

NAG C Library Function Document

nag_pde_bs_1d_means (d03nec)

1 Purpose

nag_pde_bs_1d_means (d03nec) computes average values of a continuous function of time over the remaining life of an option. It is used together with nag_pde_bs_1d_analytic (d03ndc) to value options with time-dependent parameters.

2 Specification

```
void nag_pde_bs_1d_means (double t0, double tmat, Integer ntd, const double td[],
    const double phid[], double phiav[], NagError *fail)
```

3 Description

nag_pde_bs_1d_means (d03nec) computes the quantities

$$\phi(t_0), \quad \hat{\phi} = \frac{1}{T-t_0} \int_{t_0}^T \phi(\zeta) d\zeta, \quad \bar{\phi} = \left(\frac{1}{T-t_0} \int_{t_0}^T \phi^2(\zeta) d\zeta \right)^{1/2}$$

from a given set of values **phid** of a continuous time-dependent function $\phi(t)$ at a set of discrete points **td**, where t_0 is the current time and T is the maturity time. Thus $\hat{\phi}$ and $\bar{\phi}$ are first and second order averages of ϕ over the remaining life of an option.

The function may be used in conjunction with nag_pde_bs_1d_analytic (d03ndc) in order to value an option in the case where the risk-free interest rate r , the continuous dividend q , or the stock volatility σ is time-dependent and is described by values at a set of discrete times (see Section 8.2). This is illustrated in Section 9.

4 References

None.

5 Parameters

- | | | |
|----|---|--------------|
| 1: | t0 – double | <i>Input</i> |
| | <i>On entry:</i> the current time t_0 . | |
| | <i>Constraint:</i> td [0] ≤ t0 ≤ td [ntd – 1]. | |
| 2: | tmat – double | <i>Input</i> |
| | <i>On entry:</i> the maturity time T . | |
| | <i>Constraint:</i> td [0] ≤ tmat ≤ td [ntd – 1]. | |
| 3: | ntd – Integer | <i>Input</i> |
| | <i>On entry:</i> the number of discrete times at which ϕ is given. | |
| | <i>Constraint:</i> ntd ≥ 2. | |
| 4: | td [ntd] – const double | <i>Input</i> |
| | <i>On entry:</i> the discrete times at which ϕ is specified. | |
| | <i>Constraint:</i> td [0] < td [1] < ... < td [ntd – 1]. | |

- 5: **phid**[**ntd**] – const double *Input*
On entry: **phid**[$i - 1$] must contain the value of ϕ at time **td**[$i - 1$], for $i = 1, 2, \dots, \mathbf{ntd}$.
- 6: **phiav**[3] – double *Output*
On exit: **phiav**[0] contains the value of ϕ interpolated to t_0 , **phiav**[1] contains the first-order average $\hat{\phi}$ and **phiav**[2] contains the second-order average $\bar{\phi}$, where:
- $$\hat{\phi} = \frac{1}{T-t_0} \int_{t_0}^T \phi(\zeta) d\zeta, \quad \bar{\phi} = \left(\frac{1}{T-t_0} \int_{t_0}^T \phi^2(\zeta) d\zeta \right)^{1/2}.$$
- 7: **fail** – NagError * *Input/Output*
The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **ntd** = $\langle value \rangle$.
Constraint: **ntd** ≥ 2 .

NE_INTERNAL_ERROR

Unexpected failure in internal call to nag_1d_spline_interpolant (e01bac) or nag_1d_spline_evaluate (e02bbc).

NE_NOT_STRICTLY_INCREASING

On entry, **td**[i] $\leq$ **td**[$i - 1$] for $i = \langle value \rangle$.

NE_REAL_3

On entry, **tmat** lies outside the range [**td**[0], **td**[**ntd** - 1]]: **tmat** = $\langle value \rangle$, **td**[0] = $\langle value \rangle$, **td**[**ntd** - 1] = $\langle value \rangle$.

On entry, **t0** lies outside the range [**td**[0], **td**[**ntd** - 1]]: **t0** = $\langle value \rangle$, **td**[0] = $\langle value \rangle$, **td**[**ntd** - 1] = $\langle value \rangle$.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter $\langle value \rangle$ had an illegal value.

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 consult NAG for assistance.

7 Accuracy

If $\phi \in C^4[t_0, T]$ then the error in the approximation of $\phi(t_0)$ and $\hat{\phi}$ is $O(H^4)$, where $H = \max_i (T(i+1) - T(i))$, for $i = 1, 2, \dots, \mathbf{ntd} - 1$. The approximation is exact for polynomials of degree up to 3.

