

NAG Library Function Document

nag_bessel_y1_vector (s17arc)

1 Purpose

nag_bessel_y1_vector (s17arc) returns an array of values of the Bessel function $Y_1(x)$.

2 Specification

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

void nag_bessel_y1_vector (Integer n, const double x[], double f[],
                          Integer ivalid[], NagError *fail)
```

3 Description

nag_bessel_y1_vector (s17arc) evaluates an approximation to the Bessel function of the second kind $Y_1(x_i)$ for an array of arguments x_i , for $i = 1, 2, \dots, n$.

Note: $Y_1(x)$ is undefined for $x \leq 0$ and the function will fail for such arguments.

The function is based on four Chebyshev expansions:

For $0 < x \leq 8$,

$$Y_1(x) = \frac{2}{\pi} \ln x \sum_{r=0}^x a_r T_r(t) - \frac{2}{\pi x} + \frac{x}{8} \sum_{r=0}^x b_r T_r(t), \quad \text{with } t = 2\left(\frac{x}{8}\right)^2 - 1.$$

For $x > 8$,

$$Y_1(x) = \sqrt{\frac{2}{\pi x}} \left\{ P_1(x) \sin\left(x - 3\frac{\pi}{4}\right) + Q_1(x) \cos\left(x - 3\frac{\pi}{4}\right) \right\}$$

where $P_1(x) = \sum_{r=0}^x c_r T_r(t)$,

and $Q_1(x) = \frac{8}{x} \sum_{r=0}^x d_r T_r(t)$, with $t = 2\left(\frac{8}{x}\right)^2 - 1$.

For x near zero, $Y_1(x) \simeq -\frac{2}{\pi x}$. This approximation is used when x is sufficiently small for the result to

be correct to **machine precision**. For extremely small x , there is a danger of overflow in calculating $-\frac{2}{\pi x}$ and for such arguments the function will fail.

For very large x , it becomes impossible to provide results with any reasonable accuracy (see Section 7), hence the function fails. Such arguments contain insufficient information to determine the phase of oscillation of $Y_1(x)$; only the amplitude, $\sqrt{\frac{2}{\pi x}}$, can be determined and this is returned on failure. The range for which this occurs is roughly related to **machine precision**; the function will fail if $x \gtrsim 1/\text{machine precision}$ (see the Users' Note for your implementation for details).

4 References

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

Clenshaw C W (1962) Chebyshev Series for Mathematical Functions *Mathematical tables* HMSO

5 Arguments

- 1: **n** – Integer *Input*
On entry: n , the number of points.
Constraint: $n \geq 0$.
- 2: **x[n]** – const double *Input*
On entry: the argument x_i of the function, for $i = 1, 2, \dots, n$.
Constraint: $x[i - 1] > 0.0$, for $i = 1, 2, \dots, n$.
- 3: **f[n]** – double *Output*
On exit: $Y_1(x_i)$, the function values.
- 4: **ivalid[n]** – Integer *Output*
On exit: **ivalid**[$i - 1$] contains the error code for x_i , for $i = 1, 2, \dots, n$.
ivalid[$i - 1$] = 0
 No error.
ivalid[$i - 1$] = 1
 On entry, x_i is too large. **f**[$i - 1$] contains the amplitude of the Y_1 oscillation, $\sqrt{\frac{2}{\pi x_i}}$.
ivalid[$i - 1$] = 2
 On entry, $x_i \leq 0.0$, Y_1 is undefined. **f**[$i - 1$] contains 0.0.
ivalid[$i - 1$] = 3
 x_i is too close to zero, there is a danger of overflow. On failure, **f**[$i - 1$] contains the value of $Y_1(x)$ at the smallest valid argument.
- 5: **fail** – NagError * *Input/Output*
 The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_BAD_PARAM

On entry, argument $\langle value \rangle$ had an illegal value.

NE_INT

On entry, $n = \langle value \rangle$.
 Constraint: $n \geq 0$.

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 contact NAG for assistance.

NW_INVALID

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

7 Accuracy

Let δ be the relative error in the argument and E be the absolute error in the result. (Since $Y_1(x)$ oscillates about zero, absolute error and not relative error is significant, except for very small x .)

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

$$E \simeq |xY_0(x) - Y_1(x)|\delta$$

(provided E is also within machine bounds). Figure 1 displays the behaviour of the amplification factor $|xY_0(x) - Y_1(x)|$.

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

For very small x , absolute error becomes large, but the relative error in the result is of the same order as δ .

For very large x , the above relation ceases to apply. In this region, $Y_1(x) \simeq \sqrt{\frac{2}{\pi x}} \sin\left(x - \frac{3\pi}{4}\right)$. The amplitude $\sqrt{\frac{2}{\pi x}}$ can be calculated with reasonable accuracy for all x , but $\sin\left(x - \frac{3\pi}{4}\right)$ cannot. If $x - \frac{3\pi}{4}$ is written as $2N\pi + \theta$ where N is an integer and $0 \leq \theta < 2\pi$, then $\sin\left(x - \frac{3\pi}{4}\right)$ is determined by θ only. If $x > \delta^{-1}$, θ cannot be determined with any accuracy at all. Thus if x is greater than, or of the order of, the inverse of the **machine precision**, it is impossible to calculate the phase of $Y_1(x)$ and the function must fail.

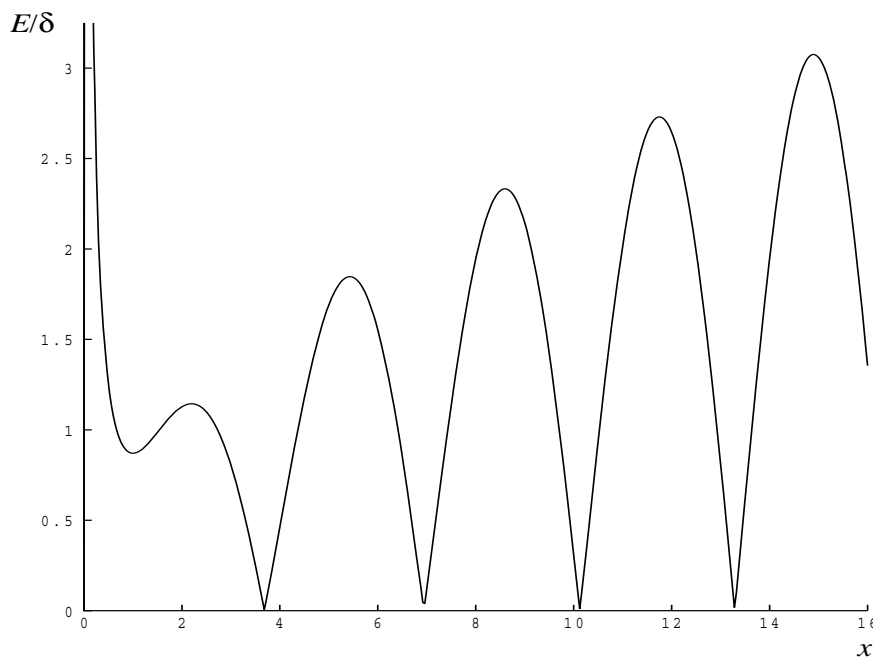


Figure 1

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

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

10.1 Program Text

