NAG Library Function Document

nag_heston_greeks (s30nbc)

1 Purpose
nag_heston_greeks (s30nbc) computes the European option price given by Heston’s stochastic volatility model together with its sensitivities (Greeks).

2 Specification
#include <nag.h>
#include <nags.h>

void nag_heston_greeks (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[], double delta[], double gamma[],
double vega[], double theta[], double rho[], double vanna[],
double charm[], double speed[], double zomma[], double vomma[],
NagError *fail)

3 Description
nag_heston_greeks (s30nbc) computes the price and sensitivities of a European option using Heston’s stochastic volatility model. The return on the asset price, $S_t$, is
\[ dS_t = (r - q)dt + \sqrt{v_t} dW^{(1)}_t \]
and the instantaneous variance, $v_t$, is defined by a mean-reverting square root stochastic process,
\[ dv_t = \kappa(\eta - v_t)dt + \sigma_v \sqrt{v_t} dW^{(2)}_t, \]
where $r$ is the risk free annual interest rate; $q$ is the annual dividend rate; $v_t$ is the variance of the asset price; $\sigma_v$ is the volatility of the volatility; $\sqrt{v_t}$ is the mean reversion rate; $\eta$ is the long term variance. $dW^{(i)}_t$, for $i = 1, 2$, denotes two correlated standard Brownian motions with
\[ \text{Cov}[dW^{(1)}_t, dW^{(2)}_t] = \rho dt. \]

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).

\begin{equation}
P_{\text{call}} = S e^{-rT} - X e^{-rT} \frac{1}{\pi} \text{Re} \left[ \int_{0+i/2}^{\infty+i/2} e^{-ik\hat{H}(k, v, T)} \frac{k^2 - ik}{k^2} dk \right],
\end{equation}

where $X = \ln(S/X) + (r - q)T$ and
\[ \hat{H}(k, v, T) = \exp \left( \frac{2\kappa \eta}{\sigma_v^2} \text{tgendgroup} \ln \left( \frac{1 - \text{he}^{-\xi}t}{1 - \text{he}^{-\xi}t} \right) + \text{vlg} \left[ \frac{1 - \text{e}^{-\xi}t}{1 - \text{e}^{-\xi}t} \right] \right), \]
\[ g = \frac{1}{2} \left( b - \xi \right), \quad h = \frac{b - \xi}{b + \xi}, \quad t = \sigma_v^2 T/2, \]
\[ \xi = \left[ b^2 + \frac{k^2 - ik}{\sigma_v^2} \right]^2. \]
\[ b = \frac{2}{\sigma_v^2} \left( (1 - \gamma + ik) \rho \sigma_e + \sqrt{\kappa^2 - (1 - \gamma) \sigma_v^2} \right) \]

with \( t = \sigma_v^2 T / 2 \). Here \( \gamma \) is the risk aversion parameter of the representative agent with \( 0 \leq \gamma \leq 1 \) and \( (1 - \gamma) \sigma_v^2 \leq \kappa^2 \). The value \( \gamma = 1 \) corresponds to \( \lambda = 0 \), where \( \lambda \) 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.

\[ P_{\text{put}} = P_{\text{call}} + X e^{-rT} - Se^{-qT}. \]

Writing the expression for the price of a call option as

\[ P_{\text{call}} = Se^{-qT} - X e^{-rT} \frac{1}{\pi} \text{Re} \left[ \int_{0+i/2}^{\infty+i/2} I(k, r, S, T, v) dk \right] \]

then the sensitivities or Greeks can be obtained in the following manner,

\[ \Delta = e^{-qT} + X e^{-rT} \frac{1}{\pi} \text{Re} \left[ \int_{0+i/2}^{\infty+i/2} (ik) I(k, r, S, T, v) dk \right], \]

\[ \text{Vega} = -X e^{-rT} \frac{1}{\pi} \text{Re} \left[ \int_{0-i/2}^{0+i/2} f_2 I(k, r, j, S, T, v) dk \right], \quad \text{where } f_2 = g \left[ \frac{1 - e^{-\alpha t}}{1 - h e^{-\alpha t}} \right], \]

\[ \text{Rho} = TX e^{-rT} \frac{1}{\pi} \text{Re} \left[ \int_{0+i/2}^{\infty+i/2} (1 + ik) I(k, r, S, T, v) dk \right]. \]

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 \( T_j, j = 1, 2, \ldots, n \).

