Using the NAG Library to calculate financial option prices in Excel

Marcin Krzysztofik and Jeremy Walton*
The Numerical Algorithms Group Ltd.

1 Introduction

This note discusses the use of algorithms from the NAG Library [18] to calculate prices for financial options. More specifically, we focus on the way in which NAG routines can be called from within a Microsoft Excel spreadsheet, and illustrate this with some examples.

Our paper is arranged as follows. The following section presents an elementary introduction to options (§ 2.1), and the variety of models that have been developed for their pricing (§ 2.2). The sensitivity of the price to various dependent parameters is also of interest; this information is contained in the partial derivatives of the price, which are known as the *Greeks* (see § 3). Section 4 presents an example of one of the option pricing models, along with its closed-form expressions for the price and the Greeks. The calculation of the results from this model (and for many others) can be performed using NAG Library routines (§ 5); we discuss this in some detail for the first of our two examples which calls a NAG routine from Excel (§ 5.1), before presenting a more complicated example (§ 5.2) to illustrate further possibilities in the determination and analysis of the behaviour of option prices—for example, the visualization of the results within Excel. Finally, we describe (§ 5.3) how the interested reader can access the examples that we have discussed, and conclude (§ 6) with some ideas for further work.

2 Options and option pricing

2.1 An introduction to options

An *option* is a contract between two parties (called the *holder* and *writer*) that gives the holder the right—but not the obligation—to buy or sell an *asset* at an agreed price (called the *strike price*) at the end of, or during, a specified period. It is a financial instrument which is an example of a *derivative*; that is, its value is based upon something else—in this case the

*infodesk@nag.co.uk

asset (which is said to *underlie* the option)[1]. The strike price is specified when the holder and seller enter into the option; the point marking the end of the specified period (when the option is said to *expire*) is called its *maturity date*. An option allowing the holder to buy the asset is a *call option*, while a *put option* allows them to sell the asset.

If the holder chooses to exercise the option, the writer is obliged to sell or buy the asset at the strike price; otherwise, the option is allowed to expire. The *type* of option determines when, how and under what circumstances the holder can exercise [11]. For example, *European*-style options may only be exercised on the maturity date, while *American*-style options can be exercised at any point before expiration. The holder decides whether or not to exercise the option based on its *payoff*, which is related to the difference between the strike price and the current value (sometimes called the *spot price*) of the asset.

European and American options are often described as *vanilla*, owing to their comparatively simple payoff model (see § 4, below). There also exist a variety of so-called *exotic* options, which involve more complicated calculations—for example, *Asian*-style options have a payoff which depends on the average price of the underlying asset over some pre-set period of time, and *Barrier*-style options cannot be exercised until the price of the underlying asset passes a specified value (or barrier).

2.2 Pricing options

Options are widely traded on financial markets, and so some method of determining the value (or price) of a given option is required. However, this is non-trivial in general, partly because it depends on several variables in addition to the value of the underlying asset—including the strike price and time to expiry, as well as the risk-free interest rate and the asset's volatility. In addition, the determination of the option price requires the construction of a model for the way in which the asset price changes over time. This process is essentially a random walk, and several quantitative techniques—usually based on stochastic calculus—have been developed for constructing these models, which are then *implemented* using a range of mathematical methods—including analytical solution, finite difference and finite element methods and Monte Carlo simulation. This note only considers the first of these methods, but we also briefly mention further work which uses some of the other implementation methods in Section 6.

[1]The underlying asset can be a piece of property, another derivative—such as a futures contract—or a stock (which is what is sometimes assumed in the notation for this article).

One of the most popular models is that of Black and Scholes [3], which leads to a partial differential equation (PDE) that can be solved analytically to give closed-form expressions for the value of a European put and call option—see § 4, below—but this is only one of a variety of models [9] for option pricing, each of which make different assumptions about the behaviour of the asset price. Many of these models have analytic solutions, which may be evaluated using NAG Library routines; we discuss this in detail in § 5, below.

3 The Greeks

Besides seeking some estimate of the behaviour of the option price itself, traders are interested in its sensitivity to changes in its dependent parameters (such as, for example, the asset price). These sensitivities can be formally represented as partial derivatives, and—because they are often denoted by Greek letters—are usually referred to as *the Greeks*.

Greeks are used by traders to help quantify risk. For example, *Delta*—see equation (1), below—is employed in so-called *delta hedging*. This is a risk-reducing strategy which involves having a short position[2] on the underlying asset and offsetting it with a long position[3] on an option, or vice versa (buying an asset and shorting an option).

3.1 Delta

Delta (Δ) is the sensitivity of the option price P to a small change in the underlying asset price, S.

$$\Delta = \frac{\partial P}{\partial S} \tag{1}$$

3.2 Vega

Vega (ν) measures the rate of change of the option price with respect to the underlying asset volatility, σ.

$$\nu = \frac{\partial P}{\partial \sigma} \tag{2}$$

[2] A trader who borrows an asset from a third party and sells it, expecting to be able to buy it back at a later date to return to the lender is said to be opening a short position. The expectation is that the price of the asset will fall between its sale and purchase, resulting in a profit for the trader.

[3] A trader who buys an asset in order to sell it at a later date is said to be opening a long position. The expectation is that the price of the asset will increase between its purchase and sale, resulting in a profit for the trader.

