NAG Library Routine Document

S30BBF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

S30BBF computes the price of a floating-strike lookback option together with its sensitivities (Greeks).

2 Specification

```
SUBROUTINE S30BBF (CALPUT, M, N, SM, S, T, SIGMA, R, Q, P, LDP, DELTA,      &
                   GAMMA, VEGA, THETA, RHO, CRHO, VANNA, CHARM, SPEED,      &
                   COLOUR, ZOMMA, VOMMA, IFAIL)
INTEGER            M, N, LDP, IFAIL
REAL (KIND=nag_wp) SM(M), S, T(N), SIGMA, R, Q, P(LDP,N), DELTA(LDP,N),     &
                   GAMMA(LDP,N), VEGA(LDP,N), THETA(LDP,N),                 &
                   RHO(LDP,N), CRHO(LDP,N), VANNA(LDP,N),                   &
                   CHARM(LDP,N), SPEED(LDP,N), COLOUR(LDP,N),               &
                   ZOMMA(LDP,N), VOMMA(LDP,N)
CHARACTER(1)       CALPUT
```

3 Description

S30BBF computes the price of a floating-strike lookback call or put option, together with the Greeks or sensitivities, which are the partial derivatives of the option price with respect to certain of the other input parameters. A call option of this type confers the right to buy the underlying asset at the lowest price, S_{min}, observed during the lifetime of the contract. A put option gives the holder the right to sell the underlying asset at the maximum price, S_{max}, observed during the lifetime of the contract. Thus, at expiry, the payoff for a call option is $S - S_{\mathrm{min}}$, and for a put, $S_{\mathrm{max}} - S$.

For a given minimum value the price of a floating-strike lookback call with underlying asset price, S, and time to expiry, T, is

$$P_{\mathrm{call}} = Se^{-qT}\Phi(a_1) - S_{\mathrm{min}}e^{-rT}\Phi(a_2) + Se^{-rT}\frac{\sigma^2}{2b}\left[\left(\frac{S}{S_{\mathrm{min}}}\right)^{-2b/\sigma^2}\Phi\left(-a_1+\frac{2b}{\sigma}\sqrt{T}\right) - e^{bT}\Phi(-a_1)\right],$$

where $b = r - q \neq 0$. The volatility, σ, risk-free interest rate, r, and annualised dividend yield, q, are constants.

The corresponding put price is

$$P_{\mathrm{put}} = S_{\mathrm{max}}e^{-rT}\Phi(-a_2) - Se^{-qT}\Phi(-a_1) + Se^{-rT}\frac{\sigma^2}{2b}\left[-\left(\frac{S}{S_{\mathrm{max}}}\right)^{-2b/\sigma^2}\Phi\left(a_1-\frac{2b}{\sigma}\sqrt{T}\right) + e^{bT}\Phi(a_1)\right].$$

In the above, Φ denotes the cumulative Normal distribution function,

$$\Phi(x) = \int_{-\infty}^{x}\phi(y)dy$$

where ϕ denotes the standard Normal probability density function

$$\phi(y) = \frac{1}{\sqrt{2\pi}}\exp\left(-y^2/2\right)$$

and

$$a_1 = \frac{\ln(S/S_m) + \left(b + \sigma^2/2\right)T}{\sigma\sqrt{T}}$$

$$a_2 = a_1 - \sigma\sqrt{T}$$

where S_m is taken to be the minimum price attained by the underlying asset, $S_{\min}$, for a call and the maximum price, $S_{\max}$, for a put.

The option price $P_{ij} = P\left(X = X_i, T = T_j\right)$ is computed for each minimum or maximum observed price in a set $S_{\min}(i)$ or $S_{\max}(i)$, $i = 1, 2, \ldots, m$, and for each expiry time in a set T_j, $j = 1, 2, \ldots, n$.

4 References

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

5 Parameters

1: CALPUT – CHARACTER(1) *Input*

On entry: determines whether the option is a call or a put.

CALPUT = 'C'
A call; the holder has a right to buy.

CALPUT = 'P'
A put; the holder has a right to sell.

Constraint: CALPUT = 'C' or 'P'.

2: M – INTEGER *Input*

On entry: the number of minimum or maximum prices to be used.

Constraint: M ≥ 1.

3: N – INTEGER *Input*

On entry: the number of times to expiry to be used.

Constraint: N ≥ 1.