The third quantity $\bar{\phi}$ is $O(H^2)$, and exact for linear functions.

8 Further Comments

8.1 Timing

The time taken is proportional to **ntd**.

8.2 Use with nag_pde_bs_1d_analytic (d03ndc)

Suppose you wish to evaluate the analytic solution of the Black–Scholes equation in the case when the risk-free interest rate r is a known function of time, and is represented as a set of values at discrete times. A call to `nag_pde_bs_1d_means` (d03nec) providing these values in **phid** produces an output array **phiav** suitable for use as the argument **r** in a subsequent call to `nag_pde_bs_1d_analytic` (d03ndc).

Time-dependent values of the continuous dividend Q and the volatility σ may be handled in the same way.

8.3 Algorithmic Details

The **ntd** data points are fitted with a cubic B-spline using the function `nag_1d_spline_interpolant` (e01bac). Evaluation is then performed using `nag_1d_spline_evaluate` (e02bbc), and the definite integrals are computed using direct integration of the cubic splines in each interval. The special case of $T = t_o$ is handled by interpolating ϕ at that point.

9 Example

This example demonstrates the use of the function in conjunction with `nag_pde_bs_1d_analytic` (d03ndc) to solve the Black–Scholes equation for valuation of a 5-month American call option on a non-dividend-paying stock with an exercise price of \$50. The risk-free interest rate varies linearly with time and the stock volatility has a quadratic variation. Since these functions are integrated exactly by `nag_pde_bs_1d_means` (d03nec) the solution of the Black–Scholes equation by `nag_pde_bs_1d_analytic` (d03ndc) is also exact.

The option is valued at a range of times and stock prices.

9.1 Program Text

```
/* nag_pde_bs_1d_means (d03nec) Example Program.
 *
 * Copyright 2001 Numerical Algorithms Group.
 *
 * Mark 7, 2001.
 */

#include <stdio.h>
#include <string.h>
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagd03.h>

#define F(I,J) f[ns*((J)-1)+(I)-1]
#define THETA(I,J) theta[ns*((J)-1)+(I)-1]
#define DELTA(I,J) delta[ns*((J)-1)+(I)-1]
#define GAMMA(I,J) gamma[ns*((J)-1)+(I)-1]
#define LAMBDA(I,J) lambda[ns*((J)-1)+(I)-1]
#define RHO(I,J) rho[ns*((J)-1)+(I)-1]

int main(void)
{
    double ds, dt, tmat, x;
    Integer i, igreek, j, ns, nt, ntd, exit_status;
    double *delta, *f, *gamma, *lambda, q[3], ra[3], *rho, *s,
        siga[3], *t, *theta, *td, *rd, *sigd, smin, smax, tmin, tmax;
    Boolean gprnt[5]={TRUE, TRUE, TRUE, TRUE, TRUE}, tdp[3];
    const char *gname[5]={"Theta", "Delta", "Gamma", "Lambda", "Rho"};
    NagError fail;
```

```

/* Skip heading in data file */

Vscanf("%*[\n] ");
exit_status = 0;

/* Read problem parameters */

Vscanf("%lf", &x);
Vscanf("%lf", &tmat);
Vscanf("%ld%ld", &ns, &nt);
Vscanf("%lf%lf", &smin, &smax);
Vscanf("%lf%lf", &tmin, &tmax);
Vscanf("%ld", &ntd);

/* Allocate memory */

if ( !(s = NAG_ALLOC(ns, double)) ||
      !(t = NAG_ALLOC(nt, double)) ||
      !(f = NAG_ALLOC(ns*nt, double)) ||
      !(theta = NAG_ALLOC(ns*nt, double)) ||
      !(delta = NAG_ALLOC(ns*nt, double)) ||
      !(gamma = NAG_ALLOC(ns*nt, double)) ||
      !(lambda = NAG_ALLOC(ns*nt, double)) ||
      !(rho = NAG_ALLOC(ns*nt, double)) ||
      !(td = NAG_ALLOC(ntd, double)) ||
      !(rd = NAG_ALLOC(ntd, double)) ||
      !(sigd = NAG_ALLOC(ntd, double)) )
{
    Vprintf("Allocation failure\n");
    exit_status = 1;
    goto END;
}

/* Read discrete times and data values for r and sigma */

for (i=0; i<ntd; i++) Vscanf("%lf", &td[i]);
for (i=0; i<ntd; i++) Vscanf("%lf", &rd[i]);
for (i=0; i<ntd; i++) Vscanf("%lf", &sigd[i]);

INIT_FAIL(fail);
Vprintf("d03nec Example Program Results\n\n");

/* Set up input parameters for d03ndc */

s[0] = smin;
s[ns-1] = smax;
t[0] = tmin;
t[nt-1] = tmax;
tdpar[0] = TRUE;
tdpar[1] = FALSE;
tdpar[2] = TRUE;
q[0] = 0.0;

ds = (s[ns-1]-s[0])/(ns-1.0);
dt = (t[nt-1]-t[0])/(nt-1.0);

/* Loop over times */

for (j=1; j<=nt; j++)
{
    t[j-1] = t[0] + (j-1)*dt;

    /* Find average values of r and sigma */

    d03nec(t[j-1], tmat, ntd, td, rd, ra, &fail);

    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from d03nec.\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }
}

