NAG Library Function Document

nag_heston_price (s30nac)

1 Purpose

nag_heston_price (s30nac) computes the European option price given by Heston's stochastic volatility model.

2 Specification

```
#include <nag.h>
#include <nags.h>
```

```
void nag_heston_price (Nag_OrderType order, Nag_CallPut option, Integer m,
     Integer n, const double x[], double s, const double t[], double sigmav,
     double kappa, double corr, double var0, double eta, double grisk,
     double r, double q, double p[], NagError *fail)
```

3 Description

nag_heston_price (s30nac) computes the price of a European option using Heston's stochastic volatility model. The return on the asset price, S, is

$$\frac{dS}{S} = (r - q)dt + \sqrt{v_t}dW_t^{(1)}$$

and the instantaneous variance, v_t, is defined by a mean-reverting square root stochastic process,

$$dv_t = \kappa(\eta - v_t)dt + \sigma_v\sqrt{v_t}dW_t^{(2)},$$

where r is the risk free annual interest rate; q is the annual dividend rate; v_t is the variance of the asset price; σ_v is the volatility of the volatility, $\sqrt{v_t}$; κ is the mean reversion rate; η is the long term variance. $dW_t^{(i)}$, for $i = 1, 2$, denotes two correlated standard Brownian motions with

$$\mathbb{C}\mathrm{ov}\left[dW_t^{(1)}, dW_t^{(2)}\right] = \rho dt.$$

The option price is computed by evaluating the integral transform given by Lewis (2000) using the form of the characteristic function discussed by Albrecher *et al.* (2007), see also Kilin (2006).

$$P_{\text{call}} = Se^{-qT} - Xe^{-rT}\frac{1}{\pi}\mathrm{Re}\left[\int_{0+i/2}^{\infty+i/2} e^{-ik\bar{X}}\frac{\hat{H}(k,v,T)}{k^2 - ik}\mathrm{d}k\right], \tag{1}$$

where $\bar{X} = \ln(S/X) + (r - q)T$ and

$$\hat{H}(k,v,T) = \exp\left(\frac{2\kappa\eta}{\sigma_v^2}\left[t\mathrm{gendgroup} - \ln\left(\frac{1 - \mathrm{he}^{-\xi\mathrm{t}}}{1 - \mathrm{h}}\right)\right] + \mathrm{v_t g}\left[\frac{1 - \mathrm{e}^{-\xi\mathrm{t}}}{1 - \mathrm{he}^{-\xi\mathrm{t}}}\right]\right),$$

$$g = \frac{1}{2}(b - \xi), \quad h = \frac{b - \xi}{b + \xi}, \quad t = \sigma_v^2 T/2,$$

$$\xi = \left[b^2 + 4\frac{k^2 - ik}{\sigma_v^2}\right]^{\frac{1}{2}},$$

$$b = \frac{2}{\sigma_v^2}\left[(1 - \gamma + ik)\rho\sigma_v + \sqrt{\kappa^2 - \gamma(1 - \gamma)\sigma_v^2}\right]$$

with $t = \sigma_v^2 T/2$. Here γ is the risk aversion parameter of the representative agent with $0 \leq \gamma \leq 1$ and $\gamma(1-\gamma)\sigma_v^2 \leq \kappa^2$. The value $\gamma = 1$ corresponds to $\lambda = 0$, where λ is the market price of risk in Heston (1993) (see Lewis (2000) and Rouah and Vainberg (2007)).

The price of a put option is obtained by put-call parity.

The option price $P_{ij} = P(X = X_i, T = T_j)$ is computed for each strike price in a set X_i, $i = 1, 2, \ldots, m$, and for each expiry time in a set T_j, $j = 1, 2, \ldots, n$.

4 References

Albrecher H, Mayer P, Schoutens W and Tistaert J (2007) The little Heston trap *Wilmott Magazine* **January 2007** 83–92

Heston S (1993) A closed-form solution for options with stochastic volatility with applications to bond and currency options *Review of Financial Studies* **6** 327–343

Kilin F (2006) Accelerating the calibration of stochastic volatility models *MPRA Paper No. 2975* http://mpra.ub.uni-muenchen.de/2975/

Lewis A L (2000) Option valuation under stochastic volatility *Finance Press, USA*

Rouah F D and Vainberg G (2007) *Option Pricing Models and Volatility using Excel-VBA* John Wiley and Sons, Inc

5 Arguments

1: **order** – Nag_OrderType *Input*

On entry: the **order** argument specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = Nag_RowMajor. See Section 3.2.1.3 in the Essential Introduction for a more detailed explanation of the use of this argument.

Constraint: **order** = Nag_RowMajor or Nag_ColMajor.

2: **option** – Nag_CallPut *Input*

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

option = Nag_Call
A call; the holder has a right to buy.

option = Nag_Put
A put; the holder has a right to sell.

Constraint: **option** = Nag_Call or Nag_Put.

