 < t_2$,

$$\begin{aligned} e^{-rt_2}v(S_{t_2}, t_2) &= \int_{t_1}^{t_2} e^{-rt}(\partial_t v + \mathcal{B}v)\,dt + \int_{t_1}^{t_2} \cdots\,dW_t + e^{-rt_1}v(S_{t_1}, t_1) \\ E[e^{-rt_2}v(S_{t_2}, t_2) \,|\, \mathcal{F}_{t_1}] &< 0 + 0 + e^{-rt_1}v(S_{t_1}, t_1). \end{aligned}$$

In other words wherever $\partial_t v + \mathcal{B}v < 0$ we lose value by holding an asset with pricing function v; the asset should always be exercised or sold immediately if (S_t, t) is a point where $\partial_t v + \mathcal{B}v < 0$. This is the situation in the exercise region $\mathcal{E}$, where $v^{\text{A-Put}}(s, t) = K - s$; to fail to exercise immediately means a loss of value.

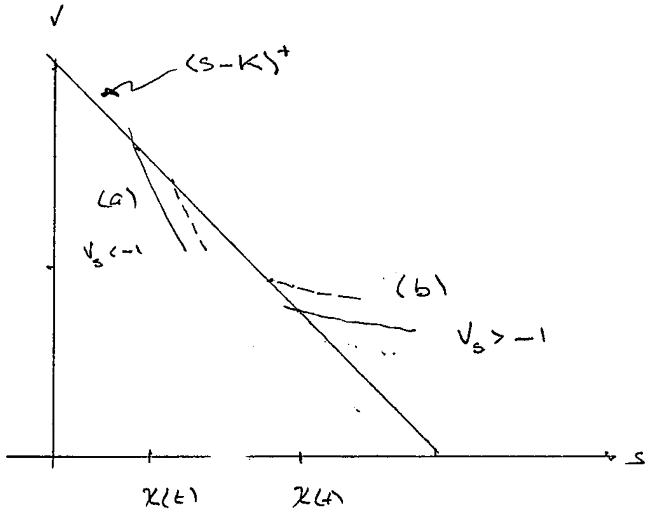
The above is not a completely rigorous argument. Itô's formula generally requires that the function v is twice continuously differentiable. That may be a reasonable assumption inside each of $\mathcal{C}$ and $\mathcal{E}$, but we should be suspicuous of such an assumption on the exercise boundary, where we know that characterization of $v^{\text{A-Put}}$ changes.

What can we say about $v^{\text{A-Put}}$ on the exercise boundary $s = \chi(t)$? Since we are going to exercise the option *on* the exercise boundary, certainly $v^{\text{A-Put}}(s, t) = (s - K)^-$ when $s = \chi(t)$ agreeing with its value inside $\mathcal{E}$. And we would naturally expect that for $s > \chi(t)$ (i.e. inside $\mathcal{C}$) but close to $\chi(t)$, we should have $v^{\text{A-Put}}(s, t) \approx (s - K)^-$. In other words we expect $v^{\text{A-Put}}$ to be continuous in the full region $0 < s$, $0 \le t \le T$. In particular it should be continuous *across* the exercise boundary.

But this is still not enough to determine where $s = \chi(t)$ actually is. We could choose virtually any reasonably smooth curve $s = G(t)$, solve the Black-Scholes equation for $G(t) < s$ with a boundary conditon $v(s, t) = (K - s)^+$ on $s = G(t)$, and then define $v(s, t) = (s - K)^-$ for $0 < s < G(t)$. This solution would satisfy every thing we have said so far about $v^{\text{A-Put}}$ with $\chi(t) = G(t)$ but would not have any connection to the true optimal exercise boundary. *What determines the true optimal exercise boundary?* The answer involves the values of $v_s^{\text{A-Put}}$ on the exercise boundary. Notice that below the boundary, in $\mathcal{E}$, $v^{\text{A-Put}} = K - s$ and so $v_s^{\text{A-Put}} = -1$. Consider a cross-section of the graph of $v^{\text{A-Put}}$ corresponding to a fixed value of $t < T$. Let's consider the possible values of $v_s^{\text{A-Put}}(\chi(t)+, t)$, i.e. the derivative evalued from the $\mathcal{C}$ side.

(a) If $v_s^{\text{A-Put}}(\chi(t)+,t) < -1$, then $v^{\text{A-Put}}$ would drop below $K-s$ for $s > \chi(t)$, which we know is not true, since $v^{\text{A-Put}} \geq K-s$.

(b) If $v_s^{\text{A-Put}}(\chi(t)+,t) > -1$, then it appears that lowering the value of $\chi(t)$ would have the effect of increasing the value of $v^{\text{A-Put}}(s,t)$ near the original $\chi(t)$. Since $v^{\text{A-Put}}(s,t)$ is the largest possible value for all choices of stopping rule, this should not be possible.



Putting these (purely heuristic) arguments together, it seems that we should have

$$v_s^{\text{A-Put}}(\chi(t)+,t) = -1.$$

In other words $v_s^{\text{A-Put}}$ should be continuous across the exercise boundary $s = \chi(t)$.

We know that $v^{\text{A-Put}}(s,t) = K-s$ for small s (below $\chi(t)$). Large values of s correspond to a large stock price S_t, in which case the stock price is nearly certain to remain large all the way until T, so that the option is nearly worthless. Thus we anticipate that $v^{\text{A-Put}}(s,t) \to 0$ as $s \to \infty$.

Collecting the above, here are the features of $v^{\text{A-Put}}$ that we anticipate.

1. $v^{\text{A-Put}}(s,T) = (s-K)^-$;
2. $v^{\text{A-Put}}(s,t)$ is continuous in $s > 0$, $0 \leq t \leq T$;
3. $\chi(t)$ is a continuous function of with $0 < \chi(t) < K$ for $t < T$;
4. for $s > \chi(t)$, $\partial_t v^{\text{A-Put}} + \mathcal{B} v^{\text{A-Put}} = 0$;
5. $v^{\text{A-Put}}(s,t) \to 0$ as $s \to +\infty$, for each $0 \leq t \leq T$;
6. for $s = \chi(t)$, $v_s^{\text{A-Put}}(\chi(t)+,t) = -1$;
7. for $s \leq \chi(t)$, $v^{\text{A-Put}} = K-s$.

We have only offered intuitive arguments for these features. Mathematical proofs are more than we can undertake here. See Jacka [29], as well as van Moerbeke [44] and McKean [40], for a careful treatment.

We observe that the exercise boundary $\chi(t)$ appears explicitly in several of the above properties. To base a numerical approximation on the above we would need to devise some method that tracks the location of the exercise boundary as part of the calculation. (Such a method is developed in Chapter 7 of [63].) But there is an alternate formulation in which $\chi(t)$ does not appear explicitly. The idea is to observe that in *both* regions $\mathcal{C}$ and $\mathcal{E}$ it is true that

- $\partial_t v^{\text{A-Put}} + \mathcal{B} v^{\text{A-Put}} \leq 0$ (in $\mathcal{E}$ this is (8.2));
- $(K-s)^+ - v^{\text{A-Put}} \leq 0$;
- in either region one of the above is an equality, so in general

$$\left[\partial_t v^{\text{A-Put}} + \mathcal{B} v^{\text{A-Put}}\right] \cdot \left[(s-K)^- - v^{\text{A-Put}}\right] = 0.$$

This formulation is called a *variational inequality.* Observe that there is no reference to the exercise boundary $s = \chi(t)$. In fact it makes no prseumption that $\mathcal{C}$ and $\mathcal{E}$ are separated by the graph of a function $s = \chi(t)$. For this reason, the variational inequality formulation applies with more general payoff functions than just $(s-K)^-$. We should be able to specify a rather arbitry payoff function $\phi(s,t)$ which gives the value of the option if exercised at time t with $S_t = s$. In general $\mathcal{C}$ and $\mathcal{E}$ could consist of several disconnected pieces; the variational inequality formulation above ought to be able to deal with that. The general form of the variational inequality is that for $t < T$ and $0 < s$,

- $\partial_t v + \mathcal{B} v \leq 0$,
- $\phi - v \leq 0$,
- $[\partial_t v + \mathcal{B} v] \cdot [\phi - v] = 0$;

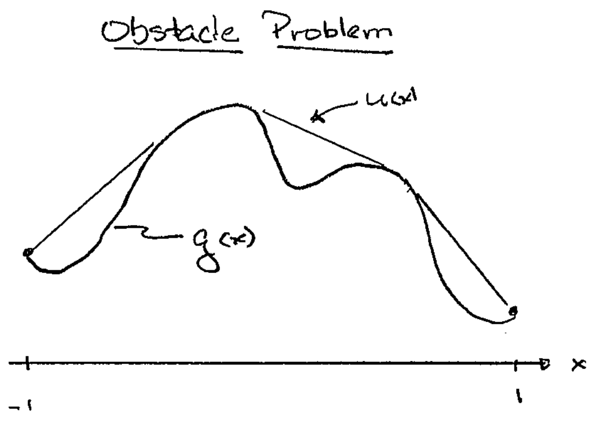
and $v(s,T) = \phi(s,T)$. If we can solve the variational inequality for $v^{\text{A-Put}}$, then we can determine the exercise region from

$$\mathcal{E} = \{(s,t) : \ v(s,t) = \phi(s,t)\}.$$

This is the approach we will follow.

We should point out that understanding what $\partial_t v + \mathcal{B} v$ actually means on the exercise boundary is a serious issue. We know from (8.2) that this expression is not continuous there. Although v_s is well-defined, v_{ss} is not continuous, so we must question if $v_{ss}(\chi(t), t)$ actually exists in the usual sense. Suffice it to say that there is a mathematically sound way to understand $\partial_t v + \mathcal{B} v$ in such situations, but it is far more than we can explain here. Details can be found in Menyi [41]. We will base our finite difference calculations on the variational inequality formulation, and once we have discretized everything no issue of differentiability will remain. In fact, based on the Markov chain interpretation of our finite difference calculations, convergence of the results to $v^{\text{A-Put}}$ can be justified independently of the variational inequality. (This sort of thing is worked out in [38].)

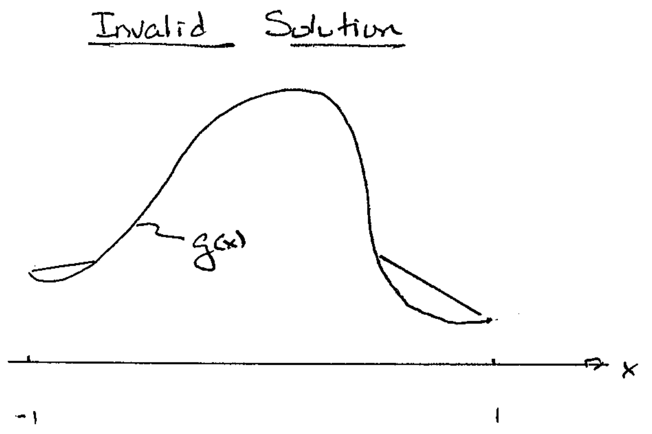
Variational inequalities are not foreign to our everyday experience. A simple application in which one arises is called the *obstacle problem.* Imagine a hill whose height is desrbied by a smooth (C^1) function $g(x)$ for $-1 \leq x \leq 1$. We are going to stretch a rope over the top of the hill and tie down the ends at $(-1, g(-1))$ and $(1, g(1))$. The rope will touch the hill at some points but be above the hill at others.



The rope will follow the graph of a curve $u(x)$, which we woud like to determine. The function $u(x)$ should be a continuously differentiable function on $[-1, 1]$ with $u(-1) = g(-1)$ and $u(1) = g(1)$, and for $-1 < x < 1$ satisfies

$$u''(x) \leq 0, \; u(x) \geq g(x), \text{ and } u''(x) \cdot (g(x) - u(x)) = 0.$$

Where the rope is in the air above the hill it shoud be perfectly straight, so $u''(x) = 0$. This can only fail where it contacts the hill, and there the hill shoud be concave, so at the contact points $u''(x) = g''(x) \leq 0$. Since it is at least as high as the hill, $u(x) \geq g(x)$. We see that this has the same form as our variational inequality above: two differential inequalities, one or the other of which has to be an equality at each x. The points where the rope leaves the ground are the "free boundary" points. Notice that it is important to insist that u' be continuous to rule out things like



See pages 108, 117 in [62].

8.2 Numerical Methods

We want to develop a finite difference methodology to approximate $v^{\text{A-Put}}$ using the variational inequality formulation above. As always, we could convert everything to heat variables $u(x, \tau)$ and $\psi(x, \tau)$ and do our approximations in that setting. We choose to stay in financial variables, but will consider the case of a general payoff function $\phi(s, t)$ in place of the specific $\phi(s, t) = (s - K)^-$ for the put.

8.2.1 Explicit Methods

First consider explicit methods. The situation is quite simple here. (The price we pay is the grid size constraint that is needed for stability.) With our usual notation $\mathbf{v}^m = [v_\cdot^m]$ and $\phi^m = [\phi_\cdot^m]$, suppose we have calculated $\mathbf{v}^m$ and are ready to solve for $\mathbf{v}^{m-1}$. The condition $\phi - v \leq 0$ discretizes as

$$\mathbf{v}^{m-1} \geq \phi^{m-1}.$$

(Again, such an inequality of two vectors means that the inequality holds for every coordinate.) Using the explicit method approximation (5.2) and our usual

$$\partial_t v(s_n, t_m) \approx \frac{v(s_n, t_m) - v(s_n, t_{m-1})}{\Delta t}$$

we see that $[\partial_t + \mathcal{B}]v \leq 0$ discretizes as

$$\mathbf{v}^{m-1} \geq (\mathbf{I} + \mathbf{B})\mathbf{v}^m + v^{\mathrm{L}}(t_m)\mathbf{b}^{\mathrm{L}} + v^{\mathrm{H}}(t_m)\mathbf{b}^{\mathrm{H}}.$$

(We will give expressions for v^{L} and v^{R} shortly.) The complementarity condition says that for each coordinate n one of the two inequalities above must be an equality. This is very easy to solve. Just calculate the right side

$$\mathbf{q} = (\mathbf{I} + \mathbf{B})\mathbf{v}^m + v^{\mathrm{L}}(t_m)\mathbf{b}^{\mathrm{L}} + v^{\mathrm{H}}(t_m)\mathbf{b}^{\mathrm{H}},$$

and then take the coordinate-wise maximum:

$$\mathbf{v}^{m-1} = \max(\phi^{m-1}, \mathbf{q}).$$

The same simple implementation will be possible for any explicit method, including space-only Markov chains.

8.2.2 Implicit Methods

We now consider implicit methods by looking specifically at the Crank-Nicholson method in financial variables. Using our approximation (5.7) and the central difference approximation for $\partial_t v$ we see that $[\partial_t + \mathcal{B}]v \le 0$ discretizes as

$$(\mathbf{I} - \frac{1}{2}\mathbf{B})\mathbf{v}^{m-1} \ge (\mathbf{I} + \frac{1}{2}\mathbf{B})\mathbf{v}^m + \frac{v^{\mathrm{L}}(t_{m-1}) + v^{\mathrm{L}}(t_m)}{2}\mathbf{b}^{\mathrm{L}} + \frac{v^{\mathrm{H}}(t_{m-1}) + v^{\mathrm{H}}(t_m)}{2}\mathbf{b}^{\mathrm{H}}.$$

We also require

$$\mathbf{v}^{m-1} \ge \phi^{m-1}$$

and the complementarity condition. These three conditions comprise what is called a *linear complementarity problem.* We need to solve this problem to determine $\mathbf{v}^{m-1}$. We will use the following following definitions to express the problem in a standard form. Let

$$\begin{aligned}
\mathbf{M} &= \mathbf{I} - \frac{1}{2}\mathbf{B},\\
\mathbf{q} &= \mathbf{M}\phi^{m-1} - \left\{(\mathbf{I} + \frac{1}{2}\mathbf{B})\mathbf{v}^m + \frac{1}{2}[v^{\mathrm{L}}(t_m) + v^{\mathrm{L}}(t_{m-1})]\mathbf{b}^{\mathrm{L}} + \frac{1}{2}[v^{\mathrm{H}}(t_m) + v^{\mathrm{H}}(t_{m-1})]\mathbf{b}^{\mathrm{H}}\right\},\\
\mathbf{z} &= \mathbf{v}^{m-1} - \phi^{m-1}.
\end{aligned}$$

With $\mathbf{M}$ and $\mathbf{q}$ given the problem is to find $\mathbf{z}$ which satisfies

$$\boxed{\begin{aligned} \mathbf{Mz} + \mathbf{q} &\ge \mathbf{0}\\ \mathbf{z} &\ge \mathbf{0}\\ (\mathbf{Mz} + \mathbf{q}) \cdot \mathbf{z} &= 0. \end{aligned}} \tag{8.3}$$

(The **0** on the right in the first two lines is the vector of all 0s, and again we mean the inequalities to be true for each component.) This is the standard form in which linear complementarity problems are usually discussed in the literature. In implicit methods for American options the linear complementarity problem replaces the problem of solving $\mathbf{Mv} = \mathbf{b}$ that we used LU factorization for in the case of European options.

8.2.3 Boundary Conditions for the American Put

To implement the above methods we will need approximations

$$v^{\mathrm{L}}(s^{\mathrm{L}}, t) \approx v^{\text{A-Put}}(s^{\mathrm{L}}, t), \quad v^{\mathrm{H}}(s^{\mathrm{H}}, t) \approx v^{\text{A-Put}}(s^{\mathrm{H}}, t).$$

for s^{L} small (close to 0) and s^{H} large. This is easy for the American put (but not necessarily so easy for a general $\phi(s,t)$). In fact for $s < \chi(t)$ we know $v^{\text{A-Put}}(s,t) = K - s$ is exact, so we should use

$$v^{\mathrm{L}}(s,t) = K - s.$$

For $S_t = s^{\mathrm{H}}$ large the probability that S_u will ever reach a value less than K is almost 0, so there is very little chance of being able to exercise the option for a positive price. Thus we should use

$$v^{\mathrm{H}}(s,t) = 0.$$

8.2.4 About Linear Complementarity Problems

The linear complementarity problem (8.3) is not as familiar as the basic linear equation, so in this section we provide a little background on it. We know the linear system $\mathbf{Mv} = \mathbf{b}$ has a unique solution for all $\mathbf{b}$ precisely when $\mathbf{M}$ is an invertible matrix. For (8.3) the analogous result requires the following definition.

Definition. *An $N \times N$ matrix $\mathbf{M}$ is a* P-matrix *if all the real eigenvalues of $\mathbf{M}$ and its principal submatricies are positive.*

(A *principal submatrix* of $\mathbf{M}$ is any matrix obtained by deleting a collection of rows and the same collection of columns.) Here is the basic existence and uniqueness theorem for linear complementarity problems.

Theorem. *The $N \times N$ (real) matrix $\mathbf{M}$ is a P-matrix if and only if for every vector $\mathbf{q}$ the linear complementarity problem* (8.3) *has a unique solution.*

A standard reference on linear complementarity problems is the book by Cottle, Pang, and Stone [14]. (Also see Huang and Pang [26] and Kinderlehrer and Stampacchia [37].) The proof of this theorem can be found there. There are several equivalent characterizations of a P-matrix which can also be found in [14]. A simple sufficient condtion for $\mathbf{M}$ to be a P-matrix is that $\mathbf{M}$ be *positive definite*. This means that

$$\mathbf{x} \cdot \mathbf{Mx} > 0 \text{ whenever } \mathbf{x} \neq \mathbf{0}. \tag{8.4}$$

So any positive definite matrix is a P-matrix. In fact (8.4) is equivalent to saying that the symmetric matrix $\mathbf{M} + \mathbf{M}^T$ is positive definite. It is easy to test whether a symmetric matrix is positive definite in MATLAB[1]. If we test the matricies $\mathbf{M} = \mathbf{I} - \frac{1}{2}\mathbf{B}$ that arise in our American option calculations, we find that virtually all of them do satisfy (8.4), although in financial variables they are *not* symmetric.

Although the proof of the above theorem would be too much of a diversion to present here, the proof of uniqueness under the stronger assumption (8.4) is very short, so we include it. Suppose $\mathbf{z}$ and $\bar{\mathbf{z}}$ are both solutions. Since $\mathbf{Mz} + \mathbf{q} \geq \mathbf{0}$ and $\bar{\mathbf{z}} \geq \mathbf{0}$, it follows that

$$(\mathbf{z} - \bar{\mathbf{z}}) \cdot (\mathbf{Mz} + \mathbf{q}) = 0 - \bar{\mathbf{z}} \cdot (\mathbf{Mz} + \mathbf{q}) \leq 0.$$

The same inequality must be true with $\mathbf{z}$ and $\bar{\mathbf{z}}$ reversed:

$$(\bar{\mathbf{z}} - \mathbf{z}) \cdot (\mathbf{M}\bar{\mathbf{z}} + \mathbf{q}) \leq 0.$$

But this is the same as

$$(\mathbf{z} - \bar{\mathbf{z}}) \cdot (\mathbf{q} + \mathbf{M}\bar{\mathbf{z}}) \leq 0.$$

Adding, we obtain $(\mathbf{z} - \bar{\mathbf{z}}) \cdot \mathbf{M}(\mathbf{z} - \bar{\mathbf{z}}) \leq 0$. But from this (8.4) implies $\mathbf{z} = \bar{\mathbf{z}}$.

