NAG Library Function Document

nag_normal_pdf_vector (g01kqc)

1 Purpose

nag_normal_pdf_vector (g01kqc) returns a number of values of the probability density function (PDF), or its logarithm, for the Normal (Gaussian) distributions.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
void nag_normal_pdf_vector (Nag_Boolean ilog, Integer lx, const double x[],
     Integer lxmu, const double xmu[], Integer lxstd, const double xstd[],
     double pdf[], Integer ivalid[], NagError *fail)
```

3 Description

The Normal distribution with mean μ_i, variance ${\sigma_i}^2$; has probability density function (PDF)

$$f(x_i,\mu_i,\sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}}e^{-(x_i-\mu_i)^2/2{\sigma_i}^2}, \quad \sigma_i > 0.$$

The input arrays to this function are designed to allow maximum flexibility in the supply of vector arguments by re-using elements of any arrays that are shorter than the total number of evaluations required. See Section 2.6 in the g01 Chapter Introduction for further information.

4 References

None.

5 Arguments

1: **ilog** – Nag_Boolean *Input*

On entry: the value of **ilog** determines whether the logarithmic value is returned in PDF.

ilog = Nag_FALSE
$f(x_i,\mu_i,\sigma_i)$, the probability density function is returned.

ilog = Nag_TRUE
$\log(f(x_i,\mu_i,\sigma_i))$, the logarithm of the probability density function is returned.

2: **lx** – Integer *Input*

On entry: the length of the array **x**.

Constraint: **lx** > 0.

3: **x**[**lx**] – const double *Input*

On entry: x_i, the values at which the PDF is to be evaluated with $x_i = \mathbf{x}[j]$, $j = (i-1) \bmod \mathbf{lx}$, for $i = 1, 2, \ldots, \max(\mathbf{lx}, \mathbf{lxstd}, \mathbf{lxmu})$.

4: **lxmu** – Integer *Input*

On entry: the length of the array **xmu**.

Constraint: **lxmu** > 0.

5: **xmu**[**lxmu**] – const double *Input*

On entry: μ_i, the means with $\mu_i =$ **xmu**$[j]$, $j = (i-1) \bmod$ **lxmu**.

6: **lxstd** – Integer *Input*

On entry: the length of the array **xstd**.

Constraint: **lxstd** > 0.

7: **xstd**[**lxstd**] – const double *Input*

On entry: σ_i, the standard deviations with $\sigma_i =$ **xstd**$[j]$, $j = (i-1) \bmod$ **lxstd**.

Constraint: **xstd**$[j-1] \geq 0.0$, for $j = 1, 2, \ldots,$ **lxstd**.

8: **pdf**$[dim]$ – double *Output*

Note: the dimension, *dim*, of the array **pdf** must be at least $\max($**lx**, **lxstd**, **lxmu**$)$.

On exit: $f(x_i, \mu_i, \sigma_i)$ or $\log(f(x_i, \mu_i, \sigma_i))$.

9: **ivalid**$[dim]$ – Integer *Output*

Note: the dimension, *dim*, of the array **ivalid** must be at least $\max($**lx**, **lxstd**, **lxmu**$)$.

On exit: **ivalid**$[i-1]$ indicates any errors with the input arguments, with

ivalid$[i-1] = 0$
No error.

ivalid$[i-1] = 1$
$\sigma_i < 0$.

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_ARRAY_SIZE

On entry, array size $=$ ⟨*value*⟩.
Constraint: **lx** > 0.

On entry, array size $=$ ⟨*value*⟩.
Constraint: **lxmu** > 0.

On entry, array size $=$ ⟨*value*⟩.
Constraint: **lxstd** > 0.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ 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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

NW_IVALID

On entry, at least one value of **xstd** was invalid.
Check **ivalid** for more information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_normal_pdf_vector (g01kqc) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example prints the value of the Normal distribution PDF at four different points x_i with differing μ_i and σ_i.

10.1 Program Text

```
/* nag_normal_pdf_vector (g01kqc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg01.h>

int main(void)
{
  /* Integer scalar and array declarations */
  Integer lx, lxmu, lxstd, i, lout;
  Integer *ivalid = 0;
  Integer exit_status = 0;

