Practical Session

Do all calculations in double precision. Working together is fine. There are plenty of questions here: don't feel you have to all of them. If something seems too easy or too difficult, feel free to skip it (some questions do build on preceding ones, though).

Note: throughout the tutorials, you can pass `NULL` as the value of the `NagError` pointer `fail` when calling any of the NAG routines.

1 Question 1

Write a function `bs` which takes the following inputs:

`S`: the initial stock price	`K`: the strike price	`T`: the maturity
`r`: the interest rate	`q`: the dividend rate	`sigma`: the volatility

The function `bs` should call `s30aac` to return the price of a simple European call option. Use your function to price a call option with the following parameters:

$S = 100$	$K = 90$	$T = 1.5$
$r = 0.03$	$q = 0.015$	$\sigma = 0.09$

1.1 Answer

You should get an answer of 12.35008695.

2 Question 2

Refer to the documentation for `c05awc`. We are now going to write a program to compute the Black Scholes implied volatility for a given call option price and set of parameters. In other words, given a call option price C and values of S, K, T, r, q, find the value of σ so that the Black Scholes formula gives the price C. Modify the function `bs` from Question 1 to have the prototype

```
double NAG_CALL bs(double sigma, Nag_Comm * comm)
```

In your `main` function, declare a `Nag_Comm` structure `comm` and allocate 6 doubles to the member `comm.user`. Define variables `S, K, T, r, q` and the target call option price `Ctarget`. Assign `S, K, T, r` and `q` to the first 5 members of `comm.user` and `Ctarget` to the 6th member.

In your function `bs`, assign the first 5 members of `comm->user` to variables `S, K, T, r` and `q` and call `s30aac` to compute the price of a call option. Return the difference between this price and the 6th member of `comm->user`.

In `main`, call `c05awc` and pass it `bs` as the function of which the zero is to be computed. Set `eps = eta = 1.0e-6` and set `nfmax = 1500`.

1. Use your program to compute the implied volatility for a (target) call option price of 12.35008695 and

   $S = 100$ $K = 90$ $T = 1.5$ $r = 0.03$ $q = 0.015$

   Use an initial guess of $\sigma = 0.15$.

2. Use your program to compute the implied volatility for a (target) call option price of 25.5 and

$$S = 100 \qquad K = 90 \qquad T = 1.5 \qquad r = 0.03 \qquad q = 0.015$$

Use an initial guess of $\sigma = 0.15$.

2.1 Answer

1. You should get an answer of 0.090001.
2. You should get an answer of 0.429938.

3 Question 3

We now turn to simple Monte Carlo simulation. We will price a simple Black Scholes call option. In your `main` function, declare variables `N, S, K, T, r` and `sigma`; two arrays `seed[6]` and `state[70]` of unsigned integers; and an array `Z` of doubles. Allocate `N` doubles for `Z` and set the first 6 elements of `seed` to the first 6 integers 1,2,3,4,5, and 6. Call `g05kfc` with `genid = Nag_MRG32k3a` and `subid = 0`, and pass in the `seed` and `state` arrays (remember to pass the **address** of `lstate`). Now call `g05skc` to generate `N` Normal random numbers (mean zero and variance one) in `Z`. Use the random numbers to compute the Monte Carlo average

$$\hat{C} = \frac{1}{N} \sum_{i=0}^{N-1} e^{-rT} \max\left\{ S \exp\left((r - q - \sigma^2/2)T + \sigma\sqrt{T} Z[\,i\,] \right) - K, 0 \right\}$$

for $N = 1000000$ and

$$S = 100 \qquad K = 90 \qquad T = 1.5 \qquad r = 0.03 \qquad q = 0.015 \qquad \sigma = 0.09$$

3.1 Answer

You should get an answer of 12.35136975.

4 Question 4

We modify the previous question to price a basket option. Add an integer `A` to the start of your `main` function, and create a correlation array `cor` of size $A \times A$. Allocate `N*A` doubles for `Z`, call `g05kfc` as before and use `g05skc` to generate `N*A` Normal random numbers (mean zero and variance one) in `Z`. Now call `f07fdc` to obtain the Cholesky factorisation of the correlation matrix `cor`: use `order=Nag_ColMajor, uplo=Nag_Lower` and `pda=A`.