### References


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


### 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: \textbf{option} – Nag_CallPut \hspace{1cm} \textit{Input}

\textit{On entry:} determines whether the option is a call or a put.

- \textbf{option} = Nag_Call
  A call; the holder has a right to buy.

- \textbf{option} = Nag_Put
  A put; the holder has a right to sell.

\textit{Constraint:} \textbf{option} = Nag_Call or Nag_Put.

3: \textbf{m} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} the number of strike prices to be used.

\textit{Constraint:} \textbf{m} \geq 1.

4: \textbf{n} – Integer \hspace{1cm} \textit{Input}

\textit{On entry:} the number of times to expiry to be used.

\textit{Constraint:} \textbf{n} \geq 1.

5: \textbf{x[m]} – const double \hspace{1cm} \textit{Input}

\textit{On entry:} \textbf{x}[i-1] must contain \(X_i\), the \(i\)th strike price, for \(i = 1,2,\ldots,\textbf{m}\).

\textit{Constraint:} \textbf{x}[i-1] \geq z \text{ and } \textbf{x}[i-1] \leq 1/z, where z = nag_real_safe_small_number, the safe range parameter, for \(i = 1,2,\ldots,\textbf{m}\).

6: \textbf{s} – double \hspace{1cm} \textit{Input}

\textit{On entry:} \(S\), the price of the underlying asset.

\textit{Constraint:} \textbf{s} \geq z \text{ and } \textbf{s} \leq 1.0/z, where z = nag_real_safe_small_number, the safe range parameter.

7: \textbf{t[n]} – const double \hspace{1cm} \textit{Input}

\textit{On entry:} \textbf{t}[i-1] must contain \(T_i\), the \(i\)th time, in years, to expiry, for \(i = 1,2,\ldots,\textbf{n}\).

\textit{Constraint:} \textbf{t}[i-1] \geq z, where z = nag_real_safe_small_number, the safe range parameter, for \(i = 1,2,\ldots,\textbf{n}\).

8: \textbf{sigmav} – double \hspace{1cm} \textit{Input}

\textit{On entry:} the volatility, \(\sigma_v\), of the volatility process, \(\sqrt{\nu_t}\). Note that a rate of 20\% should be entered as 0.2.

\textit{Constraint:} \textbf{sigmav} > 0.0.

9: \textbf{kappa} – double \hspace{1cm} \textit{Input}

\textit{On entry:} \(\kappa\), the long term mean reversion rate of the volatility.

\textit{Constraint:} \textbf{kappa} > 0.0.

10: \textbf{corr} – double \hspace{1cm} \textit{Input}

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

\textit{Constraint:} \(-1.0 \leq \textbf{corr} \leq 1.0\).

11: \textbf{var0} – double \hspace{1cm} \textit{Input}

\textit{On entry:} the initial value of the variance, \(\nu_t\), of the asset price.

\textit{Constraint:} \textbf{var0} \geq 0.0.
eta – double

On entry: $\eta$, the long term mean of the variance of the asset price.

Constraint: $\eta > 0.0$.

grisk – double

On entry: $\gamma$, the risk aversion parameter, $\gamma$, of the representative agent.

Constraint: $0.0 \leq grisk \leq 1.0$ and $grisk \times (1 - grisk) \times sigmav \times sigmav \leq kappa \times kappa$.

r – double

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

Constraint: $r \geq 0.0$.

q – double

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

Constraint: $q \geq 0.0$.

$p[m \times n]$ – double

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 $x_i$ at expiry $t_j$ for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

delta[m \times n] – double

Note: the $(i, j)$th element of the matrix is stored in

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

On exit: the $m \times n$ array delta contains the sensitivity, $\frac{\partial P}{\partial S}$, of the option price to change in the price of the underlying asset.

gamma[m \times n] – double

Note: the $(i, j)$th element of the matrix is stored in

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

On exit: the $m \times n$ array gamma contains the sensitivity, $\frac{\partial^2 P}{\partial S^2}$, of delta to change in the price of the underlying asset.

vega[m \times n] – double

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

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

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 $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. 
20: \[ \theta[m \times n] \rightarrow \text{double} \]

**Note:** where \( \text{THETA}(i, j) \) appears in this document, it refers to the array element

\[ \theta[(j-1) \times m + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor}; \]
\[ \theta[(i-1) \times n + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}. \]

*On exit:* \( \text{THETA}(i, j) \), 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 \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \), where \( b = r - q \).

