nag_elliptic_integral_rj (s21bdc)

1. Purpose

nag_elliptic_integral_rj (s21bdc) returns a value of the symmetrised elliptic integral of the third kind.

2. Specification

```
#include <nag.h>
#include <nags.h>

double nag_elliptic_integral_rj(double x, double y, double z, double r,
       NagError *fail)
```

3. Description

This function calculates an approximation to the integral

$$R_J(x,y,z,\rho) = \frac{3}{2}\int_0^\infty \frac{dt}{(t+\rho)\sqrt{(t+x)(t+y)(t+z)}}$$

where x, y, $z \geq 0$, $\rho \neq 0$ and at most one of x, y and z is zero.

If $\rho < 0$, the result computed is the Cauchy principal value of the integral.

The basic algorithm, which is due to Carlson (1978) and Carlson (1988), is to reduce the arguments recursively towards their mean by the rule:

$$\begin{array}{rcc}
x_0 & = & x, y_0 = y, z_0 = z, \rho_0 = \rho \\
\mu_n & = & (x_n + y_n + z_n + 2\rho_n)/5 \\
X_n & = & 1 - x_n/\mu_n \\
Y_n & = & 1 - y_n/\mu_n \\
Z_n & = & 1 - z_n/\mu_n \\
P_n & = & 1 - \rho_n/\mu_n \\
\lambda_n & = & \sqrt{x_n y_n} + \sqrt{y_n z_n} + \sqrt{z_n x_n} \\
x_{n+1} & = & (x_n + \lambda_n)/4 \\
y_{n+1} & = & (y_n + \lambda_n)/4 \\
z_{n+1} & = & (z_n + \lambda_n)/4 \\
\rho_{n+1} & = & (\rho_n + \lambda_n)/4 \\
\alpha_n & = & \left[\rho_n\left(\sqrt{x_n} + \sqrt{y_n} + \sqrt{z_n}\right) + \sqrt{x_n y_n z_n}\right]^2 \\
\beta_n & = & \rho_n(\rho_n + \lambda_n)^2
\end{array} .$$

For n sufficiently large,

$$\epsilon_n = \max(|X_n|, |Y_n|, |Z_n|, |P_n|) \sim 1/4^n$$

and the function may be approximated by a 5th-order power series

$$\begin{array}{rcl}
R_J(x,y,z,\rho) & = & 3\sum_{m=0}^{n-1} 4^{-m} R_C(\alpha_m, \beta_m) \\
& + & \frac{4^{-n}}{\sqrt{\mu_n^3}}\left(1 + \frac{3}{7}S_n^{(2)} + \frac{1}{3}S_n^{(3)} + \frac{3}{22}(S_n^{(2)})^2 + \frac{3}{11}S_n^{(4)} + \frac{3}{13}S_n^{(2)}S_n^{(3)} + \frac{3}{13}S_n^{(5)}\right),
\end{array}$$

where $S_n^{(m)} = (X_n^m + Y_n^m + Z_n^m + 2P_n^m)/2m$.

The truncation error in this expansion is bounded by $3\epsilon_n^6/\sqrt{(1-\epsilon_n)^3}$ and the recursion process is terminated when this quantity is negligible compared with the ***machine precision***. The function may fail either because it has been called with arguments outside the domain of definition or with arguments so extreme that there is an unavoidable danger of setting underflow or overflow.

Note: $R_J(x,x,x,x) = x^{-3/2}$, so there exists a region of extreme arguments for which the function value is not representable.

4. Parameters

x
y
z
r

Input: the arguments x, y, z and ρ of the function.
Constraint: $\mathbf{x}, \mathbf{y}, \mathbf{z} \geq 0.0$, $\mathbf{r} \neq 0.0$ and at most one of x, y and z may be zero.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_REAL_ARG_LT

On entry, **x** must not be less than 0.0: $\mathbf{x} = \langle value \rangle$.
On entry, **y** must not be less than 0.0: $\mathbf{y} = \langle value \rangle$.
On entry, **z** must not be less than 0.0: $\mathbf{z} = \langle value \rangle$.
The function is undefined.
On entry, $\langle parameters \rangle$ must not be less than $\langle value \rangle$: $\langle parameters \rangle = \langle value \rangle$.
On entry, either **r** is too close to zero, or any two of **x**, **y** and **z** are too close to zero; there is a danger of setting overflow.