The PSOR Method

There are both direct methods and iterative methods for solving linear complementarity problems. Most direct methods are "pivoting" methods. The best known of these is called Lemke's method. You can find MATLAB implementations of it on the web. The method we will use is an iterative method.

The basic idea of an iterative method is to make a rough initial guess $\mathbf{z}^{(0)} \approx \mathbf{z}$ for the solution to (8.3), and then use some kind of scheme for improving the accuracy of our guess to get a new improved guess $\mathbf{z}^{(1)}$. Then use the improvement scheme on $\mathbf{z}^{(1)}$ to get an even better approximation $\mathbf{z}^{(2)}$. This will (hopefully) produce a sequence of increasingly accurate approximations which converge to the true solution:

$$\mathbf{z}^{(0)} \to \mathbf{z}^{(1)} \to \mathbf{z}^{(2)} \to \cdots \mathbf{z}^{(k)} \cdots \to \mathbf{z} \text{ as } k \to \infty.$$

[1]As suggested in [24], if the command `[R,p]=chol(M+M')` retruns `p`$= 0$ then $\mathbf{M} + \mathbf{M}^T$ is positive definite. If it's not, `p` will be a positive integer.

To give you the idea, consider this approach applied to a simpler problem: find the solution x of $x^2 = a$. Of course $x = \sqrt{a}$, but how could we approximate it numerically? Start by rearranging the equation.

$$\begin{aligned} x^2 &= a \\ x &= \frac{a}{x} \\ 2x &= x + \frac{a}{x} \\ x &= \frac{1}{2}(x + \frac{a}{x}) \doteq h(x). \end{aligned}$$

We want the (positive) *fixed point* of h, i.e. the $x > 0$ for which $x = h(x)$. Depending on the properties of h you can often find the fixed point by iteration:

$$\begin{aligned} x_0 &= \text{ guess} \\ x_1 &= h(x_0) \\ x_2 &= h(x_1) \\ &\vdots \\ x_{k+1} &= h(x_k) \end{aligned}$$

If $\lim x_k = x$ exists, then $x = \lim x_{k+1} = \lim h(x_k) = h(\lim x_k) = h(x)$, so that x is the solution. With a decent approximation for $x_0 \approx \sqrt{a}$ this converges very quickly and is a very effective way to calculate $\sqrt{a}$. In practice we can't continue calulating x_n forever. Instead we monitor the size of the improvement $x_k - x_{k-1}$ and stop the calculation when it is acceptably small.

We want a similar approach for (8.3), namely a rearrangement of the equation so that the desired solution $\mathbf{z}$ is a fixed point $h(\mathbf{z}) = \mathbf{z}$. If our time step size Δt is small, then when we are solving (8.3) whose solution is $\mathbf{z} = \mathbf{v}^{m-1}$, the solution from the preceeding time $\mathbf{v}^m$ is probably very close and so should be a good starting value $\mathbf{z}^{(0)}$ for the fixed point iteration. So we take $\mathbf{z}^{(0)} = \mathbf{v}^m$ and then iterate $\mathbf{z}^{(k+1)} = h(\mathbf{z}^{(k)})$ and stop when the improvements get small, and take the final $\mathbf{z}^{(k)}$ as a suitable approximation for $\mathbf{v}^{m-1}$.

To find a rearrangement of (8.3) that is suitable for iteration, Start by splitting $\mathbf{M}$ into its subdiagonal ($\mathbf{E}$), diagonal ($\mathbf{D}$) and superdiagonal ($\mathbf{F}$) parts.

$$\mathbf{M} = \mathbf{E} + \mathbf{D} + \mathbf{F}.$$

Observe that for us the diagonal entries of $\mathbf{D}$ are all positive. Rearrange the complementarity problem as

$$\begin{aligned} \mathbf{Dz} + \mathbf{p}(\mathbf{z}) &\geq \mathbf{0}, \text{ where } \mathbf{p}(\mathbf{z}) = \mathbf{q} + (\mathbf{E} + \mathbf{F})\mathbf{z}; \\ \mathbf{z} &\geq \mathbf{0}; \\ (\mathbf{Dz} + \mathbf{p}(\mathbf{z})) \cdot \mathbf{z} &= 0. \end{aligned}$$

If we actually knew what $\mathbf{p}(\mathbf{z})$ was, this would be a simple linear complementarity problem to solve, because the first line is equivalent to $\mathbf{z} \geq -\mathbf{D}^{-1}\mathbf{p}(\mathbf{z})$, and the solution is given simply by

$$\mathbf{z} = \max(\mathbf{0}, -\mathbf{D}^{-1}\mathbf{p}(\mathbf{z})),$$

where the maximum is computed coordinate-wise. Of course we don't really know $\mathbf{p}(\mathbf{z})$, so the above is really a fixed-point descripton of $\mathbf{z}$. The idea would be is to start with some approximate $\mathbf{z}^{(0)}$ and then complute the sequence of iterates

$$\mathbf{z}^{(k+1)} = \max(\mathbf{0}, -\mathbf{D}^{-1}\mathbf{p}(\mathbf{z}^{(\mathbf{k})})). \tag{8.5}$$

Provided the sequence of iterates converges, $\mathbf{z}^{(k)} \to \mathbf{z}$, the limit $\mathbf{z}$ will solve the complementarity problem. This is called the *projected Jacobi method*.

The projected Jacobi method is easily implemented in MATLAB , but is not used much because other methods converge faster. A more commonly used method for calculating American option prices is the *projected successive over-relaxation (PSOR) method.* This involes a *relaxation parameter* $\omega > 0$ which must be choosen carefully. We rearrange the complementarity problem as

$$\begin{aligned}(\mathbf{E}+\frac{1}{\omega}\mathbf{D})\mathbf{z}+\mathbf{p}(\mathbf{z}) \geq \mathbf{0},\ \text{where}\ \mathbf{p}(\mathbf{z}) = \mathbf{q}+((1-\frac{1}{\omega})\mathbf{D}+\mathbf{F})\mathbf{z}; \\ \mathbf{z} \geq \mathbf{0}; \\ ((\mathbf{E}+\frac{\mathbf{1}}{\omega}\mathbf{D})\mathbf{z}+\mathbf{p}(\mathbf{z}))\cdot\mathbf{z} = 0.\end{aligned} \tag{8.6}$$

If we view $\mathbf{p}(\mathbf{z}) = [p_n]$ as known, we can use the fact that $\mathbf{E}+\frac{1}{\omega}\mathbf{D}$ is lower triangular to solve the problem directly. We do this by working from z_1 successively through z_N. If e_n and d_n are the entries in the n^{th} rows of $\mathbf{E}$ and $\mathbf{D}$ respectively, the equation for z_1 is simply

$$\frac{d_1}{\omega}z_1+p_1 \geq 0,\ z_1 \geq 0,\ \text{and}\ (\frac{d_1}{\omega}z_1+p_1)z_1 = 0.$$

The solution is simply

$$z_1 = \max(0, -\omega p_1/d_1).$$

The equations for z_n are

$$\frac{d_n}{\omega}z_n+e_nz_{n-1}+p_n \geq 0,\ z_n \geq 0,\ \text{and}\ (\frac{d_n}{\omega}z_n+e_nz_{n-1}+p_n)z_n = 0.$$

So given z_{n-1} we have

$$z_n = \max(0, -\omega(e_nz_{n-1}+p_n)/d_n).$$

(This uses the fact that $d_i > 0$.) Starting with $\mathbf{z}^{(k)}$, using this to produce $\mathbf{p} = \mathbf{p}(\mathbf{z}^{(k)})$, we then calculate z_1, z_2, $\ldots z_N$ as above. These are the corrdinates of $\mathbf{z}^{(k+1)}$, the next in the sequence of iterates of for the PSOR method.

In order to apply this method we need to pick the overrelaxation parameter ω. Normally ω is selected in the range $0 < \omega < 2$. There are theorems which guarantee that the PSOR iteration *does* converge to the solution of the complementarity problem if ω satisfies some bounds (which depend of properties of $\mathbf{M}$). See for instance Murty [43]. Usually the convergence is faster for some ω values and slower for others. Wilmott [62] suggests a simple method for automatically adjusting ω as part of the calculation, trying to keep it near its optimal value automatically. The m-file below implements his approach.

PSOR.m

```
function [z,wlast,clast]=PSOR(q,z0)
% Projected over-relaxation solver for the LC problem Mz+q>=0, z>=0.
% To initialize use PSOR(M,tol) with no output variables.
% To solve with initial estimate z0 use z=PSOR(q,z0).
% [z,w,count]=PSOR(q,z0) provides the most recent relaxation parameter w
% and teration count.  The algorithm is based on Wilmott's version with
% marching relaxation parameter.
persistent N e d DF tol w dw count
%
if nargout==0
% Initialize; q=M and z0=tol here
    N=length(q);
    e=full([0;diag(q,-1)]);
    d=full(diag(q));
    DF=spdiags(zeros(N,1),[-1],q);
    tol=z0; w=1; dw=.05; count=1000;
%
else
```

```
% Solve
    z=z0;
    count_old=count;
    er=1; count=0;
    while er>tol                  % PSOR iteration
        p=q+DF*z-d.*z/w;
        z_old=z;
        z(1)=max(0,-w*p(1)/d(1));
        for i=2:N
            z(i)=max(0,-w*(p(i)+e(i)*z(i-1))/d(i));
        end
        er=norm(z-z_old);
        count=count+1;
    end
    if count>count_old % Adjust relaxation parameter
        dw=-dw;
    end
    wlast=w;clast=count;
    w=w+dw;
end
```

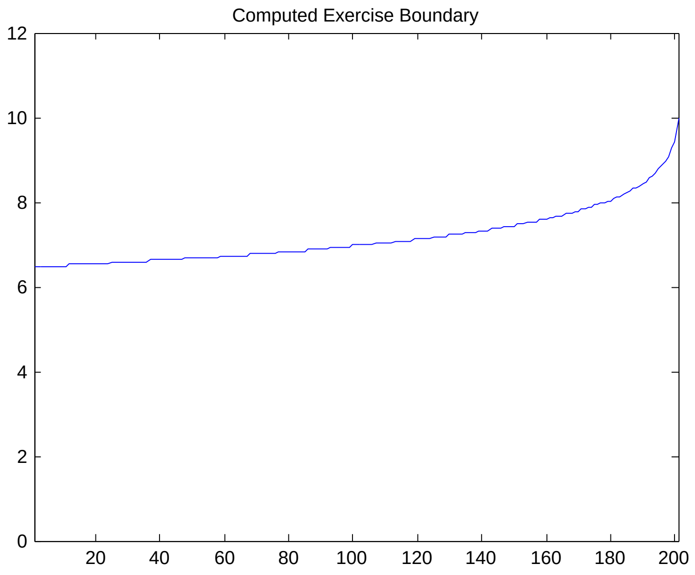
The reader interested in more discussion of the application of linear complementarity methods to American options should study [26], were different complementarity solvers are employed for different problem formulations.

As an example, we apply the above to the American put using $K = 10$, $[s^{\mathrm{L}}, s^{\mathrm{H}}] = [0, 20]$, $r = .05$, $\sigma = .3$, $T = 2$, using $N = 400$, $M = 200$, with `tol`$=10^{-4}$. The following script carries out the calculations (after all parameters are specified from the command line).

runa.m

```
% Script to solve the American BSPDE
dt=T/M;
genB                                %Generate matricies and coefficients
ML=speye(N)-B/2;
MR=speye(N)+B/2;
PSOR(ML,tol);    %Initialize
exbd=zeros(1,M);
t=T;
v=exercise(s,t)';
z=zeros(N,1);
for m=M-1:-1:0                      %Time-step loop
    q=-(MR*v+(vL(sL,t,T)*bL+vH(sH,t,T)*bH)/2);
    t=m*dt;
    phi=exercise(s,t)';
    q=q+ML*phi-(vL(sL,t,T)*bL+vH(sH,t,T)*bH)/2;
    z=PSOR(q,z);
    v=z+phi;
    exbd(m+1)=s(min(find(v>phi))); %locate exercise boundary (for put)
end
exbd=[exbd,K];
s=[sL,s,sH];                        %Include boundaries and
v=[vL(sL,t,T),v',vH(sH,t,T)];       %Convert results to rows
```

The variable `exbd(m)` records the computed approximations to the exercise boundary $\chi(t_m)$. The following plot displays the results.



8.2.5 Markov Chain Methods

Suppose we have a "space-only" Markov chain ξ_m which approximates S_t on a grid, as we discussed in Chapter 6. Viewing s_{ξ_m} as a discrete time, discrete space approximation to S_t (at $t = t_m$) it seems natural to consider the optimal stopping problem for the chain,

$$v_n^m = \sup_{0 \le \eta^* \le \eta} E[\rho^{\eta^*} \psi(s_{\xi_{\eta^*}}, t_{\eta^*}) \,|\, \xi_m = n].$$

At each gridpoint (s_n, t_m) we are faced with the alternative of stopping where we are with value $\phi(s_n, t_m)$ or continuing at least one more step, in which case the value now will be $E[\rho v_{\xi_{m+1}}^{m+1} \,|\, \xi_m = n]$. This is straight forward to calculate. We start with the values of $v_n^M = \psi(s_n, T)$ and then work backwards through time in our usual way. Given the values for $v_\cdot^{m+1}$ we calculate v_n^m as follows. First calculate the expected value

$$E[\rho v_{\xi_{m+1}}^{m+1} \,|\, \xi_m = n] = \rho \sum_{n'} p_{n,n'} v_{n'}^{m+1},$$

then take the larger of this with $\psi(s_n, t_m)$ to obtain v_n^m: For a space-only chain these are very easy to calculate:

$$v_n^m = \max(\psi(s_n, t_m), E_{(n,m)}[\rho v_{\tilde{n}}^{m+1}]).$$

When the Markov chain approximation comes from an explicit finite difference formulation, the above is the same as the calculation of Section 8.2.1. In particular, for grid sizes which allow a probabilistic interpretation of our explicit finite difference mehtod (or any other space-only approximating chain) no lineary complementarity problem is involved, making it very easy to implement.

Using a space-time chain, (ξ_k, ζ_k), the resulting equation is more complicated, because of the possibility that ξ_k can jump between the spacial nodes for the same t_m before making the jump to the next time node. Somehow we have to weigh the likelihood of reaching a node for the same t_m at which it would be optimal to stop against all the other possibilities. Again suppose we know all the optimal $v_\cdot^{m+1}$ values and we are located at $\xi_k = n$, $\zeta_k = m$. If it is not optimal to stop here, then the value at this node must be

$$\begin{aligned} v_n^m &= E[\rho^{\zeta_{k+1}-m} v_{\xi_{k+1}}^{\zeta_{k+1}} \,|\, \xi_k = n,\ \zeta_k = m] \\ &= \sum_{n'} q_{n,n'} \rho^0 v_{n'}^m + \sum_{n'} p_{n,n'} \rho^1 v_{n'}^{m+1}. \end{aligned}$$

If it is optimal to stop where we are then $v_n^m = \phi(s_n, t_m)$. In general v_n^m must be the larger of these two values. Let $\mathbf{g} = [g_n]$ where

$$-g_n = \rho \sum_{n'} p_{n,n'} v_{n'}^{m+1}.$$

Then we see that

$$v_n^m \le \sum_{n'} q_{n,n'} v_{n'}^m - g_n,$$
$$v_n^m \le \phi(s_n, t_m),$$

with one of these being an equality. Thus we again have a linear complementarity problem:

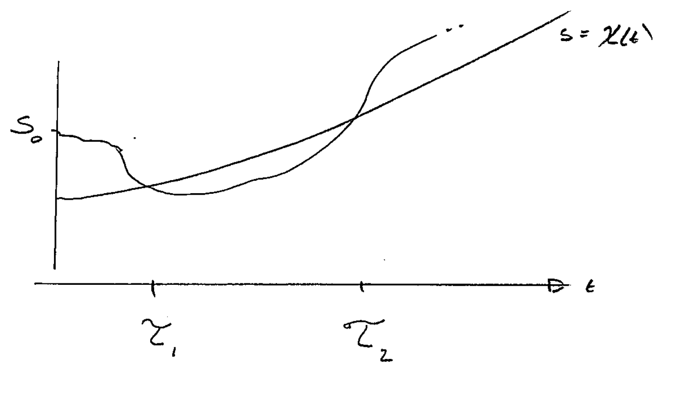
$$(\mathbf{I}-\mathbf{Q})\mathbf{v}^m + \mathbf{g} \le \mathbf{0},\ \mathbf{v}^m \le \psi^m,\ \text{with } ((\mathbf{I}-\mathbf{Q})\mathbf{v}^m + \mathbf{g}) \cdot (\mathbf{v}^m - \psi^m) = 0.$$

Linear complementarity problems arise naturally from optimal stopping problems for Markov chains.

8.3 Other Formulae for the American Put

No explicit expression is known for the pricing function $v^{\text{A-Put}}$ of the American put is known. Naturally many efforts have been made to approxximate it's values. Several are summarized in Salopeck [51]. To finish this chapter we want to describe an interesting formula connecting $v^{\text{A-Put}}$ with it's European counterpart, $v^{\text{E-Put}}$. We will give only a heuristic development, but the formula can be justified rigorously; see Carr, Jarrow and Myneni [12] for instance. Consider a $t_0 < T$ and stock price $S_{t_0} = s_0 > \chi(t_0)$ above the exercise boundary. We are going to describe a self-financing portfolio strategy which begins with value $v^{\text{A-Put}}(s_0, t_0)$ and at the final time will have value $(S_T - K)^- +$ something. Taking $e^{-r(T-t)}E[\,\cdot\,|\,\mathcal{F}_t]$ of this will give a formula relating $v^{\text{A-Put}}$ to $v^{\text{E-Put}}$.

Start by buying the American option for $v^{\text{A-Put}}(s_0, t_0)$. We will hold this untill S_t drops below the exercise boundary $\chi(t)$. At the moment when that happens ($\mathcal{T}_1$ in the figure) we exercise the option, which has value $K - S_{\mathcal{T}_1}$. We convert this to K in bonds and -1 share of stock. We hold these until the next time ($\mathcal{T}_2$ in the figure) that S_t recrosses the exercise boundary into the continuation region, at which time we buy back the option for $K - S_{\mathcal{T}_2}$.



