NAG Library Function Document

nag_fresnel_s_vector (s20aqc)

1 Purpose

nag_fresnel_s_vector (s20aqc) returns an array of values for the Fresnel integral $S(x)$.

2 Specification

```
#include <nag.h>
#include <nags.h>
void nag_fresnel_s_vector (Integer n, const double x[], double f[],
     NagError *fail)
```

3 Description

nag_fresnel_s_vector (s20aqc) evaluates an approximation to the Fresnel integral

$$S(x_i) = \int_0^{x_i} \sin\left(\frac{\pi}{2}t^2\right) dt$$

for an array of arguments x_i, for $i = 1, 2, \ldots, n$.

Note: $S(x) = -S(-x)$, so the approximation need only consider $x \geq 0.0$.

The function is based on three Chebyshev expansions:

For $0 < x \leq 3$,

$$S(x) = x^3 \sum_{r=0} a_r T_r(t), \qquad \text{with } t = 2\left(\frac{x}{3}\right)^4 - 1.$$

For $x > 3$,

$$S(x) = \frac{1}{2} - \frac{f(x)}{x}\cos\left(\frac{\pi}{2}x^2\right) - \frac{g(x)}{x^3}\sin\left(\frac{\pi}{2}x^2\right),$$

where $f(x) = \sum_{r=0} b_r T_r(t)$,

and $g(x) = \sum_{r=0} c_r T_r(t)$,

with $t = 2\left(\frac{3}{x}\right)^4 - 1$.

For small x, $S(x) \simeq \frac{\pi}{6}x^3$. This approximation is used when x is sufficiently small for the result to be correct to ***machine precision***. For very small x, this approximation would underflow; the result is then set exactly to zero.

For large x, $f(x) \simeq \frac{1}{\pi}$ and $g(x) \simeq \frac{1}{\pi^2}$. Therefore for moderately large x, when $\frac{1}{\pi^2 x^3}$ is negligible compared with $\frac{1}{2}$, the second term in the approximation for $x > 3$ may be dropped. For very large x, when $\frac{1}{\pi x}$ becomes negligible, $S(x) \simeq \frac{1}{2}$. However there will be considerable difficulties in calculating $\cos\left(\frac{\pi}{2}x^2\right)$ accurately before this final limiting value can be used. Since $\cos\left(\frac{\pi}{2}x^2\right)$ is periodic, its value is essentially determined by the fractional part of x^2. If $x^2 = N + \theta$ where N is an integer and $0 \leq \theta < 1$, then $\cos\left(\frac{\pi}{2}x^2\right)$ depends on θ and on N modulo 4. By exploiting this fact, it is possible to retain significance in the calculation of $\cos\left(\frac{\pi}{2}x^2\right)$ either all the way to the very large x limit, or at least until the integer part of $\frac{x}{2}$ is equal to the maximum integer allowed on the machine.

4 References

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

5 Arguments

1: **n** – Integer *Input*

On entry: n, the number of points.

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

2: **x**[**n**] – const double *Input*

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

3: **f**[**n**] – double *Output*

On exit: $S(x_i)$, the function values.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

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

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.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in the Essential Introduction for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in the Essential Introduction for further information.

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:

$$\epsilon \simeq \left| \frac{x \sin\left(\frac{\pi}{2}x^2\right)}{S(x)} \right| \delta.$$

Figure 1 shows the behaviour of the error amplification factor $\left| \frac{x \sin\left(\frac{\pi}{2}x^2\right)}{S(x)} \right|$.

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

For small x, $\epsilon \simeq 3\delta$ and hence there is only moderate amplification of relative error. Of course for very small x where the correct result would underflow and exact zero is returned, relative error-control is lost.

For moderately large values of x,

$$\epsilon \simeq \left| 2x \sin\left(\frac{\pi}{2}x^2\right) \right| \delta$$

and the result will be subject to increasingly large amplification of errors. However the above relation breaks down for large values of x (i.e., when $\frac{1}{x^2}$ is of the order of the ***machine precision***); in this region the relative error in the result is essentially bounded by $\frac{2}{\pi x}$.