21: \[ \rho[m \times n] \rightarrow \text{double} \]

**Note:** where \( \text{RHO}(i, j) \) appears in this document, it refers to the array element

\[ \rho[(j-1) \times m + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor}; \]
\[ \rho[(i-1) \times n + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}. \]

*On exit:* \( \text{RHO}(i, j) \), 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 \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).

22: \[ \vanna[m \times n] \rightarrow \text{double} \]

**Note:** where \( \text{VANNA}(i, j) \) appears in this document, it refers to the array element

\[ \vanna[(j-1) \times m + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor}; \]
\[ \vanna[(i-1) \times n + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}. \]

*On exit:* \( \text{VANNA}(i, j) \), contains the second-order Greek measuring the sensitivity of the first-order Greek \( \Delta_{ij} \) to change in the volatility of the asset price, i.e., \( -\frac{\partial \Delta_{ij}}{\partial \sigma} = -\frac{\partial^2 P_{ij}}{\partial \sigma^2} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).

23: \[ \charm[m \times n] \rightarrow \text{double} \]

**Note:** where \( \text{CHARM}(i, j) \) appears in this document, it refers to the array element

\[ \charm[(j-1) \times m + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor}; \]
\[ \charm[(i-1) \times n + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}. \]

*On exit:* \( \text{CHARM}(i, j) \), contains the second-order Greek measuring the sensitivity of the first-order Greek \( \Delta_{ij} \) to change in the time, i.e., \( \frac{\partial \Delta_{ij}}{\partial t} = -\frac{\partial^2 P_{ij}}{\partial \sigma \partial t} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).

24: \[ \speed[m \times n] \rightarrow \text{double} \]

**Note:** where \( \text{SPEED}(i, j) \) appears in this document, it refers to the array element

\[ \speed[(j-1) \times m + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor}; \]
\[ \speed[(i-1) \times n + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}. \]

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

25: \[ \zomma[m \times n] \rightarrow \text{double} \]

**Note:** where \( \text{ZOMMA}(i, j) \) appears in this document, it refers to the array element

\[ \zomma[(j-1) \times m + i - 1] \text{ when } \text{order} = \text{Nag\_ColMajor}; \]
\[ \zomma[(i-1) \times n + j - 1] \text{ when } \text{order} = \text{Nag\_RowMajor}. \]

*On exit:* \( \text{ZOMMA}(i, j) \), contains the third-order Greek measuring the sensitivity of the second-order Greek \( \Gamma_{ij} \) to change in the volatility of the underlying asset, i.e., \( \frac{\partial \Gamma_{ij}}{\partial \sigma} = -\frac{\partial^2 P_{ij}}{\partial \sigma^2} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).
26: **vomma[\(m \times n\)]** – double 

*Output*

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

\[vomma[(j - 1) \times m + i - 1] \text{ when } order = \text{Nag.ColMajor};\]

\[vomma[(i - 1) \times n + j - 1] \text{ when } order = \text{Nag.RowMajor}.\]

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

27: **fail** – NagError * 

*Input/Output*

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

6 **Error Indicators and Warnings**

**NE_ACCURACY**

Solution cannot be computed accurately. Check values of input arguments.

**NE_ALLOC_FAIL**

Dynamic memory allocation failed.

See Section 3.2.1.2 in the Essential Introduction for further information.

**NE_BAD_PARAM**

On entry, argument \(value\) had an illegal value.

**NE_CONVERGENCE**

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

**NE_INT**

On entry, \(m = \langle value\rangle\).

Constraint: \(m \geq 1\).

On entry, \(n = \langle value\rangle\).

Constraint: \(n \geq 1\).

**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.

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

See Section 3.6.6 in the Essential Introduction for further information.

**NE_NO_LICENCE**

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

See Section 3.6.5 in the Essential Introduction for further information.

**NE_REAL**

On entry, \(corr = \langle value\rangle\).

Constraint: \(|corr| \leq 1.0\).

On entry, \(eta = \langle value\rangle\).

Constraint: \(eta > 0.0\).
On entry, grisk = ⟨value⟩, sigmav = ⟨value⟩ and kappa = ⟨value⟩.
Constraint: 0.0 ≤ grisk ≤ 1.0 and grisk × (1.0 − grisk) × sigmav² ≤ kappa².

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 ≥ ⟨value⟩ and s ≤ ⟨value⟩.

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

On entry, var0 = ⟨value⟩.
Constraint: var0 ≥ 0.0.