```

/* nag_bessel_y1_vector (s17arc) Example Program.
 *
 * Copyright 2011, Numerical Algorithms Group.
 *
 * Mark 23 2011.
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
    Integer    exit_status = 0;
    Integer    i, n;
    double     *f = 0, *x = 0;
    Integer    *ivalid = 0;
    NagError   fail;

    INIT_FAIL(fail);

    /* Skip heading in data file */
    scanf("%*[^\\n]");

    printf("nag_bessel_y1_vector (s17arc) Example Program Results\\n");
    printf("\\n");
    printf("      x          f          ivalid\\n");
    printf("\\n");
    scanf("%ld", &n);
    scanf("%*[^\\n]");

    /* Allocate memory */
    if (!(x = NAG_ALLOC(n, double)) ||
        !(f = NAG_ALLOC(n, double)) ||
        !(ivalid = NAG_ALLOC(n, Integer)))
    {
        printf("Allocation failure\\n");
        exit_status = -1;
        goto END;
    }

    for (i=0; i<n; i++)
        scanf("%lf", &x[i]);
    scanf("%*[^\\n]");

    /* nag_bessel_y1_vector (s17arc).
     * Bessel function Y_1(x)
     */
    nag_bessel_y1_vector(n, x, f, ivalid, &fail);
    if (fail.code!=NE_NOERROR && fail.code!=NW_IVALID)
    {
        printf("Error from nag_bessel_y1_vector (s17arc).\\n%s\\n",
              fail.message);
        exit_status = 1;
        goto END;
    }

    for (i=0; i<n; i++)
        printf(" %11.3e %11.3e %4ld\\n", x[i], f[i], ivalid[i]);

    END:
    NAG_FREE(f);

```

```
NAG_FREE(x);  
NAG_FREE(ivalid);  
  
return exit_status;  
}
```

10.2 Program Data

nag_bessel_y1_vector (s17arc) Example Program Data

7

0.5 1.0 3.0 6.0 8.0 10.0 1000.0

10.3 Program Results

nag_bessel_y1_vector (s17arc) Example Program Results

x	f	ivalid
5.000e-01	-1.471e+00	0
1.000e+00	-7.812e-01	0
3.000e+00	3.247e-01	0
6.000e+00	-1.750e-01	0
8.000e+00	-1.581e-01	0
1.000e+01	2.490e-01	0
1.000e+03	-2.478e-02	0