  /* NAG structures */
  NagError fail;
  Nag_Boolean ilog;

  /* Double scalar and array declarations */
  double *x = 0, *xmu = 0, *xstd = 0, *pdf = 0;
```

```
  /* Character scalar and array declarations */
  char cilog[40];

  /* Initialize the error structure to print out any error messages */
  INIT_FAIL(fail);

  printf("nag_normal_pdf_vector (g01kqc) Example Program Results\n\n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  /* Read in the flag indicating whether logs are required */
#ifdef _WIN32
  scanf_s("%39s%*[^\n] ", cilog, (unsigned)_countof(cilog));
#else
  scanf("%39s%*[^\n] ", cilog);
#endif
  ilog = (Nag_Boolean) nag_enum_name_to_value(cilog);

  /* Read in the input vectors */
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%*[^\n] ", &lx);
#else
  scanf("%" NAG_IFMT "%*[^\n] ", &lx);
#endif
  if (!(x = NAG_ALLOC(lx, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }
  for (i = 0; i < lx; i++)
#ifdef _WIN32
    scanf_s("%lf", &x[i]);
#else
    scanf("%lf", &x[i]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%*[^\n] ", &lxmu);
#else
  scanf("%" NAG_IFMT "%*[^\n] ", &lxmu);
#endif
  if (!(xmu = NAG_ALLOC(lxmu, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }
  for (i = 0; i < lxmu; i++)
#ifdef _WIN32
    scanf_s("%lf", &xmu[i]);
#else
    scanf("%lf", &xmu[i]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

#ifdef _WIN32
```

```
  scanf_s("%" NAG_IFMT "%*[^\n] ", &lxstd);
#else
  scanf("%" NAG_IFMT "%*[^\n] ", &lxstd);
#endif
  if (!(xstd = NAG_ALLOC(lxstd, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }
  for (i = 0; i < lxstd; i++)
#ifdef _WIN32
    scanf_s("%lf", &xstd[i]);
#else
    scanf("%lf", &xstd[i]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif

  /* Allocate memory for output */
  lout = MAX(lx, MAX(lxmu, lxstd));
  if (!(pdf = NAG_ALLOC(lout, double)) ||
      !(ivalid = NAG_ALLOC(lout, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  /* Calculate probability */
  nag_normal_pdf_vector(ilog, lx, x, lxmu, xmu, lxstd, xstd, pdf, ivalid,
                        &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_normal_pdf_vector (g01kqc).\n%s\n", fail.message);
    exit_status = 1;
    if (fail.code != NW_IVALID)
      goto END;
  }

  /* Display title */
  printf("   x        xmu       xstd       pdf      ivalid\n");
  printf(" ------------------------------------------------\n");

  /* Display results */
  for (i = 0; i < lout; i++)
    printf("%6.2f    %6.2f    %6.2f    %9.3e    %3" NAG_IFMT "\n",
           x[i % lx], xmu[i % lxmu], xstd[i % lxstd], pdf[i], ivalid[i]);

END:
  NAG_FREE(x);
  NAG_FREE(xmu);
  NAG_FREE(xstd);
  NAG_FREE(pdf);
  NAG_FREE(ivalid);

  return (exit_status);
}
```

10.2 Program Data

```
nag_normal_pdf_vector (g01kqc) Example Program Data
Nag_FALSE             :: ILOG
4                     :: LX
1.0 4.0 0.1 1.0       :: X
4                     :: LXMU
0.0 2.0 0.0 0.0       :: XMU
4                     :: LXSTD
1.0 1.0 0.01 10.0     :: XSTD
```

10.3 Program Results

```
nag_normal_pdf_vector (g01kqc) Example Program Results

   x        xmu       xstd       pdf      ivalid
 ----------------------------------------------------
  1.00      0.00       1.00    2.420e-01      0
  4.00      2.00       1.00    5.399e-02      0
  0.10      0.00       0.01    7.695e-21      0
  1.00      0.00      10.00    3.970e-02      0
```

Example Program
Plots of the Gaussian Function (or Normal Distribution).