NE_REAL_ARRAY
On entry, t[⟨value⟩] = ⟨value⟩.
Constraint: t[i − 1] ≥ ⟨value⟩.

On entry, x[⟨value⟩] = ⟨value⟩.
Constraint: x[i − 1] ≥ ⟨value⟩ and x[i − 1] ≤ ⟨value⟩.

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 × |I|}, where |I| is the absolute value of the integral.

8 Parallelism and Performance
nag_heston_greeks (s30nbc) 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 function. 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 and sensitivities of a European call using Heston’s stochastic volatility model. The time to expiry is 1 year, the stock price is 100 and the strike price is 100. The risk-free interest rate is 2.5% per year, the volatility of the variance, σ_v, is 57.51% per year, the mean reversion parameter, κ, is 1.5768, the long term mean of the variance, η, is 0.0398 and the correlation between the volatility process and the stock price process, ρ, is −0.5711. The risk aversion parameter, γ, is 1.0 and the initial value of the variance, var0, is 0.0175.
10.1 Program Text

/* nag_heston_greeks (s30nbc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 23, 2011. */
*/
#include <nag.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
#ifdef NAG_COLUMN_MAJOR
#define K(I, J) (J-1)*pdp + I-1
#else
#define K(I, J) (I-1)*pdp + J-1
#endif

/* Scalars */
Integer exit_status = 0;
double corr, eta, grisk, kappa, q, r, s, sigmav, var0;
Integer i, j, pdp, m, n;
/* Arrays */
double *charm = 0, *delta = 0, *gamma = 0, *p = 0, *rho = 0,
*speed = 0, *t = 0, *theta = 0, *vanna = 0, *vega = 0,
*vomma = 0, *x = 0, *zomma = 0;
char put[8+1];
/* Nag types */
Nag_OrderType order;
Nag_CallPut putnum;
NagError fail;

INIT_FAIL(fail);

printf("nag_heston_greeks (s30nbc) Example Program Results\n");
/* Skip heading in data file */
#ifdef _WIN32
scanf_s("%*[\n]");
#else
scanf("%*[\n]");
#endif
/* Read put */
#ifdef _WIN32
scanf_s("%8s%*[\n]", put, _countof(put));
#else
scanf("%8s%*[\n]", put);
#endif

/* 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 */
#ifdef _WIN32
scanf_s("%lf%lf%lf%*[\n] ", &s, &r, &q);
#else
scanf("%lf%lf%lf%*[\n] ", &s, &r, &q);
#endif
/* Read kappa, eta, var0, sigmav, corr, grisk */
#ifdef _WIN32
scanf_s("%lf%lf%lf%*[\n] ", &kappa, &eta, &var0);
#else
scanf("%lf%lf%lf%*[\n] ", &kappa, &eta, &var0);
#endif
#ifdef _WIN32
scanf_s("%lf%lf%lf%*[\n] ", &sigmav, &corr, &grisk);
#else
scanf("%lf%lf%lf%*[\n] ", &sigmav, &corr, &grisk);
*/
/* Read m, n */
#endif

/* Read m, n */
#ifdef _WIN32
    scanf_s("%"NAG_IFMT"%"NAG_IFMT"%*[\n]", &m, &n);
#else
    scanf("%"NAG_IFMT"%"NAG_IFMT"%*[\n]", &m, &n);
#endif

if (!(charm = NAG_ALLOC(m*n, double)) ||
    !(delta = NAG_ALLOC(m*n, double)) ||
    !(gamma = NAG_ALLOC(m*n, double)) ||
    !(p = NAG_ALLOC(m*n, double)) ||
    !(rho = NAG_ALLOC(m*n, double)) ||
    !(speed = NAG_ALLOC(m*n, double)) ||
    !(t = NAG_ALLOC((n), double)) ||
    !(theta = NAG_ALLOC(m*n, double)) ||
    !(vanna = NAG_ALLOC(m*n, double)) ||
    !(vega = NAG_ALLOC(m*n, double)) ||
    !(vomma = NAG_ALLOC(m*n, double)) ||
    !(x = NAG_ALLOC((m), double)) ||
    !(zomma = NAG_ALLOC(m*n, double)) )
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}
#ifdef NAG_COLUMN_MAJOR
    order = Nag_ColMajor;
pdp = m;
#else
    order = Nag_RowMajor;
pdp = n;
#endif

for (i = 0; i < m; i++)
#ifdef _WIN32
    scanf_s("%lf", &x[i]);
#else
    scanf("%lf", &x[i]);
#endif

for (i = 0; i < n; i++)
#ifdef _WIN32
    scanf_s("%lf", &t[i]);
#else
    scanf("%lf", &t[i]);
#endif