Hence the effects of error amplification are limited and at worst the relative error loss should not exceed half the possible number of significant figures.

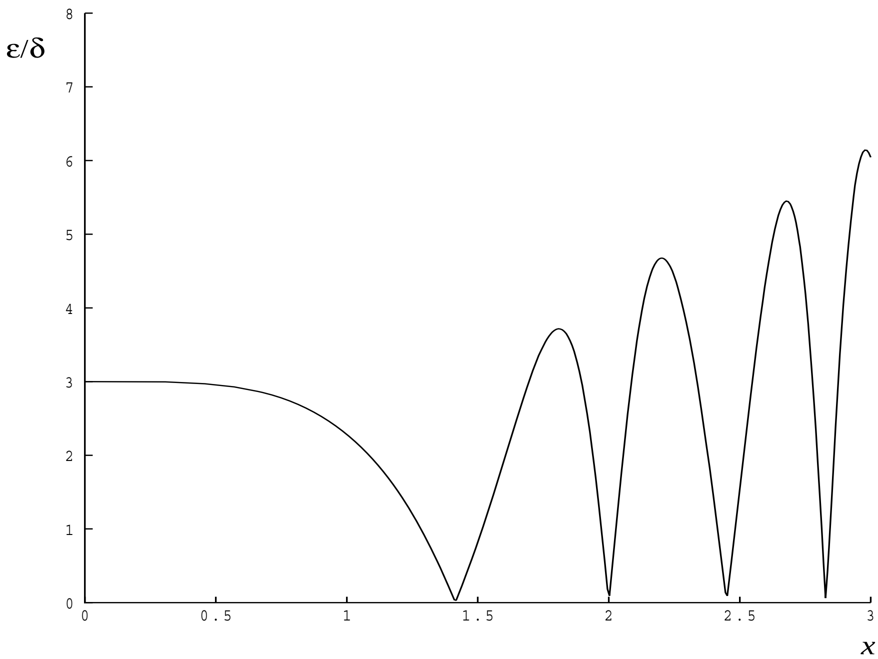

Figure 1

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_fresnel_s_vector (s20aqc) Example Program.
 *
 * Copyright 2014 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;
  NagError fail;

  INIT_FAIL(fail);

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

  printf("nag_fresnel_s_vector (s20aqc) Example Program Results\n");
  printf("\n");
  printf("     x           f\n");
  printf("\n");
#ifdef _WIN32
  scanf_s("%"NAG_IFMT"", &n);
#else
  scanf("%"NAG_IFMT"", &n);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif

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

  for (i=0; i<n; i++)
#ifdef _WIN32
    scanf_s("%lf", &x[i]);
#else
    scanf("%lf", &x[i]);
#endif
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif
```

```
  /* nag_fresnel_s_vector (s20aqc).
   * Fresnel Integral S(x)
   */
  nag_fresnel_s_vector(n, x, f, &fail);
  if (fail.code!=NE_NOERROR)
    {
      printf("Error from nag_fresnel_s_vector (s20aqc).\n%s\n",
             fail.message);
      exit_status = 1;
      goto END;
    }

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

 END:
  NAG_FREE(f);
  NAG_FREE(x);

  return exit_status;
}
```

10.2 Program Data

```
nag_fresnel_s_vector (s20aqc) Example Program Data

11

0.0 0.5 1.0 2.0 4.0 5.0 6.0 8.0 10.0 -1.0 1000.0
```

10.3 Program Results

```
nag_fresnel_s_vector (s20aqc) Example Program Results

     x           f

   0.000e+00   0.000e+00
   5.000e-01   6.473e-02
   1.000e+00   4.383e-01
   2.000e+00   3.434e-01
   4.000e+00   4.205e-01
   5.000e+00   4.992e-01
   6.000e+00   4.470e-01
   8.000e+00   4.602e-01
   1.000e+01   4.682e-01
  -1.000e+00  -4.383e-01
   1.000e+03   4.997e-01
```