3.3 Theta

Theta (Θ) is the sensitivity of the option price to a change in the time to expiry of the option, T. Because the time to expiry is always decreasing (unlike the other dependent parameters which can increase or decrease), it is conventional to take the negative of the partial derivative.

$$\Theta = -\frac{\partial P}{\partial T} \tag{3}$$

3.4 Rho

Rho (ρ) is the rate of change of the option price with respect to the *annual risk-free interest rate*, r.

$$\rho = \frac{\partial P}{\partial r} \tag{4}$$

3.5 Crho

Carryrho (*Crho*) is the sensitivity of the option price to a change in the so-called *annual cost of carry rate*, b—that is, the cost of carrying, or holding, a position. When the asset is a stock, this is defined as the difference between the risk-free interest rate and q, the stock's dividend yield; thus $b = r - q$.

$$Crho = \frac{\partial P}{\partial b} \tag{5}$$

3.6 Gamma

Gamma (Γ) is the sensitivity of *Delta* to a small change in the price of the underlying asset.

$$\Gamma = \frac{\partial \Delta}{\partial S} = \frac{\partial^2 P}{\partial S^2} \tag{6}$$

3.7 Vanna

Vanna shows how sensitive *Delta* is to a small change in the underlying asset volatility (or, equivalently, how much *Vega* will change for a small change in the asset price).

$$Vanna = \frac{\partial \Delta}{\partial \sigma} = \frac{\partial \nu}{\partial S} = \frac{\partial^2 P}{\partial S \partial \sigma} \tag{7}$$

3.8 Charm

Charm is *Delta*'s sensitivity to changes in time (or, equivalently, *Theta*'s sensitivity to small changes in the asset price).

$$Charm = -\frac{\partial \Delta}{\partial T} = \frac{\partial \Theta}{\partial S} = -\frac{\partial^2 P}{\partial S \partial T} \tag{8}$$

3.9 Speed

Speed is the sensitivity of *Gamma* to a change in the price of the underlying asset.

$$Speed = \frac{\partial \Gamma}{\partial S} = \frac{\partial^3 P}{\partial S^3} \tag{9}$$

3.10 Colour

Colour is the sensitivity of *Gamma* to a change in time (or, equivalently, the sensitivity of *Charm* to a change in the asset price).

$$Colour = -\frac{\partial \Gamma}{\partial T} = \frac{\partial Charm}{\partial S} = -\frac{\partial^3 P}{\partial S^2 \partial T} \tag{10}$$

3.11 Zomma

Zomma is the sensitivity of *Gamma* to changes in the volatility of the underlying asset (or, equivalently, *Vanna*'s sensitivity to a change in asset price).

$$Zomma = \frac{\partial \Gamma}{\partial \sigma} = \frac{\partial Vanna}{\partial S} = \frac{\partial^3 P}{\partial S^2 \partial \sigma} \tag{11}$$

3.12 Vomma

Vomma is the sensitivity of *Vega* to change in the volatility of the underlying asset.

$$Vomma = \frac{\partial \nu}{\partial \sigma} = \frac{\partial^2 P}{\partial \sigma^2} \tag{12}$$

4 The Black-Scholes model

As noted above, the Black-Scholes model [3] is a widely-used mathematical representation of the behaviour of an asset. Its application results in the so-called Black-Scholes partial differential equation for the option price, which

is assumed to depend only on asset price and time, t:

$$\frac{\partial P}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + (r-q)S\frac{\partial P}{\partial S} - rP = 0. \tag{13}$$

To solve this for P, boundary conditions must first be specified—for example,

$$\begin{aligned} P(0,t) &= 0, \forall t & (14)\\ \lim_{S\to\infty} P(S,t) &= S & (15)\\ P(S,T) &= \max(S-X,0) & (16) \end{aligned}$$

which correspond to a European call option (boundary conditions corresponding to a European put option can be specified in a similar fashion). Note that equation (16), in which X is the strike price, is the formal definition of the payoff for a call option[4].

Having specified the boundary conditions[5], the PDE can be transformed into a diffusion equation and then solved using standard methods, resulting in closed-form expressions for the prices of a European call and put option:

$$\begin{aligned} P_c &= Se^{-qT}\Phi\left(d_1\right) - Xe^{-rT}\Phi\left(d_2\right), & (17)\\ P_p &= Xe^{-rT}\Phi\left(-d_2\right) - Se^{-qT}\Phi\left(-d_1\right). & (18) \end{aligned}$$

Here,

$$d_1 = \frac{\ln\left(S/X\right) + \left(r - q + \sigma^2/2\right)T}{\sigma\sqrt{T}}, \tag{19}$$

$$d_2 = \frac{\ln\left(S/X\right) + \left(r - q - \sigma^2/2\right)T}{\sigma\sqrt{T}}, \tag{20}$$

and

$$\Phi\left(x\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-z^2/2}\,dz \tag{21}$$

is the cumulative normal distribution function. Figure 1 shows $P_c(T)$ as calculated from equation (17) for specific values of the dependent parameters. It shows how, according to this model of a European call, the price of the option decreases with the time to expiry—i.e., the smaller the interval between entering into the option and the point at which it can be exercised, the less value the option has.

[4]The payoff for a put option is $P(S,T) = \max(X-S,0)$.

[5]In addition, we make certain assumptions about the other parameters in equation (13)—for example, that σ, r and q are independent of time—which are required for an analytic solution. We return to this point below, in § 6.

