NAG Library Function Document

nag_asian_geom_greeks (s30sbc)

1 Purpose

nag_asian_geom_greeks (s30sbc) computes the Asian geometric continuous average-rate option price together with its sensitivities (Greeks).

2 Specification

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

void nag_asian_geom_greeks (Nag_OrderType order, Nag_CallPut option,
                           Integer m, Integer n, const double x[], double s, const double t[],
                           double sigma, double r, double b, double p[], double delta[],
                           double gamma[], double vega[], double theta[], double rho[],
                           double crho[], double vanna[], double charm[], double speed[],
                           double colour[], double zomma[], double vomma[], NagError *fail)
```

3 Description

nag_asian_geom_greeks (s30sbc) computes the price of an Asian geometric continuous average-rate 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. The annual volatility, \( \sigma \), risk-free rate, \( r \), and cost of carry, \( b \), are constants (see Kemna and Vorst (1990)). For a given strike price, \( X \), the price of a call option with underlying price, \( S \), and time to expiry, \( T \), is

\[
P_{\text{call}} = S e^{(r-b)T} \Phi(\bar{d}_1) - X e^{-rT} \Phi(\bar{d}_2),
\]

and the corresponding put option price is

\[
P_{\text{put}} = X e^{-rT} \Phi(-\bar{d}_2) - S e^{(r-b)T} \Phi(-\bar{d}_1),
\]

where

\[
\bar{d}_1 = \frac{\ln(S/X) + (\bar{b} + \bar{\sigma}^2/2)T}{\bar{\sigma}\sqrt{T}}
\]

and

\[
\bar{d}_2 = \bar{d}_1 - \bar{\sigma}\sqrt{T},
\]

with

\[
\bar{\sigma} = \frac{\sigma}{\sqrt{3}}, \quad \bar{b} = \frac{1}{2} \left( b - \frac{\sigma^2}{6} \right).
\]

\( \Phi \) is the cumulative Normal distribution function,

\[
\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-y^2/2)dy.
\]

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

Arguments
1: order – Nag_OrderType
   On entry: the order argument specifies the two-dimensional storage scheme being used, i.e., row-major order or column-major order. 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
   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
   On entry: the number of strike prices to be used.
   Constraint: m ≥ 1.

4: n – Integer
   On entry: the number of times to expiry to be used.
   Constraint: n ≥ 1.

5: x[m] – const double
   On entry: x[i – 1] must contain X_i, the ith strike price, for i = 1,2,…,m.
   Constraint: x[i – 1] ≥ z and x[i – 1] ≤ 1/z, where z = nag_real_safe_small_number, the safe range parameter, for i = 1,2,…,m.

6: s – double
   On entry: S, the price of the underlying asset.
   Constraint: s ≥ z and s ≤ 1.0/z, where z = nag_real_safe_small_number, the safe range parameter.

7: t[n] – const double
   On entry: t[i – 1] must contain T_i, the ith time, in years, to expiry, for i = 1,2,…,n.
   Constraint: t[i – 1] ≥ z, where z = nag_real_safe_small_number, the safe range parameter, for i = 1,2,…,n.

8: sigma – double
   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.
9:  \( r \) – 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*: \( r \geq 0.0 \).

10:  \( b \) – double

*Input*

On entry: \( b \), the annual cost of carry rate. Note that a rate of 8% should be entered as 0.08.

11:  \( p[m \times n] \) – double

*Output*

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

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

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

12:  \( \text{delta}[m \times n] \) – double

*Output*

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

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

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

13:  \( \text{gamma}[m \times n] \) – double

*Output*

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

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

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

14:  \( \text{vega}[m \times n] \) – double

*Output*

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

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

On exit: \( \text{VEGA}(i, j) \), 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., \( \frac{\partial P}{\partial \sigma} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).

15:  \( \text{theta}[m \times n] \) – double

*Output*

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

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

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., \( -\frac{\partial P}{\partial t} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \), where \( b = r - q \).
16: \texttt{rho}[^m \times n] \quad \text{double} \quad \text{Output}

Note: where \texttt{RHO}(i,j) appears in this document, it refers to the array element
\[
\text{rho}[(j-1) \times m + i - 1] \quad \text{when order} = \text{Nag\_ColMajor};
\]
\[
\text{rho}[(i-1) \times n + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]

On exit: \texttt{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 \(j = 1,2,\ldots,n\).

17: \texttt{crho}[^m \times n] \quad \text{double} \quad \text{Output}

Note: the \((i,j)\)th element of the matrix is stored in
\[
\text{crho}[(j-1) \times m + i - 1] \quad \text{when order} = \text{Nag\_ColMajor};
