16.5 Second-Order Conservative Equations

Usually when you have a system of high-order differential equations to solve it is best to reformulate them as a system of first-order equations, as discussed in §16.0. There is a particular class of equations that occurs quite frequently in practice where you can gain about a factor of two in efficiency by differencing the equations directly. The equations are second-order systems where the derivative does not appear on the right-hand side:

$$y'' = f(x, y), \qquad y(x_0) = y_0, \qquad y'(x_0) = z_0 \tag{16.5.1}$$

As usual, y can denote a vector of values.

Stoermer's rule, dating back to 1907, has been a popular method for discretizing such systems. With $h = H/m$ we have

$$\begin{aligned} y_1 &= y_0 + h[z_0 + \tfrac{1}{2}hf(x_0, y_0)] \\ y_{k+1} - 2y_k + y_{k-1} &= h^2 f(x_0 + kh, y_k), \qquad k = 1, \dots, m-1 \\ z_m &= (y_m - y_{m-1})/h + \tfrac{1}{2}hf(x_0 + H, y_m) \end{aligned} \tag{16.5.2}$$

Here z_m is $y'(x_0 + H)$. Henrici showed how to rewrite equations (16.5.2) to reduce roundoff error by using the quantities $\Delta_k \equiv y_{k+1} - y_k$. Start with

$$\begin{aligned} \Delta_0 &= h[z_0 + \tfrac{1}{2}hf(x_0, y_0)] \\ y_1 &= y_0 + \Delta_0 \end{aligned} \tag{16.5.3}$$

Then for $k = 1, \dots, m-1$, set

$$\begin{aligned} \Delta_k &= \Delta_{k-1} + h^2 f(x_0 + kh, y_k) \\ y_{k+1} &= y_k + \Delta_k \end{aligned} \tag{16.5.4}$$

Finally compute the derivative from

$$z_m = \Delta_{m-1}/h + \tfrac{1}{2}hf(x_0 + H, y_m) \tag{16.5.5}$$

Gragg again showed that the error series for equations (16.5.3)–(16.5.5) contains only even powers of h, and so the method is a logical candidate for extrapolation à la Bulirsch-Stoer. We replace mmid by the following routine stoerm:

```
      SUBROUTINE stoerm(y,d2y,nv,xs,htot,nstep,yout,derivs)
      INTEGER nstep,nv,NMAX
      REAL htot,xs,d2y(nv),y(nv),yout(nv)
      EXTERNAL derivs
      PARAMETER (NMAX=50)                 Maximum value of nv.
C     USES derivs
         Stoermer's rule for integrating y'' = f(x,y) for a system of n = nv/2 equations. On input
         y(1:nv) contains y in its first n elements and y' in its second n elements, all evaluated
         at xs. d2y(1:nv) contains the right-hand side function f (also evaluated at xs) in its
         first n elements. Its second n elements are not referenced. Also input is htot, the total
         step to be taken, and nstep, the number of substeps to be used. The output is returned
         as yout(1:nv), with the same storage arrangement as y. derivs is the user-supplied
         subroutine that calculates f.
      INTEGER i,n,neqns,nn
      REAL h,h2,halfh,x,ytemp(NMAX)
      h=htot/nstep                        Stepsize this trip.
      halfh=0.5*h
      neqns=nv/2                          Number of equations.
      do 11 i=1,neqns                     First step.
        n=neqns+i
        ytemp(n)=h*(y(n)+halfh*d2y(i))
```

```
    ytemp(i)=y(i)+ytemp(n)
enddo 11
x=xs+h
call derivs(x,ytemp,yout)          Use yout for temporary storage of derivatives.
h2=h*h
do 13 nn=2,nstep                   General step.
    do 12 i=1,neqns
        n=neqns+i
        ytemp(n)=ytemp(n)+h2*yout(i)
        ytemp(i)=ytemp(i)+ytemp(n)
    enddo 12
    x=x+h
    call derivs(x,ytemp,yout)
enddo 13
do 14 i=1,neqns                    Last step.
    n=neqns+i
    yout(n)=ytemp(n)/h+halfh*yout(i)
    yout(i)=ytemp(i)
enddo 14
return
END
```

Note that for compatibility with bsstep the arrays y and d2y are of length $2n$ for a system of n second-order equations. The values of y are stored in the first n elements of y, while the first derivatives are stored in the second n elements. The right-hand side f is stored in the first n elements of the array d2y; the second n elements are unused. With this storage arrangement you can use bsstep simply by replacing the call to mmid with one to stoerm using the same arguments; just be sure that the argument nv of bsstep is set to $2n$. You should also use the more efficient sequence of stepsizes suggested by Deuflhard:

$$n = 1, 2, 3, 4, 5, \ldots \tag{16.5.6}$$

and set KMAXX $= 12$ in bsstep.

CITED REFERENCES AND FURTHER READING:

Deuflhard, P. 1985, *SIAM Review*, vol. 27, pp. 505–535.

16.6 Stiff Sets of Equations

As soon as one deals with more than one first-order differential equation, the possibility of a *stiff* set of equations arises. Stiffness occurs in a problem where there are two or more very different scales of the independent variable on which the dependent variables are changing. For example, consider the following set of equations [1]:

$$\begin{aligned} u' &= 998u + 1998v \\ v' &= -999u - 1999v \end{aligned} \tag{16.6.1}$$

with boundary conditions

$$u(0) = 1 \qquad v(0) = 0 \tag{16.6.2}$$