NAG Library Routine Document

S30JBF

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

S30JBF computes the European option price together with its sensitivities (Greeks) using the Merton jump-diffusion model.

2 Specification

```fortran
SUBROUTINE S30JBF(CALPUT, M, N, X, S, T, SIGMA, R, LAMBDA, JVOL, P, LDP,
1 DELTA, GAMMA, VEGA, THETA, RHO, VANNA, CHARM, SPEED,
2 COLOUR, ZOMMA, VOMMA, IFAIL)
INTEGER M, N, LDP, IFAIL
DOUBLE PRECISION X(M), S, T(N), SIGMA, R, LAMBDA, JVOL, P(LDP,N),
1 DELTA(LDP,N), GAMMA(LDP,N), VEGA(LDP,N), THETA(LDP,N),
2 RHO(LDP,N), VANNA(LDP,N), CHARM(LDP,N), SPEED(LDP,N),
3 COLOUR(LDP,N), ZOMMA(LDP,N), VOMMA(LDP,N)
CHARACTER*1 CALPUT
```

3 Description

S30JBF uses Merton’s jump-diffusion model (Merton (1976)) to compute the price of a European 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. Merton’s model assumes that the asset price is described by a Brownian motion with drift, as in the Black–Scholes–Merton case, together with a compound Poisson process to model the jumps. The corresponding stochastic differential equation is,

\[
\frac{dS}{S} = (\alpha - \lambda k)dt + \sigma dW_t + dq_t.
\]

Here \(\alpha\) is the instantaneous expected return on the asset price, \(S\); \(\sigma^2\) is the instantaneous variance of the return when the Poisson event does not occur; \(dW_t\) is a standard Brownian motion; \(dq_t\) is the independent Poisson process.

This leads to the following price for a European option (see Haug (2007))

\[
P_{\text{call}} = \sum_{j=0}^{\infty} \frac{e^{-\lambda T}(\lambda T)^j}{j!} C_j(S, X, T, r, \sigma_j'),
\]

where \(T\) is the time to expiry; \(X\) is the strike price; \(r\) is the annual risk-free interest rate; \(C_j(S, X, T, r, \sigma_j')\) is the Black–Scholes–Merton option pricing formula for a European call (see S30AAF).

\[
\sigma_j' = \sqrt{z^2 + \delta^2 \left( \frac{j}{T} \right)},
\]

\[
z^2 = \sigma^2 - \lambda \delta^2,
\]

\[
\delta^2 = \frac{\sigma^2}{\lambda},
\]

where \(\sigma\) is the total volatility including jumps; \(\lambda\) is the expected number of jumps given as an average per year; \(\gamma\) is the proportion of the total volatility due to jumps.

The value of a put is obtained by substituting the Black–Scholes–Merton put price for \(C_j(S, X, T, r, \sigma_j')\).
4 References
Merton R C (1976) Option pricing when underlying stock returns are discontinuous Journal of Financial Economics 3 125–144

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 strike 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: X(M) – double precision array  
   *Input*
   
   `On entry:` X(i) must contain X_i, the i-th strike price, for i = 1, 2, ..., M.
   
   *Constraint:* X(i) ≥ z and X(i) ≤ 1/z, where z = X02AMF(), the safe range parameter, for i = 1, 2, ..., M.

5: S – double precision  
   *Input*
   
   `On entry:` S, the price of the underlying asset.
   
   *Constraint:* S ≥ z and S ≤ 1/z, where z = X02AMF(), the safe range parameter.

6: T(N) – double precision array  
   *Input*
   
   `On entry:` T(i) must contain T_i, the i-th time, in years, to expiry, for i = 1, 2, ..., N.
   
   *Constraint:* T(i) ≥ z, where z = X02AMF(), the safe range parameter, for i = 1, 2, ..., N.

7: SIGMA – double precision  
   *Input*
   
   `On entry:` σ, the annual total volatility, including jumps.
   
   *Constraint:* SIGMA > 0.0.

8: R – double precision  
   *Input*
   
   `On entry:` r, the annual risk-free interest rate, continuously compounded. Note that a rate of 5% should be entered as 0.05.
   
   *Constraint:* R ≥ 0.0.
9: LAMBDA – double precision
   Input
   On entry: $\lambda$, the number of expected jumps per year.
   Constraint: LAMBDA > 0.0.

10: JVOL – double precision
    Input
    On entry: the proportion of the total volatility associated with jumps.
    Constraint: 0.0 $\leq$ JVOL $\leq$ 1.0.

11: P(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array P contains the computed option prices.

12: LDP – INTEGER
    Input
    On entry: the first dimension of the arrays P, DELTA, GAMMA, VEGA, THETA, RHO, VANNA, CHARM, SPEED, COLOUR, ZOMMA and VOMMA as declared in the (sub)program from which S30JBF is called.
    Constraint: LDP $\geq$ M.

