NAG C Library Function Document

nag_bessel_zeros (s17alc)

1 Purpose

nag_bessel_zeros (s17alc) determines the leading N zeros of one of the Bessel functions $J_\alpha(x)$, $Y_\alpha(x)$, $J'_\alpha(x)$ or $Y'_\alpha(x)$ for real x and non-negative α.

2 Specification

```
void nag_bessel_zeros (double a, Integer n, Integer mode, double rel,
          double x[], NagError *fail)
```

3 Description

This routine attempts to find the leading N zeros of one of the Bessel functions $J_\alpha(x)$, $Y_\alpha(x)$, $J'_\alpha(x)$ or $Y'_\alpha(x)$, where x is real. When α is real, these functions each have an infinite number of real zeros, all of which are simple with the possible exception of $x = 0$. If $\alpha \geq 0$, the nth positive zero is denoted by $j_{\alpha,n}, j'_{\alpha,n}, y_{\alpha,n}$ and $y'_{\alpha,n}$, respectively, for $n = 1, 2, \ldots, N$, except that $x = 0$ is counted as the first zero of $J'_\alpha(x)$ when $\alpha = 0$. Since $J'_0(x) = -J_1(x)$, it therefore follows that $j'_{0,1} = 0$ and $j'_{0,n} = -j_{1,n-1}$, for $n = 2, 3, \ldots, N-1$. Further details can be found in Abramowitz and Stegun (1972), 9.5.

nag_bessel_zeros is based on Algol 60 procedures given by Temme N M (1979). Initial approximations to the zeros are computed from asymptotic expansions. These are then improved by higher-order Newton iteration making use of the differential equation for the Bessel functions.

4 Parameters

1: **a** – double *Input*

On entry: the order α of the function.

Constraint: $0.0 \leq$ **a** ≤ 100000.0.

2: **n** – Integer *Input*

On entry: the number N of zeros required.

Constraint: **n** ≥ 1.

3: **mode** – Integer *Input*

On entry: specifies the form of the function whose zeros are required as follows:

if **mode** $= 1$, then the zeros of $J_\alpha(x)$ are required;

if **mode** $= 2$, then the zeros of $Y_\alpha(x)$ are required;

if **mode** $= 3$, then the zeros of $J'_\alpha(x)$ are required;

if **mode** $= 4$, then the zeros of $Y'_\alpha(x)$ are required.

Constraint: $1 \leq$ **mode** ≤ 4.

4: **rel** – double *Input*

On entry: the relative accuracy to which the zeros are required.

Suggested value: the square root of the ***machine precision***.

Constraint: **rel** > 0.0.

5: **x[n]** – double *Output*

On exit: the N required zeros of the function specified by **mode**.

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

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

NE_INT

On entry, **n** = <*value*>.
Constraint: **n** $\geq$ 1.

On entry, **mode** = <*value*>.
Constraint: 1 $\leq$ **mode** $\leq$ 4.

NE_REAL

On entry, **a** = <*value*>.
Constraint: 0.0 $\leq$ **a** $\leq$ 100000.0.

On entry, **rel** = <*value*>.
Constraint: **rel** $>$ 0.0.

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.

6 Further Comments

6.1 Accuracy

If the value of **rel** is set to 10^{-d}, then the required zeros should have approximately d correct significant digits.

6.2 References

Abramowitz M and Stegun I A (1972) *Handbook of Mathematical Functions* Dover Publications (3rd Edition)

Temme N M (1979) An algorithm with Algol 60 program for the computation of the *J. Comput. Phys.* **32** 270–279

Temme N M (1976) On the numerical evaluation of the ordinary Bessel function *J. Comput. Phys.* **21** 343–350

7 See Also

None.

8 Example

To determine the leading five positive zeros of the Bessel function $J_0(x)$.

8.1 Program Text

```
/* nag_bessel_zeros (s17alc) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * NAG C Library
 *
 * Mark 6, 2000.
 */

#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nags.h>
#include <nagx02.h>

int main(void)
{

#define NMAX 100

  double a, rel;
  double *x=0;
  Integer i;
  Integer exit_status=0;
  Integer mode;
  Integer n;
  NagError fail;

  INIT_FAIL(fail);
  Vprintf("s17alc Example Program Results\n\n");

  if (! (x = NAG_ALLOC(NMAX, double)))
    {
      Vprintf("Allocation failure\n");
      exit_status=-1;
    }
  /*  Skip heading in data file */
  Vscanf("%*[^\n]");
  rel = sqrt (X02AJC);
  Vscanf("%lf %ld %ld", &a, &n, &mode);
  s17alc  (a, n, mode, rel, x, &fail);

  if (fail.code == NE_NOERROR)
    {
      Vprintf("   a   n   mode          rel\n");
      Vprintf("                   (machine-dependent)\n\n");
      Vprintf(" %4.1f%3ld%6ld       %9.1e\n\n", a, n, mode, rel);
      if (mode == 1)
        {
          Vprintf("Leading N positive zeros of J\n");
        }
      else if (mode == 2)
        {
          Vprintf("Leading N positive zeros of Y\n");
        }
      else if (mode == 3)
        {
```

```
          if (a == 0.0)
     {
       Vprintf("Leading N non-negative zeros of J'\n");
     }
          else
     {
       Vprintf("Leading N positive zeros of J'\n");
     }
            }
      else if (mode == 4)
        {
          Vprintf("Leading N positive zeros of Y'\n\n");
        }
      for (i = 0; i <= n-1; ++i)
        {
          Vprintf("x = %12.4e\n", x[i]);
        }
      Vprintf("\n");
    }
   else
    {
      Vprintf("Error from s17alc.\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
 END:
  if (x) NAG_FREE (x);
  return exit_status;
}
```

8.2 Program Data

```
s17alc Example Program Data
 0.0  5  1 : Values of a, n and mode
```

8.3 Program Results

```
s17alc Example Program Results

   a   n   mode          rel
                  (machine-dependent)

  0.0  5     1         1.1e-08

Leading N positive zeros of J
x =   2.4048e+00
x =   5.5201e+00
x =   8.6537e+00
x =   1.1792e+01
x =   1.4931e+01
```