\]
\[
\text{crho}[(i-1) \times n + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]

On exit: \texttt{DELTA}(i,j), contains the first-order Greek measuring the sensitivity of the option price \(P_{ij}\) to change in the price of the underlying asset, i.e., \(-\frac{\partial P_{ij}}{\partial S}\), for \(i = 1,2,\ldots,m\) and \(j = 1,2,\ldots,n\).

18: \texttt{vanna}[^m \times n] \quad \text{double} \quad \text{Output}

Note: where \texttt{VANNA}(i,j) appears in this document, it refers to the array element
\[
\text{vanna}[(j-1) \times m + i - 1] \quad \text{when order} = \text{Nag\_ColMajor};
\]
\[
\text{vanna}[(i-1) \times n + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]

On exit: \texttt{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., \(-\frac{\partial \Delta_{ij}}{\partial \sigma}\) \(\frac{\partial P_{ij}}{\partial S}\), for \(i = 1,2,\ldots,m\) and \(j = 1,2,\ldots,n\).

19: \texttt{charm}[^m \times n] \quad \text{double} \quad \text{Output}

Note: where \texttt{CHARM}(i,j) appears in this document, it refers to the array element
\[
\text{charm}[(j-1) \times m + i - 1] \quad \text{when order} = \text{Nag\_ColMajor};
\]
\[
\text{charm}[(i-1) \times n + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]

On exit: \texttt{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., \(-\frac{\partial \Delta_{ij}}{\partial t} = \frac{\partial^2 P_{ij}}{\partial S \partial t}\), for \(i = 1,2,\ldots,m\) and \(j = 1,2,\ldots,n\).

20: \texttt{speed}[^m \times n] \quad \text{double} \quad \text{Output}

Note: where \texttt{SPEED}(i,j) appears in this document, it refers to the array element
\[
\text{speed}[(j-1) \times m + i - 1] \quad \text{when order} = \text{Nag\_ColMajor};
\]
\[
\text{speed}[(i-1) \times n + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]

On exit: \texttt{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., \(-\frac{\partial \Gamma_{ij}}{\partial S} = -\frac{\partial^3 P_{ij}}{\partial S^2 \partial t}\), for \(i = 1,2,\ldots,m\) and \(j = 1,2,\ldots,n\).

21: \texttt{colour}[^m \times n] \quad \text{double} \quad \text{Output}

Note: where \texttt{COLOUR}(i,j) appears in this document, it refers to the array element
\[
\text{colour}[(j-1) \times m + i - 1] \quad \text{when order} = \text{Nag\_ColMajor};
\]
\[
\text{colour}[(i-1) \times n + j - 1] \quad \text{when order} = \text{Nag\_RowMajor}.
\]

On exit: \texttt{COLOUR}(i,j), contains the third-order Greek measuring the sensitivity of the second-order Greek \(\Gamma_{ij}\) to change in the time, i.e., \(-\frac{\partial \Gamma_{ij}}{\partial t} = -\frac{\partial^3 P_{ij}}{\partial S^2 \partial t}\), for \(i = 1,2,\ldots,m\) and \(j = 1,2,\ldots,n\).
22: \texttt{zomma[m x n]} – double

\textit{Output}

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

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

\textit{On exit:} \texttt{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., \( \frac{\partial \Gamma_{ij}}{\partial \sigma} = \frac{-\partial^2 V_{ij}}{\partial \sigma^2} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).

23: \texttt{vomma[m x n]} – double

\textit{Output}

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

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

\textit{On exit:} \texttt{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., \( \frac{\partial \Delta_{ij}}{\partial \sigma} = \frac{-\partial^2 \Delta_{ij}}{\partial \sigma^2} \), for \( i = 1, 2, \ldots, m \) and \( j = 1, 2, \ldots, n \).

24: \texttt{fail} – NagError *

\textit{Input/Output}

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

6 \textbf{Error Indicators and Warnings}

\textbf{NE\_ALLOC\_FAIL}

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

\textbf{NE\_BAD\_PARAM}

On entry, argument \langle value \rangle had an illegal value.

\textbf{NE\_INT}

On entry, \( m = \langle value \rangle \).
Constraint: \( m \geq 1 \).

On entry, \( n = \langle value \rangle \).
Constraint: \( n \geq 1 \).

\textbf{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.

\textbf{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.

\textbf{NE\_REAL}

On entry, \( r = \langle value \rangle \).
Constraint: \( r \geq 0.0 \).

On entry, \( s = \langle value \rangle \).
Constraint: \( s \geq \langle value \rangle \) and \( s \leq \langle value \rangle \).
On entry, $\sigma = \langle\text{value}\rangle$.
Constraint: $\sigma > 0.0$.

**NE_REAL_ARRAY**

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

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

7 **Accuracy**

The accuracy of the output is dependent on the accuracy of the cumulative Normal distribution function, $\Phi$. 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 nag_cumul_normal (s15abc) and nag_erfc (s15adc)). An accuracy close to **machine precision** can generally be expected.

8 **Parallelism and Performance**

nag_asian_geom_greeks (s30sbc) 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 of an Asian geometric continuous average-rate call with a time to expiry of 3 months, a stock price of 80 and a strike price of 97. The risk-free interest rate is 5% per year, the cost of carry is 8% and the volatility is 20% per year.

10.1 **Program Text**