NE_REAL_ARG_EQ

On entry, $\langle parameters \rangle$ must not be equal to 0.0: $\langle parameters \rangle = \langle value \rangle$.
At least two of **x**, **y** and **z** are zero and the function is undefined.
On entry, **r** must not be equal to 0.0: $\mathbf{r} = \langle value \rangle$.
The function is undefined.

NE_REAL_ARG_GT

On entry, **x** must not be greater than $\langle value \rangle$: $\mathbf{x} = \langle value \rangle$.
On entry, **y** must not be greater than $\langle value \rangle$: $\mathbf{y} = \langle value \rangle$.
On entry, **z** must not be greater than $\langle value \rangle$: $\mathbf{z} = \langle value \rangle$.
On entry, $|\mathbf{r}|$ must not be greater than $\langle value \rangle$: $|\mathbf{r}| = \langle value \rangle$.

6. Further Comments

If the argument **r** is equal to any of the other arguments, the function reduces to the integral R_D, computed by nag_elliptic_integral_rd (s21bcc).

Symmetrised elliptic integrals returned by functions nag_elliptic_integral_rj, nag_elliptic_integral_rc (s21bac), nag_elliptic_integral_rf (s21bbc) and nag_elliptic_integral_rd (s21bcc) can be related to the more traditional canonical forms (see Abramowitz and Stegun (1968)), as described in the Chapter Introduction.

6.1. Accuracy

In principle the function is capable of producing full ***machine precision***. However round off errors in internal arithmetic will result in slight loss of accuracy. This loss should never be excessive as the algorithm does not involve any significant amplification of round off error. It is reasonable to assume that the result is accurate to within a small multiple of the ***machine precision***.

6.2. References

Abramowitz M and Stegun I A (1968) *Handbook of Mathematical Functions* Dover Publications, New York ch 17.

Carlson B C (1978) *Computing Elliptic Integrals by Duplication* Department of Physics, Iowa State University (Preprint).

Carlson B C (1988) A Table of Elliptic Integrals of the Third Kind *Math. Comp.* **51** 267–280.

7. See Also

nag_elliptic_integral_rc (s21bac)
nag_elliptic_integral_rf (s21bbc)
nag_elliptic_integral_rd (s21bcc)

8. Example

This example program simply generates a small set of non-extreme arguments which are used with the function to produce the table of low accuracy results.

8.1. Program Text

```
/* nag_elliptic_integral_rj(s21bdc) Example Program
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 2 revised, 1992.
 */

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

main ()
{
  double r, rj, x, y, z;
  Integer ix, iy, iz;

  Vprintf("s21bdc Example Program Results\n");
  Vprintf("      x      y      z      r       s21bdc  \n");
  for ( ix = 1; ix <= 3; ix++){
    x = ix*0.5;
    for ( iy=ix; iy<=3; iy++){
      y = iy*0.5;
      for ( iz=iy; iz<=3; iz++){
        z = iz*0.5;
        r = 2.0;
        rj = s21bdc(x, y, z, r, NAGERR_DEFAULT);
        Vprintf (" %7.2f%7.2f%7.2f%7.2f%12.4f\n", x, y, z, r, rj);
      }
    }
  }
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

None.

8.3. Program Results

```
s21bdc Example Program Results
      x      y      z      r       s21bdc
    0.50   0.50   0.50   2.00      1.1184
    0.50   0.50   1.00   2.00      0.9221
    0.50   0.50   1.50   2.00      0.8115
    0.50   1.00   1.00   2.00      0.7671
    0.50   1.00   1.50   2.00      0.6784
    0.50   1.50   1.50   2.00      0.6017
    1.00   1.00   1.00   2.00      0.6438
    1.00   1.00   1.50   2.00      0.5722
    1.00   1.50   1.50   2.00      0.5101
    1.50   1.50   1.50   2.00      0.4561
```