6.9 Fresnel Integrals, Cosine and Sine Integrals

Fresnel Integrals

The two Fresnel integrals are defined by

$$C(x) = \int_0^x \cos\left(\frac{\pi}{2}t^2\right) dt, \qquad S(x) = \int_0^x \sin\left(\frac{\pi}{2}t^2\right) dt \tag{6.9.1}$$

The most convenient way of evaluating these functions to arbitrary precision is to use power series for small x and a continued fraction for large x. The series are

$$\begin{aligned} C(x) &= x - \left(\frac{\pi}{2}\right)^2 \frac{x^5}{5 \cdot 2!} + \left(\frac{\pi}{2}\right)^4 \frac{x^9}{9 \cdot 4!} - \cdots \\ S(x) &= \left(\frac{\pi}{2}\right) \frac{x^3}{3 \cdot 1!} - \left(\frac{\pi}{2}\right)^3 \frac{x^7}{7 \cdot 3!} + \left(\frac{\pi}{2}\right)^5 \frac{x^{11}}{11 \cdot 5!} - \cdots \end{aligned} \tag{6.9.2}$$

There is a complex continued fraction that yields both $S(x)$ and $C(x)$ simultaneously:

$$C(x) + iS(x) = \frac{1+i}{2} \operatorname{erf} z, \qquad z = \frac{\sqrt{\pi}}{2}(1-i)x \tag{6.9.3}$$

where

$$\begin{aligned} e^{z^2} \operatorname{erfc} z &= \frac{1}{\sqrt{\pi}} \left(\frac{1}{z+} \; \frac{1/2}{z+} \; \frac{1}{z+} \; \frac{3/2}{z+} \; \frac{2}{z+} \cdots \right) \\ &= \frac{2z}{\sqrt{\pi}} \left(\frac{1}{2z^2+1-} \; \frac{1 \cdot 2}{2z^2+5-} \; \frac{3 \cdot 4}{2z^2+9-} \cdots \right) \end{aligned} \tag{6.9.4}$$

In the last line we have converted the "standard" form of the continued fraction to its "even" form (see §5.2), which converges twice as fast. We must be careful not to evaluate the alternating series (6.9.2) at too large a value of x; inspection of the terms shows that $x = 1.5$ is a good point to switch over to the continued fraction.

Note that for large x

$$C(x) \sim \frac{1}{2} + \frac{1}{\pi x} \sin\left(\frac{\pi}{2}x^2\right), \qquad S(x) \sim \frac{1}{2} - \frac{1}{\pi x} \cos\left(\frac{\pi}{2}x^2\right) \tag{6.9.5}$$

Thus the precision of the routine `frenel` may be limited by the precision of the library routines for sine and cosine for large x.

```
#include <math.h>
#include "complex.h"
#define EPS 6.0e-8
#define MAXIT 100
#define FPMIN 1.0e-30
#define XMIN 1.5
#define PI 3.1415927
#define PIBY2 (PI/2.0)
```

Here EPS is the relative error; MAXIT is the maximum number of iterations allowed; FPMIN is a number near the smallest representable floating-point number; XMIN is the dividing line between using the series and continued fraction.

```
#define TRUE 1
#define ONE Complex(1.0,0.0)

void frenel(float x, float *s, float *c)
```