13: DELTA(LDP,N) – double precision array
    Output
    On exit: the leading M and 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.

14: GAMMA(LDP,N) – double precision array
    Output
    On exit: the leading M and 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.

15: VEGA(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array VEGA contains the sensitivity, $\frac{\partial P}{\partial \sigma}$, of the option price to change in the volatility of the underlying asset.

16: THETA(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array THETA contains the sensitivity, $-\frac{\partial P}{\partial T}$, of the option price to change in the time to expiry of the option.

17: RHO(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array RHO contains the sensitivity, $\frac{\partial P}{\partial r}$, of the option price to change in the annual risk-free interest rate.

18: VANNA(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array VANNA contains the sensitivity, $\frac{\partial^2 P}{\partial S \partial \sigma}$, of VEGA to change in the price of the underlying asset or, equivalently, the sensitivity of DELTA to change in the volatility of the asset price.

19: CHARM(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array CHARM contains the sensitivity, $-\frac{\partial^2 P}{\partial S \partial T}$, of DELTA to change in the time to expiry of the option.

20: SPEED(LDP,N) – double precision array
    Output
    On exit: the leading M and N part of the array SPEED contains the sensitivity, $\frac{\partial^3 P}{\partial S^3}$, of GAMMA to change in the price of the underlying asset.
21: COLOUR(LDP,N) – double precision array
   *Output*
   On exit: the leading M and N part of the array COLOUR contains the sensitivity, \( \frac{\partial P}{\partial \sigma^2 \partial T} \), of GAMMA to change in the time to expiry of the option.

22: ZOMMA(LDP,N) – double precision array
   *Output*
   On exit: the leading M and N part of the array ZOMMA contains the sensitivity, \( \frac{\partial P}{\partial \sigma} \), of GAMMA to change in the volatility of the underlying asset.

23: VOMMA(LDP,N) – double precision array
   *Output*
   On exit: the leading M and N part of the array VOMMA contains the sensitivity, \( \frac{\partial^2 P}{\partial \sigma^2} \), of VEGA to change in the volatility of the underlying asset.

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 2.3 in the Essential Introduction for details.
   On exit: IFAIL = 0 unless the routine detects an error (see Section 6).
   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.

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 ≠ 'C' or 'P'.

IFAIL = 2
On entry, M ≤ 0.

IFAIL = 3
On entry, N ≤ 0.

IFAIL = 4
On entry, X(i) < z or X(i) > 1/z, where z = X02AMF(), the safe range parameter.

IFAIL = 5
On entry, S < z or S > 1/z, where z = X02AMF(), the safe range parameter.

IFAIL = 6
On entry, T(i) < z, where z = X02AMF(), the safe range parameter.

IFAIL = 7
On entry, SIGMA ≤ 0.0.
IFAIL = 8
On entry, R < 0.0.

IFAIL = 9
On entry, LAMBDA ≤ 0.0.

IFAIL = 10
On entry, JVOL < 0.0 or JVOL ≥ 1.0.

IFAIL = 12
On entry, LDP < M.

7 Accuracy
The accuracy of the output is dependent on the accuracy of the cumulative Normal distribution function, $\Phi$, occurring in $C_j$. 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 Further Comments
None.

9 Example
This example computes the price of two European calls with jumps. The time to expiry is 6 months, the stock price is 100 and strike prices are 80 and 90 respectively. The number of jumps per year is 5 and the percentage of the total volatility due to jumps is 25%. The risk-free interest rate is 8% per year while the total volatility is 25% per year.