4: SM(M) – REAL (KIND=nag_wp) array *Input*

On entry: SM(i) must contain $S_{\min}(i)$, the ith minimum observed price of the underlying asset when CALPUT = 'C', or $S_{\max}(i)$, the maximum observed price when CALPUT = 'P', for $i = 1, 2, \ldots,$ M.

Constraints:

SM(i) $\geq z$ and SM(i) $\leq 1/z$, where $z =$ X02AMF(), the safe range parameter, for $i = 1, 2, \ldots,$ M;
if CALPUT = 'C', SM(i) $\leq S$, for $i = 1, 2, \ldots,$ M;
if CALPUT = 'P', SM(i) $\geq S$, for $i = 1, 2, \ldots,$ M.

5: S – REAL (KIND=nag_wp) *Input*

On entry: S, the price of the underlying asset.

Constraint: S $\geq z$ and S $\leq 1.0/z$, where $z =$ X02AMF(), the safe range parameter.

6: T(N) – REAL (KIND=nag_wp) array *Input*

On entry: T(i) must contain T_i, the ith time, in years, to expiry, for $i = 1, 2, \ldots, \mathrm{N}$.

Constraint: T(i) $\geq z$, where $z = \mathrm{X02AMF}()$, the safe range parameter, for $i = 1, 2, \ldots, \mathrm{N}$.

7: SIGMA – REAL (KIND=nag_wp) *Input*

On entry: σ, the volatility of the underlying asset. Note that a rate of 15% should be entered as 0.15.

Constraint: SIGMA > 0.0.

8: R – REAL (KIND=nag_wp) *Input*

On entry: the annual risk-free interest rate, r, continuously compounded. Note that a rate of 5% should be entered as 0.05.

Constraint: $\mathrm{R} \geq 0.0$ and $\mathrm{abs}(\mathrm{R} - \mathrm{Q}) > 10 \times \mathrm{eps} \times \max(\mathrm{abs}(\mathrm{R}), 1)$, where $\mathrm{eps} = \mathrm{X02AJF}()$, the ***machine precision***.

9: Q – REAL (KIND=nag_wp) *Input*

On entry: the annual continuous yield rate. Note that a rate of 8% should be entered as 0.08.

Constraint: $\mathrm{Q} \geq 0.0$ and $\mathrm{abs}(\mathrm{R} - \mathrm{Q}) > 10 \times \mathrm{eps} \times \max(\mathrm{abs}(\mathrm{R}), 1)$, where $\mathrm{eps} = \mathrm{X02AJF}()$, the ***machine precision***.

10: P(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: P(i, j) contains P_{ij}, the option price evaluated for the minimum or maximum observed price $S_{\mathrm{min}}(i)$ or $S_{\mathrm{max}}(i)$ at expiry T_j for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

11: LDP – INTEGER *Input*

On entry: the first dimension of the arrays P, DELTA, GAMMA, VEGA, THETA, RHO, CRHO, VANNA, CHARM, SPEED, COLOUR, ZOMMA and VOMMA as declared in the (sub)program from which S30BBF is called.

Constraint: LDP $\geq$ M.

12: DELTA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: the leading $\mathrm{M} \times \mathrm{N}$ part of the array DELTA contains the sensitivity, $\frac{\partial P}{\partial S}$, of the option price to change in the price of the underlying asset.

13: GAMMA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: the leading $\mathrm{M} \times \mathrm{N}$ part of the array GAMMA contains the sensitivity, $\frac{\partial^2 P}{\partial S^2}$, of DELTA to change in the price of the underlying asset.

14: VEGA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: VEGA(i, j), contains the first-order Greek measuring the sensitivity of the option price P_{ij} to change in the volatility of the underlying asset, i.e., $\frac{\partial P_{ij}}{\partial \sigma}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

15: THETA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: THETA(i, j), contains the first-order Greek measuring the sensitivity of the option price P_{ij} to change in time, i.e., $-\frac{\partial P_{ij}}{\partial T}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$, where $b = r - q$.

16: RHO(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: RHO(i,j), contains the first-order Greek measuring the sensitivity of the option price P_{ij} to change in the annual risk-free interest rate, i.e., $-\frac{\partial P_{ij}}{\partial r}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

17: CRHO(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: CRHO(i,j), contains the first-order Greek measuring the sensitivity of the option price P_{ij} to change in the annual cost of carry rate, i.e., $-\frac{\partial P_{ij}}{\partial b}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$, where $b = r - q$.