Our bond holdings have grown in value to

$$Ke^{r(\mathcal{T}_2 - \mathcal{T}_1)}.$$

We only need to sell K in bonds for our transaction at time $\mathcal{T}_2$, leaving us with bonds of value

$$Ke^{r(\mathcal{T}_2 - \mathcal{T}_1)} - K = Ke^{r\mathcal{T}_2} \int_{\mathcal{T}_1}^{\mathcal{T}_2} re^{-ru}\, du.$$

By time T this will have grown in value to

$$Ke^{rT}\int_{\mathcal{T}_1}^{\mathcal{T}_2} re^{-ru}\,du.$$

We continue to do this, selling the option when S_t enters the exercise region, and buying it back when S_t enters to contiuation region. When we reach the final time we will have earned a total of

$$Ke^{rT}\int_{t_0}^{T} re^{-ru}\,1_{S_u<\chi(u)}\,du, \tag{8.7}$$

beyond the terminal value of $(K-S_T)^+$. Our policy (which is self-financing) yields a terminal value of

$$X = (S_T-K)^+ + Ke^{rT}\int_{t_0}^{T} re^{-ru}\,1_{S_u<\chi(u)}\,du.$$

Now we would be correct to question whether this portfolio managment strategy can really be carried out, because S_t will actually cross $\chi(t)$ infinitely many times, if at all. So our strategy would require infinitely many sell and buy transactions. That's why this is only a heuristic development. But the formula is correct even if our explanation is not supportable. The conclusion is that

$$\begin{aligned} v^{\text{A-Put}}(s_0,t_0) &= e^{rt_0}E_{s_0,t_0}[e^{-rT}(K-S_T)^+ + K\int_{t_0}^{T} re^{-ru}\,1_{S_u<\chi(u)}\,du] \\ &= v^{\text{E-Put}}(s_0,t_0) + e^{rt_0}E_{s_0,t_0}[K\int_{t_0}^{T} re^{-ru}\,1_{S_u<\chi(u)}\,du]. \end{aligned}$$

The last term is the *early exercise premium.* We can work out a more explicit expression for it.

$$e^{rt_0}E_{s_0,t_0}\left[K\int_{t_0}^{T} re^{-ru}\,1_{S_u<\chi(u)}\,du\right] = \int_{t_0}^{T} Kre^{-r(u-t_0)}P_{s_0,t_0}(S_u<\chi(u))\,du.$$

The P here is the risk-neutral probability, with respect to which we know

$$S_u = s_0e^{(r-\sigma^2/2)(u-t_0)+\sigma(W_u-W_{t_0})}.$$

Now $W_u-W_{t_0}$ is $\sqrt{u-t_0}\,Y$ for a standard normal random variable Y. So, $P_{s_0,t_0}(S_u<\chi(u))$ is the probability that

$$\begin{gathered} s_0e^{(r-\sigma^2/2)(u-t_0)+\sigma\sqrt{u-t_0}Y} < \chi(u) \\ (r-\sigma^2/2)(u-t_0)+\sigma\sqrt{u-t_0}Y < \log(\chi(u)/s_0), \end{gathered}$$

or

$$Y < \frac{\log(\chi(u)/s_0)-(r-\sigma^2/2)(u-t_0)}{\sigma\sqrt{u-t_0}}.$$

So we have

$$P_{s_0,t_0}(S_u<\chi(u)) = \mathcal{N}\left(\frac{\log(\chi(u)/s_0)-(r-\sigma^2/2)(u-t_0)}{\sigma\sqrt{u-t_0}}\right).$$

Putting the pieces together, here is the formula we have derived.

$$v^{\text{A-Put}}(s,t) = v^{\text{E-Put}}(s,t) + \int_{t}^{T} Kre^{-r(u-t)}\,\mathcal{N}\left(\frac{\log(\chi(u)/s)-(r-\sigma^2/2)(u-t)}{\sigma\sqrt{u-t}}\right)du. \tag{8.8}$$

If we knew the exercise boundary $\chi(t)$ then (8.8) would be an exact formula connecting the American and European pricing functions, and could be evaluated by numerical integration. If we have bounds on $\chi(t)$, like $b_\infty \le \chi(t) \le K$ (where b_∞ is the exercise boundary for the perpetual American put from the problems below), then using the bounds in place of $\chi(t)$ in (8.8) will give us bounds on $v^{\text{A-Put}}$.

Another way to exploit (8.8) is to consider it specifically for $s = \chi(t)$. Since $v^{\text{A-Put}}(\chi(t), t) = K - \chi(t)$ we have the equation

$$K - \chi(t) = v^{\text{E-Put}}(\chi(t), t) + \int_t^T Kre^{-r(u-t)} \mathcal{N}\left(\frac{\log(\chi(u)/\chi(t)) - (r - \sigma^2/2)(u-t)}{\sigma\sqrt{u-t}}\right) du.$$

This is an integral equation for $\chi(t)$ for $t \le T$, and could be used as the basis of an approach to numerically approximate $\chi(t)$. (A simpler integral equation for $\chi(t)$ results from using (8.8) together with $v_s^{\text{A-Put}}(\chi(t), t)) = -1$; see [12].)

The exercise boundary $\chi(t)$ is known to be continuous and increasing, with $\chi(T) = K$; see [29] for instance. It appears from our finite difference calculations above to have a vertical tangent at the point of contact. Several efforts have been made to find approximations to $\chi(t)$ for $t \approx T$. It turns out that a good approximation (for $t \approx T$) is

$$\chi(t) \approx K - K\sigma\sqrt{(T-t)\log(T-t)}.$$

A development of this can be found in Jiang [31] among other places. Also in Jiang [31] (see §3.6) is an unusual formula connecting the American put and call with each other, via a price inversion and interchange of interest and dividend rates.

8.4 Problems

Problem 8.A
Use an up-wind discretization of the Black-Scholes equation, with step sizes choosen to insure stability according to the probabilistic interpretation of Chapter 6. Implement the explicit method based on this approach to approximate the value of an American put.
.... `uwAM`

Problem 8.B
To understand our free boundary formulation of the value of American put options, it is helpful to consider the *perpetual* American put, which is an American put which never expires: $T = \infty$. Its market value should be given as a function of the stock price alone: $v(s) = v^{\text{PAP}}(s)$. The exercise boundary $s = \chi(t)$ of the usual American put (page 71) becomes just a constant χ^{P} in the perpetual version. The description of $v(s)$ becomes the following. $v(s)$ is a continuous function of $s > 0$ and $\chi^{\text{P}} = b$ is a constant such that the following hold.

1. $v(s)$ solves $\frac{1}{2}\sigma^2 s^2 v''(s) + rsv'(s) - rv(s) = 0$ for $s > b$;
2. $v(s) = (s - K)^-$ for $0 < s \le b$;
3. $\lim_{s \to b^+} v'(s) = -1$;
4. $\lim_{s \to \infty} v(s) = 0$.

Determine the value of χ^{P} by directly solving this problem. Specifically, carry out the following:

- First, check that there are two values of γ for which $v(s) = s^\gamma$ solves $\frac{1}{2}\sigma^2 s^2 v''(s) + rsv'(s) - rv(s) = 0$. One of them is $\gamma = 1$, the other is $\gamma = M$ for a certain constant M. Find M. According to the general theory of linear ordinary differential equations, this means that the solutions of the differential equation for $v(s)$ in item 1 above are
$$v(s) = c_1 s + c_2 s^M$$
for some choice of constants c_1, c_2.
- Assuming that $0 \le b \le K$, use items 2 and 4 above to determine values of c_1 and c_2. Show that there is a unique value of b for which item 3 is satisfied. This value of b is χ^{P}.

Next we are going to see that item 3 characterizes the optimality of χ^{P}. Let $v(s)$ be as determined by items 1, 2, and 4 for an arbitrary $0 \leq b \leq K$. For any $s > K$ show that the value of $0 \leq b \leq K$ which maximizes $v(s)$ is the same $b = \chi^{\mathrm{P}}$ that you found in b). Take $\sigma = .3$, $r = .05$, $K = 10$. Calculate χ^{P}. Pick values $0 < b_1 < \chi^{\mathrm{P}} < b_2 < K$ and for each of $b = b_1$, $b = \chi^{\mathrm{P}}$ and $b = b_2$ plot $v(s)$ for $b_i \leq s \leq 15$, together with $(K - s)^+$. (You should get a picture like the one on page 73.)

It can be shown (though you are not being asked to) that the formula for $v^{\mathrm{PAP}}(s)$ derived above ($v(s)$ for $b = \chi^{\mathrm{P}}$) is the limit $\lim_{T\to\infty} v^{\text{A-Put}}(s, 0)$ of the usual American put's value as the time to go becomes infinite, and that $\chi^{\mathrm{P}} = \lim_{T\to\infty} \chi(0)$ is likewise the limiting value of its exercise boundary when we are far from expiry.

. . . . $\boxed{\texttt{K}}$

Problem 8.C
Revise `runa.m` so that it keeps a record (like `exbd` for the exercise boundary) of the value of the relaxation parameter `w` and interation count used in each call to `PSOR`. (Note that `PSOR.m` is written so that it will return these vaues if asked.) Run your script with the same parameters as in the notes. Produce graphs of the lists of `w` and `count` values. What are the average values of w and count for this calculation?

. . . . $\boxed{\texttt{runtest}}$

Problem 8.D
Consider an American option with exercise value $\phi(s,t) = s^\gamma$, where $\gamma > 0$ is a parameter. Using Problem 2.D you can write down an explicit formula for the price $v(s,t)$ of the European version of this option: $v(s,T) = s^\gamma$. Show that if $\gamma > 1$ you would never exercise the American option early. What if $\gamma < 1$?

. . . . $\boxed{\texttt{power}}$

Problem 8.E
Suppose that $\mathbf{M} = \mathbf{E} + \mathbf{D} + \mathbf{F}$ is a tridiagonal (real) matrix with subdiagonal entries e_i, diagonal entries d_i and superdiagonal entries f_i. Show that if $\mathbf{M}$ satisfies (8.4) then all $d_i > 0$. Suppose that $\mathbf{z}^{(0)}, \mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(k)}, \ldots$ is a sequence of vectors produced by the PSOR method applied to our standard linear complementarity problem (8.3). Suppose that the limit $\lim_{k\to\infty} \mathbf{z}^{(k)} = \mathbf{z}$ exists. Show that $\mathbf{z}$ *does* solve the complementarity problem. (Hint: If you let $k \to \infty$ in the description of the PSOR method you will get a collection of equations satisfied by the coordinates z_i and some auxiliary variables y_i.)

. . . . $\boxed{\texttt{L}}$

Problem 8.F
The linear complementarity problem that describes American options on stocks paying continuous dividends is

$$\begin{aligned} v_t + \mathcal{B}^D v &\leq 0 \\ \phi - v &\leq 0 \\ (v_t + \mathcal{B}^D v) \cdot (\phi - v) &= 0, \end{aligned}$$

where $\mathcal{B}^D$ is as in Chapter 7, and $\phi(s,t)$ is the formula for the value of the option when exercised. Develop Matlab code to apply the PSOR method in the Crank-Nicholson formulation to compute a solution to this for a call option. Modify `runa.m` as appropriate to plot the optimal exercise boundary $s = \chi(t)$. Try it for $r = .08$, $\delta = .05$, $\sigma = .3$, $K = 20$ over a period of $T = 1$. Plot both your rough approximation to the exercise boundary $\chi(t)$ with respect to t, and the resulting $v(s,0)$ with respect to the stock price s. Mark the approximate exercise boundary point on this latter graph.

. . . . $\boxed{\texttt{P}}$

Chapter 9

Asian Options

We now turn our attention to *path-dependent* options. By this we mean that V_T depends on the history of the stock price, $\{S_t : 0 \leq t \leq T\}$, not just S_T alone. These generally arise by taking the terminal value for our usual call or put, $(S_T - K)^{\pm}$ and replacing either K or S_T by some value computed from the history of S_t. In the next chapter we consider lookback options, which use a historical maximum (or minimum) stock price Z_T rather than an average. In this chapter we will discuss *Asian* options, in which we use some sort of average stock price, A_T. For instance, the *average price*[1] *call* has terminal value

$$V_T = (A_T - K)^+,$$

and the *average strike put* has

$$V_T = (S_T - A_T)^-.$$

There are several ways we might compute an average stock price.

- The continuous arithmetic average: $A_T = \frac{1}{T}\int_0^T S_t\,dt$.
- A discrete arithmetic average: $A_T = \frac{1}{R}\sum_1^R S_{T_i}$ $(0 \leq T_1 < \cdots T_R \leq T)$.
- A discrete geometric average: $A_T = \left(\prod_1^R S_{T_i}\right)^{\frac{1}{R}} = \exp(\frac{1}{R}\sum_1^R \log(S_{T_i}))$.
- The continuous geometric average: $A_T = \exp(\frac{1}{T}\int_0^T \log(S_t)\,dt)$.

Although geometric averages are not really used in financial markets, they have been considered in the literature because explicit price formulas can be developed for them, which can then be used as a "control variate" in a Monte Carlo approach. Observe that the geometric average is always bounded above by the corresponding arithmetic average. This is because $\exp(\cdot)$ is a convex function, and so

$$\exp(\frac{1}{N}\sum_1^N \log(S_{T_i})) \leq \frac{1}{N}\sum_1^N S_{T_i}.$$

The continuous version is a limit of this. An average price call would therefore have a higher value with an arithmetic average than it would for a geometric average. So explicit formulas in the case of geometric averages can be used as lower bounds for the arithmetic averaged versions. Having made these observations we will limit our discussion to arithmetic averages.

We first consider the PDE approach to pricing Asian options. Then we mention briefly two other treatments, and then consider how we might handle discrete arithmetic averages.

[1]Also called "fixed strike".

9.1 Continuous Arithmetic Averages

Our general risk-neutral valuation formula says that

$$V_t = E[e^{-r(T-t)}X \mid \mathcal{F}_t],$$

where X is the random variable giving the value at expiry. Previously we had $X = \phi(S_T)$, a function of the final stock price. This is no longer true for Asians. But if we define the "running integral"

$$Y_t = \int_0^t S_u \, du,$$

then the arithmetic average is $A_T = \frac{1}{T}Y_T$. So we can write the terminal value as a function of the pair S_T, Y_T:

$$\phi(S_T, Y_T).$$

For instance, the average strike put would use

$$\phi(s, y) = (s - y/T)^-.$$

We will argue that the option's market price must be a function $v(s, y, t)$ evalued at (S_t, Y_t, t),

$$V_t = v(S_t, Y_t, t),$$

and will derive a generalization of the Black-Scholes equation which describes the pricing function $v(s, y, t)$. Then we will consider how we might calculate the appropriate solutions.

We begin with $S_T = S_t \frac{S_T}{S_t}$. The key observation is that for $t < T$ we have that S_T/S_t is independent of $\mathcal{F}_t$, because

$$S_T/S_t = e^{(r-\sigma^2/2)(T-t)+\sigma(W_T-W_t)}.$$

A general fact about random variables is that when W and Z are independent, W being $\mathcal{G}$-measurable and Z being independent of $\mathcal{G}$, then

$$E[f(W, Z) \mid \mathcal{G}] = \int f(W, z) p_Z(z) \, dz, \tag{9.1}$$

where $p_Z(\cdot)$ is the density of Z. In other words the conditional expectation "integrates out" the independent random variable Z. (See the appendix.) This explains why, in the standard European case,

$$V_t = e^{-r(T-t)}E[\phi(S_t \cdot [S_T/S_t]) \mid \mathcal{F}_t]$$

is expressible as some function of S_t and t. In the case of Asian options we also have dependence on Y_T. But we can write

$$\begin{aligned} Y_T &= Y_t + \int_t^T S_u \, du \\ &= Y_t + S_t \left[\int_t^T S_u/S_t \, du \right]. \end{aligned}$$

In the second line the integral is also independent of $\mathcal{F}_t$. So in

$$V_t = e^{-r(T-t)}E[\phi(S_t \cdot [S_T/S_t], Y_t + S_t \left[\int_t^T S_u/S_t \, du \right]) \mid \mathcal{F}_t]$$

the S_T/S_t and $\int_t^T S_u/S_t \, du$ will integrate out, so this should be a function of S_t, Y_t and t:

$$V_t = v(S_t, Y_t, t).$$

Whatever Asian option we are interested in, we want to find the pricing function $v(s, y, t)$ associated with it. Observe that v is now a function of *three* variables. We need the appropriate generalization of the Black-Scholes equation. The essential property is that

$$e^{-rt}V_t = e^{-rt}v(S_t, Y_t, t)$$

should be a martingale. Itô's formula will lead us to what we want. Becuase

$$dS_t = rS_t\,dt + \sigma S_t\,dW_t \text{ and } dY_t = S_t\,dt$$

we have that

$$(dY_t)^2 = 0 = dY_t\,dS_t \text{ and } (dS_t)^2 = \sigma^2\,dt.$$

Using these we calculate that

$$\begin{aligned} d[e^{-rt}v(S_t, Y_t, t)] &= e^{-rt}v_s\,dS_t + e^{-rt}\frac{1}{2}v_{ss}\,(dS_t)^2 + e^{-rt}v_y\,dY_t + e^{-rt}(v_t - rv)\,dt \\ &= e^{-rt}\sigma v_s S_t\,dW_t + e^{-rt}\left[\frac{1}{2}\sigma^2 S_t^2 v_{ss} + rS_t v_s + S_t v_y + v_t - rv\right]\,dt. \end{aligned}$$

(All the v and its partial derivatives are evaluated at (S_t, Y_t, t) of course.) In order for this to be a martingale, the $[\cdots]\,dt$ term in the second line must be 0. Thus the *Asian Black-Scholes eqaution* which characterizes the pricing function v for an Asian option based on the continuous arithmetic average must be

$$\boxed{\frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + sv_y - rv + v_t = 0.} \tag{9.2}$$

Note the new term sv_y compared to our former (1.5). We will abbreviate this as

$$\mathcal{B}^{\mathrm{A}}v + \partial_t v = 0,$$

where

$$\mathcal{B}^{\mathrm{A}}v = \frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + sv_y - rv.$$

The ranges of the variables are $0 < s$ and $0 \le t \le T$ as before, and $0 < y$ (because $Y_t = \int_0^t S_u\,du > 0$).

The v we seek is the solution of (9.2) with the terminal value

$$v(s, y, T) = \phi(s, y),$$

for the specific option we are studying, *and* some sort of growth condition as $s \to 0, \infty$ and $y \to 0, \infty$. In particular, it turns out that if $|\phi(s, y, t)| \le c_1 s + c_2 y + c_3$ for some constants c_i, then the risk neutral valuation formula implies that for some contants c_i',

$$|v(s, y, t)| \le c_1' s + c_2' y + c_3'.$$

(See Problem 9.B.) Such a growth condition identifies the unique solution of (9.2) and the terminal conditions.

We can produce a few explicit solutions of (9.2). (See the Problem 9.C.) These allow us to write down put-call parity relations. For the *average strike* case,

$$v^{\mathrm{Call}}(s, y, T) = (s - y/T)^+, \quad v^{\mathrm{Put}}(s, y, T) = (s - y/T)^-.$$

So $v^{\mathrm{Call}} - v^{\mathrm{Put}}$ is the solution of (9.2) with terminal value $\phi = s - y/T$. This is easily checked to be

$$v^{\mathrm{Call}} - v^{\mathrm{Put}} = s - \frac{s}{rT}(1 - e^{-r(T-t)}) - \frac{1}{T}e^{-r(T-t)}y.$$

For the *average price* case,

$$v^{\mathrm{Call}}(s, y, T) = (y/T - K)^+, \quad v^{\mathrm{Put}}(s, y, T) = (y/T - K)^-.$$

So $v^{\text{Call}} - v^{\text{Put}}$ is the solution of (9.2) with terminal value $\phi = y/T - K$. This is checked to be

$$v^{\text{Call}} - v^{\text{Put}} = \frac{s}{rT}(1 - e^{-r(T-t)}) - e^{-r(T-t)}(K - \frac{y}{T}).$$

In light of these we concentrate on calls in what follows.

A finite difference approach to solving (9.2) would invole a 3-dimensional grid $(s_n, y_k, t_m) = (n\Delta s, k\Delta y, m\Delta t)$ and approximate values

$$v^m_{n,k} \approx v(s_n, y_k, t_m).$$

We would then form a discretized approximation to the equation (9.2), along the lines of what we did previously. Observe that there are two space dimensions, s and y. For each time t_m, instead of a vector $[v^m_{\cdot}]$ as before, we now have a $N \times K$ matrix $[v^m_{\cdot,\cdot}]$ of values. The finite difference equation would give us a way of calculating $\mathbf{V}^{m-1} = [v^{m-1}_{\cdot,\cdot}]$ from $\mathbf{V}^m = [[v^m_{\cdot,\cdot}]$. The finite difference approximations to (9.2) can be arranged as a matrix equation in the form

$$\mathbf{M}\mathbf{V}^{m-1}\mathbf{G} = \mathbf{N}\mathbf{V}^m\mathbf{H} + \mathbf{B}$$

for appropriate matricies $\mathbf{G}$, $\mathbf{H}$, and $\mathbf{B}$. Work out the details ... Since there is no v_{yy} term in the equation, it is particularly important that v_y is discretized appropriately. Since $y > 0$ we should use an up-wind approximation such as

$$v_y(s_n, y_k, t_m) \approx \frac{v^m_{n,k+1} - v^m_{n,k}}{\Delta y}.$$

In addition, we would either need to find approximate formulas to use for v around the four sides (boundary) of our grid rectangle $s_0 \leq s \leq s_{N+1}$, $y_0 \leq y \leq y_{K+1}$, or else decide on alternative boundary conditions, along the lines we talked about in Section 5.3. Although there are new levels of complexity and detail, it is possible to implement such a calculation in MATLAB. Fortunately, in the particular cases of interest to us there are ways to eliminate one spacial variable, and bring the problem back to the level we are more comfortable with.