```c
/* nag_asian_geom_greeks (s30sbc) Example Program. *
 * Copyright 2014 Numerical Algorithms Group. *
 * Mark 9, 2009. */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <nag.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 b, r, s, sigma;
    double *charm = 0, *colour = 0, *crho = 0, *delta = 0, *gamma = 0;
    double *p = 0, *rho = 0, *speed = 0, *t = 0, *theta = 0, *vanna = 0;
```
double *vega = 0, *vomma = 0, *x = 0, *zomma = 0;
/* Character scalar and array declarations */
char put[8+1];
Nag_OrderType order;

INIT_FAIL(fail);

printf("nag_asian_geom_greeks (s30sbc) Example Program Results\n");
printf("Asian Option: Geometric Continuous Average-Rate\n");

/* Skip heading in data file */
#if _WIN32
scanf_s("%*[\n]");
#else
scanf("%*[\n]");
#endif
/* Read put */
#if _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, sigma, r, b */
#if _WIN32
scanf_s("%lf%lf%lf%lf%*[\n] ", &s, &sigma, &r, &b);
#else
scanf("%lf%lf%lf%lf%*[\n] ", &s, &sigma, &r, &b);
#endif
/* Read m, n */
#if _WIN32
scanf_s("%"NAG_IFMT"%"NAG_IFMT"%*[\n] ", &m, &n);
#else
scanf("%"NAG_IFMT"%"NAG_IFMT"%*[\n] ", &m, &n);
#endif
#ifdef NAG_COLUMN_MAJOR
#define CHARM(I, J) charm[(J-1)*m + I-1]
#define COLOUR(I, J) colour[(J-1)*m + I-1]
#define CRHO(I, J) crho[(J-1)*m + I-1]
#define DELTA(I, J) delta[(J-1)*m + I-1]
#define GAMMA(I, J) gamma[(J-1)*m + I-1]
#define P(I, J) p[(J-1)*m + I-1]
#define RHO(I, J) rho[(J-1)*m + I-1]
#define SPEED(I, J) speed[(J-1)*m + I-1]
#define THETA(I, J) theta[(J-1)*m + I-1]
#define VANNA(I, J) vanna[(J-1)*m + I-1]
#define VEGA(I, J) vega[(J-1)*m + I-1]
#define VOMMA(I, J) vomma[(J-1)*m + I-1]
#define ZOMMA(I, J) zomma[(J-1)*m + I-1]
#define order = Nag_ColMajor;
#else
#define CHARM(I, J) charm[(I-1)*n + J-1]
#define COLOUR(I, J) colour[(I-1)*n + J-1]
#define CRHO(I, J) crho[(I-1)*n + J-1]
#define DELTA(I, J) delta[(I-1)*n + J-1]
#define GAMMA(I, J) gamma[(I-1)*n + J-1]
#define P(I, J) p[(I-1)*n + J-1]
#define RHO(I, J) rho[(I-1)*n + J-1]
#define SPEED(I, J) speed[(I-1)*n + J-1]
#define THETA(I, J) theta[(I-1)*n + J-1]
#define VANNA(I, J) vanna[(I-1)*n + J-1]
#define VEGA(I, J) vega[(I-1)*n + J-1]
#define VOMMA(I, J) vomma[(I-1)*n + J-1]
#define ZOMMA(I, J) zomma[(I-1)*n + J-1]
#define order = Nag_RowMajor;
#endif
if (!(charm = NAG_ALLOC(m*n, double)) ||
! (colour = NAG_ALLOC(m*n, double)) ||

Mark 25 s30sbc
! (crho = 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;
}

/* Read array of strike/exercise prices, X */
for (i = 0; i < m; i++)
#ifdef _WIN32
 scanf_s("%lf ", &x[i]);
#else