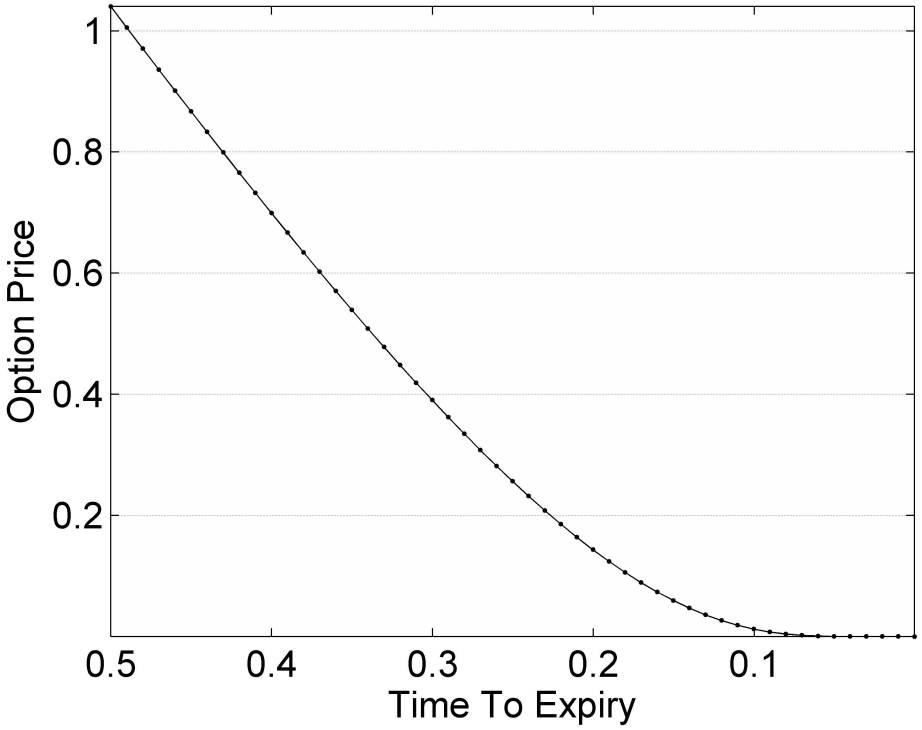

Figure 1: Option price as a function of expiry time for a call option, as given by the Black-Scholes formula [equation (17)] with $S = 55$, $\sigma = 30\%$, $r = 5\%$, $q = 0$ and $X = 70$.

We can also use equation (17) to calculate $P_c(S)$ (at fixed T), which is displayed in Figure 2, along with the option payoff[6], given by equation (16). Figure 2 shows that—again, for this model of a European call—the price of the option increases from zero as the strike price is approached (recall that S is the price of the underlying asset at the time the option is entered into), approaching the payoff asymptotically as S is increased. Looking at figures 1 and 2 together, it can be seen that, as T is decreased towards zero (i.e. the maturity date is approached), the $P_c(S)$ curve moves closer to the option payoff, as is to be expected.

Given a closed-form solution for the option price, explicit formulae for the Greeks may be readily found. For example, those corresponding to the Black-Scholes model (obtained by substituting equations (17) and (18) into

[6]The payoff is also sometimes known as the option *intrinsic value*, and the difference between the price and the intrinsic value—that is, the distance between the two curves in Figure 2—is the option *time value*.

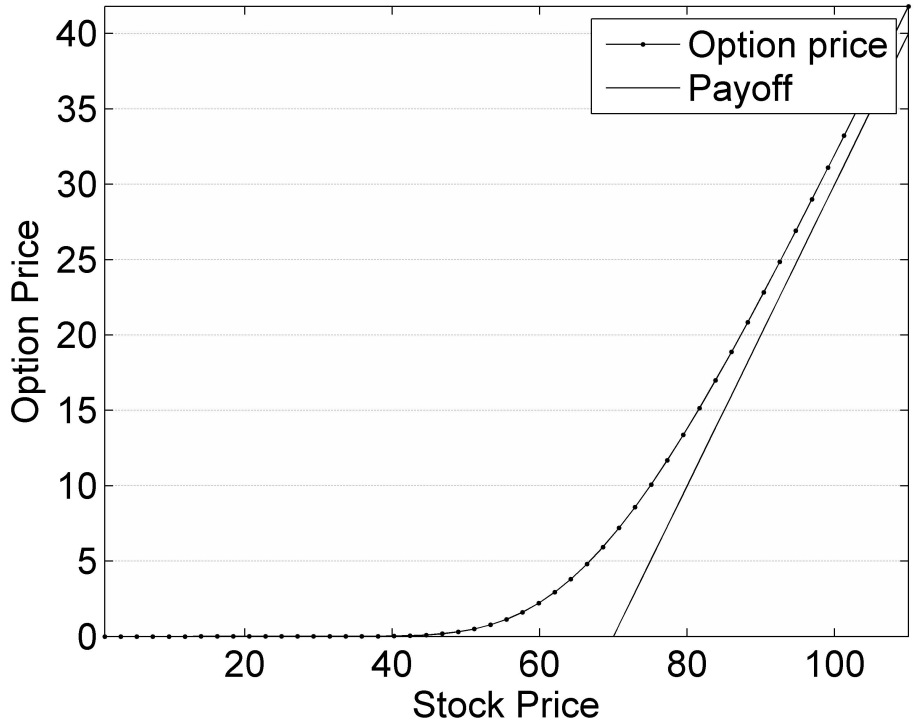

Figure 2: Option price as a function of asset—here assumed to be a stock—price for a call option, as given by the Black-Scholes formula [equation (17)] with $T = 0.5$ years, $\sigma = 30\%$, $r = 5\%$, $q = 0$, and $X = 70$. Also shown is the payoff for the option, given by equation (16).

the expressions of § 3) are given in Table 1. We use these to plot $\Theta(T)$ (which represents the negative of the slope of the curve in Figure 1), shown in Figure 3, while taking the slope of $P_c(S)$ gives $\Delta(S)$, which is plotted in Figure 4.