9.2 Average Price Call

For the average price call we want to consider $\phi(s, y) = (\frac{y}{T} - K)^+$. We first make an important observation. Suppose that at some $t < T$ we have $\frac{1}{T}Y_t - K \geq 0$. Since $Y_t = \int_0^t S_u\,du$ is increasing we know with certaintity (probability 1) that

$$Y_T/T - K \geq Y_t/T - K > 0.$$

So the call option *will* be in the money at expiry. The corresponding put will be worthless with probability 1. Our put-call parity formula now implies the following exact expression for v for such y values:

$$v(s, y, t) - 0 = \frac{s}{rT}\left(1 - e^{-r(T-t)}\right) - e^{-r(T-t)}(K - \frac{y}{T}), \text{ if } y \geq TK. \tag{9.3}$$

A change of variables will reduce the problem to one space dimension. Consider the new variable

$$x = \frac{K - y/T}{s}.$$

If we write

$$v(s, y, t) = sc(x, t),$$

then we find (Problem 9.D) that (9.2) is equivalent to

$$\frac{1}{2}\sigma^2 x^2 c_{xx} - (\frac{1}{T} + rx)c_x + c_t = 0. \tag{9.4}$$

The terminal values can be written

$$v(s, y, T) = (y/T - K)^+ = s \cdot \left(\frac{y/T - K}{s}\right)^+ = s(-x)^+ = sx^-,$$

so we want the solution of (9.4) with terminal values

$$c(x,T) = x^-.$$

Moreover when $x \le 0$ our explict formula (9.3) applies, so

$$sc(x,t) = \frac{s}{rT}\left(1 - e^{-r(T-t)}\right) - e^{-r(T-t)}(K - \frac{y}{T}) = s \cdot \left[\frac{1}{rT}\left(1 - e^{-r(T-t)}\right) - e^{-r(T-t)}x\right].$$

Thus for $x \le 0$ we have the *exact* formula

$$c(x,t) = \frac{1}{rT}\left(1 - e^{-r(T-t)}\right) - e^{-r(T-t)}x.$$

Another interesting observation is that K does not appear in the equation (9.4), the terminal values, nor the boundary values $c(0,t)$. Thus $c(x,t)$ must be be independent of K. (The conversion from x, t back to s, y, t does involve K, so this independence from K is not true for v.)

To compute $c(x,t)$ by finite differences we only need approximate boundary values for large x. Suppose we hold s and y fixed and let $K \to \infty$. This corresponds to $x \to +\infty$ in $c(x,t)$. With $S_t = s$ and $Y_t = y$ given, the probability of $Y_T > TK$ will approach 0 as $K \to \infty$. It follows that

$$0 = \lim_{K\to\infty} v(s,y,t) = \lim_{x\to\infty} sc(x,t).$$

We deduce that

$$\lim_{x\to\infty} c(x,t) = 0.$$

Using $c(x,t) \approx 0$ as boundary conditions for large x we have all we need to carry out a finite difference computation.

9.3 Average Strike Call

Now we consider the terminal value $\phi(s,y) = (s - y/T)^+$. Define the new variable

$$z = y/s > 0.$$

Observe that if

$$v(s,y,t) = s \cdot w(z,t),$$

then for v to solve (9.2) is equivalent to w solving

$$\frac{1}{2}\sigma^2 z^2 w_{zz} + (1 - rz)w_z + w_t = 0. \tag{9.5}$$

The terminal values for v

$$v(s,y,T) = (s - y/T)^+ = s \cdot (1 - z/T)^+$$

correspond to terminal values for w:

$$w(z,T) = (1 - z/T)^+.$$

So to find the pricing function v for the average strike call, we can apply our finite difference approach to (9.5). All we need are approximate boundary conditions for $z = 0$ and z large.

9.3.1 Boundary Condtions for Large z

Since $z = y/s$, large z corresponds to both to $s \to 0$ with $y > 0$ fixed, and to $y \to \infty$ while $s > 0$ is fixed. The second of these is more revealing about w for large z. Suppose we hold s at a fixed value and think about the value $v(s,y,t)$ of the option as $y \to \infty$. For the option to expire with positive value we need $S_T > Y_T/T$.

If $Y_t = y$, then $Y_T/T > y/T$. So starting from $S_t = s$ for the option to expire in the money it is necessary that $S_T > y/T$. For s fixed, the probability of this happening $\to 0$ as $y \to \infty$. Thus we must have

$$s \lim_{z\to\infty} w(z,t) = \lim_{y\to\infty} v(s,y,t) = 0.$$

We conclude that

$$\lim_{z\to\infty} w(z,t) = 0.$$

Therefore $w(z,t) \approx 0$ for large z.

9.3.2 Boundary Conditions for $z = 0$

This is more difficult - it is hard to produce an explicit approximate expression for $w(0,t)$. This is a situation where an alternative boundary conditon seems more reliable. Wilmott [62] suggests that at $z = 0$ we should use

$$w_z(0,t) + w_t(0,t) = 0.$$

This is the result of simply substituting $z - 0$ in (9.5). It can be supported by the following argument. From the problems at the end of the chapter we know that $v = s \cdot w$ is bounded. If we presume that $\lim_{z\to 0} z^2 w_{zz} = c$ exists, then for $z \approx 0$ we would have $w_{zz} \approx c/z^2$ which suggests that w is something like $c\log(z)$. Unless $c = 0$ this would mean that $v = z \cdot w(z,t)$ is unbounded as $y \to 0$ with $s > 0$ fixed. Since we know this is not the case, it would have to be that $z^2 w_{zz} \to c = 0$ as $z \to 0$. So taking the limit in (9.5) as $z \to 0$ leads us to $w_z + w_t = 0$. This is not a rigourus justification, but it seems reasonable.

To implement this in the Crank-Nicholson approach, we would add a new discrete equation for $z_0 = 0$, using

$$\begin{aligned} w_t(0,t_{m-1/2}) &= \frac{w_0^m - w_0^{m-1}}{\Delta t} + \mathcal{O}(\Delta t^2) \\ w_z(0,t_{m-1/2}) &= \frac{1}{2} w_z(0,t_m) + \frac{1}{2} w_z(0,t_{m-1}) + \mathcal{O}(\Delta z^2), \end{aligned}$$

approximating each w_z term by

$$w_z(0,t_m) = \frac{w_1^m - w_0^m}{\Delta z} + \mathcal{O}(\Delta z).$$

A higher order (but one sided) difference approximation to w_z could be used for an $\mathcal{O}(\Delta z^2)$ approximation. As mentioned previously, the price we pay for this increased accuracy is that the resulting system is no longer tridiagonal.

9.3.3 Bounds and Approximate Formulae

There have been numerous efforts to find approximate or semi-explicit formulae for asian options. We simply wish to mention two of the most important. Rogers and Shi [48] develop lower bounds which turn out to be fairly close to the true value. Geman and Yor [21] develope a formula which is explicit except for the inversion of a Laplace transform, for which there are good numerical methods. Additional references of this type can be found in [45] and [62]. Develop some of this ...

9.4 Discrete Arithmetic Averages

We now turn our attention now to the use of discrete arithmetic averages, options with a terminal value $\phi(S_T, A_T)$ where

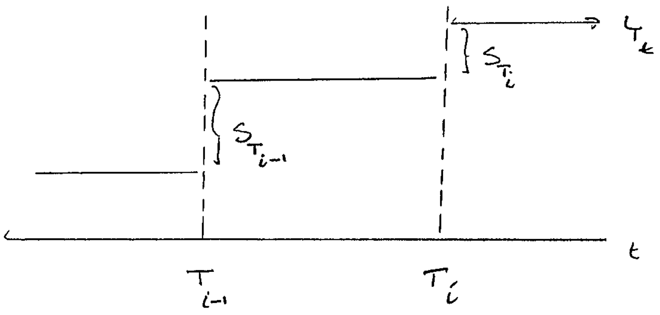
$$A_T = \frac{1}{R} \sum_1^R S_{T_i},$$

where $0 < T_1 < \cdots < T_R$ being the *record times.* There are some differences from the continuously-sampled case, but many similarities as well. Now we will use Y_t for the "running sum" of stock prices at the record times up to t:

$$Y_t = \sum_{T_i \le t} S_{T_i}.$$

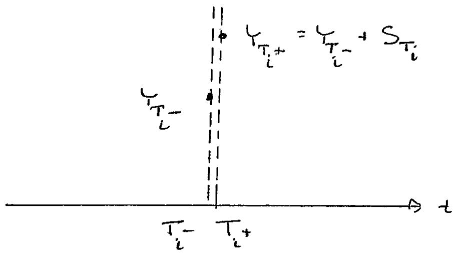
This Y_t is piecewise constant making upward jumps at the record times:

$$Y_{T_i+} = Y_{T_i-} + S_{T_i}.$$



Again we expect the market value to be given by some function $v(S_t, Y_t, t)$. As for discrete dividends, we need to be careful about how our calculations handle the jumps at the record times. On an interval (T_{i-1}, T_i) between record times Y_t is constant ($dY_t = 0$) so by our usual reasoning that $e^{-rt}v(S_t, Y_t, t)$ is a martingale, Itô's formula together with $dY_t = 0$ leads us to the conclusion that the usual Black-Scholes equation applies, with no v_y term. At a record time $v(S_t, Y_t, t)$ must be continuous.

$$\begin{aligned} v(S_{T_i}, Y_{T_i-}, T_i-) &= v(S_{T_i}, Y_{T_i+}, T_i+) \\ &= v(S_{T_i}, Y_{T_i-} + S_{T_i}, T_i+). \end{aligned}$$



This gives us the jump condition for v across a record time:

$$v(s, y, T_i-) = v(s, y + s, T_i+). \tag{9.6}$$

Finite difference calculations in financial variables are easy to implement here. We would set up a 3-dimensional grid $(s_n, y_k, t_m) = (n\Delta s, k\Delta s, m\Delta t)$ and approximate values

$$v_{n,k}^m \approx v(s_n, y_k, t_m).$$

There is every reason to use $\Delta s = \Delta y$, and we will assume that in what follows. We will work on the finite grid $s^{\mathrm{L}} = s_0 < s_n < s_{N+1} = s^{\mathrm{R}}$ and $y^{\mathrm{L}} = y_0 < y_k < y_{L+1} = y^{\mathrm{H}}$ and will need approximate formulas

$$\begin{aligned} v^{\mathrm{L}}(s^{\mathrm{L}}, y, t) &\approx v(s^{\mathrm{L}}, y, t) \\ v^{\mathrm{H}}(s^{\mathrm{H}}, y, t) &\approx v(s^{\mathrm{H}}, y, t) \\ v^{\mathrm{R}}(s, y, t) &\approx v(s, y, t) \text{ for } y \ge y^{\mathrm{H}}. \end{aligned}$$

(No formula for $y \le y^{\mathrm{L}}$ will be needed.) We will come back to the choice of formulas for these. First we want to see how the finite difference calculations are organized. For each time node t_m our approximate values $v^m_{n,k}$ ($n = 1, \dots, N$, $k = 1, \dots, L$) are organized into an $N \times L$ matrix $\mathbf{V}^m = [v^m_{\cdot,\cdot}]$. Each column corresponds to a single y_k value.

We start at $t_M = T$ and fill in the values of $\mathbf{V}^M$ using the desired formula for terminal values,

$$\begin{aligned}\phi(s, y) &= (s - y/R)^{\pm} \text{ for an average strike call or put,}\\ &= (y/R - K)^{\pm} \text{ for an average price call or put.}\end{aligned}$$

Next we want to use one of our methods, Crank-Nicholson say, to step backwards through the time nodes, $\mathbf{V}^M \to \mathbf{V}^{M-1} \to \mathbf{V}^{M-2} \cdots$ until we reach the larget record time $T_i = t_{m_i}$. These backwards time steps are computed using the Crank-Nicholson method. Since there are no v_y terms in the equation in this itme interval, the calculations apply to each column separately. We can do the calculations on all columns simultaneously:

$$\mathbf{M}\mathbf{V}^{m-1} = \mathbf{N}\mathbf{V}^m + \mathbf{B},$$

where $\mathbf{B}$ is the matrix of columns of the boundary terms:

$$\mathbf{B} = \begin{bmatrix} \frac{1}{2}a_1(v^{m-1}_{0,1} + v^m_{0,1}) & \frac{1}{2}a_1(v^{m-1}_{0,2} + v^m_{0,2}) & \cdots & \frac{1}{2}a_1(v^{m-1}_{0,L} + v^m_{0,L}) \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \\ \frac{1}{2}c_N(v^{m-1}_{N+1,1} + v^m_{N+1,1}) & \frac{1}{2}c_N(v^{m-1}_{N+1,2} + v^m_{N+1,2}) & \cdots & \frac{1}{2}c_N(v^{m-1}_{N+1,L} + v^m_{N+1,L}) \end{bmatrix}.$$

We will use

$$\mathbf{v}^{\mathrm{L}:m} = [v^{\mathrm{L}}(s^{\mathrm{L}}, y_1, t_m), \cdots, v^{\mathrm{L}}(s^{\mathrm{L}}, y_L, t_m)]$$

for the *row* vector of boundary values along the *bottom* of our grid box, and similarly

$$\mathbf{v}^{\mathrm{H}:m} = [v^{\mathrm{H}}(s^{\mathrm{H}}, y_1, t_m), \cdots, v^{\mathrm{H}}(s^{\mathrm{H}}, y_L, t_m)]$$

for the row of values along the top. We can use this with the notation $\mathbf{b}^{\mathrm{L}}$ and $\mathbf{b}^{\mathrm{H}}$ from page 50 to write

$$\mathbf{B} = \mathbf{b}^{\mathrm{L}}(\mathbf{v}^{\mathrm{L}:m-1} + \mathbf{v}^{\mathrm{L}:m})/2 + \mathbf{b}^{\mathrm{H}}(\mathbf{v}^{\mathrm{H}:m-1} + \mathbf{v}^{\mathrm{H}:m})/2.$$

Though $\mathbf{B}$ is cumbersome to write, the calculation is easy to execute in our usual way (mixing MATLAB and mathematical notation): for $\mathbf{M} =$`LU`,

$$\mathbf{V}^{m-1} = \mathtt{U}\backslash\mathtt{L}\backslash(\mathbf{N}\mathbf{V}^m + \mathbf{B}).$$

Having reached the m_i corresponding to the record time T_i, the values of $\mathbf{V}^{m_i}$ have the approximate values for $v(\cdot, \cdot, T_i+)$. Before proceeding we implement the jump conditon (9.6):

$$v(s_n, y_k, t_m-) = v(s_n, y_k + s_n, t_m+).$$

The values on the right are the "old" values; the values on the left are the "new" values. Because $\Delta s = \Delta y$, we implement the jump conditon as follows.

$$v^m_{n,k} = \begin{cases} v^m_{n,k+n} & \text{if } k + n \le L \\ v^{\mathrm{R}}(s_n, y_k + s_n, t_m) & \text{if } k + n > L. \end{cases}$$

Now $\mathbf{V}^{m_i}$ contains the desired values for t_m-. We resume the Crank-Nicholson calculations until we reach the next largest record time $t_{m_i-1} = T_{i-1}$. Then implement the jump condition again, and continue. We repeat this until we reach $t_0 = 0$.

To close our discussion of these calculations we will discuss what formulas[2] to use for v^{L}, v^{H} and v^{R} in the average price case. The natural choice for s_0 is 0. For $S_t = s_0 = 0$ we have $S_u = 0$ for all $t \le u \le T$. Therefore $Y_T = Y_t = y$, and

$$V_T = (\frac{y}{R} - K)^+,$$

[2]We would like all of these to accept a vector of y_k values and return the corresponding vector of v values.

so that $V_t = e^{-t(T-t)}(\frac{y}{R} - K)^+$. Thus we should use

$$v^{\mathrm{L}}(0, y, t) = e^{-r(T-t)}(\frac{y}{R} - K)^+.$$

In the continuously averaged case, both the large s and large y approximations used the explicit formula from the put-call parity relationship. So we need the discrete version of that.

$$V_T^{\mathrm{c}} - V_T^{\mathrm{p}} = \frac{Y_T}{R} - K,$$

so

$$\begin{aligned} V_t^{\mathrm{c}} - V_t^{\mathrm{p}} &= e^{-r(T-t)} E\left[\frac{1}{R}\sum S_{T_i} - K \,|\, \mathcal{F}_t\right] \\ &= e^{-r(T-t)}\left(\frac{1}{R}\sum E[S_{T_i} \,|\, \mathcal{F}_t] - K\right). \end{aligned}$$

Now

$$E[S_{T_i} \,|\, \mathcal{F}_t] = \begin{cases} S_{T_i} & \text{if } T_i < t \\ e^{r(T_i - t)} S_t & \text{if } t \le T_i. \end{cases}$$

Therefore

$$\begin{aligned} V_t^{\mathrm{c}} - V_t^{\mathrm{p}} &= e^{-r(T-t)}\frac{1}{R}\sum_{T_i \le t} S_{T_i} + e^{-r(T-t)}\frac{1}{R}\sum_{t < T_i} e^{r(T_i - t)} S_t - e^{-r(T-t)} K \\ &= e^{-r(T-t)}\frac{1}{R} Y_t + \frac{1}{R}\left[\sum_{t<T_i} e^{-r(T-T_i)}\right] S_t - e^{-r(T-t)} K \end{aligned}$$

To express this more concisely, define

$$\rho(t) = \frac{1}{R}\left[\sum_{t<T_i} e^{-r(T-T_i)}\right],$$

provided there is at least one record time after t: $t < T_R \le T$. If no record times remain, $T_R < t$, then

$$\rho(t) = 0.$$

Thus the put-call parity formula is

$$V_t^{\mathrm{c}} - V_t^{\mathrm{p}} = e^{-r(T-t)}\frac{1}{R} Y_t + \rho(t) S_t - e^{-r(T-t)} K.$$

Now we are ready to finish our discussion of approximate boundary formulas for the average price Asian call, using a discretely sampled average. Just as for the continuously sampled case, if ever $Y_t \ge RK$, then we know with certainty that $Y_t \ge RK$ and so the corresponding put is worthless. Therefore we have the following *exact* formula for all $y \ge RK$:

$$v^{\mathrm{R}}(s, y, t) = v^{\mathrm{c}} - v^{\mathrm{p}} = e^{-r(T-t)}\frac{1}{R} y + \rho(t) s - e^{-r(T-t)} K.$$

Finally consider large s, for y, t fixed. Given that $S_t = s$ is very large, the probability is nearly 1 that S_{T_i} will be very large for any $t < T_i$ which remain. So the put has value nearly 0 and the call is given approximaty by the same formula as above, again provided there is at least one record time T_i after t.

$$v^{\mathrm{H}}(s, y, t) = v^{\mathrm{c}} - v^{\mathrm{p}} = e^{-r(T-t)}\frac{1}{R} y + \rho(t) s - e^{-r(T-t)} K.$$

If $T_R < t < T$, so no record times remain, then Y_t will not change between t and the final time T. So $V_T = (\frac{1}{R} Y_t - K)^+$ is known today. The option is essentially a bond with face value V_T, so its value today is

$$v^{\mathrm{H}}(s, y, t) = e^{-r(T-t)}(\frac{1}{R} Y_t - K)^+.$$

9.5 Problems

Problem 9.A
Work out the explicit formulas for average price call and puts using a continuous geometric average.
. . . . $\boxed{\texttt{Geometric}}$

Problem 9.B
The risk-neutral probability measure was chosen so that $e^{-rt}S_t$ is a martingale. In particular that means that for $t < u$

$$E[S_u|\, \mathcal{F}_t] = e^{r(u-t)}S_t.$$

Let $Y_t = \int_0^t S_u\, du$ and compute

$$E[Y_T|\, \mathcal{F}_t] \tag{9.7}$$

as an expression in terms of r, S_t, Y_t, and $t < T$. (Hint: It is legitimate to interchange $\int \cdot\, du$ and $E[\cdot|\, \mathcal{F}_t]$:

$$E[\int \cdot\, du|\, \mathcal{F}_t] = \int E[\cdot|\, \mathcal{F}_t]\, du.$$

Use this and the formula for $E[S_u|\, \mathcal{F}_t]$ above.) Suppose

$$|\phi(s, y)| \le c_1 s + c_2 y + c_3 \tag{9.8}$$

(where c_i are constants) and that

$$v(S_t, Y_t, t) = e^{rt}E[\phi(S_T, Y_T)e^{-rT}\,|\, \mathcal{F}_t]. \tag{9.9}$$

Show that

$$|v(s, y, t)| \le \tilde{c}_1 s + \tilde{c}_2 y + \tilde{c}_3$$

for some other constants $\tilde{c}_i$. In other words among all the possible solutions of (9.2), those that correspond to the value of an Asian option whose final value is $V_T = \phi(S_T, Y_T)$, where ϕ is some function satisfying (9.8), have the property that $v(s, y, t)$ likewise grows at most linearly in s and y.
. . . . $\boxed{\texttt{U}}$

Problem 9.C
Since each $v^{(i)}$ for $1 = 1, \ldots, 6$ from Section 2.6 have no y-dependence, they must also solve (9.2). (Explain this.) Show that the following also solves (9.2):

$$v^{(6)}(s, y, t) = e^{-r(T-t)}(s - ry)$$

By finding a linear combination of $v^{(i)}$ which has the correct value at $t = T$, find the explicit formulas for

1. The solution with $v(s, y, T) = y$;
2. the put-call parity expression $v^{\text{Call}} - v^{\text{Put}}$ for the case of average strike Asian options;
3. the put-call parity expression $v^{\text{Call}} - v^{\text{Put}}$ for the case of average rate Asian options.

