NAG C Library Function Document

nag_prob_vavilov (g01euc)

1 Purpose

nag_prob_vavilov (g01euc) returns the value of the Vavilov distribution function \( \Phi_V(\lambda; \kappa, \beta^2) \).

It is intended to be used after a call to nag_init_vavilov (g01zuc).

2 Specification

```c
double nag_prob_vavilov (double x, const double comm_arr[])
```

3 Description

nag_prob_vavilov (g01euc) evaluates an approximation to the Vavilov distribution function \( \Phi_V(\lambda; \kappa, \beta^2) \) given by

\[
\Phi_V(\lambda; \kappa, \beta^2) = \int_{-\infty}^{\lambda} \phi_V(\lambda; \kappa, \beta^2) d\lambda,
\]

where \( \phi(\lambda) \) is described in nag_prob_density_vavilov (g01muc). The method used is based on Fourier expansions. Further details can be found in Schorr (1974).

4 References


5 Parameters

1: \( x \) – double

\( \text{Input} \)

\( \text{On entry:} \) the argument \( \lambda \) of the function.

2: \( \text{comm_arr}[322] \) – const double

\( \text{Input} \)

\( \text{On entry:} \) this must be the same parameter \( \text{comm_arr} \) as returned by a previous call to nag_init_vavilov (g01zuc).

6 Error Indicators and Warnings

None.

7 Accuracy

At least 5 significant digits are usually correct.

8 Further Comments

nag_prob_vavilov (g01euc) can be called repeatedly with different values of \( \lambda \) provided that the values of \( \kappa \) and \( \beta^2 \) remain unchanged between calls. Otherwise, nag_init_vavilov (g01zuc) must be called again. This is illustrated in Section 9.
9  Example

The example program evaluates $\Phi_V(\lambda; \kappa, \beta^2)$ at $\lambda = 0.1$, $\kappa = 2.5$ and $\beta^2 = 0.7$, and prints the results.

9.1  Program Text

```c
/* nag_prob_vavilov (g01euc) Example Program.  
*  Copyright 2002 Numerical Algorithms Group.  
*  Mark 7, 2002.  */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg01.h>
#include <nagx02.h>

int main(void)
{
    double c1, c2, x, rkappa, beta2, xl, xu, y;
    Integer exit_status, mode;
    NagError fail;

define WKMAX 322

    double comm_arr[WKMAX];
    mode = 1;
    INIT_FAIL(fail);
    exit_status = 0;
    c1 = -X02ALC;
    c2 = -X02ALC;
    Vprintf(" g01euc Example Program Results\n\n");

    while (scanf("%lf%lf%lf\n", &x, &rkappa, &beta2) != EOF)
    {
        if ((rkappa != c1) || (beta2 != c2 ))
        {
            g01zuc(rkappa, beta2, mode, &xl, &xu, comm_arr, &fail);
            if (fail.code != NE_NOERROR)
            {
                Vprintf("Error from g01zuc.\n", fail.message);
                exit_status = 1;
                goto END;
            }
        }
        y = g01euc(x, comm_arr);
        Vprintf(" X  Rkappa  Beta2  Y\n");
        Vprintf(" %3.1f %3.1f %3.1f %12.4e\n", x, rkappa, beta2, y);
        c1 = rkappa;
        c2 = beta2;
    }
    END:
    return exit_status;
}
```

9.2 Program Data

g01euc Example Program Data
  0.1  2.5  0.7 : Values of X, RKAPPA and BETA2

9.3 Program Results

g01euc Example Program Results

<table>
<thead>
<tr>
<th>X</th>
<th>Rkappa</th>
<th>Beta2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2.5</td>
<td>0.7</td>
<td>9.9982e-01</td>
</tr>
</tbody>
</table>