3: **m** – Integer *Input*

On entry: the number of strike prices to be used.

Constraint: **m** ≥ 1.

4: **n** – Integer *Input*

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

Constraint: **n** ≥ 1.

5: **x**[**m**] – const double *Input*

On entry: **x**$[i-1]$ must contain X_i, the ith strike price, for $i = 1, 2, \ldots, $**m**.

Constraint: **x**$[i-1] \geq z$ and **x**$[i-1] \leq 1/z$, where z = nag_real_safe_small_number, the safe range parameter, for $i = 1, 2, \ldots, $**m**.

6: **s** – double *Input*

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

Constraint: **s** $\geq z$ and **s** $\leq 1.0/z$, where $z =$ nag_real_safe_small_number, the safe range parameter.

7: **t**[**n**] – const double *Input*

On entry: **t**$[i-1]$ must contain T_i, the ith time, in years, to expiry, for $i = 1, 2, \ldots,$ **n**.

Constraint: **t**$[i-1] \geq z$, where $z =$ nag_real_safe_small_number, the safe range parameter, for $i = 1, 2, \ldots,$ **n**.

8: **sigmav** – double *Input*

On entry: the volatility, σ_v, of the volatility process, $\sqrt{v_t}$. Note that a rate of 20% should be entered as 0.2.

Constraint: **sigmav** > 0.0.

9: **kappa** – double *Input*

On entry: κ, the long term mean reversion rate of the volatility.

Constraint: **kappa** > 0.0.

10: **corr** – double *Input*

On entry: the correlation between the two standard Brownian motions for the asset price and the volatility.

Constraint: $-1.0 \leq$ **corr** ≤ 1.0.

11: **var0** – double *Input*

On entry: the initial value of the variance, v_t, of the asset price.

Constraint: **var0** ≥ 0.0.

12: **eta** – double *Input*

On entry: η, the long term mean of the variance of the asset price.

Constraint: **eta** > 0.0.

13: **grisk** – double *Input*

On entry: the risk aversion parameter, γ, of the representative agent.

Constraint: $0.0 \leq$ **grisk** ≤ 1.0 and **grisk** $\times$ $(1.0 -$ **grisk**$)$ $\times$ **sigmav** $\times$ **sigmav** $\leq$ **kappa** $\times$ **kappa**.

14: **r** – double *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.

15: **q** – double *Input*

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

Constraint: **q** ≥ 0.0.

16: **p**[**m** × **n**] – double *Output*

Note: where **P**(i,j) appears in this document, it refers to the array element

p$[(j-1) \times$ **m** $+ i - 1]$ when **order** = Nag_ColMajor;
p$[(i-1) \times$ **n** $+ j - 1]$ when **order** = Nag_RowMajor.

On exit: **P**(i,j) contains P_{ij}, the option price evaluated for the strike price $\mathbf{x}_i$ at expiry $\mathbf{t}_j$ for $i = 1, 2, \ldots,$ **m** and $j = 1, 2, \ldots,$ **n**.

17: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_CONVERGENCE

Quadrature has not converged to the specified accuracy. However, the result should be a reasonable approximation.

NE_INT

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

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

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

NE_REAL

On entry, **corr** = ⟨*value*⟩.
Constraint: |**corr**| ≤ 1.0.

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

On entry, **grisk** = ⟨*value*⟩, **sigmav** = ⟨*value*⟩ and **kappa** = ⟨*value*⟩.
Constraint: $0.0 \leq$ **grisk** ≤ 1.0 and **grisk** $\times (1.0 -$ **grisk**$) \times$ **sigmav**$^2 \leq$ **kappa**^{2}.

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

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

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

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

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

On entry, **var0** $= \langle value \rangle$.
Constraint: **var0** ≥ 0.0.

NE_REAL_ARRAY

On entry, $\mathbf{t}[\langle value \rangle] = \langle value \rangle$.
Constraint: $\mathbf{t}[i-1] \geq \langle value \rangle$.

On entry, $\mathbf{x}[\langle value \rangle] = \langle value \rangle$.
Constraint: $\mathbf{x}[i-1] \geq \langle value \rangle$ and $\mathbf{x}[i-1] \leq \langle value \rangle$.

7 Accuracy

The accuracy of the output is determined by the accuracy of the numerical quadrature used to evaluate the integral in (1). An adaptive method is used which evaluates the integral to within a tolerance of $\max(10^{-8}, 10^{-10} \times |I|)$, where $|I|$ is the absolute value of the integral.

8 Parallelism and Performance

nag_heston_price (s30nac) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please 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 European call using Heston's stochastic volatility model. The time to expiry is 6 months, the stock price is 100 and the strike price is 100. The risk-free interest rate is 5% per year, the volatility of the variance, σ_v, is 22.5% per year, the mean reversion parameter, κ, is 2.0, the long term mean of the variance, η, is 0.01 and the correlation between the volatility process and the stock price process, ρ, is 0.0. The risk aversion parameter, γ, is 1.0 and the initial value of the variance, **var0**, is 0.01.