(Your answer for item 1 should be consistent with the expression you derived for (9.7) above. Your answers for items 2 and 3 should agree with the expressions we earlier in this chapter.)
. . . . $\boxed{\texttt{V}}$

Problem 9.D
Show that if $x = \frac{K-y/T}{s}$ and functions $v(s, y, t)$ and $c(x, t)$ which are related to each other according to

$$v(s, y, t) = sc(x, t),$$

then v solves the Asian version (9.2) of the Black-Scholes equation if and only if c solves (9.4). Find an example of an explicit solution c of (9.4) which involves both variables x and t.

.... `W`

Problem 9.E
The computation of an Asian fixed-strike put or call based on a discrete time arithmetic average can be carried out in terms of the same change of variables as in Section 9.2. In this problem you are asked to work out some details of such an approach. Specifically, answer the following questions:

1. What PDE should $c(x,t)$ solve between record times: $T_{i-1} < t < T_i$?
2. If we solve for approximate values $c_n^m \approx c(x_n, t_m)$ on a grid, how would the "jump condition" (9.6) appear in the calculations?
3. What constraints on the step sizes Δx and Δt are appropriate?
4. In the case of a call, what terminal and boundary values should be used for $c(x,t)$?

.... `Z`

Problem 9.F
There is an alternate change of variables for the average price call. Let

$$\theta = s/y = 1/z.$$

Write

$$v(s,y,t) = y \cdot u(\theta,t)$$

and show that the appropriate equation for $u(\theta,t)$ is

$$\frac{1}{2}\sigma^2\theta^2 u_{\theta\theta} + \theta(r-\theta)u_\theta + u_t + (x-r)u = 0,$$

and the terminal values for u are

$$u(\theta,T) = (\theta - \frac{1}{T})^+.$$

For $\theta = 0$ we should use $u(0,t) = 0$. Can you speculate on a derivative boundary conditon for large θ based on our $w_{zz} \approx 0$ for small z above?

.... `apalt`

Problem 9.G
Find coefficient formulas a_n, b_n, c_n so that the Crank-Nicolson method applied to equation (9.4) above takes the same form (5.7) as our previous calculations for the Black-Scholes equation. (Simply revise the derivation leading to (5.1) to obtain the new $\mathbf{B}$, $\mathbf{b}^{\mathrm{L}}$, and $\mathbf{b}^{\mathrm{H}}$ that correspond to (9.4).)

.... `X`

Problem 9.H
Write MATLAB code to compute the solution $c(x,t)$ to equation (9.4) associated with a average price Asian call option, using the explicit boundary conditions that we developed in class. Use your computed values to plot $v(s,0,0)$ versus s and compare it to the price of the standard European call $v^{\text{E-Call}}(s,0)$ with the same parameters. Try a few different choices for the parameter values. Based on what you observe, conjecture a relationship between the Asian and standard European call options.

Writing the code should not be too bad; you can just modify `genB.m` and `cnf.m` appropriately. It would be a good idea to change variable names to match the notation we have used in the notes: `x` instead of `s`, `cL.m` instead of `vL.m` and so forth. That will help minimize confusion. To check that your code is working, you could first try it on the explicit solution from Problem 9.D before you set up the terminal and boundary conditions for the average price (which means the same as "fixed strike") Asian call.

There are several things to think about in order to use your calculated values of $c(x,0)$ to answer the rest of the problem. You will probably calculate $c(x_n,0)$ for a uniformly spaced vector of x_n values: $0 = x^{\mathrm{L}} = x_0 < x_1 < \cdots x_N < x_{N+1} = x^{\mathrm{H}}$. You will need to decide what x^{H} to use. You don't want it so small that the boundary condition at x^{H} is inconsistent with the the other values of $c(x_n,0)$ for large n. Nor do you want x^{H} so large that most of your $c(x_n,0)$ values are wasted calculations that just end up agreeing with the approximate boundary values. Plot your computed $c(x,0)$ values and experiment with x^{H} to see.

Once you are satisfied with x^{H}, you need to convert your x_n and $c(x_n,0)$ values to corresponding s_n and $v(s_n,0,0)$ values. The order of the resulting s_n values will be reversed, and they will be unevenly spaced. You will probably get some very large s_n values. (In fact $x_0 = 0$ corresponds to $s_0 = \infty$!) If you plot $v(s,0,0)$ versus s, you won't want to include the full range of s values, because the "important" part of the graph would only be a tiny part of it. You can either plot only some of the s and v values, or else plot them all and then use an `axis` command to limit the part of the plot that is displayed. You may find it convenient to produce the value of $v(s^{\mathrm{ref}},0,0)$ for some reference s^{ref}-value for comparison purposes as you vary the parameters (as per the problem's instructions). Since s^{ref} is not likely to be one of the s_n values you will need to interpolate from the s_n, $v(s_n,0,0)$ values. That is easy to do with an interpolation command, like `interp1(s,v,sref)`.

As a check if you run the calculation using $r = .05$, $\sigma = .2$, $T = 2$, with $K = 10$, you should a value $v(10,0,0) \approx .87$.

. . . . $\boxed{\texttt{Y}}$

Problem 9.I

Find the out-call parity relation for average strike options using discretely computed arithmetic averages. If we write $v(s,y,t) = sw(z,t)$, $z = y/s$ as we did for continuous time averages, describe the jump condition at record times, initial values, and boundary conditions for $z = 0$ and $z \to \infty$.

. . . . $\boxed{\texttt{das}}$

Chapter 10

Lookbacks

In this chapter we consider options whose value at expiry depends on both S_T and the historical maximum (or minimum) stock price. These are generally called *lookback* options. For instance a *floating strike put* uses

$$V_T = Z_T - S_T,$$

with $Z_T = \max_{0\le t\le T} S_t$ being the maximum stock price to date. Other variants are described below.

10.1 Lookback Varieties

The historical maximum (or minimum) stock price can be computed either using *continuous sampling*,

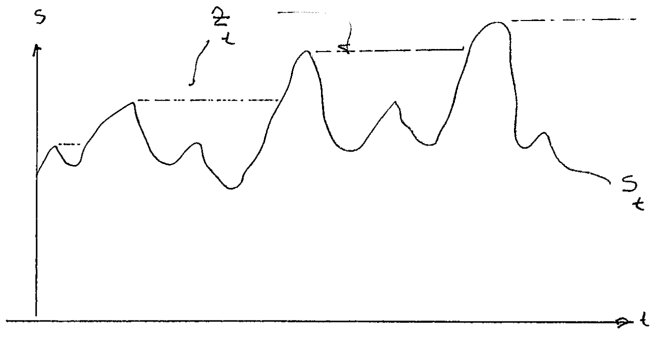
$$Z_t = \max_{0\le u\le t} S_u \quad (\text{ or } Z_t^{\min} = \min_{0\le u\le t} S_u \ ),$$

or using *discrete sampling* based on a prescribed sequence of *record times* $0 \le T_1 < T_2 < \cdots$,

$$Z_t = \max_{T_i\le t} S_{T_i} \quad (\text{ or } Z_t^{\min} = \min_{T_i\le t} S_{T_i} \ ).$$

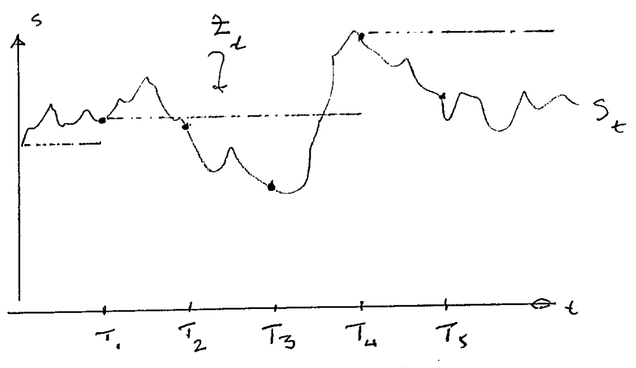
Discrete sampling is more realistic for describing real-world financial calculations, but continuous sampling is more natural mathematically. If the record times are many and closely spaced then we would expect continuous sampling to provide a good approximation. We will consider both in our discussion below.

Although only Z_T is needed when we actually determine the final value V_T, the evolution of Z_t as t progresses from $t = 0$ to $t = T$ is important for our calculations of V_0. In the case of continuous sampling Z_t is a continuous nondecreasing stochastic process with many flat sections in its typical path:



This makes it a rather different kind of stochastic process than we have encountered before.

In the discretely sampled case Z_t is constant between record times T_i and makes upward jumps at the record times themselves:



Our discussion of this will be somewhat similar to that of discrete dividend payments in Chapter 7 and discretely averaged Asian option in Chapter 9; things will be very Black-Scholes-like between the record times, and some sort of discontinuity has to be worked out at the record times themselves.

Given the choice of discrete or continuous sampling, there are serveral varieties of lookback options that are usually considered.

- *Floating Strike Put* (also called a *Lookback Put*):

$$V_T = (S_T - Z_T)^-,$$

  where Z_T is maximum stock price (either continuously or discretely sampled). In the continuously sampled case $Z_T \geq S_T$ always holds, so $(S_T - Z_T)^- = Z_T - S_T$, but in the discretely sampled case, if T is not a record time T_i then $S_T > Z_T$ is possible so we need to keep the $(\cdot)^-$.

- *Floating Strike Call* (also called a *Lookback Call*):

$$V_T = (S_T - Z_T^{\min})^+,$$

  The $(\cdot)^+$ is unnecessary in the continuously sampled case.

- *Floating Price Put* (also called a *Forward Lookback Put*):

$$V_T = (Z_T^{\min} - K)^-.$$

- *Floating Price Call* (also called a *Forward Lookback Call*):

$$V_T = (Z_T - K)^+.$$

These are all of the form

$$V_T = \phi(Z_T, S_T)$$

where ϕ is one of the functions $\phi(s, z) = (s - z)^{\pm}$ or $\phi(s, z) = (z - k)^{\pm}$. In general we are interested in computing the pricing function $v(s, z, t)$ which gives the option's market value in terms of the current values of the S_t, Z_t:

$$V_t = v(S_t, Z_t, t).$$

We will consider PDE descriptions of v first. For continuous sampling explicit formulae for v are available for all of option types above. That will be discussed in the final section.

10.2 The Lookback PDE

We first consider the continuously sampled case. We always have $S_t \le Z_t$. As always the risk-neutral pricing formula tells us that

$$e^{-rt}v(S_t, Z_t, t)$$

is a martingale. We know the terminal value,

$$v(s, z, T) = z - s.$$

In those time intervals where $S_t < Z_t$ we know Z_t is constant, so we expect that

$$d[e^{-rt}v(S_t, Z_t, t)] = e^{-rt}[\frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + v_t - rv]\,dt + (\cdots)\,dW_t,$$

with no terms involving v_z. Thus we anticipate that for $s < z$ and $t < T$ we should have our usual Black-Scholes equation for $v = v(s, z, t)$:

$$\frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + v_t - rv = 0.$$

But something new will be involved for $s = z$. To see what it should be we need to know how to deal with dZ_t when we work out the stochastic differential of our risk neutral martingale. It is not simply some new formula of the form $dZ_t = (\cdots)\,dt + (\cdots)\,dW_t$. Z_t is a different kind of stochastic process than we have encountered before, a process of "bounded variation." Integrals $\int \cdots dZ_t$ are Rieman-Stieltjes type, closer to $\int \cdots dt$ than to Ito's stochastic integrals $\int \cdots dW_t$. The details of stochastic calculus with processses of bounded variation thrown in is more complicated than we can try to summarize here. (The courageous reader may want to consult Rogers & Williams [50].) In the end however it does turn out that a version of Ito's formula still holds, if in addition to the basic differential rules $(dt)^2 = dt\,dW_t = 0$ and $(dW_t)^2 = dt$ we add

$$(dZ_t)^2 = dt\,dZ_t = dZ_t\,dW_t = 0.$$

If we are willing to take all this for granted, then we are led to the following stochastic differential expression for our risk-neutral martingale.

$$d[e^{-rt}v(S_t, Z_t, t)] = e^{-rt}v_z\,dZ_t + e^{-rt}\left[\frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + v_t - rv\right]dt + (\cdots)\,dW_t.$$

To be a martingale it is necessary that *both* the dt and dZ_t components vanish. In other words the dt and dZ_t are incompatible, just like dt and dW_t are, so these terms can't cancel. They must be 0 individually. When $S_t < Z_t$ we have $dZ_t = 0$, but when $S_t = Z_t$ we have $dZ_t \ne 0$ so it must be that its coefficient $e^{-rt}v_z$ in the equation above vanishes. This leads us to the following PDE description of $v(s, z, t)$.

$$\boxed{\begin{aligned} \frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + v_t - rv &= 0 && \text{for } 0 < s < z,\ t < T,\\ v_z &= 0 && \text{for } s = z,\\ v(s, z, 0) &= z - s && \text{for } t = T. \end{aligned}} \tag{10.1}$$

In the discretely sampled case there is no requirement that $s \le z$, and since Z_t is contant between record times the appropriate PDE is just the usual Black-Scholes equation ((10.1) with no v_y term). But there are discontinuities at the record times $t = T_i$. We need to develope the appropriate jump conditions. The maximum observed stock price has a jump at T_i:

$$Z_{T_i+} = \max(Z_{T_i-}, S_{T_i}).$$

Since the market price $V_t = v(S_t, Z_t, t)$ must be continuous in t, it follows that

$$v(s, z, T_i-) = v(s, \max(z, s), T_i+). \tag{10.2}$$

This describes how the values of v are to be adjusted as a calculation moves from values of $t > T_i$ to values $t < T_i$.

To continue with this PDE formulation we would need approximate boundary conditions, as well as the difficulties of coping with the wedge-shaped domain of (10.1). For the particular case of floating strike options we can make a simplifying variable change, as we did for Asians. The next section deals specifically with that case.

10.3 Floating Strike Puts

We will focus on the floating strike put, and will consider both the continuous and discretely sampled versions. The terminal value function is

$$\phi(s,z) = (s-z)^-.$$

A stochastic argument will lead us to the desired change of variables.

$$v(S_t, Z_t, t) = V_t = e^{-r(T-t)}E[Z_T - S_T \,|\, \mathcal{F}_t].$$

We know $S_t = S_0 e^{(r-\sigma^2/2)t+\sigma^2 W_t}$. For $t \le u$ we can write

$$S_u = S_t G_{t,u},$$

where

$$G_{t,u} = S_u/S_t = e^{(r-\sigma^2/2)(u-t)+\sigma^2(W_u-W_t)}.$$

The important observation is that $G_{t,u}$ is independent of $\mathcal{F}_t$ because $W_u - W_t$ is. Now write

$$\begin{aligned} Z_T &= \max(Z_t, S_t \max_{t\le u\le T} G_{t,u}) \\ &= Z_t \max(1, \xi_t \max_{t\le u\le T} G_{t,u}), \text{ where } \xi_t = S_t/Z_t. \end{aligned}$$

Likewise

$$S_t = S_t G_{t,T} = Z_t \xi_t G_{t,T}.$$

So we find that

$$\begin{aligned} V_t &= e^{-r(T-t)}E[Z_T - S_T \,|\, \mathcal{F}_t] \\ &= e^{-r(T-t)}E[Z_t \max(1, \xi_t \max_{t\le u\le T} G_{t,u}) - Z_t\xi_t G_{t,T} \,|\, \mathcal{F}_t] \\ &= e^{-r(T-t)}Z_t E[\max(1, \xi_t \max_{t\le u\le T} G_{t,u}) - \xi_t G_{t,T} \,|\, \mathcal{F}_t]. \end{aligned}$$

Now ξ_t is $\mathcal{F}_t$ measurable, and both $G_{t,T}$ and $\max_{t\le u\le T} G_{t,u}$ are independent of $\mathcal{F}_t$. This implies (see (9.1)) that the conditional expectation above is of the form

$$E[\max(1, \xi_t \max_{t\le u\le T} G_{t,u}) - \xi_t G_{t,T} \,|\, \mathcal{F}_t] = g(\xi_t, t)$$

for some function $g(\cdot,\cdot)$. We find then that

$$V_t = Z_t e^{-r(T-t)} g(\xi_t, t) = Z_t w(\xi_t, t)$$

for some function $w(\xi, t)$ of two variables. In brief, we should look for $v(s,z,t)$ in the form

$$v(s,z,t) = z w(\xi, t), \text{ where } \xi = s/z. \tag{10.3}$$

This agrument is also valid in the discretely sampled case if we replace $\max_{t\le u\le T} G_{t,u}$ with $\max_{t<T_i\le T} G_{t,T_i}$.

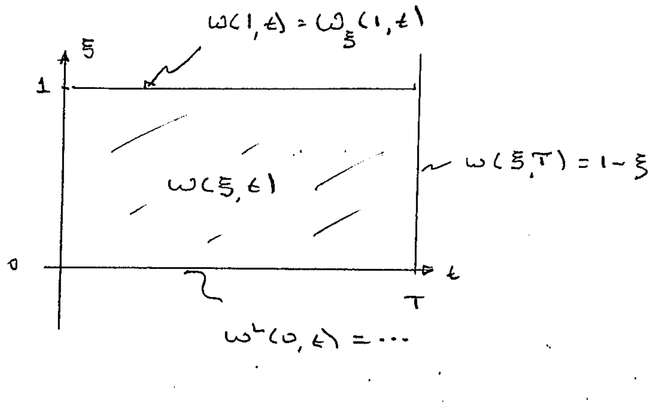
10.3.1 Continuous Sampling

In the case of continuous sampling, the limitation to $0 < s < z$ means we are only interested the function $w(\xi, t)$ defined in the strip $0 < \xi \le 1$, $t \le T$. The interior of this region, $\xi < 1$, corresponds to $s < z$. The Black-Scholes PDE converts easily: $v(s, z, t) = zw(s/z, t)$ so

$$\begin{aligned} 0 &= \frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s - rv + v_t \\ &= \frac{1}{2}\sigma^2 s^2 z\, z^{-2} w_{\xi\xi} + rsz\, z^{-1} w_\xi + rzw - zw_t \\ &= z[\frac{1}{2}\sigma^2 \xi^2 w_{\xi\xi} + r\xi w_\xi - rw + w_t]. \end{aligned}$$

In other words, in the strip $0 < \xi < 1$, $t < T$ the function $w(\xi, t)$ again satisfies the Black-Scholes PDE but now in the new variables (ξ, t):

$$0 = \frac{1}{2}\sigma^2 \xi^2 w_{\xi\xi} + r\xi w_\xi - rw + w_t.$$



Terminal Conditions

The terminal conditions are

$$zw(\xi, T) = v(s, z, T) = z - s = z(1 - \xi), \quad \text{so } w(\xi, T) = 1 - \xi.$$

Boundary Conditions

For boundary conditions, Problem 10.A will provide approximate values for $\xi \approx 0$ which can be used for lower boundary conditions.

The boundary at $\xi = 1$ corresponds to $s = z$ which is where we require $v_z = 0$. Since $v(s, z, t) = zw(s/z, t)$ we have

$$v_z = w + z(\frac{-s}{z^2})w_\xi = w - \xi w_\xi,$$

so the boundary condition for w at $\xi = 1$ is

$$w(1, t) = w_\xi(1, t).$$

We can incorporate this boundary condition into finite difference calculations just as we did previously.

(Recall that for discrete sampling we had $w(\xi, T_m-) = \xi w(1, T_m+)$ for $\xi > 1$. Thus $w_\xi(1, T_m-) = w(1, T_m+)$ just after each update. If we think of continuous sampling as the limit of discrete sampling as the record times fill up a dense set in $[0, T]$, it seems natural that the $w_\xi(1, T_m-) = w(1, T_m+)$ at record times should lead to $w_\xi(1, t) = w(1, t)$ at all $t < T$ in the continuous case.)