9.1 Program Text
* S30JBF Example Program Text
* Mark 22 Release. NAG Copyright 2007.
* .. Parameters ..
INTEGER NIN, NOUT
PARAMETER (NIN=5,NOUT=6)
INTEGER LDP, MMAX, NMAX
PARAMETER (LDP=10,MMAX=10,NMAX=10)
* .. Local Scalars ..
DOUBLE PRECISION JVOL, LAMBDA, R, S, SIGMA
INTEGER I, IFAIL, J, M, N
CHARACTER PUT
* .. Local Arrays ..
DOUBLE PRECISION CHARM(LDP,NMAX), COLOUR(LDP,NMAX),
+ DELTA(LDP,NMAX), GAMMA(LDP,NMAX), P(LDP,NMAX),
+ RHO(LDP,NMAX), SPEED(LDP,NMAX), T(NMAX),
+ THETA(LDP,NMAX), VANNA(LDP,NMAX), VEGA(LDP,NMAX),
+ VOMMA(LDP,NMAX), X(MMAX), ZOMMA(LDP,NMAX)
* .. External Subroutines ..
EXTERNAL S30JBF
* .. Executable Statements ..
WRITE (NOUT,*) 'S30JBF Example Program Results'
WRITE (NOUT,*)
WRITE (NOUT,*) 'Merton Jump-Diffusion Model'
* Skip heading in data file
READ (NIN,*) PUT
* Read problem parameters.
READ (NIN,*) R, SIGMA, R, JVOL
READ (NIN,*) M, N
*  IF (M.LE.MMAX .AND. N.LE.NMAX) THEN
  *  Read array of strike/exercise prices, X
  READ (NIN,*) (X(I),I=1,M)
  *  Read array of times to expiry
  READ (NIN,*) (T(I),I=1,N)
  *
  IFAIL = 1
  *
  CALL S30JBF(PUT,M,N,X,S,T,SIGMA,R,LAMBDA,JVOL,P,LDP,DELTA,
  + GAMMA,VEGA,THETA,RHO,VANNA,CHARM,SPEED,COLOUR,
  + ZOMMA,VOMMA,IFAIL)
  *
  IF (IFAIL.EQ.0) THEN
    IF (PUT.EQ.'C' .OR. PUT.EQ.'c') THEN
      WRITE (NOUT,*) 'European Call :'  
    ELSE IF (PUT.EQ.'P' .OR. PUT.EQ.'p') THEN
      WRITE (NOUT,*) 'European Put :'  
    END IF
    *
    WRITE (NOUT,'(A,1X,F8.4)') ' Spot = ', S
    WRITE (NOUT,'(A,1X,F8.4)') ' Volatility = ', SIGMA
    WRITE (NOUT,'(A,1X,F8.4)') ' Rate = ', R
    WRITE (NOUT,'(A,1X,F8.4)') ' Jumps = ', LAMBDA
    WRITE (NOUT,'(A,1X,F8.4)') ' Jump vol = ', JVOL
    *
    WRITE (NOUT,*)
    DO 60 J = 1, N
        WRITE (NOUT,*)
        WRITE (NOUT,*)
        WRITE (NOUT,99999) T(J)
        WRITE (NOUT,*)
        '
        Strike Price Delta Gamma Vega Theta'
        +
        '/ Rho'  
        DO 20 I = 1, M
            WRITE (NOUT,99998) X(I), P(I,J), DELTA(I,J),
            + GAMMA(I,J), VEGA(I,J), THETA(I,J), RHO(I,J)
        20 CONTINUE
    *
    WRITE (NOUT,*)
    + ' Strike Price Vanna Charm Speed Colour Zomma'
    +  
    '/ Vomma'
    DO 40 I = 1, M
        WRITE (NOUT,99997) X(I), P(I,J), VANNA(I,J),
        + CHARM(I,J), SPEED(I,J), COLOUR(I,J), ZOMMA(I,J),
        + VOMMA(I,J)
    40 CONTINUE
  *
  60 CONTINUE
  ELSE
    WRITE (NOUT,*)
    WRITE (NOUT,99996) IFAIL
  END IF
END IF
 *
99999 FORMAT (1X,'Time to Expiry : ',1X,F8.4)
99998 FORMAT (1X,7(F8.4,1X))
99997 FORMAT (1X,8(F8.4,1X))
99996 FORMAT (1X,' ** S30JBF returned with IFAIL = ',I5)
END
9.2 Program Data

S30JBF Example Program Data
'C' : Call = 'C', Put = 'P'
5.0 : LAMBDA (jumps)
100.0 0.25 0.08 0.25 : S, SIGMA, R, JVOL
2 1 : M, N
80.0
90.0 : X(I), I = 1,2,...M
0.5 : T(I), I = 1,2,...N

9.3 Program Results

S30JBF Example Program Results

Merton Jump-Diffusion Model
European Call :
Spot = 100.0000
Volatility = 0.2500
Rate = 0.0800
Jumps = 5.0000
Jump vol = 0.2500

Time to Expiry : 0.5000

<table>
<thead>
<tr>
<th>Strike</th>
<th>Price</th>
<th>Delta</th>
<th>Gamma</th>
<th>Vega</th>
<th>Theta</th>
<th>Rho</th>
</tr>
</thead>
<tbody>
<tr>
<td>80.0000</td>
<td>23.6090</td>
<td>0.9431</td>
<td>0.0064</td>
<td>8.1206</td>
<td>-7.6718</td>
<td>35.3480</td>
</tr>
<tr>
<td>90.0000</td>
<td>15.4193</td>
<td>0.8203</td>
<td>0.0149</td>
<td>18.5256</td>
<td>-9.9695</td>
<td>33.3037</td>
</tr>
</tbody>
</table>

Strike Price Vanna Charm Speed Colour Zomma Vomma
80.0000 23.6090 -0.6334 0.1080 -0.0006 -0.0035 0.0315 70.6824
90.0000 15.4193 -0.7726 0.0770 -0.0009 0.0109 -0.0186 49.7161

__________________________________________________________________________________________________________________________