10.1 Program Text

```
/* nag_heston_price (s30nac) Example Program.
 *
 * Copyright 2009, Numerical Algorithms Group.
 *
 * Mark 9, 2009.
 */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
  /* Integer scalar and array declarations */
  Integer        exit_status = 0;
  Integer        i, j, m, n;
  NagError       fail;
  Nag_CallPut    putnum;
  /* Double scalar and array declarations */
  double         corr, eta, gamma, kappa, q, r, s, sigmav, var0;
  double         *p = 0, *t = 0, *x = 0;
```

```
/* Character scalar and array declarations */
char         put[8+1];
Nag_OrderType order;

INIT_FAIL(fail);

printf("nag_heston_price (s30nac) Example Program Results\n");
printf("Heston's Stochastic volatility Model\n\n");
/* Skip heading in data file */
scanf("%*[^\n] ");
/* Read put */
scanf("%8s%*[^\n] ", put);
/*
 * nag_enum_name_to_value (x04nac).
 * Converts NAG enum member name to value
 */
putnum = (Nag_CallPut) nag_enum_name_to_value(put);
/* Read s, r, q */
scanf("%lf%lf%lf%*[^\n] ", &s, &r, &q);
/* Read kappa,eta,var0,sigmav,corr,gamma */
scanf("%lf%lf%lf%lf%lf%lf%*[^\n] ",
       &kappa, &eta, &var0, &sigmav, &corr, &gamma);
/* Read m, n */
scanf("%ld%ld%*[^\n] ", &m, &n);
  #ifdef NAG_COLUMN_MAJOR
  #define P(I, J) p[(J-1)*m + I-1]
order = Nag_ColMajor;
  #else
  #define P(I, J) p[(I-1)*n + J-1]
order = Nag_RowMajor;
  #endif
if (!(p = NAG_ALLOC(m*n, double)) ||
    !(t = NAG_ALLOC(n, double)) ||
    !(x = NAG_ALLOC(m, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }
/* Read array of strike/exercise prices, X */
for (i = 0; i < m; i++)
  scanf("%lf ", &x[i]);
scanf("%*[^\n] ");
for (i = 0; i < n; i++)
  scanf("%lf ", &t[i]);
scanf("%*[^\n] ");
/*
 * nag_heston_price (s30nac)
 * Heston's model option pricing formula
 */
nag_heston_price(order, putnum, m, n, x, s, t, sigmav, kappa, corr,
                 var0, eta, gamma, r, q, p, &fail);
if (fail.code != NE_NOERROR)
  {
    printf("Error from nag_heston_price (s30nac).\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
  }
if (putnum == Nag_Call)
  printf("%s\n\n", "European Call :");
else if (putnum == Nag_Put)
  printf("%s\n\n", "European Put :");
printf("%s%8.4f\n", "  Spot                   = ", s);
printf("%s%8.4f\n", "  Volatility of vol      = ", sigmav);
printf("%s%8.4f\n", "  Mean reversion         = ", kappa);
printf("%s%8.4f\n", "  Correlation            = ", corr);
printf("%s%8.4f\n", "  Variance               = ", var0);
printf("%s%8.4f\n", "  Mean of variance       = ", eta);
printf("%s%8.4f\n", "  Risk aversion          = ", gamma);
printf("%s%8.4f\n", "  Rate                   = ", r);
```

```
  printf("%s%8.4f\n", "  Dividend               = ", q);
  printf("\n");
  printf("%s\n", "   Strike    Expiry   Option Price");
  for (i = 1; i <= m; i++)
    for (j = 1; j <= n; j++)
      printf("%9.4f %9.4f %11.4f\n", x[i-1], t[j-1], P(i, j));

 END:
  NAG_FREE(p);
  NAG_FREE(t);
  NAG_FREE(x);

  return exit_status;
}
```

10.2 Program Data

```
nag_heston_price (s30nac) Example Program Data
 Nag_Call                 : Nag_Call or Nag_Put
 100.0 0.05 0.0           : s, r, q
 2.0 0.01 0.01 0.225 0.0 1.0 : kappa, eta, var0, sigmav, corr, grisk
 1 1                      : m, n
 100.0                    : X(I), I = 1,2,...n
 0.5                      : T(I), I = 1,2,...m
```

10.3 Program Results

```
nag_heston_price (s30nac) Example Program Results
Heston's Stochastic volatility Model

European Call :

  Spot                    = 100.0000
  Volatility of vol       =   0.2250
  Mean reversion          =   2.0000
  Correlation             =   0.0000
  Variance                =   0.0100
  Mean of variance        =   0.0100
  Risk aversion           =   1.0000
  Rate                    =   0.0500
  Dividend                =   0.0000

   Strike    Expiry   Option Price
 100.0000    0.5000      4.0851
```