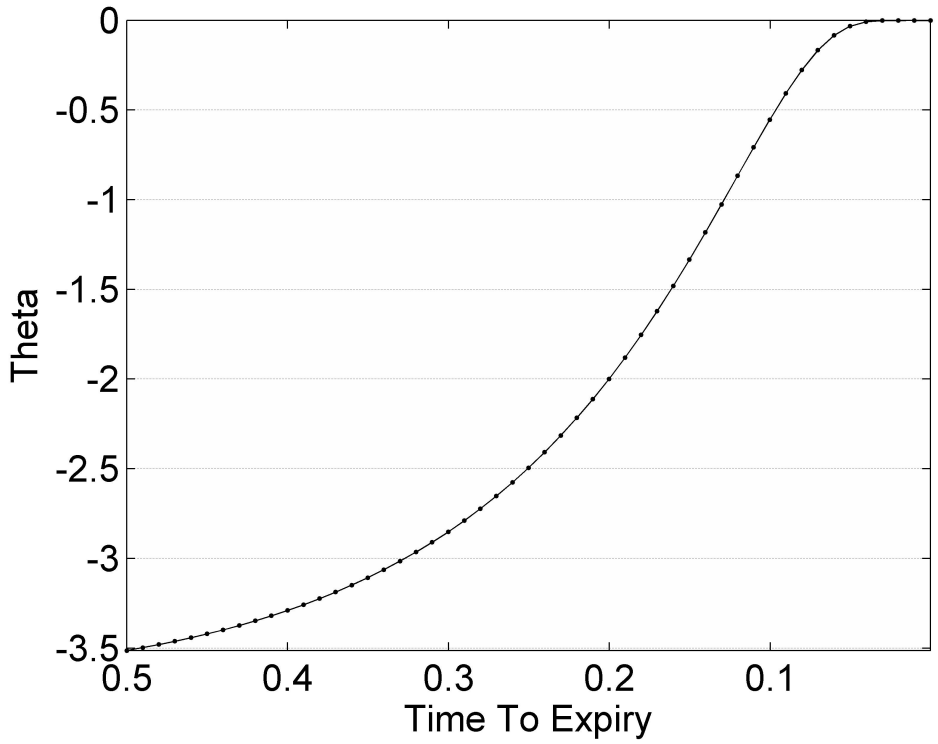

Figure 3: *Theta* as a function of expiry time for a call option, as given by the Black-Scholes formula with the parameters of Figure 1.

5 NAG Library option pricing routines

Chapter S of the NAG Library [18] is devoted to approximations of special functions in mathematics and physics, and contains several routines for calculating these. The latest release of the library includes a number of option pricing functions in this chapter, arising from a number of different models (see Table 2). Beginning with the comparatively simple formula derived from the Black-Scholes[7] model of § 4, the chapter offers a range of routines for determining the prices of options (both vanilla and exotic) which can be called from within developers' applications. We note in passing that, for reasons of convenience, each pricing formula—with a few exceptions—is offered in two routines: one that calculates the Greeks and one that doesn't (the exceptions are those models—Bjerksund and Stensland, Standard Barrier and Heston—for which estimates of the Greeks are not available in the current

[7]Historically, these formulae are often labelled as Black-Scholes-Merton, which is the notation used by the NAG Library documentation, as reflected in Table 2.

Greek	Formula
Δ_c	$e^{-qT}\Phi(d_1)$
Δ_p	$-e^{-qT}\Phi(-d_1)$
Γ	$\frac{\Phi'(d_1)e^{-qT}}{S\sigma\sqrt{T}}$
ν	$Se^{-qT}\Phi'(d_1)\sqrt{T}$
Θ_c	$-\frac{Se^{-qT}\Phi'(d_1)\sigma}{2\sqrt{T}} + qSe^{-qT}\Phi(d_1) - rXe^{-rT}\Phi(d_2)$
Θ_p	$-\frac{Se^{-qT}\Phi'(d_1)\sigma}{2\sqrt{T}} - qSe^{-qT}\Phi(-d_1) + rXe^{-rT}\Phi(-d_2)$
ρ_c	$TXe^{-rT}\Phi(d_2)$
ρ_p	$-TXe^{-rT}\Phi(-d_2)$
$Crho_c$	$TSe^{-qT}\Phi(d_1)$
$Crho_p$	$-TSe^{-qT}\Phi(-d_1)$
$Vanna$	$\frac{-e^{-qT}d_2}{\sigma}\Phi'(d_1)$
$Charm_c$	$-e^{-qT}\left[\Phi'(d_1)\left(\frac{r-q}{\sigma\sqrt{T}} - \frac{d_2}{2T}\right) - q\Phi(d_1)\right]$
$Charm_p$	$-e^{-qT}\left[\Phi'(d_1)\left(\frac{r-q}{\sigma\sqrt{T}} - \frac{d_2}{2T}\right) + q\Phi(-d_1)\right]$
$Speed$	$\frac{\Gamma\cdot\left(1+\frac{d_1}{\sigma\sqrt{T}}\right)}{S}$
$Colour$	$\Gamma\cdot\left(q + \frac{(r-q)d_1}{\sigma\sqrt{T}} + \frac{1-d_1d_2}{2T}\right)$
$Zomma$	$\Gamma\cdot\left(\frac{d_1d_2-1}{\sigma}\right)$
$Vomma$	$\nu\cdot\left(\frac{d_1d_2}{\sigma}\right)$

Table 1: Expressions for the Greeks in the Black-Scholes model. *Delta*, *Theta*, *Rho*, *Crho* and *Charm* are different for call and put options; the other Greeks are independent of this distinction.

release of the library).