Computes the Fresnel integrals $S(x)$ and $C(x)$ for all real x.

```
{
    void nrerror(char error_text[]);
    int k,n,odd;
    float a,ax,fact,pix2,sign,sum,sumc,sums,term,test;
    fcomplex b,cc,d,h,del,cs;

    ax=fabs(x);
    if (ax < sqrt(FPMIN)) {                Special case: avoid failure of convergence
        *s=0.0;                                 test because of underflow.
        *c=ax;
    } else if (ax <= XMIN) {               Evaluate both series simultaneously.
        sum=sums=0.0;
        sumc=ax;
        sign=1.0;
        fact=PIBY2*ax*ax;
        odd=TRUE;
        term=ax;
        n=3;
        for (k=1;k<=MAXIT;k++) {
            term *= fact/k;
            sum += sign*term/n;
            test=fabs(sum)*EPS;
            if (odd) {
                sign = -sign;
                sums=sum;
                sum=sumc;
            } else {
                sumc=sum;
                sum=sums;
            }
            if (term < test) break;
            odd=!odd;
            n += 2;
        }
        if (k > MAXIT) nrerror("series failed in frenel");
        *s=sums;
        *c=sumc;
    } else {                               Evaluate continued fraction by modified
        pix2=PI*ax*ax;                          Lentz's method (§5.2).
        b=Complex(1.0,-pix2);
        cc=Complex(1.0/FPMIN,0.0);
        d=h=Cdiv(ONE,b);
        n = -1;
        for (k=2;k<=MAXIT;k++) {
            n += 2;
            a = -n*(n+1);
            b=Cadd(b,Complex(4.0,0.0));
            d=Cdiv(ONE,Cadd(RCmul(a,d),b));    Denominators cannot be zero.
```

```
            cc=Cadd(b,Cdiv(Complex(a,0.0),cc));
            del=Cmul(cc,d);
            h=Cmul(h,del);
            if (fabs(del.r-1.0)+fabs(del.i) < EPS) break;
        }
        if (k > MAXIT) nrerror("cf failed in frenel");
        h=Cmul(Complex(ax,-ax),h);
        cs=Cmul(Complex(0.5,0.5),
            Csub(ONE,Cmul(Complex(cos(0.5*pix2),sin(0.5*pix2)),h)));
        *c=cs.r;
        *s=cs.i;
    }
    if (x < 0.0) {                              Use antisymmetry.
        *c = -(*c);
        *s = -(*s);
    }
}
```


Cosine and Sine Integrals

The cosine and sine integrals are defined by

$$\begin{aligned} \mathrm{Ci}(x) &= \gamma + \ln x + \int_0^x \frac{\cos t - 1}{t}\,dt \\ \mathrm{Si}(x) &= \int_0^x \frac{\sin t}{t}\,dt \end{aligned} \tag{6.9.6}$$

Here $\gamma \approx 0.5772\ldots$ is Euler's constant. We only need a way to calculate the functions for $x > 0$, because

$$\mathrm{Si}(-x) = -\,\mathrm{Si}(x), \qquad \mathrm{Ci}(-x) = \mathrm{Ci}(x) - i\pi \tag{6.9.7}$$

Once again we can evaluate these functions by a judicious combination of power series and complex continued fraction. The series are

$$\begin{aligned} \mathrm{Si}(x) &= x - \frac{x^3}{3\cdot 3!} + \frac{x^5}{5\cdot 5!} - \cdots \\ \mathrm{Ci}(x) &= \gamma + \ln x + \left(-\frac{x^2}{2\cdot 2!} + \frac{x^4}{4\cdot 4!} - \cdots\right) \end{aligned} \tag{6.9.8}$$

The continued fraction for the exponential integral $E_1(ix)$ is

$$\begin{aligned} E_1(ix) &= -\,\mathrm{Ci}(x) + i[\mathrm{Si}(x) - \pi/2] \\ &= e^{-ix}\left(\frac{1}{ix+}\;\frac{1}{1+}\;\frac{1}{ix+}\;\frac{2}{1+}\;\frac{2}{ix+}\cdots\right) \\ &= e^{-ix}\left(\frac{1}{1+ix-}\;\frac{1^2}{3+ix-}\;\frac{2^2}{5+ix-}\cdots\right) \end{aligned} \tag{6.9.9}$$