 scanf("%lf ", &x[i]);
#endif
#ifdef _WIN32
 scanf_s("%*[\n] ");
#else
 scanf("%*[\n] ");
#endif

/* Read array of times to expiry */
for (i = 0; i < n; i++)
#ifdef _WIN32
 scanf_s("%lf ", &t[i]);
#else
 scanf("%lf ", &t[i]);
#endif
#ifdef _WIN32
 scanf_s("%*[\n] ");
#else
 scanf("%*[\n] ");
#endif

/* nag_asian_geom_greeks (s30sbc) */
/* Asian option: geometric continuous average rate pricing formula */
/* with Greeks */

nag_asian_geom_greeks(order, putnum, m, n, x, s, t, sigma, r, b, p,
 delta, gamma, vega, theta, rho, crho, vanna,
 charm, speed, colour, zomma, vomma, &fail);
if (fail.code != NE_NOERROR)
{
 printf("Error from nag_asian_geom_greeks (s30sbc).\n", fail.message);
 exit_status = 1;
 goto END;
}

if (putnum == Nag_Call)
 printf("\n Asian Call :");
else if (putnum == Nag_Put)
 printf("\n Asian Put :");
 printf("\n Spot = %.8f\n", s);
 printf("\n Volatility = %.8f\n", sigma);
 printf("\n Rate = %.8f\n", r);
 printf("\n Cost of carry = %.8f\n", b);
 printf("\n\n");
for (j = 1; j <= n; j++)
{
 printf("\n Time to Expiry : %.8f\n", t[j-1]);
 printf("\n Strike Price Delta Gamma Vega \n" "Theta Rho CRho\n");
for (i = 1; i <= m; i++)
    printf("%8.4f %8.4f %8.4f %8.4f %8.4f %8.4f %8.4f %8.4f
", 
    x[i-1], P(i, j), DELTA(i, j), GAMMA(i, j), VEGA(i, j), 
    THETA(i, j), RHO(i, j), CRHO(i, j));

printf("\n\nVanna Charm Speed ",
    "Colour Zomma Vomma\n\n";)
for (i = 1; i <= m; i++)
    printf("%26.4f %8.4f %8.4f %8.4f %8.4f %8.4f\n", 
    VANNA(i, j), 
    CHARM(i, j), SPEED(i, j), COLOUR(i, j), ZOMMA(i, j), 
    VOMMA(i, j));

END:
NAG_FREE(charm);
NAG_FREE(colour);
NAG_FREE(crho);
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(x);
NAG_FREE(zomma);

return exit_status;
}

10.2 Program Data
nag_asian_geom_greeks (s30sbc) Example Program Data
Nag_Call : Nag_Call or Nag_Put
80.0 0.2 0.05 0.08 : s, sigma, r, b
1 1 : m, n
97.0 : X(I), I = 1,2,...m
0.25 : T(I), I = 1,2,...n

10.3 Program Results
nag_asian_geom_greeks (s30sbc) Example Program Results
Asian Option: Geometric Continuous Average-Rate

Asian Call :

\begin{tabular}{cccccccc}
\textbf{Spot} & \textbf{Volatility} & \textbf{Rate} & \textbf{Cost of carry} \\
80.0000 & 0.2000 & 0.0500 & 0.0800 \\
\end{tabular}

\begin{tabular}{cccccccc}
\textbf{Time to Expiry} : & 0.2500 \\
\textbf{Strike Price} & Delta & Gamma & Vega & Theta & Rho & CRho \\
97.0000 & 0.0010 & 0.0008 & 0.0006 & 0.0638 & -0.0281 & 0.0079 & 0.0081 \\
\textbf{Vanna Charm Speed Colour Zomma Vomma} \\
0.0443 & -0.0196 & 0.0004 & -0.0122 & 0.0272 & 3.1893 \\
\end{tabular}