The advantages of using a NAG routine (as opposed to user-developed code) are well-known[8]: the implementation of every algorithm has been extensively tested for accuracy, reliability and stability, and is accessible from a variety of programming environments. We illustrate this here by presenting two examples showing how the NAG option pricing routines can be called from within Microsoft Excel. We use Visual Basic for Applications (VBA)—which is the standard environment for adding functionality to Excel—and the NAG Fortran Library[9] in the form of a Dynamic Link Library (DLL); it is then straightforward [19] to call Library routines from within a VBA program which, in the simplest case, would be a mere wrapper around the

[8]Although these advantages could perhaps be viewed as marginal in the case of the formulae of § 4, it should be noted that the expressions for prices and the Greeks in the more sophisticated models are increasingly complicated.

[9]Option pricing routines are also available in the NAG C Library and the NAG Library for SMP and Multicore.

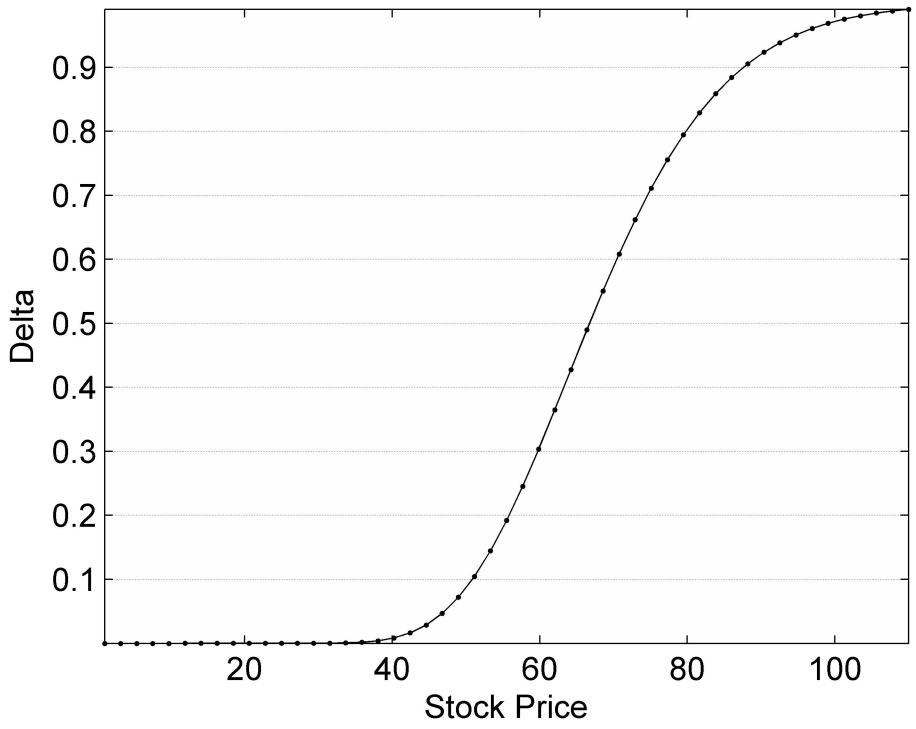

Figure 4: *Delta* as a function of asset price for a call option, as given by the Black-Scholes formula with the parameters of Figure 2.

NAG function.

5.1 Example 1

The first example is a simple VBA program which calls the S30ABF routine to calculate the price of a European put option and the Greeks using the Black-Scholes formula of § 4. The code[10], shown in Figure 5, can be inserted [19] into a Module Sheet of an Excel Workbook, allowing the function `NAG_option_price(X,S,T,Sigma,r,q)` to be called from within a spreadsheet in the workbook.

Some technical details about the code may be helpful. In VBA, the `Declare` statement is used to include the definition of a DLL function; we use it [19] near the beginning of our program to define the interface to S30ABF, which is to be found in the NAG DLL `FLDLL244M_nag.dll`. The syntax of the `Declare` statement for each function in the NAG Library is part of the distribution of the DLL (and is also available on the Web at [20]), and so can be cut and pasted into our program. The rest of the code deals with the declaration of other parameters, calling the NAG routine, checking for errors it encounters (all of which in this case concern incorrect input values) and transferring the results to an array which is passed back to the spreadsheet.

The parameters for `NAG_option_price` are `X` (strike price), `S` (underlying asset price), `T` (time to expiry), `Sigma` (volatility), `r` (annual risk-free interest rate) and `q` (dividend yield). Values for these variables can be entered into the spreadsheet and passed into the function, which returns an array (the `output` variable in the program) that contains (see Figure 5) the option price and the Greeks. Figure 6 is a screenshot showing `NAG_option_price` being called from a spreadsheet[11].

We note in passing that the NAG option pricing routines are set up to calculate P for multiple values of X and T. Thus for example, in Figure 5, the third and fourth parameters in the call to S30ABF are `m` and `n`, defined respectively as the number of strike prices and the the number of expiry times to be used. Values for these are passed into the routine via the arrays `X(m)` and `T(n)`, while the output is returned in the two-dimensional arrays

[10] See § 5.3 for details about obtaining a copy of the code.

[11] Note that `NAG_option_price` returns an array of values. In order to call a VBA function which returns an array (and to have those values displayed in the spreadsheet), we first select the range of cells (in this case, thirteen of them) into which the array will be written, type in the formula—here, `NAG_option_price(C4,C5,C6,C7,C8,C9)`—and then press `CTRL SHIFT ENTER` together (as opposed to pressing `ENTER`, which would only enter the first element of the array into the first cell).