```

```

    }

    d03nec(t[j-1], tmat, ntd, td, sigd, siga, &fail);

    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from d03nec.\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }

    /* Loop over stock prices */

    for (i=1; i<=ns; i++)
    {
        s[i-1] = s[0] + (i-1)*ds;

        /* Evaluate analytic solution of Black-Scholes equation */

        d03ndc(Nag_AmericanCall, x, s[i-1], t[j-1], tmat, tdpar, ra,
            q, siga, &F(i,j), &THETA(i,j), &DELTA(i,j), &GAMMA(i,j),
            &LAMBDA(i,j), &RHO(i,j), &fail);

        if (fail.code != NE_NOERROR)
        {
            Vprintf("Error from d03ndc.\n%s\n", fail.message);
            exit_status = 1;
            goto END;
        }
    }
}

/* Output option values */

Vprintf("\n");
Vprintf("Option Values\n");
Vprintf("-----\n");
Vprintf(" Stock Price | Time to Maturity (months)\n");
Vprintf(" | ");
for (i=0; i<nt; i++) Vprintf(" %11.4e", 12.0*(tmat-t[i]));
Vprintf("\n");
for (i=0; i<64; i++) Vprintf("-");
Vprintf("\n");
for (i=1; i<=ns; i++)
{
    Vprintf(" %11.4e | ", s[i-1]);
    for (j=1; j<=nt; j++) Vprintf(" %11.4e", F(i,j));
    Vprintf("\n");
}

for (igreek = 0; igreek < 5; igreek++)
{
    if (!gprnt[igreek]) continue;

    Vprintf("\n");
    Vprintf("%s\n", gname[igreek]);
    for (i=0; i<(Integer)strlen(gname[igreek]); i++) Vprintf("-");
    Vprintf("\n");
    Vprintf(" Stock Price | Time to Maturity (months)\n");
    Vprintf(" | ");
    for (i=0; i<nt; i++) Vprintf(" %11.4e", 12.0*(tmat-t[i]));
    Vprintf("\n");
    for (i=0; i<64; i++) Vprintf("-");
    Vprintf("\n");

    for (i=1; i<=ns; i++)
    {
        Vprintf(" %11.4e | ", s[i-1]);
        switch (igreek)
        {

```

```

        case 0:
            for (j=1; j<=nt; j++) Vprintf(" %11.4e", THETA(i,j));
            break;
        case 1:
            for (j=1; j<=nt; j++) Vprintf(" %11.4e", DELTA(i,j));
            break;
        case 2:
            for (j=1; j<=nt; j++) Vprintf(" %11.4e", GAMMA(i,j));
            break;
        case 3:
            for (j=1; j<=nt; j++) Vprintf(" %11.4e", LAMBDA(i,j));
            break;
        case 4:
            for (j=1; j<=nt; j++) Vprintf(" %11.4e", RHO(i,j));
            break;
        default:
            break;
    }
    Vprintf("\n");
}
}
END:
if (s) NAG_FREE(s);
if (t) NAG_FREE(t);
if (f) NAG_FREE(f);
if (theta) NAG_FREE(theta);
if (delta) NAG_FREE(delta);
if (gamma) NAG_FREE(gamma);
if (lambda) NAG_FREE(lambda);
if (rho) NAG_FREE(rho);
if (td) NAG_FREE(td);
if (rd) NAG_FREE(rd);
if (sigd) NAG_FREE(sigd);

return exit_status;
}

```

9.2 Program Data

d03nec Example Program Data

```

50.
0.4166667
21 4
0.0 100.
0.0 0.125
6
0.00 0.10 0.20
0.30 0.40 0.50
0.10 0.11 0.12
0.13 0.14 0.15
0.30 0.46 0.54
0.54 0.46 0.30

