CITED REFERENCES AND FURTHER READING:

Erdélyi, A., Magnus, W., Oberhettinger, F., and Tricomi, F.G. 1953, *Higher Transcendental Functions*, Vol. II, (New York: McGraw-Hill). [1]

Gradshteyn, I.S., and Ryzhik, I.W. 1980, *Table of Integrals, Series, and Products* (New York: Academic Press). [2]

Carlson, B.C. 1977, *SIAM Journal on Mathematical Analysis*, vol. 8, pp. 231–242. [3]

Carlson, B.C. 1987, *Mathematics of Computation*, vol. 49, pp. 595–606 [4]; 1988, *op. cit.*, vol. 51, pp. 267–280 [5]; 1989, *op. cit.*, vol. 53, pp. 327–333 [6]; 1991, *op. cit.*, vol. 56, pp. 267–280. [7]

Bulirsch, R. 1965, *Numerische Mathematik*, vol. 7, pp. 78–90; 1965, *op. cit.*, vol. 7, pp. 353–354; 1969, *op. cit.*, vol. 13, pp. 305–315. [8]

Carlson, B.C. 1979, *Numerische Mathematik*, vol. 33, pp. 1–16. [9]

Carlson, B.C., and Notis, E.M. 1981, *ACM Transactions on Mathematical Software*, vol. 7, pp. 398–403. [10]

Carlson, B.C. 1978, *SIAM Journal on Mathematical Analysis*, vol. 9, p. 524–528. [11]

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), Chapter 17. [12]

Mathews, J., and Walker, R.L. 1970, *Mathematical Methods of Physics*, 2nd ed. (Reading, MA: W.A. Benjamin/Addison-Wesley), pp. 78–79.

6.12 Hypergeometric Functions

As was discussed in §5.14, a fast, general routine for the the complex hypergeometric function ${}_2F_1(a,b,c;z)$, is difficult or impossible. The function is defined as the analytic continuation of the hypergeometric series,

$$\begin{aligned}{}_2F_1(a,b,c;z) = 1 &+ \frac{ab}{c}\frac{z}{1!} + \frac{a(a+1)b(b+1)}{c(c+1)}\frac{z^2}{2!} + \cdots \\ &+ \frac{a(a+1)\dots(a+j-1)b(b+1)\dots(b+j-1)}{c(c+1)\dots(c+j-1)}\frac{z^j}{j!} + \cdots\end{aligned} \tag{6.12.1}$$

This series converges only within the unit circle $|z| < 1$ (see [1]), but one's interest in the function is not confined to this region.

Section 5.14 discussed the method of evaluating this function by direct path integration in the complex plane. We here merely list the routines that result.

Implementation of the function `hypgeo` is straightforward, and is described by comments in the program. The machinery associated with Chapter 16's routine for integrating differential equations, `odeint`, is only minimally intrusive, and need not even be completely understood: use of `odeint` requires one zeroed global variable, one function call, and a prescribed format for the derivative routine `hypdrv`.

The function `hypgeo` will fail, of course, for values of z too close to the singularity at 1. (If you need to approach this singularity, or the one at ∞, use the "linear transformation formulas" in §15.3 of [1].) Away from $z = 1$, and for moderate values of a, b, c, it is often remarkable how few steps are required to integrate the equations. A half-dozen is typical.

Sample page from NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (ISBN 0-521-43108-5)
Copyright (C) 1988-1992 by Cambridge University Press. Programs Copyright (C) 1988-1992 by Numerical Recipes Software.
Permission is granted for internet users to make one paper copy for their own personal use. Further reproduction, or any copying of machine-readable files (including this one) to any server computer, is strictly prohibited. To order Numerical Recipes books, diskettes, or CDROMs
visit website http://www.nr.com or call 1-800-872-7423 (North America only), or send email to trade@cup.cam.ac.uk (outside North America).

```
#include <math.h>
#include "complex.h"
#include "nrutil.h"
#define EPS 1.0e-6                                Accuracy parameter.

fcomplex aa,bb,cc,z0,dz;                          Communicates with hypdrv.

int kmax,kount;                                   Used by odeint.
float *xp,**yp,dxsav;

fcomplex hypgeo(fcomplex a, fcomplex b, fcomplex c, fcomplex z)
```