Routine	Description	Origin
S30AAF	Black-Scholes-Merton option pricing formula	[3, 15]
S30ABF	Black-Scholes-Merton option pricing formula with Greeks	[3, 15]
S30BAF	Floating-strike lookback option pricing formula	[8]
S30BBF	Floating-strike lookback option pricing formula with Greeks	[8]
S30CAF	Binary option: cash-or-nothing option pricing formula	[23]
S30CBF	Binary option: cash-or-nothing option pricing formula with Greeks	[23]
S30CCF	Binary option: asset-or-nothing option pricing formula	[23]
S30CDF	Binary option: asset-or-nothing option pricing formula with Greeks	[23]
S30FAF	Standard barrier option pricing formula	[9]
S30JAF	Jump-diffusion, Merton model, option pricing formula	[9, 16]
S30JBF	Jump-diffusion, Merton model, option pricing formula with Greeks	[9, 16]
S30NAF	Heston model option pricing formula	[1, 10, 13, 24]
S30QCF	American option: Bjerksund and Stensland pricing formula	[2, 7]
S30SAF	Asian option: geometric continuous average rate option pricing formula	[12]
S30SBF	Asian option: geometric continuous average rate option pricing formula with Greeks	[12]

Table 2: Option pricing routines in Mark 24 of the NAG Fortran Library.

```
'each parameter must be declared explicitly
Option Explicit
'arrays are numbered from 1 (default is 0)
Option Base 1

'declaration of NAG routine S30ABF (provided by NAG)
Declare Sub S30ABF Lib "FLDLL244M_nag.dll" ( _
ByVal CALPUT As String, ByVal CALPUTLength As Long, _
ByRef m As Long, ByRef n As Long, ByRef X As Double, _
ByRef S As Double, ByRef T As Double, ByRef Sigma As Double, _
ByRef r As Double, ByRef q As Double, ByRef P As Double, _
ByRef LDP As Long, ByRef Delta As Double, _
ByRef Gamma As Double, ByRef Vega As Double, _
ByRef Theta As Double, ByRef Rho As Double, _
ByRef Crho As Double, ByRef Vanna As Double, _
ByRef Charm As Double, ByRef Speed As Double, _
ByRef Colour As Double, ByRef Zomma As Double, _
ByRef Vomma As Double, ByRef IFAIL As Long )

'This function calculates an option price using a NAG routine
Function NAG_option_price( _
ByRef X As Double, ByRef S As Double, _
ByRef T As Double, ByRef Sigma As Double, _
ByRef r As Double, ByRef q As Double) As Variant

'declaration of parameters
'the option price
Dim P As Double
'the Greeks
Dim Delta As Double, Gamma As Double, Vega As Double
Dim Theta As Double, Rho As Double, Crho As Double
Dim Vanna As Double, Charm As Double, Speed As Double
Dim Colour As Double, Zomma As Double, Vomma As Double
'the error-checking parameter
Dim IFAIL As Long:  IFAIL = 1
'the array that will return option price and Greeks
Dim output(13, 1) As Double

'error handling mechanism
On Error GoTo error_handler:

'a call to the NAG routine
'for details look at the documentation
Call S30ABF( _
  "P", 1, 1, 1, X, S, T, Sigma, r, q, P, 1, _
  Delta, Gamma, Vega, Theta, Rho, Crho, Vanna, _
  Charm, Speed, Colour, Zomma, Vomma, IFAIL)

'catching of possible errors
Select Case IFAIL
Case 4:
  Err.Raise Number:=4, Description:= _
  "On entry, X < z or X > 1/z, where z is " _
  & "the safe range parameter."
Case 5:
  Err.Raise Number:=5, Description:= _
  "On entry, S < z or S > 1/z, where z is " _
  & "'the safe range parameter."
Case 6:
  Err.Raise Number:=6, Description:= _
  "On entry, T < z, where z is " _
  & "the safe range parameter."
Case 7:
  Err.Raise Number:=7, Description:= _
  "On entry, Sigma <= 0.0."
Case 8:
  Err.Raise Number:=8, Description:= _
  "On entry, r <= 0.0."
Case 9:
  Err.Raise Number:=9, Description:= _
  "On entry, q <= 0.0."
End Select

'the results are transferred into an array
output(1, 1) = P
output(2, 1) = Delta
output(3, 1) = Gamma
output(4, 1) = Vega
output(5, 1) = Theta
output(6, 1) = Rho
output(7, 1) = Crho
output(8, 1) = Vanna
output(9, 1) = Charm
output(10, 1) = Speed
output(11, 1) = Colour
output(12, 1) = Zomma
output(13, 1) = Vomma

'the function returns an array of values
NAG_option_price = output
Exit Function

'return a message in case of an error
error_handler:
  NAG_option_price = Err.Description
Exit Function

End Function
```

Figure 5: VBA function which calls S30ABF to calculate the price of a European put option and the Greeks using the Black-Scholes-Merton formula.

