ASSIGNMENT 2 Using Taylor Integrator

Ignacio Coca

We create a file *eq_os_var.eq* with 6 ODE. The first two are the "position" (x, y), and the remaining 4 are variational equations

```
eq_os_var.eq:

diff(x1,t)=x2;
diff(x2,t)=-x1;
diff(x3,t)=x5;
diff(x4,t)=x6;
diff(x5,t)=-x3;
diff(x6,t)=-x4;
----------------
```

Next we modify the file named *main_os_flow.f* in order to adequate it to our problem. The new file is named *main_os_flow_var.f*

```
Main_os_flow_var.f
c*************************************************************
c
c  MAIN_OS_FLOW_VAR.f
c
c       We integrate the harmonic oscillator field with Taylor
c       from t=ti up to t=tmax
c       idir= +1 (integration forward in time); =-1 (backward)
c       np= number of intermediate points (apart from the initial one)
c           that we want to write on the file orbit.d. If np=1
c           only the initial and final points are written
c
c  input: xi,ti,tmax,idir,np
c*************************************************************
        implicit real*8 (a-h,o-z)
        parameter (n=6)
        dimension xi(n),x(n)

        dimension A(2,2)

        open(10,file='orbit.d',status='unknown')
        write(*,*) 'Initial condition'
        read(*,*) (xi(i),i=1,n)
```

```
        write(*,*) 'ti,tmax,np (number of points)'
        read(*,*)ti,tmax,np
c particular example integration up to t=pi
c        pi=4.d0*datan(1.d0)
c        tmax=pi/2.d0
        if (tmax.ge.ti)then
c         'idir (=1 forward in time, =-1 backward)'
            idir=1
        else
            idir=-1
        endif
        do i=1,n
         x(i)=xi(i)
        enddo
         write(*,*)ti,'   initial t, initial cond:'
         write(*,*)(x(i),i=1,n)
c REMARK: xinctime positive
       xinctime=dabs(tmax-ti)/np
         write (10,*)ti,(x(ii),ii=1,n)
       do 20 i=1,np
         call flow(ti,n,x,idir,xinctime)
         write (10,*)ti,(x(ii),ii=1,n)

  ham=(x(1)*x(1)+x(2)*x(2))/2.d0
        do 20 i=1,np
         call flow(ti,n,x,idir,xinctime)
         ham_new=(x(1)*x(1)+x(2)*x(2))/2.d0
         dif=dabs(ham-ham_new)
         if (dif.gt.1.D-11) then
           write(*,*) 'Problem in first integral'
           stop
         endif
         write (10,*)ti,(x(ii),ii=1,n)


 20    continue

       A(1,1)=x(3)
       A(1,2)=x(4)
       A(2,1)=x(5)
```

```
       A(2,2)=x(6)

       call det(A,deta,2)
       write(*,*) deta, '<-the determinant of A'


       write(*,*)ti,'  final t,  final point:'
       write(*,*)(x(i),i=1,n)
        end


       subroutine flow(t,n,x,idir,xinctemps)
       IMPLICIT REAL*8 (A-H,O-Z)
       dimension x(n)
       tmax=t+idir*xinctemps
c
c parameters for the integration
c
        hab=0.1e-16
        hre=0.1e-16
        pabs=dlog10(hab)
        prel=dlog10(hre)
c Option of control of step
        istep=1
        ht=0.d0
 1        CALL taylor_f77_eq_os_var_(t,x,idir,istep,pabs,prel,
     &  tmax,ht,iordre,ifl)
c        write(10,100) t,(x(i),i=1,n)
        if (idir.eq.1.and.t.lt.tmax)go to 1
        if (idir.eq.-1.and.t.gt.tmax)go to 1
c check t=tmax
        if (dabs(t-tmax).le.1.d-13)return
        write(*,*)'problems in taylor'
        stop
c 100     format(f15.8,2f22.15)
        return
        end
----------------
```

The changes done in this file are:

- The amplification of dimension from 2 to 6, since our system now has 4 variational equations.
- A subroutine to check the first integral.
- The creation of a matrix A of dimension $2x2$, that will be $(x3, x4; x5, x6)$
- The creation of a subroutine where we compute the existence of the determinant. It is implemented before the subroutine flow, and runs a call to the *det* function.
- The call function to call $taylor\_(...)\_var\_(...)$

This way, after running taylor, we are ready to run the program.

The inputs for it (initial conditions) are:

- $(x, y) = (1, 0)$
- variational equations: $x3 = 1, x4 = 0, x5 = 0, x6 = 1$
- $ti = 0$
- $tmax = 6.28$ $(2.d0 * pi)$
- $np = 100$

Once the orbit is created from ti to $tmax$, we plot (x, y) in gnuplot, obtaining the following graphic:

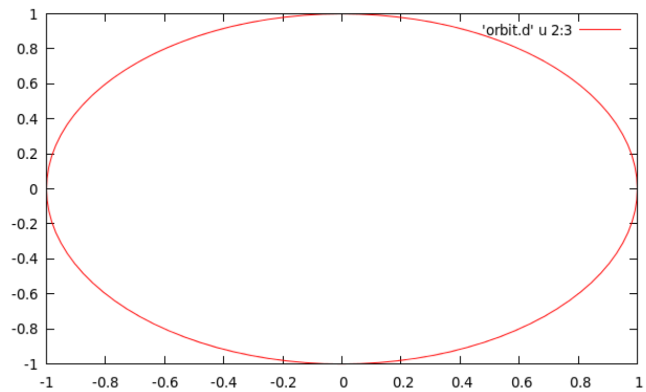

Figure 1: Orbit

PART 2 We edit a file called *eq_lorenz.eq* with 3 ODE:

```
eq_lorenz.eq:

diff(x1,t)=sigma*(x2-x1);
diff(x2,t)=rho*x1-x2-x1*x3;
diff(x3,t)=x1*x2-beta*x3;

sigma=10;
rho=28;
beta=8./3.;
----------------
```