The "even" form of the continued fraction is given in the last line and converges twice as fast for about the same amount of computation. A good crossover point from the alternating series to the continued fraction is $x = 2$ in this case. As for the Fresnel integrals, for large x the precision may be limited by the precision of the sine and cosine routines.

```
#include <math.h>
#include "complex.h"
#define EPS 6.0e-8               Relative error, or absolute error near a zero of Ci(x).
#define EULER 0.57721566         Euler's constant γ.
#define MAXIT 100                Maximum number of iterations allowed.
#define PIBY2 1.5707963          π/2.
#define FPMIN 1.0e-30            Close to smallest representable floating-point number.
#define TMIN 2.0                 Dividing line between using the series and continued frac-
#define TRUE 1                       tion.
#define ONE Complex(1.0,0.0)
```

```
void cisi(float x, float *ci, float *si)
```
Computes the cosine and sine integrals $\mathrm{Ci}(x)$ and $\mathrm{Si}(x)$. $\mathrm{Ci}(0)$ is returned as a large negative number and no error message is generated. For $x < 0$ the routine returns $\mathrm{Ci}(-x)$ and you must supply the $-i\pi$ yourself.

```
{
    void nrerror(char error_text[]);
    int i,k,odd;
    float a,err,fact,sign,sum,sumc,sums,t,term;
    fcomplex h,b,c,d,del;

    t=fabs(x);
    if (t == 0.0) {                          Special case.
        *si=0.0;
        *ci = -1.0/FPMIN;
        return;
    }
    if (t > TMIN) {                          Evaluate continued fraction by modified
        b=Complex(1.0,t);                        Lentz's method (§5.2).
        c=Complex(1.0/FPMIN,0.0);
        d=h=Cdiv(ONE,b);
        for (i=2;i<=MAXIT;i++) {
            a = -(i-1)*(i-1);
            b=Cadd(b,Complex(2.0,0.0));
            d=Cdiv(ONE,Cadd(RCmul(a,d),b));  Denominators cannot be zero.
            c=Cadd(b,Cdiv(Complex(a,0.0),c));
            del=Cmul(c,d);
            h=Cmul(h,del);
            if (fabs(del.r-1.0)+fabs(del.i) < EPS) break;
        }
        if (i > MAXIT) nrerror("cf failed in cisi");
        h=Cmul(Complex(cos(t),-sin(t)),h);
        *ci = -h.r;
        *si=PIBY2+h.i;
    } else {                                 Evaluate both series simultaneously.
        if (t < sqrt(FPMIN)) {               Special case: avoid failure of convergence
            sumc=0.0;                            test because of underflow.
            sums=t;
        } else {
            sum=sums=sumc=0.0;
            sign=fact=1.0;
            odd=TRUE;
            for (k=1;k<=MAXIT;k++) {
                fact *= t/k;
                term=fact/k;
                sum += sign*term;
                err=term/fabs(sum);
                if (odd) {
                    sign = -sign;
                    sums=sum;
                    sum=sumc;
                } else {
                    sumc=sum;
                    sum=sums;
```

```
            }
            if (err < EPS) break;
            odd=!odd;
        }
        if (k > MAXIT) nrerror("maxits exceeded in cisi");
    }
    *si=sums;
    *ci=sumc+log(t)+EULER;
  }
  if (x < 0.0) *si = -(*si);
}
```

CITED REFERENCES AND FURTHER READING:

Stegun, I.A., and Zucker, R. 1976, *Journal of Research of the National Bureau of Standards*, vol. 80B, pp. 291–311; 1981, *op. cit.*, vol. 86, pp. 661–686.

Abramowitz, M., and Stegun, I.A. 1964, *Handbook of Mathematical Functions*, Applied Mathematics Series, Volume 55 (Washington: National Bureau of Standards; reprinted 1968 by Dover Publications, New York), Chapters 5 and 7.

6.10 Dawson's Integral

Dawson's Integral $F(x)$ is defined by

$$F(x) = e^{-x^2} \int_0^x e^{t^2}\, dt \tag{6.10.1}$$

The function can also be related to the complex error function by

$$F(z) = \frac{i\sqrt{\pi}}{2} e^{-z^2} \left[1 - \operatorname{erfc}(-iz)\right]. \tag{6.10.2}$$

A remarkable approximation for $F(x)$, due to Rybicki [1], is

$$F(z) = \lim_{h \to 0} \frac{1}{\sqrt{\pi}} \sum_{n\ \mathrm{odd}} \frac{e^{-(z-nh)^2}}{n} \tag{6.10.3}$$

What makes equation (6.10.3) unusual is that its accuracy increases *exponentially* as h gets small, so that quite moderate values of h (and correspondingly quite rapid convergence of the series) give very accurate approximations.

We will discuss the theory that leads to equation (6.10.3) later, in §13.11, as an interesting application of Fourier methods. Here we simply implement a routine based on the formula.

It is first convenient to shift the summation index to center it approximately on the maximum of the exponential term. Define n_0 to be the even integer nearest to x/h, and $x_0 \equiv n_0 h$, $x' \equiv x - x_0$, and $n' \equiv n - n_0$, so that

$$F(x) \approx \frac{1}{\sqrt{\pi}} \sum_{\substack{n'=-N \\ n'\ \mathrm{odd}}}^{N} \frac{e^{-(x'-n'h)^2}}{n' + n_0}, \tag{6.10.4}$$