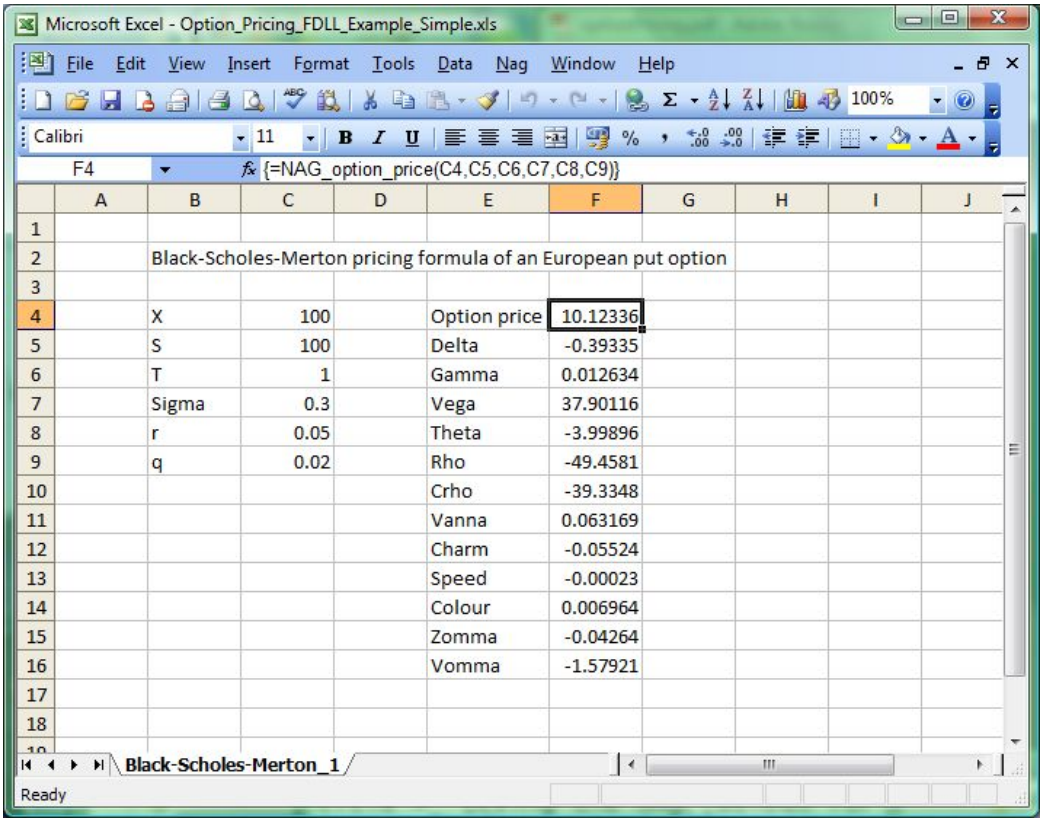

Figure 6: Calling the function `NAG_option_price` from within an Excel spreadsheet.

`P(m,n)`, `Delta(m,n)`, etc[12]. In our example program, we only calculate a single price and set of Greeks, so these variables are hardwired to `1`, and `P`, `Delta`, etc are defined as scalars.

5.2 Example 2

The second example (see Figure 7) is a more complicated VBA program which allows the user to interactively select the option pricing model, choose between a call and put option and set values for the dependent parameters. After calling the appropriate NAG routine, Excel's plotting capabilities are used in the worksheet to visualize the dependency of the price and the Greek on either the strike price or the time to expiry. The user can choose the Greek which is to be plotted, select the plots' x-axis and set the range of x

[12]The leading dimension of these arrays is specified by the variable `LDP` (the twelfth parameter in the call to S30ABF), which must be greater than or equal to `m`.

values to be plotted.

Note that the pricing model selected in the example displayed in Figure 7 is the so-called cash-or-nothing formula [23], as calculated by the S30CAF routine. This type of option pays a fixed amount, K, at expiration if the asset price is greater than strike price for a call option, and vice-versa for a put option. The prices of a call and a put option are then [25]:

$$P_c = Ke^{-rT}\Phi(d_2), \tag{22}$$
$$P_p = Ke^{-rT}\Phi(-d_2). \tag{23}$$

[cf equations (17) and (18)]. Comparison of the upper plot in Figure 7 with Figure 2 illustrates the qualitative difference between different types of option.

5.3 Availability

An Excel workbook which contains the program of Example 1, together with an extended version which returns option prices and Greeks that correspond to multiple strike prices and expiry times, is available for download at [21].

A live demo of Example 2—including audio commentary, and a discussion of the contents of the program—is available at [22]. Users who would like a copy of the Excel workbook for Example 2 are asked to contact NAG at support@nag.co.uk.

6 Conclusions and future work

In this note, we have looked at calculation of financial option prices in Excel using the NAG Library. We have discussed the mechanism by which NAG routines can be accessed from within an Excel worksheet, and illustrated this with some examples.

Other aspects of this area remain to be studied, including the exploration of how the NAG routines can be called from within other environments such as MATLAB [6, 26]. In addition, alternative methods for estimating option prices can be implemented using routines from the NAG Library; these include Monte Carlo simulation [14] and the numerical solution [17] of equation (13) and similar PDEs arising from other models such as Heston's [10]. More specifically, we note the existence of a library routine [4] in chapter D03 (which is devoted to the numerical solution of PDEs) for the solution of the Black-Scholes PDE using a finite-difference scheme. This implementation method allows the relaxation of some of the assumptions associated