```

9.3 Program Results

d03nec Example Program Results

Option Values

Stock Price		Time to Maturity (months)			
		5.0000e+00	4.5000e+00	4.0000e+00	3.5000e+00

0.0000e+00		0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
5.0000e+00		3.3671e-13	7.7404e-14	7.3210e-15	2.0179e-16
1.0000e+01		5.2088e-07	2.4281e-07	7.2216e-08	1.1540e-08
1.5000e+01		2.6607e-04	1.6753e-04	8.0943e-05	2.7179e-05
2.0000e+01		8.9697e-03	6.6505e-03	4.1780e-03	2.0942e-03
2.5000e+01		8.3647e-02	6.8467e-02	5.0375e-02	3.2105e-02
3.0000e+01		3.8221e-01	3.3331e-01	2.7117e-01	2.0119e-01
3.5000e+01		1.1298e+00	1.0275e+00	8.9292e-01	7.3146e-01
4.0000e+01		2.5164e+00	2.3541e+00	2.1380e+00	1.8699e+00

4.5000e+01		4.6249e+00	4.4110e+00	4.1267e+00	3.7700e+00
5.0000e+01		7.4287e+00	7.1797e+00	6.8531e+00	6.4449e+00
5.5000e+01		1.0830e+01	1.0564e+01	1.0221e+01	9.7996e+00
6.0000e+01		1.4707e+01	1.4436e+01	1.4097e+01	1.3689e+01
6.5000e+01		1.8937e+01	1.8671e+01	1.8348e+01	1.7968e+01
7.0000e+01		2.3421e+01	2.3164e+01	2.2860e+01	2.2514e+01
7.5000e+01		2.8080e+01	2.7833e+01	2.7550e+01	2.7234e+01
8.0000e+01		3.2857e+01	3.2620e+01	3.2354e+01	3.2064e+01
8.5000e+01		3.7713e+01	3.7484e+01	3.7233e+01	3.6963e+01
9.0000e+01		4.2620e+01	4.2398e+01	4.2158e+01	4.1904e+01
9.5000e+01		4.7561e+01	4.7344e+01	4.7112e+01	4.6868e+01
1.0000e+02		5.2523e+01	5.2310e+01	5.2084e+01	5.1848e+01

Theta

Stock Price	Time to Maturity (months)			
	5.0000e+00	4.5000e+00	4.0000e+00	3.5000e+00

0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
5.0000e+00	-8.9082e-12	-3.4507e-12	-5.0884e-13	-2.1236e-14
1.0000e+01	-7.2097e-06	-5.5915e-06	-2.5721e-06	-6.1830e-07
1.5000e+01	-2.2499e-03	-2.3259e-03	-1.7227e-03	-8.6349e-04
2.0000e+01	-4.9483e-02	-5.9355e-02	-5.6562e-02	-4.1921e-02
2.5000e+01	-3.1200e-01	-4.0620e-01	-4.4765e-01	-4.1683e-01
3.0000e+01	-9.8578e-01	-1.3408e+00	-1.6092e+00	-1.7186e+00
3.5000e+01	-2.0479e+00	-2.8395e+00	-3.5745e+00	-4.1390e+00
4.0000e+01	-3.2501e+00	-4.5165e+00	-5.8147e+00	-7.0323e+00
4.5000e+01	-4.3144e+00	-5.9349e+00	-7.6762e+00	-9.4488e+00
5.0000e+01	-5.0802e+00	-6.8543e+00	-8.7919e+00	-1.0815e+01
5.5000e+01	-5.5225e+00	-7.2603e+00	-9.1500e+00	-1.1104e+01
6.0000e+01	-5.7006e+00	-7.2722e+00	-8.9491e+00	-1.0625e+01
6.5000e+01	-5.7014e+00	-7.0446e+00	-8.4366e+00	-9.7565e+00
7.0000e+01	-5.6037e+00	-6.7093e+00	-7.8142e+00	-8.7951e+00
7.5000e+01	-5.4653e+00	-6.3555e+00	-7.2107e+00	-7.9170e+00
8.0000e+01	-5.3218e+00	-6.0329e+00	-6.6903e+00	-7.1974e+00
8.5000e+01	-5.1920e+00	-5.7627e+00	-6.2736e+00	-6.6481e+00
9.0000e+01	-5.0833e+00	-5.5487e+00	-5.9563e+00	-6.2492e+00
9.5000e+01	-4.9969e+00	-5.3857e+00	-5.7234e+00	-5.9700e+00
1.0000e+02	-4.9306e+00	-5.2651e+00	-5.5570e+00	-5.7797e+00