We run Taylor to obtain the time stepper, and create a file called $main\_lorenz_flow.f$ as follow:

```
main_lorenz_flow:


        implicit real*8 (a-h,o-z)
        parameter (n=3)
        dimension xi(n),x(n)
        open(10,file='orbit.d',status='unknown')
        write(*,*) 'Initial condition x(1),x(2)'
        read(*,*) (xi(i),i=1,n)
```

```
        write(*,*) 'ti,tmax,np (number of points)'
        read(*,*)ti,tmax,np
c particular example integration up to t=pi
c        pi=4.d0*datan(1.d0)
c        tmax=pi/2.d0
        if (tmax.ge.ti)then
c          'idir (=1 forward in time, =-1 backward)'
            idir=1
        else
            idir=-1
        endif
        do i=1,n
         x(i)=xi(i)
        enddo
         write(*,*)ti,'   initial t, initial cond:'
         write(*,*)(x(i),i=1,n)
c REMARK: xinctime positive
       xinctime=dabs(tmax-ti)/np
         write (10,*)ti,(x(ii),ii=1,n)
       do 20 i=1,np
         call flow(ti,n,x,idir,xinctime)
         write (10,*)ti,(x(ii),ii=1,n)
 20    continue

       write(*,*)ti,'  final t,  final point:'
       write(*,*)(x(i),i=1,n)
        end


       subroutine flow(t,n,x,idir,xinctemps)
       IMPLICIT REAL*8 (A-H,O-Z)
       dimension x(n)
       tmax=t+idir*xinctemps
c
c parameters for the integration
c
        hab=0.1e-16
        hre=0.1e-16
        pabs=dlog10(hab)
        prel=dlog10(hre)
```

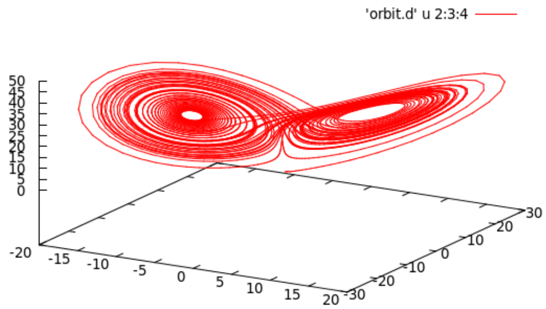

Figure 2: (x,y,z)

```
c Option of control of step
        istep=1
        ht=0.d0
 1        CALL taylor_f77_eq_lorenz_(t,x,idir,istep,pabs,prel,
     &  tmax,ht,iordre,ifl)
c        write(10,100) t,(x(i),i=1,n)
        if (idir.eq.1.and.t.lt.tmax)go to 1
        if (idir.eq.-1.and.t.gt.tmax)go to 1
c check t=tmax
        if (dabs(t-tmax).le.1.d-13)return
        write(*,*)'problems in taylor'
        stop
c 100     format(f15.8,2f22.15)
        return
        end
----------------
```

Introducing the initial conditions to the executable file of the program $((x, y, z) = (0., 1., 0.)$, $ti = 0.$, $tmax = 60$, $np = 4000)$ and plotting the orbits with gnuplot we find:

Plot 1: (x, y, z): Figure 2

Plot 2: (x, z): Figure 3

Plot 3: (t, y): Figure 4

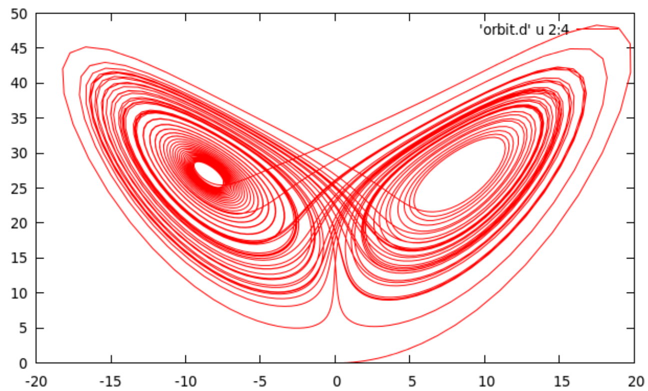

Figure 3: (x,z)

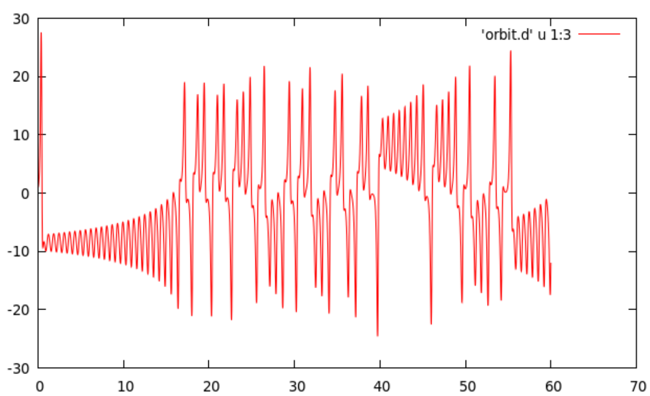

Figure 4: (t,y)