10.3.2 Discrete Sampling

We have already observed that Z_t is constant between the record times T_i, and that $v(s,z,t)$ should satisfy the standard Black-Scholes PDE in the variables (s,t) for each value of z:

$$\frac{1}{2}\sigma^2 s^2 v_{ss} + rsv_s + v_t - rv = 0, \text{ for } T_{i-1} < t < T_i.$$

As in the continuous case, this is equivalent to

$$\frac{1}{2}\sigma^2 \xi^2 w_{\xi\xi} + r\xi w_\xi + w_t - rw = 0.$$

We also need to work out the appropriate reformulation of the jump condition (10.2) in terms of w. First observe that

$$Z_{T_i+} = \max(Z_{T_i-}, S_{T_i}) = \max(1, S_{T_i}/Z_{T_i-})Z_{T_i-} = \max(1, \xi_{T_i-})Z_{T_i-},$$

and therefore

$$\xi_{T_i+} = \frac{S_{T_i}}{Z_{T_i+}} = \frac{S_{T_i}}{\max(1,\xi_{T_i-})Z_{T_i-}} = \min(1, 1/\xi_{T_i-})\xi_{T_i-} = \min(\xi_{T_i-}, 1).$$

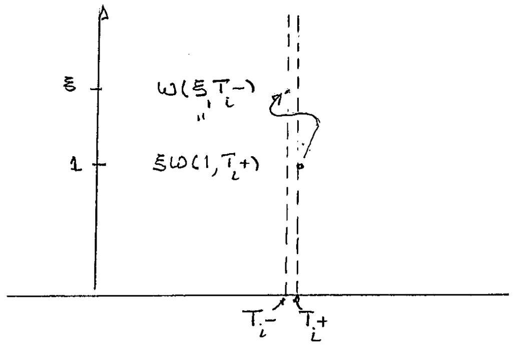
The continuity requirement becomes

$$\begin{aligned} v(S_{T_i}, Z_{T_i-}, T_i-) &= v(S_{T_i}, Z_{T_i+}, T_i+) \\ Z_{T_i-}w(\xi_{T_i-}, T_i-) &= Z_{T_i+}w(\xi_{T_i+}, T_i+) \\ &= \max(1, \xi_{T_i-})Z_{T_i-}w(\min(\xi_{T_i-}, 1), T_i+), \end{aligned}$$

or simply that

$$\begin{aligned} w(\xi, T_i-) &= \max(1,\xi)w(\min(1,\xi), T_i+) \\ &= \begin{cases} \xi w(1, T_i+) & \text{for } \xi > 1 \\ w(\xi, T_i+) & \text{for } \xi \leq 1. \end{cases} \end{aligned}$$

The following figure illustrates.



In order to implement this, it would be *very helpful* to arrange for $\xi = 1$ to be a grid point.

Terminal Conditions

Assume that $T = T_R$ is a record time. Then the final value is

$$v(s,z,T) = z - s = z(1-\xi),$$

and therefore we would use

$$w(\xi, T_R) = 1 - \xi$$

to get finite difference calculations started.

If the last record time is prior to the terminal time, $T_R < T$, then for $T_R \leq t \leq T$ we have $Z_t = Z_{T_R}$. The option's value final value is $(S_T - Z_{T_R})^-$, essentially a put on the stock, with exercise price Z_{T_R}. Its value for at T_R is therefore $v^{\mathrm{Put};T,K=Z_{T_R}}$. This is the value we should use for $v(S_{T_R}, Z_{T_R+}, T_R+)$. This can be expressed in the form $Z_{T_R} w(\xi_{T_R}, T_R+)$. We should start our calculations with the values from this formula, immediately apply the jump condition to obtain the values of $w(\xi_{T_R}, T_R-)$ and then proceed with the finite difference calculations until we reach the next largest record time, implement the jump conditions again, and continue.

Boundary Conditions

Problem 10.A will develope boundary conditions to use for $\xi = 0$. We focus on an approximate formula for $\xi \to \infty$. Suppose at least one record time is still to come, $T_{i-1} < t < T_i$. (Otherwise we have an exact formula, above.) On this time interval $Z_t = Z_{T_{i-1}}$ is constant, so for $\xi = S_t/Z_t \to \infty$ means that $s = S_t$ is very large compared to $Z_{T_{i-1}}$. That means the probability is nearly 1 that S_{T_i} will be greater than $Z_{T_{i-1}}$, in which case $Z_{T_i} = S_{T_i}$. So with near certainty the value at T_i will be

$$V_{T_i} = Z_{T_i} w(1, T_i) = S_{T_i} w(1, T_i).$$

Therefore the value at t will very nearly equal

$$V_t = S_t w(1, T_i).$$

Thus for $T_{i-1} < t < T_i$, an approximate formula as $\xi \to \infty$ is

$$z w(\xi, t) \approx s w(1, T_i) = z \xi w(1, T_i),$$

or simply

$$w(\xi, t) \approx \xi w(1, T_i).$$

As we calculate for values of t decreasing through this interval we will have the value of $w(1, T_i)$ available for use. We note that Wilmott [61] has suggested using the derivative boundary condition $w_\xi(\xi, t) = \frac{1}{\xi} w(\xi, t)$ on the upper boundary $\xi = \xi_b$. We see that this is consistent with our approximate formula.

10.4 Explicit Formula

When Z_T is continuously sampled, explicit formulas exist for all the lookback versions described above; see Conze and Viswanathan [13]. This is because the joint distribution of (Z_T, S_T) conditional on $\mathcal{F}_t$ is known. So if $V_T = \phi(Z_T, S_T)$ for some function $\phi(z, s)$, we can (in principal) work out a formula for V_t based on the risk-neutral valuation formula

$$V_t = e^{rt} E[\phi(Z_T, S_T) e^{-rT} \,|\, \mathcal{F}_t]$$

by integrating ϕ with respect to the known conditional distribution of (Z_T, S_T), provided we can get through the details of the integration. In this section we want to chart our way through these calculations, and work out at least some of the particulars.

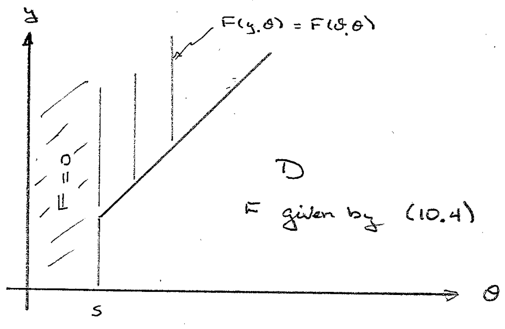
We are interested in the conditional joint distribution function

$$P(S_t \leq y \text{ and } Z_T \leq \theta \,|\, \mathcal{F}_t).$$

The conditioning makes this dependent on $s = S_t$ and $z = Z_t$, and we are only interested in $0 < s \leq z$. Since $z = Z_t \leq Z_T$, the above probability is 0 if $\theta < z$. And if $z \leq \theta$, then $Z_T \leq \theta$ if and only if $\sup_{t \leq u \leq T} S_u \leq \theta$. Thus the dependence on z is only whether or not $z \leq \theta$. We will assume $t = 0$, $Z_0 = S_0 = s$ and work out

$$F(y, \theta) = P(S_T \leq y \text{ and } Z_T \leq \theta).$$

(For the general case of $s < z$ and $0 < t < T$ we just multiply by $1_{[0,\theta]}(z)$ and replace T by $T - t$.)



Notice that $F(y,\theta) = 0$ if $\theta < s$, and since $S_T \le Z_T$ it is constant in $y \ge \theta$. So we will concentrate on $0 < y \le \theta$ and $0 < s \le \theta$.

Because (as (10.4) will confirm) F is continuous we have

$$\begin{aligned} F(y,\theta) &= P(S_T \le y \text{ and } Z_T \le \theta) \\ &= P(S_T < y \text{ and } Z_T < \theta) \\ &= P(S_T < y \text{ and } T < \mathcal{T}^\theta), \text{ using the notation of Section 3.2} \\ &= E[1_{(0,y)}(S_T);\ T < \mathcal{T}^\theta]. \end{aligned}$$

This is the expected terminal value of an up & out barrier option with terminal value function $\phi(s) = 1_{(0,y)}(s)$. So $F(y,\theta) = e^{rT} v(s,0)$ where v is the pricing function for that barrier option. We can find v by working out the development of Section 3.2. We observe that since $y \le \theta$ the truncated terminal value (3.5) is again $1_{(0,y)}(s)$.

$$1_{(0,y)}(s) = 1_{(0,1)}(s/y) = v^{(2)}(s/y, T),$$

Therefore $v^\theta(s,t) = v^{(2)}(s/y,t)$, and so

$$v(s,t) = v^{(2)}\left(\frac{s}{y}, t\right) - \left(\frac{s}{\theta}\right)^\kappa v^{(2)}\left(\frac{\theta^2}{sy}, t\right).$$

We deduce the following formula for the joint distribution function.

$$\boxed{F(y,\theta) = \mathcal{N}\left(-d^{(2)}\left(\frac{s}{y},0\right)\right) - \left(\frac{s}{\theta}\right)^\kappa \mathcal{N}\left(-d^{(2)}\left(\frac{\theta^2}{sy},0\right)\right);\ 0 < y \le \theta,\ 0 < s \le \theta.} \tag{10.4}$$

Observe that we get $F = 0$ for $s = \theta$ as well as in the limit as $y \downarrow 0$ (because $d^{(2)} \to \infty$).

In principle $f(y,\theta) = \partial_y \partial_\theta F(y,\theta)$ would produce a formula for the joint density of (S_T, Z_T). Then, for a given lookback terminal value function ϕ we calculate the initial price of the option (if $S_0 = s$, $Z_0 = 0$) by working out the integral

$$v(s,0,0) = e^{-rT} E[\phi(S_T, Z_T)] = e^{-rT} \iint_D \phi(y,\theta) f(y,\theta)\, dy\, d\theta,$$

where $D = \{(y,\theta) : 0 < y \le \theta,\ s \le \theta\}$.

10.4.1 The Mean of the Maximal Price

To illustrate, consider again the floating strike put. We want to find

$$v(s,0,0) = e^{-rT} E[Z_T - S_T] = e^{-rT} E[Z_T] - s.$$

So all we really need is $E[Z_T]$. By taking $y = \theta$ in (10.4) we get the distribution function of Z_T:

$$\begin{aligned} F^{Z_T}(\theta) &= P(Z_T \le \theta) \\ &= \begin{cases} 0 & \text{if } \theta \le s \\ \mathcal{N}\left(-d^{(2)}\left(\frac{s}{\theta}, 0\right)\right) - \left(\frac{s}{\theta}\right)^{\kappa} \mathcal{N}\left(-d^{(2)}\left(\frac{\theta}{s}, 0\right)\right) & \text{if } s < \theta. \end{cases} \end{aligned} \tag{10.5}$$

Recall from (2.23) that $\kappa = 1 - \frac{2r}{\sigma^2}$, and that

$$d^{(2)}(u, 0) = \frac{\ln(u) + (r - \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}.$$

It will be more convenient to work in terms of

$$\zeta = \log(Z_T/s) \text{ and } x = \log(\theta/s).$$

This makes ζ a nonnegative random variable, with distribution function

$$F^{\zeta}(x) = P(\zeta \le x) = \mathcal{N}\left(\frac{x - (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right) - e^{-\kappa x}\mathcal{N}\left(\frac{-x - (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right).$$

Since $Z_T = s e^{\zeta}$, what we are after is

$$v(s, 0, 0) = e^{-rT}E[Z_T] - s = s(e^{-rT}E[e^{\zeta}] - 1).$$

We could differentiate to find an expression for the density of ζ, but it is more convenient work directly in terms of the distribution functions. We can do that with the following "integration by parts" formula, valid for any random variable X with a continuous distribution function $F^X(x)$ and any real numbers b, c:

$$c\int_b^\infty e^{cx}(1 - F^X(x))\,dx = E[e^{cX};\, X \ge b] - e^{cb}(1 - F^X(b)). \tag{10.6}$$

If we apply the formula with $c = 1$ and $b = 0$ we obtain

$$\begin{aligned} E[e^{\zeta}] &= 1 + \int_0^\infty e^x\left(1 - F^{\zeta}(x)\right)\,dx \\ &= 1 + \int_0^\infty e^x\left(1 - \mathcal{N}\left(\frac{x - (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right)\right)dx + \int_0^\infty e^{(1-\kappa)x}\mathcal{N}\left(\frac{-x - (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right)dx \\ &= 1 + \int_0^\infty e^x\left(1 - \mathcal{N}\left(\frac{x - (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right)\right)dx + \int_0^\infty e^{(1-\kappa)x}\left(1 - \mathcal{N}\left(\frac{x + (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right)\right)dx. \end{aligned}$$

Each of these integrals can be resolved by using (10.6) again in conjunction with (2.26). For instance, in the first integral we can write

$$F^X(x) = \mathcal{N}\left(\frac{x - (r - \sigma^2/2)T}{\sigma\sqrt{T}}\right)$$

where $X = \sigma\sqrt{T}\,Y + (r - \sigma^2/2)T$ and Y is a standard normal random variable. So in (10.6) we can write

$$E[e^X;\, X \ge 0] = e^{(r-\sigma^2/2)T}E\left[e^{\sigma\sqrt{T}\,Y};\, Y \ge -\frac{(r - \sigma^2/2)T}{\sigma\sqrt{T}}\right],$$

for which we can use the formula (2.26). This leads to a formula for the first integral, and the second is worked out in a similar manner. One has to be careful in working out the details, but the point is that we *do* a strategy to produce a formula. When worked out this leads to

$$v(s, 0, 0) = s \cdot \left\{e^{-rT}\left(1 - \frac{\sigma^2}{2r}\right)\mathcal{N}\left(\frac{(-r + \sigma^2/2)T}{\sigma\sqrt{T}}\right) + \frac{\sigma^2}{2r}\mathcal{N}\left(\frac{(r + \sigma^2/2)T}{\sigma\sqrt{T}}\right) - \mathcal{N}\left(\frac{-(r + \sigma^2/2)T}{\sigma\sqrt{T}}\right)\right\}.$$

This is the formula *only* for the initial price of the put, $t = 0$ with $Z_0 = s$. The general formula for $0 < t < T$ the formula also involves $z = Z_t$; you can find it as (14) in [13]. Our point here is just to illustrate that it is possible to work out such formulas with the tools we have developed. References such as [13] and [45] which develop the formulas in more generality resort to some additional devices to simplify the development.

10.5 Problems

Problem 10.A

If we write the value function for a lookback put in the form $v(s, z, t) = z\, w(\xi, t)$ with $\xi = s/z$, then the appropriate boundary values for w when $\xi = 0$ are

$$w(0, t) = e^{-r(T-t)}.$$

Explain why this is correct, regardless of whether Z_t is a discretely or continuously sampled maximum stock price up to time t.

. . . . $\boxed{\texttt{AA}}$

Chapter 11

Monte Carlo Methods

Finite difference methods can be used effectively in up to 3 space dimensions. But they become unwieldly beyond that point. (We have seen that path-dependent options for a single stock lead us to 2 space dimensions. If we were to consider multiple stocks that would also produce multiple space dimensions.) The conventional wisdom is that in higher dimensions Monte Carlo methods are preferable. (See Tavella [56].) Here we offer a brief look at the general idea of Monte Carlo methods.

According to the risk-neutral formula, the price v of an option (for given current stock price $S_{t_0} = s_0$) involves the mean of a certain random variable:

$$v = e^{-r(T-t_0)}E[X].$$

The plan is to generate a sequence $X_1, X_2, \ldots, X_n$ of i.i.d. random variables each with the distribution of X, and then take the sample mean as an approximation to the true mean $m = E[X]$:

$$m \approx \frac{1}{N}\sum_{1}^{N} X_i.$$

The validity of this is based on the Strong Law of Large Numbers, which says that $\frac{1}{N}\sum_1^N X_i \to m$ almost surely as $N \to \infty$. Before using this for an option pricing calculation we will talk about the idea in general.

11.1 Some History

The earliest known instance of a Monte Carlo calculation seems to be in 1777 when the French scientist Georges-Louis Leclerc, Comte de Buffon, flipped a coin 2084 times as part of an experiment to resolve the St. Petersburg problem. The St. Petersburg problem involves an infinite mean, so that is not a very good example for us. But he also designed an experiment intended to produce an approximate value for π. This has come to be known as *Buffon's Needle Problem*: rule parallel lines on the floor separated by a units. Take a "needle" or piece of wire ℓ units long and throw it randomly on the floor. The problem is to find the probability that the wire will land crossing or touching one of the lines. The answer (for a "short" needle: $\ell < a$) works out to be

$$p = \frac{2\ell}{a\pi}.$$

In 1864, Asaph Hall, an astronomer at the U. S. Naval Observatory, decided to carry out the the experiment. He engaged a recovering sea caption (O. C. Fox) to repeatedly throw a wire on a floor with lines ruled on it and count the number of times the wire landed on a line. Each throw of the wire determines a random variable X_i according to

$$X_i = \begin{cases} 1 & \text{if it lands on a line} \\ 0 & \text{if it lands not touching a line.} \end{cases}$$

Using the estimate $p \approx \frac{1}{N}\sum_1^N X_i$ leads to an approximate value of π. One of his experiments used $\ell = 3$ and $a = 4$, and throwing the wire $N = 530$ times he found that $\sum_1^N X_i = 253$. This implies that

$$\frac{2\cdot 3}{4\cdot \pi} \approx \frac{253}{530}, \text{ which says that } \pi \approx \frac{2\cdot 3\cdot 530}{4\cdot 253} = 3.14229\cdots$$

The technique was suggested again in a more serious context by S. Ulam, a mathematician working on the development of the atomic bomb at Los Alamos durring World War II. He said that it first occurred to him as a quick way to estimate the probability of a successful solitare hand. In 1946 he proposed it to John von Neumann as a potentially useful tool for the Los Alamos work, and the two of them started working out details. All the work at Los Alamos was very secret, so the work needed a code name. Von Neumann chose "Monte Carlo" as the code name, after the mediterranean city famous for it's gambling casinos. Ulam later wrote that he thought the catchy name contributed to the spread of interest in the technique.

By the 1950's Monte Carlo calculations were being used by the Rand Corporation for a number of strategic studies. This required a source of random numbers, but computers were still in their infancy and not yet ready for that task. To fill that need the Rand Corporation developed an electronic system for producing random integers (digits $0, 1, \dots, 9$ actually) and used it in 1955 to publish a large collection of random numbers as the book, A MILLION RANDOM DIGITS WITH 100,000 NORMAL DEVIATES, which you can still find in many university libraries. Since then much has been written about computer algorithms to generate random numbers. It is an important topic, since bad random numbers will lead to bad Monte Carlo results. We will not delve into that, but will rely on the algorithms built into MATLAB's `rand` and `randn` commands to produce random numers for us.

11.2 Accuracy

The sample mean

$$\hat{m} = \frac{1}{N}\sum_1^N X_i$$

produces an approximation to the true value m that we seek. It will never be exact. Moreover $\hat{m}$ is itself a random variable. If we repeat the calculation with new values of X_i we will get a different result for $\hat{m}$. Thus it is important to understand some basic estimates related to the accuracy of the results.

Suppose for the moment that we know the true mean and variance of the X_i.

$$m = E[X_i], \quad \sigma^2 = \mathrm{Var}[X_i] = E[(X_i - m)^2].$$

We choose N and do our Monte Carlo experiment to obtain a value for $\hat{m} = \frac{1}{N}\sum_1^N X_i$. If we want our estimate $\hat{m}$ to be accurate to within a specified error,

$$|m - \hat{m}| < \epsilon,$$

what is the probability that we succeeded? To calculate this exactly would be a difficult calculation using the distribution of X_i, much harder than calculating the mean m itself. But we can use the Central Limit Theorem to get an approximate probability. The Central Limit Theorem says that

$$\frac{1}{\sigma\sqrt{N}}\sum_1^N (X_i - m) \Rightarrow Y,$$

where Y is a standard normal random variable. In particular this means that

$$P(-\delta < \frac{1}{\sigma\sqrt{N}}\sum_1^N (X_i - m) < \delta) \to \int_{-\delta}^{\delta} \frac{1}{\sqrt{2\pi}} e^{-y^2/2}\, dy = 2\mathcal{N}(\delta) - 1.$$

Now $-\delta < \frac{1}{\sigma\sqrt{N}}\sum_1^N (X_i - m) < \delta$ is equivalent to

$$|\hat{m} - m| < \frac{\delta\sigma}{\sqrt{N}}.$$

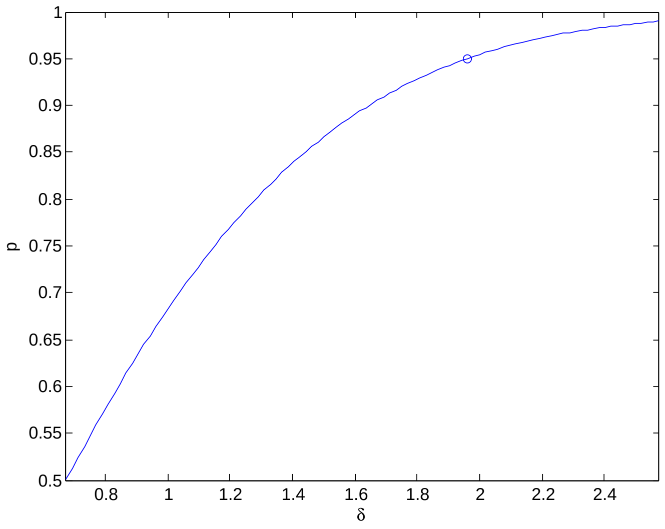
So we have the approximation (for large N)

$$P\left(|m-\hat{m}| < \frac{\delta\sigma}{\sqrt{N}}\right) \approx 2\mathcal{N}(\delta) - 1.$$

This is used in the following way. Pick a probability or "confidence level," say $p = .95$. That determines a corresponding value of delta, in our case $\delta \approx 1.96$ corresponds to $2\mathcal{N}(\delta) - 1 = .95$. We can then say with "confidence level of p" that $|m - \hat{m}| < \epsilon = \delta\sigma/\sqrt{N}$. I.e.

$$\hat{m} - \epsilon < m < \hat{m} + \epsilon.$$

The interval $[\hat{m} - \epsilon, \hat{m} + \epsilon]$, with $\epsilon = 1.96\sigma/\sqrt{N}$ is called the *95% confidence interval* for m. A higher probability requires a larger δ which means a bigger confidence interval (i.e. less accuracy). Probability of .99 uses $\delta = 2.5758$. Probability of .5 uses $\delta = .6745$. (These values are calculated from $\delta = \mathcal{N}^{-1}(\frac{1+p}{2})$ using MATLAB's `norminv` to evaluate $\mathcal{N}^{-1}$.) Here is a graph of $p = 2\mathcal{N}(\delta) - 1$, with the point corresponding to $p = .95$ marked.