Delta

Stock Price		Time to Maturity (months)			
		5.0000e+00	4.5000e+00	4.0000e+00	3.5000e+00

0.0000e+00		0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
5.0000e+00		1.6086e-12	3.8832e-13	3.9572e-14	1.2111e-15
1.0000e+01		8.9933e-07	4.3972e-07	1.4063e-07	2.4884e-08
1.5000e+01		2.3975e-04	1.5810e-04	8.1943e-05	3.0366e-05
2.0000e+01		4.9150e-03	3.8095e-03	2.5596e-03	1.4100e-03
2.5000e+01		3.0345e-02	2.5906e-02	2.0311e-02	1.4153e-02
3.0000e+01		9.6991e-02	8.7980e-02	7.5946e-02	6.1231e-02
3.5000e+01		2.0863e-01	1.9675e-01	1.8053e-01	1.5957e-01
4.0000e+01		3.4875e-01	3.3719e-01	3.2158e-01	3.0109e-01
4.5000e+01		4.9361e-01	4.8480e-01	4.7356e-01	4.5924e-01
5.0000e+01		6.2450e-01	6.1931e-01	6.1363e-01	6.0735e-01
5.5000e+01		7.3200e-01	7.3000e-01	7.2907e-01	7.2954e-01
6.0000e+01		8.1439e-01	8.1462e-01	8.1681e-01	8.2145e-01
6.5000e+01		8.7440e-01	8.7589e-01	8.7961e-01	8.8602e-01
7.0000e+01		9.1650e-01	9.1850e-01	9.2260e-01	9.2911e-01
7.5000e+01		9.4522e-01	9.4726e-01	9.5107e-01	9.5679e-01
8.0000e+01		9.6441e-01	9.6624e-01	9.6946e-01	9.7406e-01
8.5000e+01		9.7704e-01	9.7856e-01	9.8111e-01	9.8461e-01
9.0000e+01		9.8526e-01	9.8646e-01	9.8839e-01	9.9094e-01
9.5000e+01		9.9057e-01	9.9148e-01	9.9290e-01	9.9470e-01
1.0000e+02		9.9397e-01	9.9464e-01	9.9567e-01	9.9691e-01

Gamma

Stock Price		Time to Maturity (months)			
		5.0000e+00	4.5000e+00	4.0000e+00	3.5000e+00

0.0000e+00		0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
5.0000e+00		7.2334e-12	1.8390e-12	2.0276e-13	6.9267e-15
1.0000e+01		1.4139e-06	7.2829e-07	2.5205e-07	4.9786e-08
1.5000e+01		1.8932e-04	1.3153e-04	7.3756e-05	3.0494e-05
2.0000e+01		2.2528e-03	1.8392e-03	1.3360e-03	8.2017e-04
2.5000e+01		8.6933e-03	7.8126e-03	6.6135e-03	5.1251e-03
3.0000e+01		1.8099e-02	1.7264e-02	1.6056e-02	1.4350e-02
3.5000e+01		2.5953e-02	2.5691e-02	2.5315e-02	2.4683e-02
4.0000e+01		2.9260e-02	2.9618e-02	3.0194e-02	3.0968e-02
4.5000e+01		2.8046e-02	2.8736e-02	2.9814e-02	3.1368e-02
5.0000e+01		2.4005e-02	2.4715e-02	2.5793e-02	2.7346e-02
5.5000e+01		1.8950e-02	1.9500e-02	2.0296e-02	2.1401e-02
6.0000e+01		1.4105e-02	1.4449e-02	1.4903e-02	1.5476e-02
6.5000e+01		1.0054e-02	1.0221e-02	1.0396e-02	1.0555e-02
7.0000e+01		6.9401e-03	6.9861e-03	6.9806e-03	6.8890e-03
7.5000e+01		4.6779e-03	4.6538e-03	4.5552e-03	4.3505e-03
8.0000e+01		3.0978e-03	3.0414e-03	2.9096e-03	2.6800e-03
8.5000e+01		2.0250e-03	1.9598e-03	1.8291e-03	1.6205e-03
9.0000e+01		1.3114e-03	1.2499e-03	1.1365e-03	9.6637e-04
9.5000e+01		8.4362e-04	7.9138e-04	7.0024e-04	5.7052e-04
1.0000e+02		5.4033e-04	4.9856e-04	4.2893e-04	3.3442e-04