18: VANNA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: VANNA(i,j), contains the second-order Greek measuring the sensitivity of the first-order Greek Δ_{ij} to change in the volatility of the asset price, i.e., $-\frac{\partial \Delta_{ij}}{\partial T} = -\frac{\partial^2 P_{ij}}{\partial S \partial \sigma}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

19: CHARM(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: CHARM(i,j), contains the second-order Greek measuring the sensitivity of the first-order Greek Δ_{ij} to change in the time, i.e., $-\frac{\partial \Delta_{ij}}{\partial T} = -\frac{\partial^2 P_{ij}}{\partial S \partial T}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

20: SPEED(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: SPEED(i,j), contains the third-order Greek measuring the sensitivity of the second-order Greek Γ_{ij} to change in the price of the underlying asset, i.e., $-\frac{\partial \Gamma_{ij}}{\partial S} = -\frac{\partial^3 P_{ij}}{\partial S^3}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

21: COLOUR(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: COLOUR(i,j), contains the third-order Greek measuring the sensitivity of the second-order Greek Γ_{ij} to change in the time, i.e., $-\frac{\partial \Gamma_{ij}}{\partial T} = -\frac{\partial^3 P_{ij}}{\partial S \partial T}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

22: ZOMMA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: ZOMMA(i,j), contains the third-order Greek measuring the sensitivity of the second-order Greek Γ_{ij} to change in the volatility of the underlying asset, i.e., $-\frac{\partial \Gamma_{ij}}{\partial \sigma} = -\frac{\partial^3 P_{ij}}{\partial S^2 \partial \sigma}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

23: VOMMA(LDP, N) – REAL (KIND=nag_wp) array *Output*

On exit: VOMMA(i,j), contains the second-order Greek measuring the sensitivity of the first-order Greek Δ_{ij} to change in the volatility of the underlying asset, i.e., $-\frac{\partial \Delta_{ij}}{\partial \sigma} = -\frac{\partial^2 P_{ij}}{\partial \sigma^2}$, for $i = 1, 2, \ldots, \mathrm{M}$ and $j = 1, 2, \ldots, \mathrm{N}$.

24: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, CALPUT = ⟨*value*⟩ was an illegal value.

IFAIL = 2

On entry, M = ⟨*value*⟩.
Constraint: M ≥ 1.

IFAIL = 3

On entry, N = ⟨*value*⟩.
Constraint: N ≥ 1.

IFAIL = 4

On entry, SM(⟨*value*⟩) = ⟨*value*⟩.
Constraint: ⟨*value*⟩ $\leq$ SM(i) $\leq$ ⟨*value*⟩ for all i.

On entry with a call option, SM(⟨*value*⟩) = ⟨*value*⟩.
Constraint: for call options, SM(i) $\leq$ ⟨*value*⟩ for all i.

On entry with a put option, SM(⟨*value*⟩) = ⟨*value*⟩.
Constraint: for put options, SM(i) $\geq$ ⟨*value*⟩ for all i.

IFAIL = 5

On entry, S = ⟨*value*⟩.
Constraint: S $\geq$ ⟨*value*⟩ and S $\leq$ ⟨*value*⟩.

IFAIL = 6

On entry, T(⟨*value*⟩) = ⟨*value*⟩.
Constraint: T(i) $\geq$ ⟨*value*⟩ for all i.

IFAIL = 7

On entry, SIGMA = ⟨*value*⟩.
Constraint: SIGMA > 0.0.

IFAIL = 8

On entry, R = ⟨*value*⟩.
Constraint: R ≥ 0.0.

IFAIL = 9

On entry, Q = ⟨*value*⟩.
Constraint: Q ≥ 0.0.

IFAIL = 11

On entry, LDP = ⟨*value*⟩ and M = ⟨*value*⟩.
Constraint: LDP $\geq$ M.

IFAIL = 12

On entry, R = ⟨*value*⟩ and Q = ⟨*value*⟩.
Constraint: $|\mathrm{R} - \mathrm{Q}| > 10 \times \mathrm{eps} \times \max(|\mathrm{R}|, 1)$, where eps is the ***machine precision***.

IFAIL = −99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL = −399

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL = −999

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

The accuracy of the output is dependent on the accuracy of the cumulative Normal distribution function, Φ. This is evaluated using a rational Chebyshev expansion, chosen so that the maximum relative error in the expansion is of the order of the ***machine precision*** (see S15ABF and S15ADF). An accuracy close to ***machine precision*** can generally be expected.

