NAG Library Function Document
nag_prob_density_vavilov (g01muc)

1 Purpose
nag_prob_density_vavilov (g01muc) returns the value of the Vavilov density function \( \phi_V(\lambda; \kappa, \beta^2) \).
It is intended to be used after a call to nag_init_vavilov (g01zuc).

2 Specification
#include <nag.h>
#include <nagg01.h>
double nag_prob_density_vavilov (double x, const double comm_arr[])

3 Description
nag_prob_density_vavilov (g01muc) evaluates an approximation to the Vavilov density function \( \phi_V(\lambda; \kappa, \beta^2) \) given by
\[
\phi_V(\lambda; \kappa, \beta^2) = \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} e^{\lambda s} f(s; \kappa, \beta^2) \, ds,
\]
where \( \kappa > 0 \) and \( 0 \leq \beta^2 \leq 1 \), \( c \) is an arbitrary real constant and
\[
f(s; \kappa, \beta^2) = C(\kappa, \beta^2) \exp\left\{ s \ln \kappa + (s + \kappa\beta^2) \left[ \ln\left(\frac{s}{\kappa}\right) + E_1\left(\frac{s}{\kappa}\right)\right] - \kappa \exp\left(-\frac{s}{\kappa}\right) \right\}.
\]
\( E_1(x) = \int_0^x t^{-1}(1 - e^{-t}) \, dt \) is the exponential integral, \( C(\kappa, \beta^2) = \exp\{\kappa(1 + \gamma\beta^2)\} \) and \( \gamma \) is Euler’s constant.
The method used is based on Fourier expansions. Further details can be found in Schorr (1974).
For values of \( \kappa \leq 0.01 \), the Vavilov distribution can be replaced by the Landau distribution since \( \lambda_V = (\lambda_L - \ln \kappa)/\kappa \). For values of \( \kappa \geq 10 \), the Vavilov distribution can be replaced by a Gaussian distribution with mean \( \mu = \gamma - 1 - \beta^2 - \ln \kappa \) and variance \( \sigma^2 = (2 - \beta^2)/2\kappa \).

4 References

5 Arguments
1:  x – double
    Input
    On entry: the argument \( \lambda \) of the function.

2:  comm_arr[322] – const double
    Communication Array
    On entry: this must be the same argument comm_arr as returned by a previous call to nag_init_vavilov (g01zuc).

6 Error Indicators and Warnings
None.
7 Accuracy
At least five significant digits are usually correct.

8 Parallelism and Performance
Not applicable.

9 Further Comments
nag_prob_density_vavilov (g01muc) 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 10.

10 Example
This example evaluates $\phi_V(\lambda; \kappa, \beta^2)$ at $\lambda = 2.5$, $\kappa = 0.4$ and $\beta^2 = 0.1$, and prints the results.

10.1 Program Text
/* nag_prob_density_vavilov (g01muc) Example Program.*/
/* Copyright 2014 Numerical Algorithms Group.*/
/* Mark 7, 2002.*/
#include <stdio.h>
#include <nag.h>
#include <naq_stdlib.h>
#include <nagg01.h>
#include <nagx02.h>

int main(void)
{
    /* Scalars */
    double c1, c2, x, rkappa, beta2, x1, xu, y;
    Integer exit_status, mode;
    NagError fail;
#define WKMAX 322
    double comm_arr[WKMAX];
    mode = 0;
    INIT_FAIL(fail);
    exit_status = 0;
    /* nag_real_largest_number (x02alc).
     The largest positive model number */
    c1 = -nag_real_largest_number;
    /* nag_real_largest_number (x02alc), see above. */
    c2 = -nag_real_largest_number;
    printf(" nag_prob_density_vavilov (g01muc) Example Program Results\n\n");
    /* Skip heading in data file */
    #ifdef _WIN32
        scanf_s("%*[\n] ");
    #else
        scanf("%*[\n] ");
    #endif
}
```c
#ifdef _WIN32
    while (scanf_s("%lf%lf%lf%*[\n] ", &x, &rkappa, &beta2) != EOF)
#else
    while (scanf("%lf%lf%lf%*[\n] ", &x, &rkappa, &beta2) != EOF)
#endif
{
    if ((rkappa != c1) || (beta2 != c2))
    {
        /* nag_init_vavilov (g01zuc).
           * Initialization function for
           * nag_prob_density_vavilov (g01muc) and
           * nag_prob_vavilov (g01euc)
           */
        nag_init_vavilov(rkappa, beta2, mode, &xl, &xu, comm_arr, &fail);
        if (fail.code != NE_NOERROR)
        {
            printf("Error from nag_init_vavilov (g01zuc).
                   ", fail.message);
            exit_status = 1;
            goto END;
        }
    }
    /* nag_prob_density_vavilov (g01muc).
       * Vavilov density function \( \phi_V(\lambda;kappa)beta^2 \)
       */
    y = nag_prob_density_vavilov(x, comm_arr);
    printf(" X   Rkappa   Beta2   Y\n\n");
    printf(" %3.1f %3.1f %3.1f %13.4e\n", x, rkappa, beta2, y);
    c1 = rkappa;
    c2 = beta2;
}
END:
return exit_status;
}
```

10.2 Program Data

**nag_prob_density_vavilov (g01muc) Example Program Data**

2.5 0.4 0.1 : Values of X, RKAPPA and BETA2

10.3 Program Results

**nag_prob_density_vavilov (g01muc) Example Program Results**

<table>
<thead>
<tr>
<th>X</th>
<th>Rkappa</th>
<th>Beta2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5</td>
<td>0.4</td>
<td>0.1</td>
<td>8.3675e-02</td>
</tr>
</tbody>
</table>