The value of ϵ corresponding to $p = .5$ is sometimes called the *probable error*. This value, $\epsilon = .6745\sigma/\sqrt{N}$, is the error level for which $|m - \hat{m}| > \epsilon$ and $|m - \hat{m}| < \epsilon$ are equally likely. Thus it constitutes an indication of the size of the typical error.

Whatever confidence level p we select, we see that the accuracy $\epsilon = \delta\sigma/\sqrt{N}$ is proportional to $\sigma/\sqrt{N}$. To get one more digit of accuracy (with the same confidence level) we would need to increase N by a factor of 100. If $\sigma = 1$ for instance and we wanted to know m to within $\epsilon = 10^{-5}$ with 95% confidence, we would need to use about 38 billion samples! The upshot is that it is hard to get high accuracy from a Monte Carlo calculation.

11.3 Variance Reduction

There are various refinements that can be made that improve this accuracy somewhat. We will illustrate two of them below using a simple example.

11.3.1 An Example

If U is a uniformly distributed random variable on $[0,1]$, then

$$\pi = E[4\sqrt{1-U^2}] = \int_0^1 4\sqrt{1-u^2}\,du. \tag{11.1}$$

We will do a Monte Carlo calculation to estimate π from this formula. Suppose we use `rand` to generate a vector of $N = 1000$ independent realizations U_i. We form $X_i = 4\sqrt{1-U_i^2}$ and then compute the sample mean. The result of one such calculation was

$$3.1779 = \frac{1}{N}\sum_1^N X_i.$$

In order to use this to produce a confidence interval we need the variance of $X = 4\sqrt{1-U^2}$. This is easy to work out ($\sigma^2 = \frac{32}{3} - \pi^2 \approx .7971$) but we want to be able to do these calculations without needing to work out means and variances by hand. An effective technique is to use the sample variance

$$\hat{\sigma}^2 = \frac{1}{N-1}\sum_1^N (X_i - \hat{m})^2. \tag{11.2}$$

in place of the true variance σ^2. (The Strong Law of Large Numbers implies that $\hat{\sigma}^2 \to \sigma^2$ as $N \to \infty$.) Using $\hat{\sigma}$ we can produce approximate confidence intervals. The following simple code will analyze a list of random samples, returning the sample mean and printing a brief summary, including the 95% confidence interval, in the command window.

MCanl.m

```
function m = MCanl(data)
%MCanl(data) returns the sample mean and prints a simple
%summary of the data and 95% confidence error.
N=length(data);
m=mean(data);
v=var(data);
fprintf('\nSample variance=%g\n',v)
fprintf('Probable error=%g\n',.6745*sqrt(v/N))
fprintf('95%% interval=[%g, %g]\n',1.96*sqrt(v/N)*[-1,1]+m)
end
```

Using it for the X_i of our test calculation we found $\hat{\sigma} = .7746$ and so calculated the 95% confidence interval to be

$$[3.12333, 3.23244].$$

This gives us the first digit of π with reasonable certainty. To get the next digit we need to increase N by a factor of 100, to $N = 10^5$. Repeating the calculation with this N gives $\hat{m} = 3.1399$ with a 95% confidence interval of

$$[3.13435, 3.14545].$$

We got our second digit, but we can see that further accuracy improvements will be increasingly expensive. This is typical of Monte Carlo calculations. They will effectively produce a ball-park figure but do *not* easily produce high accuracy results.

11.3.2 Antithetic Variates

It is natural to simply increase N to improve the accuray of our Monte Carlo estimate. But the basic acccuracy relation $\epsilon = \delta\sigma/\sqrt{N}$ suggests a second approach: reduce σ. One way to do that is the use of *antithetic variates*. In our example above, observe that if U is unifromly distributed on $[0, 1]$, then so is $\tilde{U} = 1 - U$. If we formed $\tilde{X} = 4\sqrt{1 - \tilde{U}^2}$, then X and $\tilde{X}$ will have the same mean and variance. If we use both of them at the same time,

$$Y = \frac{1}{2}(X + \tilde{X}),$$

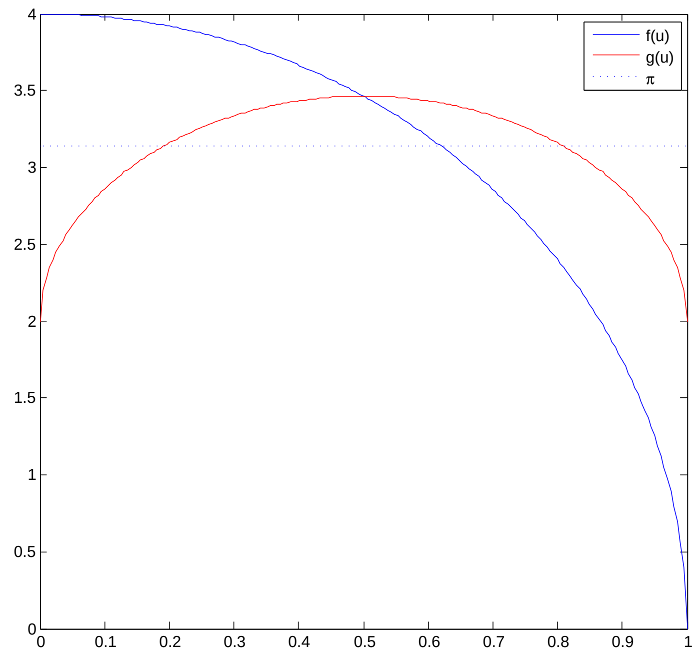
it turns out that while the mean is still the same, the variance is significantly less. If we write it this way

$$\begin{aligned} X &= f(U), \text{ where } f(u) = 4\sqrt{1-u^2} \\ Y &= g(U), \text{ where } g(u) = \frac{1}{2}(4\sqrt{1-u^2} + 4\sqrt{1-(1-u)^2}), \end{aligned}$$

we can see why. Since both means are π, the variances are

$$\begin{aligned} \mathrm{Var}[X] &= \int_0^1 (f(u) - \pi)^2 \, du \approx .7971 \\ \mathrm{Var}[Y] &= \int_0^1 (g(u) - \pi)^2 \, du \approx .1097. \end{aligned}$$

The variances measure how far the functions f and g are from the constant function with value π. If we look at the graphs of f and g we can see that g has a "flatter" graph than f; it stays closer to the mean overall. That accounts for its smaller variance.



A trial calculation with this antithetic approach, using $N = 1000$, produced $\hat{m} = 3.1444$ with a sample variance $\hat{\sigma} = 0.108762$ and a 95% confidence interval $[3.12392, 3.1648]$. This is a significant improvement over the original approach.

The features that make this effective are

1. the symmetry of the uniform distribution: $\tilde{U} = 1 - U$ has the same distribution as U itself, and

2. the monotonicity of $f(u) = 4\sqrt{1-u^2}$.

When we form $X = f(U)$ and $\tilde{X} = f(\tilde{U}) = f(1-U)$ *using the same copy of* U, X decreases as U increases while $\tilde{X}$ increases as U increases. So in $Y = \frac{1}{2}(X + \tilde{X})$ the increasing/decreasing effects counteract each other, at least partially, producing a random variable with a "flatter" dependence on U and therefore a smaller variance. We will see that these same features are present for some option pricing problems, which means that in those cases an antithetic variant approach will be useful.

11.3.3 Control Variates

Another technique to obtain improved accuracy from a Monte Carlo calculation of the mean of X is to find another random variable $\tilde{X}$ whose mean $m_{\tilde{X}} = E[\tilde{X}]$ we *do* know and such that $X \approx \tilde{X}$. We then use Monte Carlo to estimate

$$Z = X - (\tilde{X} - m_{\tilde{X}}). \tag{11.3}$$

The mean is still the same, $E[Z] = E[X] - E[\tilde{X}] + m_{\tilde{X}} = E[X]$. The variance

$$\mathrm{Var}[Z] = \mathrm{Var}[X - \tilde{X}]$$

will be small if $X \approx \tilde{X}$ is a good approximation. The approximating random variable $\tilde{X}$ is called a *control variate.* The success of this approach depends on finding a good choice for $\tilde{X}$.

In our example, we can easily calculate all the moments of U:

$$E[U^n] = \int_0^1 u^n\,du = \frac{1}{n+1}.$$

So if $\tilde{X}$ is a polynomial function of U we will be able to calculate its mean. One idea is to simply use $\tilde{X} = 4(1-U^2)$ as an approximation to $X = 4\sqrt{1-U^2}$. We have $m_{\tilde{X}} = 8/3$. Trying this (with $N = 1000$) we obtained $\hat{m} = 3.1650$, $\hat{\sigma} = .34967$ with a 95% confidence interval of $[3.12835, 3.20165]$. That's better than the basic approach (with $N = 1000$) but not as good as as the antithetic calculation.

To make the point that the use of a control variate can be effective, consider

$$\tilde{X} = \frac{4}{81\sqrt{3}}(142 - 19U + 16U^2 - 200U^3 + 224U^4 - 128U^5).$$

This is the result of using the fifth degree Taylor polynomial approximation to $f(u) = 4\sqrt{1-u^2}$ at $u_0 = \frac{1}{2}$. We are able to work out this approximation for our simple example, but we would not be able to do anything comparable in a more complex setting. However it will illustrate the power of the method when a good approximating control variate is available. Using the moments of U above we can work out that $m_{\tilde{X}} = \frac{742}{135\sqrt{3}}$. Applying Monte Carlo to Z as in (11.3), with $N = 1000$, we obtained $\hat{m} = 3.1398$ with a 95% confidence interval of $[3.13381, 3.14588]$. This is significantly better than the other estimates, but that is to be expected with such a "fine-tuned" control variate.

The following script will carry out all four of the calculations we have described for our example (11.1).

pimc.m

```
%Demonstration script for Monte Carlo Estimation of pi.
U=rand(1,N);
%
disp('Basic Method:')
X=4*sqrt(1-U.^2);
MCanl(X)
%
disp('Antithetic Method:')
Y=(X+4*sqrt(1-(1-U).^2))/2;
MCanl(Y)
```

```
%
disp('Simple Control Variate:')
Z=X-4*(1-U.^2)+8/3;
MCanl(Z)
%
disp('High Order Control Variate:')
Z5=X-4*(142-19*U+16*U.^2-200*U.^3+224*U.^4-128*U.^5)/(81*sqrt(3))+742/(135*sqrt(3));
MCanl(Z5)
```

Once a choice of $\tilde{X}$ has been made, a further refinement is to use $\theta\tilde{X}$ instead of $\tilde{X}$, where the parameter θ is chosen to minimize the variance $\mathrm{Var}[X-\theta\tilde{X}]$. The optimal choice of θ can be shown to be $\theta^* = \mathrm{Cov}[X,\tilde{X}]/\mathrm{Var}[\tilde{X}]$ and this can be estimated once the collection of random samples for X and $\tilde{X}$ have been generated, and then used in the final estimation of $E[Z] = E[X-\theta^*(\tilde{X}-m_{\tilde{X}})]$. We will not develop that refinement further. There are several other techniques for improving the accuracy of Monte Carlo calculations, which you can read about in the many references on the method.

11.4 A European Call

We now turn our attention to the use of Monte Carlo to estimate option prices, starting first with a European call. Of course we have no need of a Monte Carlo calculation here. For the case of a general terminal value function $\phi(s)$ our finite difference methods are quite effective, and for piecewise linear $\phi(s)$ we have explicit formulas. But developing a Monte Carlo calculation here is a simple first step toward the more complicated cases of path-dependent options.

$$v(s,0) = e^{-rT}E[\phi(S_T)\,|\,S_0=s].$$

We just need a sequence of random variables with the distribution of

$$S_T = se^{(r-\sigma^2/2)T+\sigma W_T}.$$

This is easy. MATLAB's `randn` will generate a vector of independent samples of standard normal random variables Y. Then $\sqrt{T}Y$ will have the distribution of W_T, which we can use in the above formula for S_T. So the random variable whose mean we want is

$$X = e^{-rT}\phi(se^{(r-\sigma^2/2)T+\sigma\sqrt{T}Y}).$$

When $\phi(s)$ is monotone (as it is for both calls and puts) we should get some benefit from an antithetic approach. Since $\tilde{Y}=-Y$ is also standard normal, we should average X above with

$$\tilde{X} = e^{-rT}\phi(se^{(r-\sigma^2/2)T-\sigma\sqrt{T}Y}).$$

The following script will execute both methods for a call.

EuroCall.m

```
%Script for basic and antithetic Monte Carlo evaluation of a European call option.
Y=randn(1,N);
ST=s*exp((r-sigma^2/2)*T+sigma*sqrt(T)*Y);
STa=s*exp((r-sigma^2/2)*T-sigma*sqrt(T)*Y);
VT=max(0,ST-K);
VTa=max(0,STa-K);
disp('Basic Method:')
MCanl(exp(-r*T)*VT)
disp('Antithetic Method:')
MCanl(exp(-r*T)*(VT+VTa)/2)
disp('Exact Value:')
BSCall(s,T,K)
```

Trying this out, we see that the antithetic version is generally better, especially for larger s values (where the monotonicity of $\phi(s) = (s-K)^+$ is stronger).

11.5 Path-Dependent Options

Finally let's consider path-dependent options. Since these are more difficult to address by finite differences, we might take a Monte Carlo approach more seriously here. The Monte Carlo calculations are also more complicated, but still quite manageable.

To be specific, let's look at an Asian option, starting from time $t = 0$ with $S_0 = s$. If $Y_T = \int_0^T S_u\,du$, we are interested in the contingent claim with terminal value $V_T = \phi(S_T, Y_T)$ for some function $\phi(s, y)$:

$$V_0 = e^{-rT} E[\phi(S_T, Y_T) \,|\, \mathcal{F}_t],$$

We will need to generate a sequence of random trials for (S_T, Y_T). For a single realization of this pair we will need a discretely sampled path for S_t:

$$\mathbf{S} = [S_0,\ S_{t_1},\ \dots,\ S_{t_M}].$$

Recall the formula $S_t = se^{(r-\sigma^2/2)t+\sigma W_t}$. We start with $\mathbf{Y} = [0,\ Y_1,\ \dots,\ Y_M]$ using M independent standard normal random variables Y_i. For $\Delta t = T/M$ we can build a discretely sampled Brownian path as

$$W_0 = 0,\ W_{t_1} = \sqrt{\Delta t}\,Y_1,\ W_{t_2} = \sqrt{\Delta t}\,(Y_1 + Y_2),\ W_{t_M} = \sqrt{\Delta t}\sum_1^m Y_i,\ \dots.$$

Matlab's `cumsum(Y)` command will compute the vector of cumulative sums the entires of a vector `Y`, so this is easy to do: `W=sqrt(dt)*cumsum(Y)`. This gives us a discretely sampled Brownian path $\mathbf{W} = [W_0,\ W_{t_1},\ \dots W_{t_M}]$. Let $\mathbf{T} = [0,\ t_1,\ \dots,\ t_M]$. We can now form a discretely sampled S_t path:

$$\mathbf{S} = se^{(r-\sigma^2/2)\mathbf{T}+\sigma\mathbf{W}},$$

the exponential being applied coordinate-by-coordinate. The last entry of $\mathbf{S}$ will be S_T. For Y_T we simply use

$$Y_T = \int_0^T S_t\,dt \approx \texttt{dt*trapz(S)}.$$

The preceding produces *just one* realization of the pair (S_T, Y_T). For a Monte Carlo calculation we need a large number N of independent realizations. You might think to just write a `for`-loop to repeat the above N times. But Matlab is designed so we can avoid that. The commands `cumsum` and `trapz` will accept a matrix argument, in which case they will operate on each column individually. So if we start with an $M \times N$ matrix of independent stand normal random variables, we can skip the `for`-loop and get all N copies of S_T and Y_T at once. The following script demonstrates this, using our known solution $v^{(6)}$ for demonstration purposes.

Asian.m

```
% Script for Monte Carlo calculation of Asian option.
% Values for r, sigma, T, s, N, and M must be prescribed.
dt=T/M;
W=sqrt(dt)*cumsum([zeros(1,N);randn(M,N)]);
S=s*exp((r-sigma^2/2)*linspace(0,T,M+1)'*ones(1,N)+sigma*W);
ST=S(M+1,:); YT=trapz(S)*dt;
%
VT=ST-r*YT; %v(6)(s,y,T) for demonstration
MCanl(exp(-r*T)*VT)
exp(-r*T)*s %known true value = v(6)(s,0,0)
```

Lookbacks would be quite similar, just with the maximum (or minimum) of S_t in place of the integral. For the average or floating price cases (both Asians and lookbacks) the terminal value is monotone in Y_T (or Z_T) so that an antithetic modification should produce some improvement. For Asians, the price for a geometric average makes a good choice of control variate (because an explicit pricing formula is available). American options are particularly hard to approach with a Monte Carlo calculation; it is not apparent how the optimization of the stopping rule can by included in a simulation-based approach. Even so, there have been serious efforts to price American options by Monte Carlo. The survey paper of P. Boyle, M. Broadie, and P. Glasserman [8] is a good place to start if you want to pursue the applications of Monte Carlo to option pricing further.

11.6 Problems

Problem 11.A
Based on the results of the Hall-Fox needle throwing experiment, find the 95% confidence interval for $p = \frac{2\ell}{a\pi}$. (Use the sample variance (11.2).) Based on that what would be a 95% confidence interval for π itself?
. . . . $\boxed{\texttt{Fox}}$

Problem 11.B
Modify the script `Asian.m` to estimate the price of the average strike Asian call. Use it to estimate the price at $t = 0$ if $S_0 = 10$ if the parameters are $r = .03$, $\sigma = .2$, $S_0 = 10$ and the option expires at $T = 2$. Run it more than once. (Turn in a listing of your script and the results you obtained.)
. . . . $\boxed{\texttt{MCAsian}}$

Problem 11.C
Modify `Asian.m` to estimate the price of the lookback option which pays 1 if $Z_T < K$ and 0 otherwise. Try it out with $r = .03$, $\sigma = .2$, $K = 20$, and $T = 2$, assuming that $S_0 = 10$. Run your script more than once. This is the same as the price of an certain barrier option. Based on that calculate the exact value of the price. (Turn in a listing of your script and the results you obtained.)
. . . . $\boxed{\texttt{MCBarrier}}$

Appendix: Matlab

There are many resources for those just getting started with Matlab. The Matlab software includes a number of video "demos". (The Getting Started with Matlab demo is a good one if you've never used Matlab before.) If you want a written introduction, Mathworks has a Getting Started Guide [39] which is available as a pdf file for free. (Chapter 2 covers what a beginner would need.) A particularly nice book is [24]. Extensive description of individual commands is available from "Product Help" under Matlab's help menu[1].

We will assume the reader is familiar with basic command syntax and the use of the command window. Our purpose here is to give brief descriptions of those features that are important for our use of Matlab in this book.

A .1 M-Files: Functions and Scripts

Entering commands one-by-one in the command window is fine for simple tasks, but for more complicated tasks you will want to write programs, which are called m-files for Matlab. These can be one of two types: scripts and functions. The files themselves are simply text files and can be edited with any text editor, or with the file editor that is part of Matlab.