We need to multiply each A-dimensional vector of Normal random numbers by the Cholesky factorisation matrix. For this, call `f16yfc` with `order=Nag_ColMajor, side=Nag_LeftSide, uplo=Nag_Lower, trans=Nag_NoTrans, diag=Nag_NonUnitDiag` and `alpha=1`. For `a` pass in the matrix `cor` and for `b` pass in the array `Z`; set `m=A, n=N, pda=A` and `pdb=A`. Finally, use the (now correlated) Normal random numbers to compute the Monte Carlo sum

$$\hat{C} = \frac{1}{N} \sum_{i=0}^{N-1} e^{-rT} \max\left\{ \frac{1}{A} X_i - K, 0 \right\}$$

where

$$X_i = \sum_{a=0}^{A-1} S_a \exp\left(\left(r - \tfrac{1}{2}\sigma_a^2\right) T + \sigma_a \sqrt{T} Z[\,iA + a\,] \right)$$

for $N = 50000$, $A = 5$, $K = 80$, $T = 1.5$, $r = 0.03$ and

$$S = \begin{bmatrix} 100 & 90 & 80 & 70 & 60 \end{bmatrix}$$
$$\sigma = \begin{bmatrix} 0.04 & 0.06 & 0.07 & 0.10 & 0.15 \end{bmatrix}$$
$$\texttt{cor} = \begin{pmatrix} 1.0 & 0.3 & 0.4 & -0.7 & 0.0 \\ 0.3 & 1.0 & 0.5 & -0.3 & 0.1 \\ 0.4 & 0.5 & 1.0 & -0.2 & -0.5 \\ -0.7 & -0.3 & -0.2 & 1.0 & 0.4 \\ 0.0 & 0.1 & -0.5 & 0.4 & 1.0 \end{pmatrix}$$

4.1 Answer

You should get an answer of 3.760712586.

5 Question 5

We now consider the Heston model. Use `s30nac` to price a call option in the Heston model with the following parameters:

$$\begin{array}{lll} S = 100 & K = 90 & T = 1.5 \\ r = 0.03 & q = 0.015 & \sigma_v = 0.09 \\ \kappa = 1 & \eta = 0.15 & v_0 = 0.15 \\ \rho = 0.5 & \gamma = 0.5 & \end{array}$$

5.1 Answer

You should get an answer of 23.5895499.

6 Question 6

We now look at how to calibrate the Heston model. This is a much simplified version on what one has to do in practice, but the basic ideas are the same. Look at the documentation for `e04unc`. We are going to do a 3 parameter calibration with 3 input points (i.e. $n = m = 3$). The parameters from the Heston model we will calibrate will be r, σ_v and η. All other parameters will be assumed known.

Create an `objfun` function with prototype

```
void NAG_CALL objfun(Integer m, Integer n, const double x[],
                     double f[], double fjac[],
                     Integer tdfjac, Nag_Comm *comm)
```

which can be passed to `e04unc`. Inside the function, assign the first three values of `x` to variables `r, sigmav` and `eta`. From the `Nag_Comm` pointer `comm`, read all the remaining values `K, S, T, kappa, var0, rho, grisk, q` from the member `comm->user` (remember there will be 3 values of K). Now call `s30nac` to get the prices of the `m` call options and write them into `f`.

In your `main` function, create a `NAG_E04_Opt` structure `options`, initialise it with `e04xxc` and set `options.obj_deriv = Nag_FALSE`. This means we will not have to compute derivatives of the Heston model: the solver will instead compute the derivatives numerically. The three points we will calibrate to are

$$(K_1, C_1) = (90, 23.238062)$$
$$(K_2, C_2) = (100, 18.375622)$$
$$(K_3, C_3) = (110, 14.492988)$$

Create a `Nag_Comm` structure `comm` and assign an array of 10 doubles to `comm.user`. Load the 3 strikes K_1, K_2, K_3, as well as all the other parameters $S, T, q, \kappa, v_0, \rho, \gamma$ into `comm.user`. Put the values C_1, C_2, C_3 into an array `y`. Now create an array `bl` of lower bounds for r, σ_v and η and give each element a value of `0.0001`. Create a similar array `bu` for the upper bounds and give r and η upper bounds of 100. Look at the `s30nac` documentation and work out what the upper bound of η should be (it's best not to formulate this as a non-linear constraint, so work out what the linear equivalent is). Call `e04unc` with the following parameter values:

$$S = 100 \qquad T = 1.5 \qquad q = 0.015 \qquad \kappa = 1$$
$$v_0 = 0.15 \qquad \rho = 0.5 \qquad \gamma = 0.5$$

Use an initial guess for r, σ_v and η of

$$r = 0.03 \qquad \sigma_v = 0.09 \qquad \eta = 0.15$$

6.1 Answer

You should get an answer of $r = 0.070000$, $\sigma_v = 0.210001$ and $\eta = 0.049999$ with a final objective value of 5.7439487e-24.