Lambda

Stock Price		Time to Maturity (months)			
		5.0000e+00	4.5000e+00	4.0000e+00	3.5000e+00

0.0000e+00		0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
5.0000e+00		3.6558e-11	8.6441e-12	8.6672e-13	2.6259e-14
1.0000e+01		2.8583e-05	1.3693e-05	4.3098e-06	7.5495e-07
1.5000e+01		8.6115e-03	5.5645e-03	2.8375e-03	1.0404e-03
2.0000e+01		1.8217e-01	1.3832e-01	9.1376e-02	4.9748e-02
2.5000e+01		1.0984e+00	9.1808e-01	7.0676e-01	4.8574e-01
3.0000e+01		3.2931e+00	2.9214e+00	2.4708e+00	1.9584e+00
3.5000e+01		6.4272e+00	5.9173e+00	5.3025e+00	4.5851e+00
4.0000e+01		9.4643e+00	8.9101e+00	8.2604e+00	7.5135e+00
4.5000e+01		1.1481e+01	1.0941e+01	1.0323e+01	9.6323e+00
5.0000e+01		1.2132e+01	1.1617e+01	1.1026e+01	1.0367e+01
5.5000e+01		1.1588e+01	1.1091e+01	1.0498e+01	9.8169e+00
6.0000e+01		1.0265e+01	9.7801e+00	9.1734e+00	8.4486e+00
6.5000e+01		8.5872e+00	8.1198e+00	7.5104e+00	6.7621e+00
7.0000e+01		6.8747e+00	6.4363e+00	5.8487e+00	5.1188e+00
7.5000e+01		5.3194e+00	4.9219e+00	4.3812e+00	3.7109e+00
8.0000e+01		4.0081e+00	3.6599e+00	3.1840e+00	2.6009e+00
8.5000e+01		2.9578e+00	2.6623e+00	2.2597e+00	1.7754e+00
9.0000e+01		2.1474e+00	1.9036e+00	1.5741e+00	1.1870e+00
9.5000e+01		1.5392e+00	1.3429e+00	1.0806e+00	7.8078e-01
1.0000e+02		1.0923e+00	9.3740e-01	7.3341e-01	5.0711e-01

Rho

Stock Price		Time to Maturity (months)			
		5.0000e+00	4.5000e+00	4.0000e+00	3.5000e+00

0.0000e+00		0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
5.0000e+00		3.2110e-12	6.9908e-13	6.3513e-14	1.7073e-15
1.0000e+01		3.5302e-06	1.5579e-06	4.4470e-07	6.9214e-08
1.5000e+01		1.3876e-03	8.2648e-04	3.8273e-04	1.2492e-04
2.0000e+01		3.7221e-02	2.6077e-02	1.5671e-02	7.6142e-03
2.5000e+01		2.8124e-01	2.1719e-01	1.5247e-01	9.3836e-02
3.0000e+01		1.0531e+00	8.6478e-01	6.6907e-01	4.7709e-01
3.5000e+01		2.5718e+00	2.1971e+00	1.8086e+00	1.4156e+00
4.0000e+01		4.7641e+00	4.1750e+00	3.5750e+00	2.9673e+00
4.5000e+01		7.3281e+00	6.5270e+00	5.7279e+00	4.9280e+00
5.0000e+01		9.9152e+00	8.9196e+00	7.9427e+00	6.9774e+00
5.5000e+01		1.2262e+01	1.1095e+01	9.9592e+00	8.8448e+00
6.0000e+01		1.4232e+01	1.2915e+01	1.1637e+01	1.0383e+01
6.5000e+01		1.5791e+01	1.4348e+01	1.2942e+01	1.1557e+01
7.0000e+01		1.6973e+01	1.5424e+01	1.3907e+01	1.2403e+01
7.5000e+01		1.7838e+01	1.6204e+01	1.4594e+01	1.2987e+01

8.0000e+01		1.8457e+01	1.6755e+01	1.5067e+01	1.3376e+01
8.5000e+01		1.8890e+01	1.7135e+01	1.5387e+01	1.3629e+01
9.0000e+01		1.9189e+01	1.7393e+01	1.5599e+01	1.3790e+01
9.5000e+01		1.9393e+01	1.7567e+01	1.5738e+01	1.3891e+01
1.0000e+02		1.9531e+01	1.7683e+01	1.5827e+01	1.3954e+01