A .1.1 Scripts

A script is nothing more than a predefined sequence of commands, typed just as you would enter at the command line. The commands are simply read from an m-file and executed when you type the name of the m-file at the command line. The only special feature is the provision for including comments. Matlab treats anything that follows the percent sign % as a comment which will be ignored when the commands are executed. This is how we include documentation in an m-file. The initial group of comment lines at the top of the file play a special role; they are what Matlab will display if you ask for help on the m-file. Putting a concise description of the m-file's purpose here can be very helpful later if you need a quick reminder about how to use it. We saw one example of a script on page 36. Here is another.

rune.m

```
%Script for test calculations using efd to calculate call values.
%Initial and boundary values provided by u0(*), uL(*) and uh(*).
%xL and xH are converted form sL and sH.
[junk,x]=hf(0,[sL,sH],0,0); % Convert [sL,sH] to xL, xH.
xL=x(1);
xH=x(2);
% Run explicit method
efd
% Compute true values: BSCall and convert to heat variables
[junk,s]=fh(0,x,0,0);
true=hf(BSCall(s,T,K),s,0,T);
```

[1]Or online at `http://www.mathworks.com/help/techdoc/index.html`.

```
% Display results
plot(x,true,'r')
hold on
plot(x,u,'*')
% Print summary
title('Explict Finite Difference Results (in heat variables)')
hold off
fprintf('\n[xL,xH]=[%g , %g]\n',xL,xH)
fprintf('dx=%g, using N+1=%g x-intervals.\n',dx,N+1)
fprintf('T=%g, dt=%g, using M=%g time steps.\n',T,dt,M)
fprintf('alpha=%g\n',alpha)
fprintf('Minimal M for stability=%g\n',ceil(2*T/(dx^2)))
fprintf('Maximum error=%g\n\r',max(abs(u-true)))
```

The initial block of comment lines are what will be displayed if you type `help rune` at the command line. This particular script is what we used to carry out the test calculations for the explicit method in Section 4.3. Before running this script we make sure (from the command line) that the variables `sigma`, `r`, and `K` are defefined as global and that all the required variables have been given the desired values: `T, N, M, sL, sH`. We run the script by just entering `efd` at the command prompt. The script will produce values for `xL, xH` before invoking `efd`. It obtains the true solution from `BSCsll` and conversion to heat variables by `hf`. The results are plotted and some summary information is printed in the command window. When it is finished, `u` will contain the results and all the output will have been displayed on the screen.

It is important to note the semicolon after each of the executable statements. This keeps MATLAB from printing the result of that line to your screen each time it is executed. Leaving them out can be helpful for debugging, but you really do want them for your finished version. Note that a script can call another script; here `rune` calls `efd`.

A .1.2 Functions

A function m-file defines a new named function, which takes prescribed inputs and returns an output determined by the calculations contained in the m-file. Once defined you can use the new named function just as you would any other MATLAB function. As an example, here is an m-file to define the function `BSCall` which evaluates the standard Black-Scholes formula (3.2).

BSCall.m

```
function c=BSCall(s,tau,K)
%BSCall(s,tau,K) evaluates the standard Black-Scholes
%formula. s (which may be a vector) is the stock price(s), tau is
%the time to expiry, K is the exercise price.  The volatility sigma
%and the risk-free rate r are supplied as global variables.
global sigma r
d1=(log(s/K)+(r+sigma^2/2)*tau)/(sigma*sqrt(tau));
d2=d1-sigma*sqrt(tau);
c=(s.*erfc(-d1/sqrt(2))-K*exp(-r*tau)*erfc(-d2/sqrt(2)))/2;
%Note that erfc(-x/sqrt(2))/2 gives the standard normal distribution
%function.
```

The first line of the file identifies this m-file as the defintion of a function `BSCall`, and specifies the variable names which will be used *inside the m-file* to refer to the input arguments (`s, tau, K`) and output value (`c`). The Black-Scholes forumla also requires values for `sigma` and `r`. Since these parameters will always be the same throughout any calculation we have chosen to make these global variables (see Section A .3). That makes them available to all m-files without having to pass them back and forth as arguments all the time.

The initial group of of comment lines (`%...`) following the `function` line at the top of the file are what MATLAB will display if you type `help BSCall` at the command line.

The executable part of `BSCall` is only three lines long, and should be easy to recognize as implementing the formula from (3.2). As the comment lines explain, this code allows the input argument `s` to be a vector of values, in which case the output `c` will be the corresponding vector of values of the Balck-Scholes formula. In the notation of ..., if `s` $= [s_1, \ldots, s_n]$ then the output of `BSCall` with `tau` $= T - t$ will be the vector of values

$$[v^{\text{Call,K}}(s_1, T-t), \ldots, v^{\text{Call,K}}(s_n, T-t)].$$

In other words you get a whole vector of values of $v^{\text{Call,K}}$ with a single call to `BSCall` – you don't need to get them one at a time with a `for`-loop. This is ability to "vectorize" caluculations is one of MATLAB's great strengths, and should be taken advantage whenever possible!

A typical use of `BSCall` is in the script `efd` above. Note that as invoked in `efd` the middle argument was `T` while in `BSCall.m` the middle variable was called `tau`. You can put whatever you want in the argument positions when you invoke a funtion. Those values will be placed into the argument variables (`s,tau,K`) for use inside the m-file. In general the variables used inside the m-file are not acessible from the command line or other m-files. If at the command line you have defined variables such as `s,K,d1,` ... with the same names as those in the m-file, the command line versions and m-file versions are completely separate. You might think of the m-file as a separate little world whose only connection to the outside world is what is passed in as arguments and what it sent out as the returned value. (Global variables are the one exception to this; we will talk more about that below.) Any variables in an m-file, like `d1,d2,c` above, must be provided with values by the sequence of commands in the m-file. They are just empty when the m-file is evoked, and cleared out when the m-file is done. (Permanent variables are an exception to this; see below.)

A .2 Some General Advice

We collect here some general advice for programming in MATLAB.

- Wherever possible use vector operations instead of `for`-loops. Scalar operations are vectorized by using the "dot" form of the command. For instance if we have two vectors `x`$= [x(i)]$ and `y`$= y(i)]$, and we want to compute `z`$= [z(i)] = [x(i) \cdot y(i)]$, the way to do it is `z=x.*y`, not with a `for`-loop. There will be times when there is no alternative to a `for`-loop, but avoid them when you can.
- Include explanatory comments in your programs; it can make modifying them later much easier.
- Calculations that you will use repeatedly in several places are best written as separate m-files. Breaking up a project into smaller tasks will help organize your project.
- When you start developing a program, first organize what you want it to do on paper. Decide on variable names, how you might want to make repeated tasks into separate m-files.
- As you write your m-files, especially if you are using commands you are not too familiar with, it is good to experiment and test each line of code from the command line and with simple examples, to be sure it behaves the way you think it does. Don't try to write the whole thing at once and expect it to work properly.

You need to make sure MATLAB can find the m-files you have written. It will always look in your current working directory. The command `pwd` will tell you what the working directory is. You can change it by navigating to wherever you want using the directory window on the left side of the standard MATLAB display. (Or using the `cd` command, which works the same way as the UNIX command of the same name.) So if you create a directory for each project you are working on, you can put the m-files you create for that project in that folder. However some m-files you will want to use on more than one project. For instance we will want to use `hf.m`, `fh.m`, `BSCall.m` for several different projects. So instead of putting new copies of them in each project folder, you can put them in a separate reference directory and then add that directory to the *search path*, which is the list of directories that MATLAB searches when looking for m-files. (You can do this from Set Path under the File menu, or from the command line using the `addpath`. See MATLAB's documentation for more details.)

A .3 Global and Persistent Variables.

The variables defined inside a function m-file are not accessible from outside. So if the calculation the m-file performs requires some parameters, then those parameters would need to be included in the variables used to call the function. However there is a way around this using *global* variables. For instance, the parameters r and σ occur in many of our calculations so we might find it convenient to avoid having to always include them in the list of arguments for every function. If we declare these to be global variables then they will be acessible from the command line and any m-file (which also declares them to be global). To make these global variables, at the command line enter the following command.

```
global sigma r
```

Then include a similar `global` statement at the top of any m-file you want to have access to these variables. We saw an example of the in `BSCall.m` above. As another example, the following function gives the terminal value of a call option, which depends on the parameter K.

vT.m

```
function v=vT(s)
%v0(s) evaluates the terminal conditions, in financial variables,
%for the Black-Scholes equation.
global K
v=max(s-K,0); %for a call
end
```

This function is going to be called by one of our finite difference methods, but the code for that doesn't need to even know about the existence of K — only `vT` needs to know about that. The `global K` statement gives the function access to the same value of `K` as the command line, so if you give the declaration `global K` at the command line and give it a value (before `vT` is called) then `vT` will use that value when it is called. Note that the `global K` statement needs to be issued at the command line as well as being in the text of any `m`-file which needs to use it.

Any variables that are defined inside a function m-file (like `d1` and `d2` in `BSCall` above) will be cleared from memory at the end of each call to the function. In some cases you may want to preserve the values of internally defined variables from one function call to the next. A good example is `PSOR.m` in Chapter 8 — there were several variables `N, e, d, DF, tol, w, dw, count` whose values we wanted to preserve from one call to the function to the next. We accomplished this by declaring those variables to be *persistent*, and writing the code to avoid re-creating them on subsequent calls.

Appendix: Itô's Formula and Martingales

This appendix briefly summarizes facts about Brownian motion, Itô's formula, martingales. For more on these topics readers might consult [36], [45], [50] or other more advanced texts on stochastic processes.

Brownian motion, W_t, is a stochastic process with these properties.

1. $W_0 = 0$.
2. For any $0 \leq s < t$, $W_t - W_s$ is a normal random variable with mean 0 and variance $t - s$.
3. For any increasing selection of times, $0 = t_0 < t_1 < \cdots < t_n$, the increments $\Delta W_{t_i} = W_{t_i} - W_{t_{-1}}$ are independent of each other.
4. As a function of $t \geq 0$, W_t is continuous.

We generally denote by $\mathcal{F}_t$ the σ-algebra generated by W_u for $0 \leq u \leq t$. For $t < u$ the third bullet above says that $W_u - W_t$ is independent of $\mathcal{F}_t$. We can take advantage of that to compute many of the conditional expectations encountered in these notes, using a general *separation formula* for conditional expectations. Suppose we want

$$E[X \mid \mathcal{F}_t].$$

If we can write X as a function of some other random variables,

$$X = g(Y, Z),$$

where Y is measurable with respect to (i.e. dependent on) $\mathcal{F}_t$ and Z is independent of $\mathcal{F}_t$, then the conditional expectation is given by

$$E[g(Y, Z) \mid \mathcal{F}_t] = h(Y),$$

where

$$h(y) = E[g(y, Z)].$$

In words, we average out the Z-dependence of $g(y, Z)$ to get a function h of y alone, and then plug the random variable Y into that. (A special case is $g(y, z) = yz$ for which the above leads to $E[YZ \mid \mathcal{F}_t] = Y E[Z]$.) This is used several places in the notes above: Sections 2.6.2, 9.1, and 10.3. (The calculation on page 10 could also have been expressed this way.)

The notion of stochastic integrals with respect to W_t and other stochastic processes allows the development of a stochastic calculus, with which calculations with stochastic processes can be carried out in much the same way that "freshman calculus" is used to work with conventional deterministic functions. Itoô's formula provides a sort of stochastic chain rule, which is the key to many calculations using stochastic calculus. This is usually expressed in terms of stochastic differentials. We simply state Itô's formula as formal rule. Suppose $f(x_1, \ldots, x_n)$ is a C^2 function and $X^{(1)}(t), \ldots, X_t^{(n)}$ are stochastic processes with stochastic differentials $dX_t^{(i)}$. Then $f(\mathbf{X}_t) = f(X^{(1)}(t), \ldots, X_t^{(n)})$ has a stochastic differential given by

$$df(\mathbf{X}_t) = \sum_{i=1}^n f_{x_i}(\mathbf{X}_t)\, dX_t^{(i)} + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n f_{x_i, x_j}(\mathbf{X}_t)\, dX_t^{(i)} dX_t^{(j)}.$$

When the $dX_t^{(i)}$ can be expressed in terms of dW_t and dt, the products $dX_t^{(i)} dX_t^{(j)}$ are determined from simple product rules such as $dt\, dW_t = dt\, dt = 0$, $dW_t\, dW_t = dt$. In general $dX_t^{(i)} dX_t^{(j)}$ is the differential of the quadratic covariation process. For an accurate understanding, consult an advanced stochastic processes text such as references such as [27] or [50].

Bibliography

[1] F. AitSahlia and P. Carr, *American options: a comparison of numerical methods*, in [49], pp. 67–87.

[2] B. Alziary, J.-P. Decamps, and P.-F. Koehl, A P.D.E. approach to Asian options: analytical and numerical evidence, Journal of Banking and Finance, v. 21 (1997), pp. 613-640.

[3] S. Asmusssen and P. W. Glyn, Stochastic Simulation and Analysis, Springer, NY, 2007.

[4] A. Bensousan, *On the theory of options pricing*, Acta. Appl. Math. **2** (1984), pp. 139–158.

[5] T. Björk, Arbitrage Theory in Continuous Time, Oxford Univ. Press, Oxford, 1998. *This is the text for Math 5725.*

[6] F. Black and M. Scholes, *The pricing of options and corporate liabilities*, J. Political Economy v. 81 (1973), pp. 637–654.

[7] J. F. Botha and G. F. Pinder, Fundamental Concepts in the Numerical Solution of Differential Equations, Wiley, New York , 1983.

[8] P. Boyle, M. Broadie, and P. Glasserman, *Monte Carlo methods for security pricing*, J. Economic Dynamics and Control 21 (1997), pp. 1267–1321.

[9] P. Brandimarte, Numerical Methods in Fianace: A MATLAB-based Introduction, Wiley, 2002. *Discusses and illustrates a wide range of numerical techniques in finance in the specific context of Matlab.*

[10] M. Brennan and E. Schwartz, *The valuation of American put options*, J. Finance **32** (1977), pp. 449–462.

[11] M. Broadie and J. Detemple, *Recent advances in numerical methods for pricing derivative securities*, in [49], pp. 43–66.

[12] P. Carr, R. Jarrow, and R. Myneni, *Alternaitve characterization of American put options*, Math. Finance v. 2 (1992), pp. 87–106.

[13] A. Conze and Viswanathan, *Path dependent options: the case of lookback options*, J. Finance v. 41 (1991), pp. 1893–1907.

[14] R. W. Cottle, J. S. Pang, and R. E. Stone, The Linear Complementarity Problem, Academic Press, Boston, 1992.

[15] J.-C.Duan, G. Gauthier and J.-G. Simonato, *A Markov chain method for pricing contingent claims*, in Stochastic Modeling and Optimization with Applications in Queues, Finance and Supply Chains (Yao, Shang, Zhou editors), Springer, 2003.

[16] J.-C.Duan, G. Gauthier, and J.-G. Simonato, *American option pricing under GARCH by a Markov chain approximation*, Journal of Economic Dynamics and Control Volume 25, Issue 11, November 2001, pp. 1689–1718.

[17] C. M. Elliott and J. R. Ockendon, Weak and Variational Methods for Moving Boundary Problems, Pitman, Boston, 1982.

[18] E. Fournié, J. M. Lasry, and N. Touzi, *Monte Carlo methods for sotchastic volatility models*, in [49], pp. 146–164.

[19] A. Friedman, *Parabolic variational inequalities in one space dimension and smoothness of the free boundary*, J. Funct. Anal. **18** (1975), pp. 151–176.

[20] G. Fusari and A. Roncoroni, Implementing Models in Quantative Finance: Methods and Cases, Springer, Berlin, 2008.

[21] Geman and Yor, *Pricing and hedging double barrier options: a probabilistic approach*, Math. Finance **6** (1996). pp. 365–378.

[22] D. J. Higham, An Introduction to Financial Option Valuation, Cambridge Univ. Press, 2004.

[23] D. J. Higham, Nine ways to implement the binomial method for option valuation in Matlab, SIAM **44** (2002), pp. 661–677.

[24] D. J. Higham and N. J. Higham, Matlab Guide, SIAM, Philadelphia, 2005.

[25] J. C. Hull, Options, Futures, & Other Derivatives (4th edition), Prentice-Hall, NJ, 1989.

[26] J. Huang and J.-S. Pang, *Option pricing and linear complementarity*, J. Comp. Finance, **2** (1998), pp. 31–60.

[27] N. Ikeda and S. Watanabe, Stochastic Differential Equations and Diffusion Processes, North-Holland, Amsterdam, 1989.

[28] K. Itô and H. P. McKean, Diffusion Processes and their Sample Paths, Springer-Verlag, Berlin, 1974.

[29] S. D. Jacka, *Optimal stopping and the American put*, Math. Finance **1** (1991), pp. 1–14.

[30] P. Jaillet, D. Lamberton and B. Lapeyre, *variational inequalities and the pricing of American options*, Acta Appl. Math. **21** (1990), pp. 263–289.

[31] L. Jiang, Mathematical Modeling and Methods of Option Pricing, World Scientific, Singapore, 2005.

[32] F. John, Partial Differential Equations (3rd edition), Springer-Verlag, New York, 1978.

[33] H. E. Johnson, *An analytic approximation for the American put price*, J. Fin. Quant. Anal. **18** (1983), pp. 141–148.

[34] I. Karatzas, *On the pricing of American options*, Appl. Math. Opt. **17** (1988), pp. 37–60.

[35] I. Karatzas, *Optimization problems in the theory of continuous trading*, SIAM J. Cont. Opt. **27** (1989), pp. 1221-1259.

[36] I. Karatzas and S. Shreve, Methods of Mathematical Finance, Springer, New York, 1998.

[37] D. Kinderlehrer and G. Stampacchia, An Introduction to Variational Inequalities and their Applications, Academic Press, New York, 1980.

[38] H. J. Kushner and P. Dupuis, Numerical Methods for Stochastic Control Problems in Continuous Time, Springer, New York, 2001.

[39] Matlab Getting Strated Guide, The Mathworks, Natic, MA, 2009. Available at

`http://www.mathworks.com/help/techdoc/matlab_product_page2.html`

[40] H. P. McKean, *A free boundary problem for the heat equation arising from a problem in mathematical economics*, Industrial Management Review **6** (1965), pp. 32–39.

[41] R. Menyi, *Pricing of the American option*, Ann. Prob. **2** (1992), pp. 1-23.

[42] R. C. Merton, *Theory of rational option pricing*, Bell J. Econ. Management Sci. **4** (1973), pp. 141–183.

[43] K. G. Murty and F.-T. Yu, Linear Complementarity, Linear and Nonlinear Programming, Internet Edition:

`http://ioe.engin.umich.edu/people/fac/books/murty/linear_complementarity_webbook/`

[44] P. van Moerbeke, *On optimal stopping and free boundary problems*, Arch. Rat. Mech. Anal. **60** (1976), pp. 101–148.

[45] M. Musiela and M. Rutkowski, Martingale Methods in Financial Modeling, Springer, Berlin, 1998.

[46] M. Parkinson, *Option pricing: the American put*, J. Business **50** (1997), pp.21–36.

[47] R. D. Richtmyer and K. W. Morton, Difference Methods for Initial Value Problems (2nd edition), Wiley, New York, 1967.

[48] L. C. G. Rogers and Z. Shi, *The value of an Asian option*, J. Appl. Prob. **32** (199), pp. 1077-1088.

[49] L. C. G. Rogers and D. Talay (ed.s), Numerical Methods in Finance, Cambridge Univ. Press, 1997.

[50] L. C. G. Rogers and D. Williams, Diffusions, Markov Processes and Martingales (2 vol.s), Cambridge Univ. Press, 2000.

[51] D. M. Salopek, American Put Options, Addison Wesley Longman, Essex, 1997.

[52] R. Seydel, Tools for Computational Finance, Springer-Verlag, Berlin, 2002.

[53] W. Shaw, Modelling Financial Derivatives with Mathematica, Cambridge Univ. Press, 1999.

[54] G. D. Smith, Numerical Solution of Partial Differential Equations: Finite Difference Methods, Clarendon Press, Oxford, 1985.

[55] J. Stoer and R. Bulirsch, Introduction to Numerical Analysis (2nd ed.), Springer-Verlag, New York, 1993.

[56] D. Tavella, Quantitative Methods in Derivatives Pricing: an Introduction to Computational Fiance, J. Wiley, N.Y., 2002.

[57] D. Tavella and C. Randall, Pricing Financial Instruments: The Finite Difference Method, J. Wiley & Sons, New York, 2000.

[58] R. J. Vanderbi and M. Ç. Pinar, Pricing American Perpetual Warrants by Linear Programming, SIAM Rev. v. 51 (no.4) 2009, pp. 767–782.

[59] R. S. Varga, Matrix Iterative Analysis, Springer-Verlag, Berlin, 2009.

[60] D. V. Widder, The Heat Equation, Academic Press, New York, 1975.

[61] P. Wilmott, Derivatives: The Theory and Practice of Financial Engineering, Wiley, Chichester, 1998.

[62] P. Wilmott, S. Howison, and J. Dewynne, The Mathematics of Financial Derivatives: A Student Introduction, Cambridge Univ. Press, 1995.

[63] Y,-I. Zhu, X. Wu, and I.-L. Chang, Derivative Securities and Difference Methods, Springer, NY, 2004.