/* nag_heston_greeks (s30nbc).  
 Heston's model option pricing formula with Greeks  */
    nag_heston_greeks(order, putnum, m, n, x, s, t, sigmav, kappa, corr, var0,
            eta, grisk, r, q, p, delta, gamma, vega, theta, rho, vanna,
            charm, speed, zomma, vomma, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_heston_greeks (s30nbc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
printf("Heston's Stochastic volatility Model
");
switch (putnum)
{
    case Nag_Call:
        printf("European Call :
")
        break;
    case Nag_Put:
        printf("European Put :
")
}
printf(" Spot = %10.4f
", s);
printf(" Volatility of vol = %10.4f
", sigmav);
printf(" Mean reversion = %10.4f
", kappa);
printf(" Correlation = %10.4f
", corr);
printf(" Mean of variance = %10.4f
", var0);
printf(" Risk aversion = %10.4f
", grisk);
printf(" Rate = %10.4f
", r);
printf(" Dividend = %10.4f

", q);
for (j = 1; j <= n; j++)
{
    printf("Time to Expiry : %8.4f
", t[j-1]);
    for (i = 1; i <= m; i++)
        printf("%10.4f %10.4f %10.4f %10.4f %10.4f %10.4f %10.4f
", x[i-1],
            p[K(i, j)], delta[K(i, j)], gamma[K(i, j)], vega[K(i, j)],
            theta[K(i, j)], rho[K(i, j)]);
    printf("Vanna", "Charm", "Speed", "Zomma", "Vomma");
    for (i = 1; i <= m; i++)
        printf("%21s %10.4f %10.4f %10.4f %10.4f %10.4f
", "", vanna[K(i, j)],
            charm[K(i, j)], speed[K(i, j)], zomma[K(i, j)], vomma[K(i, j)]);
}
END:
NAG_FREE(charm);
NAG_FREE(delta);
NAG_FREE(gamma);
NAG_FREE(p);
NAG_FREE(rho);
NAG_FREE(speed);
NAG_FREE(t);
NAG_FREE(theta);
NAG_FREE(vanna);
NAG_FREE(vega);
NAG_FREE(vomma);
NAG_FREE(x);
NAG_FREE(zomma);
return exit_status;

10.2 Program Data

nag_heston_greeks (s30nbc) Example Program Data
Nag_Call : CallPut option
    100.0 0.025 0.0 : s, r, q
    1.5768 0.0398 0.0175 : kappa, eta, var0
    0.5751 -0.5711 1.0 : sigmav, corr, grisk
    1 : m, n
    100.0 : x[i], i = 0,...,n-1
    1.0 : t[i], i = 0,...,m-1
# 10.3 Program Results

**nag_heston_greeks (s30nbc) Example Program Results**

Heston’s Stochastic volatility Model

**European Call:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spot</td>
<td>100.0000</td>
</tr>
<tr>
<td>Volatility of vol</td>
<td>0.5751</td>
</tr>
<tr>
<td>Mean reversion</td>
<td>1.5768</td>
</tr>
<tr>
<td>Correlation</td>
<td>-0.5711</td>
</tr>
<tr>
<td>Variance</td>
<td>0.0175</td>
</tr>
<tr>
<td>Mean of variance</td>
<td>0.0398</td>
</tr>
<tr>
<td>Risk aversion</td>
<td>1.0000</td>
</tr>
<tr>
<td>Rate</td>
<td>0.0250</td>
</tr>
<tr>
<td>Dividend</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

**Time to Expiry:** 1.0000

<table>
<thead>
<tr>
<th>Strike</th>
<th>Price</th>
<th>Delta</th>
<th>Gamma</th>
<th>Vega</th>
<th>Theta</th>
<th>Rho</th>
<th>Vanna</th>
<th>Charm</th>
<th>Speed</th>
<th>Zomma</th>
<th>Vomma</th>
<th>Vomma</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.0000</td>
<td>7.2743</td>
<td>0.6945</td>
<td>0.0251</td>
<td>52.5461</td>
<td>-4.9969</td>
<td>62.1735</td>
<td>-0.5643</td>
<td>-0.0321</td>
<td>-0.0023</td>
<td>-0.1976</td>
<td>-321.0780</td>
<td></td>
</tr>
</tbody>
</table>