NAG C Library Function Document

nag_complex_bessel_j_seq (s18gkc)

1 Purpose

nag_complex_bessel_j_seq (s18gkc) returns a sequence of values for the Bessel functions $J_{\alpha+n-1}(z)$ or $J_{\alpha-n+1}(z)$ for complex z, non-negative $\alpha < 1$ and $n = 1, 2, \ldots, |N| + 1$.

2 Specification

```
void nag_complex_bessel_j_seq (Complex z, double a, Integer nl, Complex b[],
    NagError *fail)
```

3 Description

nag_complex_bessel_j_seq (s18gkc) evaluates a sequence of values for the Bessel function of the first kind $J_\alpha(z)$, where z is complex and non-zero and α is the order with $0 \leq \alpha < 1$. The $(|N| + 1)$-member sequence is generated for orders $\alpha, \alpha + 1, \ldots, \alpha + |N|$ when $N \geq 0$. Note that $+$ is replaced by $-$ when $N < 0$. For positive orders the function may also be called with $z = 0$, since $J_q(0) = 0$ when $q > 0$. For negative orders the formula

$$J_{-q}(z) = \cos(\pi q) J_q(z) - \sin(\pi q) Y_q(z)$$

is used to generate the required sequence. The appropriate values of $J_q(z)$ and $Y_q(z)$ are obtained by calls to nag_complex_bessel_j (s17dec) and nag_complex_bessel_y (s17dcc).

4 References

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

5 Parameters

1: **z** – Complex *Input*

On entry: the argument z of the function.

Constraint: $\mathbf{z} \neq (0.0, 0.0)$ when $\mathbf{nl} < 0$.

2: **a** – double *Input*

On entry: the order α of the first member in the required sequence of function values.

Constraint: $0.0 \leq \mathbf{a} < 1.0$.

3: **nl** – Integer *Input*

On entry: the value of N.

Constraint: $\text{abs}(\mathbf{nl}) \leq 101$.

4: **b**[dim] – Complex *Output*

Note: the dimension, dim, of the array **b** must be at least $\text{abs}(\mathbf{nl}) + 1$.

On exit: with **fail.code** = **NE_NOERROR** or **NW_SOME_PRECISION_LOSS**, the required sequence of function values: **b**$[n-1]$ contains $J_{\alpha+n-1}(z)$ if $\mathbf{nl} \geq 0$ and $J_{\alpha-n+1}(z)$ otherwise, for $n = 1, 2, \ldots, \text{abs}(\mathbf{nl}) + 1$.

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

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, abs(**nl**) = ⟨*value*⟩.
Constraint: abs(**nl**) ≤ 101.

On entry, **z** = (0.0, 0.0) when **nl** < 0: **nl** = ⟨*value*⟩.

NE_OVERFLOW_LIKELY

Computation abandoned due to the likelihood of overflow.

NE_REAL

On entry, **a** = ⟨*value*⟩.
Constraint: **a** < 1.0.

On entry, **a** = ⟨*value*⟩.
Constraint: **a** ≥ 0.0.

NE_TERMINATION_FAILURE

Computation abandoned due to failure to satisfy the termination condition.

NE_TOTAL_PRECISION_LOSS

Computation abandoned due to total loss of precision.

NW_SOME_PRECISION_LOSS

Computation completed but some precision has been lost.

NE_BAD_PARAM

On entry, parameter ⟨*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 consult NAG for assistance.

7 Accuracy

All constants in nag_complex_bessel_y (s17dcc) and nag_complex_bessel_j (s17dec) are specified to approximately 18 digits of precision. If t denotes the number of digits of precision in the floating-point arithmetic being used, then clearly the maximum number of correct digits in the results obtained is limited by $p = \min(t, 18)$. Because of errors in argument reduction when computing elementary functions inside nag_complex_bessel_y (s17dcc) and nag_complex_bessel_j (s17dec), the actual number of correct digits is limited, in general, by $p - s$, where $s \approx \max(1, |\log_{10}|z||, |\log_{10}|\alpha||)$ represents the number of digits lost due to the argument reduction. Thus the larger the values of $|z|$ and $|\alpha|$, the less the precision in the result.

8 Further Comments

None.

9 Example

The example program evaluates $J_0(z), J_1(z), J_2(z)$ and $J_3(z)$ at $z = 0.6 - 0.8i$, and prints the results.

9.1 Program Text

```
/* nag_complex_bessel_j_seq (s18gkc) Example Program
 *
 * Copyright 2002 Numerical Algorithms Group.
 *
 * Mark 7, 2002.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
  Complex z, b[20];
  double a, alpha;
  Integer i, nl;

  Integer exit_status = EXIT_SUCCESS;
  NagError fail;

  INIT_FAIL(fail);

  /* Skip heading in data file */
  Vscanf("%*[^\n]");
  Vprintf("s18gkc Example Program Results\n");
  while (scanf(" (%lf,%lf) %lf %ld%*[^\n] ", &z.re, &z.im, &a, &nl) != EOF)
    {
      s18gkc(z, a, nl, b, &fail);
      if (fail.code == NE_NOERROR)
        {
          Vprintf("         z             a       nl\n");
          Vprintf(" (%7.3f,%7.3f)  %lf    %ld\n\n", z.re, z.im, a, nl);
          Vprintf("Requested values of J_alpha(Z)\n\n");
          alpha = a;
          Vprintf("     alpha              J_alpha(z)\n");
          for (i = 0; i < ABS(nl) + 1; i++)
            {
              Vprintf("%12.4e   (%12.4e,%12.4e)\n", alpha, b[i].re, b[i].im);
              if (nl > 0)
                alpha += 1.0;
              else
                alpha -= 1.0;
            }
        }
      else
        {
          Vprintf("Error from s18gkc.\n%s\n", fail.message);
          exit_status = 1;
          goto END;
        }
    }
 END:
  return exit_status;
}
```

9.2 Program Data

```
s18gkc Example Program Data
 ( 0.6,-0.8)   0.0   3  - Values of z, a and nl
```

9.3 Program Results

```
s18gkc Example Program Results
         z             a       nl
 (  0.600, -0.800)  0.000000    3

Requested values of J_alpha(Z)
```

```
    alpha             J_alpha(z)
 0.0000e+00   (  1.0565e+00,  2.4811e-01)
 1.0000e+00   (  3.5825e-01, -3.7539e-01)
 2.0000e+00   ( -2.5974e-02, -1.2538e-01)
 3.0000e+00   ( -1.9369e-02, -8.6380e-03)
```