NAG Toolbox

nag_specfun_bessel_i0_real_vector (s18as)

1 Purpose

nag_specfun_bessel_i0_real_vector (s18as) returns an array of values of the modified Bessel function $I_0(x)$.

2 Syntax

```
[f, ivalid, ifail] = nag_specfun_bessel_i0_real_vector(x, 'n', n)
[f, ivalid, ifail] = s18as(x, 'n', n)
```

3 Description

nag_specfun_bessel_i0_real_vector (s18as) evaluates an approximation to the modified Bessel function of the first kind $I_0(x_i)$ for an array of arguments x_i, for $i = 1, 2, \ldots, n$.

Note: $I_0(-x) = I_0(x)$, so the approximation need only consider $x \geq 0$.

The function is based on three Chebyshev expansions:

For $0 < x \leq 4$,

$$I_0(x) = e^x \sum_{r=0} a_r T_r(t), \qquad \text{where } t = 2\left(\frac{x}{4}\right) - 1.$$

For $4 < x \leq 12$,

$$I_0(x) = e^x \sum_{r=0} b_r T_r(t), \qquad \text{where } t = \frac{x-8}{4}.$$

For $x > 12$,

$$I_0(x) = \frac{e^x}{\sqrt{x}} \sum_{r=0} c_r T_r(t), \qquad \text{where } t = 2\left(\frac{12}{x}\right) - 1.$$

For small x, $I_0(x) \simeq 1$. This approximation is used when x is sufficiently small for the result to be correct to ***machine precision***.

For large x, the function must fail because of the danger of overflow in calculating e^x.

4 References

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

5 Parameters

5.1 Compulsory Input Parameters

1: **x**(**n**) – REAL (KIND=nag_wp) array

The argument x_i of the function, for $i = 1, 2, \ldots, \mathbf{n}$.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the dimension of the array **x**.

n, the number of points.

Constraint: $\mathbf{n} \geq 0$.

5.3 Output Parameters

1: **f**(**n**) – REAL (KIND=nag_wp) array

$I_0(x_i)$, the function values.

2: **ivalid**(**n**) – INTEGER array

ivalid(i) contains the error code for x_i, for $i = 1, 2, \ldots, \mathbf{n}$.

ivalid$(i) = 0$
No error.

ivalid$(i) = 1$
x_i is too large. $\mathbf{f}(i)$ contains the approximate value of $I_0(x_i)$ at the nearest valid argument. The threshold value is the same as for **ifail** $= 1$ in nag_specfun_bessel_i0_real (s18ae), as defined in the Users' Note for your implementation.

3: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$ (*warning*)

On entry, at least one value of **x** was invalid.
Check **ivalid** for more information.

ifail $= 2$

Constraint: $\mathbf{n} \geq 0$.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

Let δ and ϵ be the relative errors in the argument and result respectively.

If δ is somewhat larger than the ***machine precision*** (i.e., if δ is due to data errors etc.), then ϵ and δ are approximately related by:

$$\epsilon \simeq \left|\frac{xI_1(x)}{I_0(x)}\right|\delta.$$

Figure 1 shows the behaviour of the error amplification factor

$$\left|\frac{xI_1(x)}{I_0(x)}\right|.$$

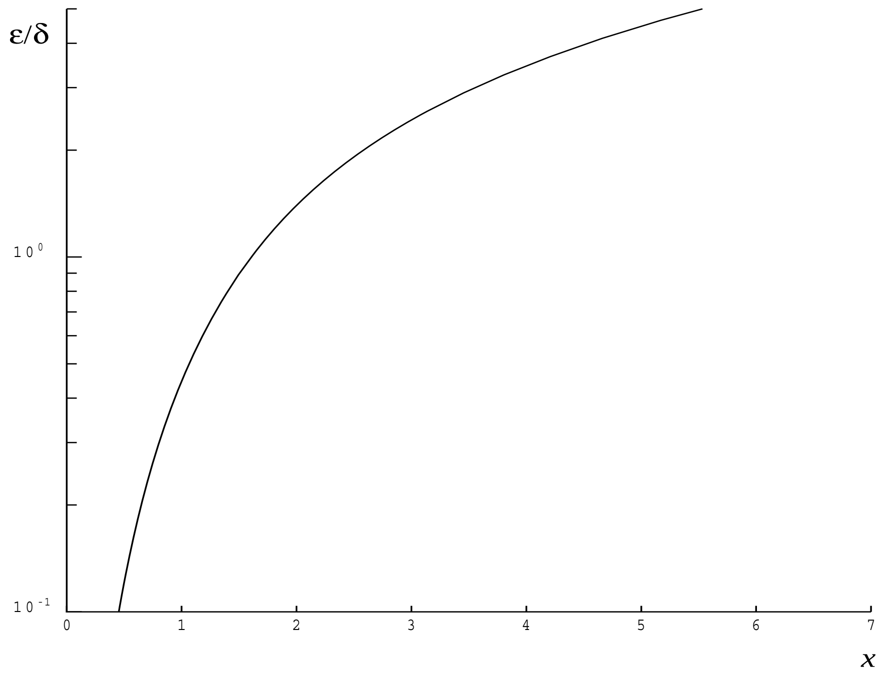

Figure 1

However if δ is of the same order as ***machine precision***, then rounding errors could make ϵ slightly larger than the above relation predicts.

For small x the amplification factor is approximately $\frac{x^2}{2}$, which implies strong attenuation of the error, but in general ϵ can never be less than the ***machine precision***.

For large x, $\epsilon \simeq x\delta$ and we have strong amplification of errors. However, for quite moderate values of x ($x > \hat{x}$, the threshold value), the function must fail because $I_0(x)$ would overflow; hence in practice the loss of accuracy for x close to $\hat{x}$ is not excessive and the errors will be dominated by those of the standard function exp.

8 Further Comments

None.

9 Example

This example reads values of **x** from a file, evaluates the function at each value of x_i and prints the results.

9.1 Program Text

```
function s18as_example

fprintf('s18as example results\n\n');

x = [0; 0.5; 1; 3; 6; 8; 10; 15; 20; -1];

[f, ivalid, ifail] = s18as(x);

fprintf('     x           I_0(x)   ivalid\n');
for i=1:numel(x)
  fprintf('%12.3e%12.3e%5d\n', x(i), f(i), ivalid(i));
end
```

9.2 Program Results

```
s18as example results

     x           I_0(x)   ivalid
  0.000e+00   1.000e+00    0
  5.000e-01   1.063e+00    0
  1.000e+00   1.266e+00    0
  3.000e+00   4.881e+00    0
  6.000e+00   6.723e+01    0
  8.000e+00   4.276e+02    0
  1.000e+01   2.816e+03    0
  1.500e+01   3.396e+05    0
  2.000e+01   4.356e+07    0
 -1.000e+00   1.266e+00    0
```