8 Parallelism and Performance

S30BBF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

None.

10 Example

This example computes the price of a floating-strike lookback put with a time to expiry of 6 months and a stock price of 87. The maximum price observed so far is 100. The risk-free interest rate is 6% per year and the volatility is 30% per year with an annual dividend return of 4%.

10.1 Program Text

```
    Program s30bbfe

!     S30BBF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, s30bbf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: q, r, s, sigma
      Integer                          :: i, ifail, j, ldp, m, n
      Character (1)                    :: calput
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: charm(:,:), colour(:,:), crho(:,:),  &
```

```
                                          delta(:,:), gamma(:,:), p(:,:),      &
                                          rho(:,:), sm(:), speed(:,:), t(:),   &
                                          theta(:,:), vanna(:,:), vega(:,:),   &
                                          vomma(:,:), zomma(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'S30BBF Example Program Results'

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) calput
      Read (nin,*) s, sigma, r, q
      Read (nin,*) m, n

      ldp = m
      Allocate (charm(ldp,n),colour(ldp,n),crho(ldp,n),delta(ldp,n), &
        gamma(ldp,n),p(ldp,n),rho(ldp,n),sm(m),speed(ldp,n),t(n),theta(ldp,n), &
        vanna(ldp,n),vega(ldp,n),vomma(ldp,n),zomma(ldp,n))

      Read (nin,*)(sm(i),i=1,m)
      Read (nin,*)(t(i),i=1,n)

      ifail = 0
      Call s30bbf(calput,m,n,sm,s,t,sigma,r,q,p,ldp,delta,gamma,vega,theta, &
        rho,crho,vanna,charm,speed,colour,zomma,vomma,ifail)

      Write (nout,*)
      Write (nout,*) 'Floating-Strike Lookback'

      Select Case (calput)
      Case ('C','c')
        Write (nout,*) 'European Call :'
      Case ('P','p')
        Write (nout,*) 'European Put :'
      End Select

      Write (nout,99997) '  Spot       = ', s
      Write (nout,99997) '  Volatility = ', sigma
      Write (nout,99997) '  Rate       = ', r
      Write (nout,99997) '  Dividend   = ', q

      Write (nout,*)

      Do j = 1, n
        Write (nout,*)
        Write (nout,99999) t(j)
        Write (nout,*) 'S-Max/Min     Price     Delta     Gamma' // &
          '      Vega      Theta      Rho      CRho'

        Do i = 1, m
          Write (nout,99998) sm(i), p(i,j), delta(i,j), gamma(i,j), vega(i,j), &
            theta(i,j), rho(i,j), crho(i,j)
        End Do

        Write (nout,*) 'S-Max/Min     Price     Vanna     Charm' // &
          '     Speed     Colour    Zomma     Vomma'

        Do i = 1, m
          Write (nout,99998) sm(i), p(i,j), vanna(i,j), charm(i,j), &
            speed(i,j), colour(i,j), zomma(i,j), vomma(i,j)
        End Do

      End Do

99999 Format (1X,'Time to Expiry : ',1X,F8.4)
99998 Format (8(1X,F9.4))
99997 Format (A,1X,F8.4)
    End Program s30bbfe
```

10.2 Program Data

```
S30BBF Example Program Data
 'P'                    : Call = 'C', Put = 'P'
 87.0 0.3 0.06 0.04     : S, SIGMA, R, Q
 1 1                    : M, N
 100.0                  : SM(I), I = 1,2,...M
 0.5                    : T(I), I = 1,2,...N
```

10.3 Program Results

```
 S30BBF Example Program Results

 Floating-Strike Lookback
 European Put :
  Spot       =   87.0000
  Volatility =    0.3000
  Rate       =    0.0600
  Dividend   =    0.0400


 Time to Expiry :    0.5000
 S-Max/Min      Price      Delta      Gamma       Vega      Theta        Rho       CRho
  100.0000   18.3530    -0.3560     0.0391    45.5353   -11.6139   -32.8139   -23.6374
 S-Max/Min      Price      Vanna      Charm      Speed     Colour      Zomma      Vomma
  100.0000   18.3530     1.9141    -0.6199     0.0007     0.0221    -0.0648    76.1292
```