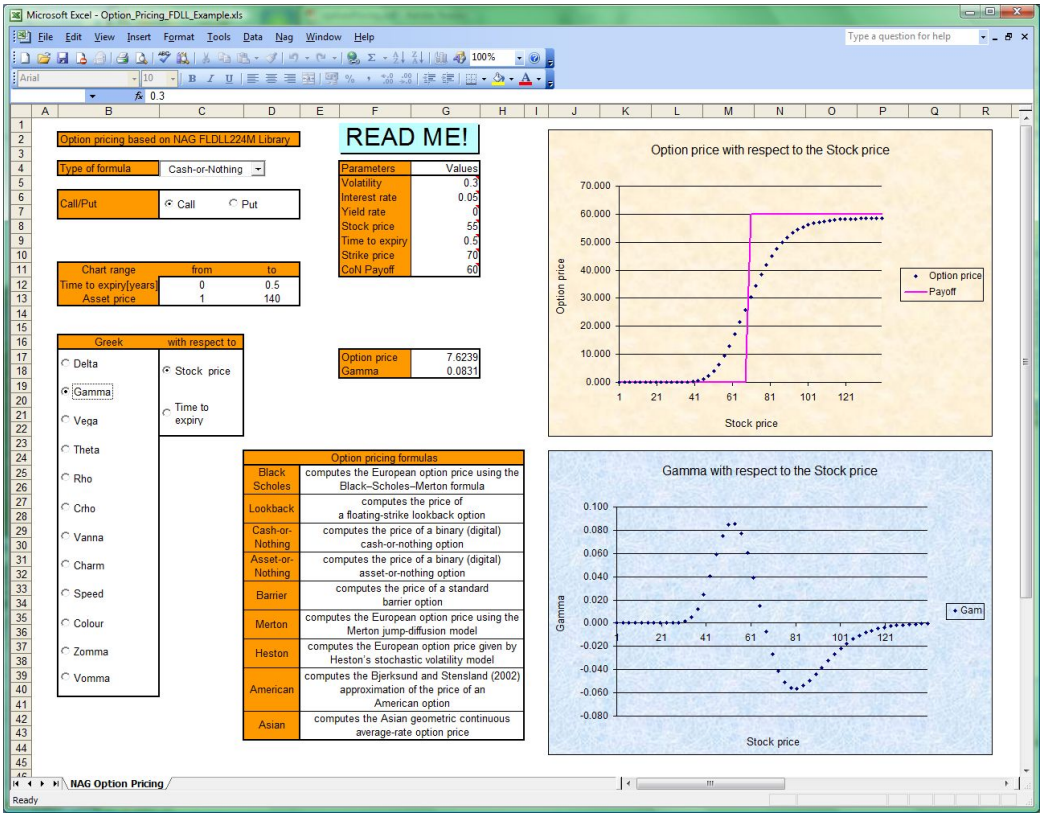

Figure 7: An example Excel worksheet that uses several NAG option pricing routines. The user can select the method, choose between a call and put option (top left) and enter values for the dependent parameters (top centre). The results from the calculation are displayed in the graphs on the right (option price above, Greek below). The user can edit various aspects of the graphs (including selecting the Greek to be plotted) via Excel Form controls (bottom left).

with the derivation of the analytic solution to equation (13)—for example, the parameters σ, r and q can now vary with time. (There may also be differences in performance between a numerical solution and the evaluation of an analytic expression.) Finally, for the sake of completeness, we mention a D03 routine [5] that evaluates the analytic solution to equation (13); this differs from S30ABF in that it attempts to allow for time dependent parameters by using their average instantaneous values in the calculation of the price and the Greeks.

Acknowledgements

We thank Robert Tong, Mick Pont, Lawrence Mulholland, David Sayers and John Holden for helpful discussions whilst preparing this note.

References

[1] H. Albrecher, P. Mayer, W. Schoutens, and J. Tistaert. The little Heston trap. *Wilmott Magazine*, January 2007, 2007.

[2] P. Bjerksund and G. Stensland. Closed form valuation of American options. Discussion paper 2002/09, NHH Bergen, Norway, 2002.

[3] F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81:637–654, 1973.

[4] NAG Library Routine Document: D03NCF. http://www.nag.co.uk/numeric/fl/nagdoc_fl24/pdf/D03/d03ncf.pdf.

[5] NAG Library Routine Document: D03NDF. http://www.nag.co.uk/numeric/fl/nagdoc_fl24/pdf/D03/d03ndf.pdf.

[6] N. Esteves, N. Fenton, and J. Walton. Using the NAG Toolbox for MATLAB—Part 4. Industry article, NAG Ltd, 2009. http://www.nag.co.uk/IndustryArticles/usingtoolboxmatlabpart4.asp.

[7] A. Genz. Numerical computation of rectangular bivariate and trivariate normal and t probabilities. *Statistics and Computing*, 14:151–160, 2004.

[8] B. M. Goldman, H. B. Sosin, and M. A. Gatto. Path dependent options: buy at the low, sell at the high. *Journal of Finance*, 34:1111–1127, 1979.

[9] E. G. Haug. *The Complete Guide to Option Pricing Formulas*. McGraw-Hill, 2nd edition, 2007.

[10] S. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6:347–343, 1993.

[11] M. S. Joshi. *The Concepts and Practice of Mathematical Finance*. Cambridge University Press, 2003.

[12] A. Kemna and A. Vorst. A pricing method for options based on average asset values. *Journal of Banking and Finance*, 14, 1990.

[13] F. Kilin. Accelerating the calibration of stochastic volatility models. MPRA Paper No. 2975, 2006. http://mpra.ub.uni-muenchen.de/2975/.

[14] M. Krzysztofik. Monte Carlo methods in finance—application of NAG random number generators in option pricing using MS Excel and VBA. In preparation, 2010.

[15] R. C. Merton. Theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4:141–183, 1973.

[16] R. C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3:125–144, 1976.

[17] L. Mulholland. Personal communication, 2010.

[18] The NAG Library. http://www.nag.co.uk/numeric/numerical_libraries.asp.

[19] NAG DLLs and Microsoft Excel and Microsoft Visual Basic for Applications. http://www.nag.com/numeric/callingDLLsfromotherlang.asp.

[20] Declare statements for NAG Fortran Library DLL Mark 24. http://www.nag.co.uk/numeric/NAGExcelExamples/vb6_headers.zip.

[21] NAG and Microsoft Excel—Expand your capabilities. http://www.nag.co.uk/numeric/nagandexcel.asp.

[22] Webinars highlighting NAG in Excel functions. http://www.nag.co.uk/numeric/nagexcelexamples/webinars.asp.

[23] E. Reiner and M. Rubinstein. Unscrambling the binary code. *Risk*, 4, 1991.

[24] F. D. Rouah and G. Vainberg. *Option Pricing Models and Volatility using Excel-VBA*. John Wiley and Sons, Inc, 2007.

[25] NAG Library Routine Document: S30CAF. http://www.nag.co.uk/numeric/fl/nagdoc_fl24/pdf/S/s30caf.pdf.

[26] J. Walton and M. Krzysztofik. Using the NAG Toolbox for MATLAB—Part 5, 2010.