Complex hypergeometric function ${}_2F_1$ for complex a, b, c, and z, by direct integration of the hypergeometric equation in the complex plane. The branch cut is taken to lie along the real axis, Re $z > 1$.

```
{
    void bsstep(float y[], float dydx[], int nv, float *xx, float htry,
        float eps, float yscal[], float *hdid, float *hnext,
        void (*derivs)(float, float [], float []));
    void hypdrv(float s, float yy[], float dyyds[]);
    void hypser(fcomplex a, fcomplex b, fcomplex c, fcomplex z,
        fcomplex *series, fcomplex *deriv);
    void odeint(float ystart[], int nvar, float x1, float x2,
        float eps, float h1, float hmin, int *nok, int *nbad,
        void (*derivs)(float, float [], float []),
        void (*rkqs)(float [], float [], int, float *, float, float,
        float [], float *, float *, void (*)(float, float [], float [])));
    int nbad,nok;
    fcomplex ans,y[3];
    float *yy;

    kmax=0;
    if (z.r*z.r+z.i*z.i <= 0.25) {                Use series...
        hypser(a,b,c,z,&ans,&y[2]);
        return ans;
    }
    else if (z.r < 0.0) z0=Complex(-0.5,0.0);     ...or pick a starting point for the path
    else if (z.r <= 1.0) z0=Complex(0.5,0.0);         integration.
    else z0=Complex(0.0,z.i >= 0.0 ? 0.5 : -0.5);
    aa=a;                                         Load the global variables to pass pa-
    bb=b;                                             rameters "over the head" of odeint
    cc=c;                                             to hypdrv.
    dz=Csub(z,z0);
    hypser(aa,bb,cc,z0,&y[1],&y[2]);              Get starting function and derivative.
    yy=vector(1,4);
    yy[1]=y[1].r;
    yy[2]=y[1].i;
    yy[3]=y[2].r;
    yy[4]=y[2].i;
    odeint(yy,4,0.0,1.0,EPS,0.1,0.0001,&nok,&nbad,hypdrv,bsstep);
```

The arguments to odeint are the vector of independent variables, its length, the starting and ending values of the dependent variable, the accuracy parameter, an initial guess for stepsize, a minimum stepsize, the (returned) number of good and bad steps taken, and the names of the derivative routine and the (here Bulirsch-Stoer) stepping routine.

```
    y[1]=Complex(yy[1],yy[2]);
    free_vector(yy,1,4);
    return y[1];
}
```

Sample page from NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (ISBN 0-521-43108-5)
Copyright (C) 1988-1992 by Cambridge University Press. Programs Copyright (C) 1988-1992 by Numerical Recipes Software.
Permission is granted for internet users to make one paper copy for their own personal use. Further reproduction, or any copying of machine-readable files (including this one) to any server computer, is strictly prohibited. To order Numerical Recipes books, diskettes, or CDROMs
visit website http://www.nr.com or call 1-800-872-7423 (North America only), or send email to trade@cup.cam.ac.uk (outside North America).

```
#include "complex.h"
#define ONE Complex(1.0,0.0)

void hypser(fcomplex a, fcomplex b, fcomplex c, fcomplex z, fcomplex *series,
    fcomplex *deriv)
```
Returns the hypergeometric series ${}_2F_1$ and its derivative, iterating to machine accuracy. For $|z| \leq 1/2$ convergence is quite rapid.
```
{
    void nrerror(char error_text[]);
    int n;
    fcomplex aa,bb,cc,fac,temp;

    deriv->r=0.0;
    deriv->i=0.0;
    fac=Complex(1.0,0.0);
    temp=fac;
    aa=a;
    bb=b;
    cc=c;
    for (n=1;n<=1000;n++) {
        fac=Cmul(fac,Cdiv(Cmul(aa,bb),cc));
        deriv->r+=fac.r;
        deriv->i+=fac.i;
        fac=Cmul(fac,RCmul(1.0/n,z));
        *series=Cadd(temp,fac);
        if (series->r == temp.r && series->i == temp.i) return;
        temp= *series;
        aa=Cadd(aa,ONE);
        bb=Cadd(bb,ONE);
        cc=Cadd(cc,ONE);

    }
    nrerror("convergence failure in hypser");
}
```

```
#include "complex.h"
#define ONE Complex(1.0,0.0)

extern fcomplex aa,bb,cc,z0,dz;        Defined in hypgeo.

void hypdrv(float s, float yy[], float dyyds[])
```
Computes derivatives for the hypergeometric equation, see text equation (5.14.4).
```
{
    fcomplex z,y[3],dyds[3];

    y[1]=Complex(yy[1],yy[2]);
    y[2]=Complex(yy[3],yy[4]);
    z=Cadd(z0,RCmul(s,dz));
    dyds[1]=Cmul(y[2],dz);
    dyds[2]=Cmul(Csub(Cmul(Cmul(aa,bb),y[1]),Cmul(Csub(cc,
        Cmul(Cadd(Cadd(aa,bb),ONE),z)),y[2])),
        Cdiv(dz,Cmul(z,Csub(ONE,z))));
    dyyds[1]=dyds[1].r;
    dyyds[2]=dyds[1].i;
    dyyds[3]=dyds[2].r;
    dyyds[4]=dyds[2].i;
}
```

CITED REFERENCES AND FURTHER READING:

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). [1]

Sample page from NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (ISBN 0-521-43108-5)
Copyright (C) 1988-1992 by Cambridge University Press. Programs Copyright (C) 1988-1992 by Numerical Recipes Software.
Permission is granted for internet users to make one paper copy for their own personal use. Further reproduction, or any copying of machine-readable files (including this one) to any server computer, is strictly prohibited. To order Numerical Recipes books, diskettes, or CDROMs
visit website http://www.nr.com or call 1-800-872-7423 (North America only), or send email to trade@cup.cam.ac.uk (outside North America).