ASSIGNMENT 3 Computation of a Poincare Section

Ignacio Coca

We first implement the computation of $n - crossing > 1$ using the following code:

```
c
c  MAIN_OS_SEC1.f
c
c*****************************************************************
c       We integrate the harmonic oscillator field with Taylor
c       up to the   FIRST  crossing with the Poincare section: y=0
c                   -----
c
c !!!!   You should enter the code to integrate up to a given
c   'n_crossing'  crossing with the Poincare section: y=0   !!!!!
c
c*****************************************************************
        implicit real*8 (a-h,o-z)
        parameter (n=2)
        dimension yf(n),x(n)
        open(10,file='orbit.d',status='unknown')
        write(*,*) 'Initial condition x(1),...,x(n)'
        read(*,*) (x(i),i=1,n)
        write(*,*) 'idir?'
        read(*,*) idirorig
        write(*,*) 'n_crossing'
        read(*,*) ncross
c
c we assume initial time t=0.d0
c
        t=0.d0
C        write(10,*)t,(x(i),i=1,2)

        DO i=1,ncross
        call poinc1(n,x,yf,tfinal,idirorig)
        x=yf
end DO
        end


C******************************************************************************
```

```
c Input:
c n dimension of the vectors yi and yf
c yi initial point
c idirorig: +1 integration forwards in time; -1 backwards
c yf final point
c tfinal final time
c
C*****************************************************************************
        SUBROUTINE POINC1(n,YI,YF,tfinal,idirorig)
        IMPLICIT REAL*8 (A-H,O-Z)
        DIMENSION YI(n),YF(n),DGG(n),F(n)
           icont=0
        idir=idirorig
c
c  we assume initial time t=0.
c
        ti=0.D0
C   DETERMINATION OF THE FIRST PASSAGE OF THE ORBIT THROUGH y=0
C
        CALL SECCIO(YI,GG,DGG)
        IF(DABS(GG).LT.1.D-9)GG=0.d0
        GA=GG
        hab=.1e-16
        hre=.1e-16
        pabs=dlog10(hab)
        prel=dlog10(hre)
        istep=1
c reasonable step:
        pas=0.4d0
        ht=0.d0
        t=ti
c |tmax| must be big enough
1        tmax=t+idir*pas
        CALL taylor_f77_eq_os_(t,yi,idir,istep,pabs,prel,
     &  tmax,ht,iordre,ifl)
c computation of first integral to be done
C
        CALL SECCIO(YI,GG,DGG)
        IF(GG*GA.LT.0.D0)go to 22
C        write(10,*)tfinal,(yi(ii),ii=1,2)
```

```
      GA=GG
      GO TO 1
C
C   REFINEMENT OF THE INTERSECTION POINT YF(*) USING NEWTON'S METHOD
C   TO GET A ZERO OF THE FUNCTION GG (SEE SUBROUTINE SECCIO)
C
 22   continue
      icont=icont+1
      if (icont.gt.20)then
          write(*,*)'problems finding the section'
          stop
      endif
      CALL FIELD(T,YI,N,F)
      P=0.D0
      DO 3 I=1,N
3     P=P+F(I)*DGG(I)
      H=-GG/P
c check p is not (or very close to) 0:  to be done
      if (h.ge.0.d0)idir=1
      if (h.lt.0.d0)idir=-1
      tmax=t+h
c         write(*,*)icont,'  refining: h and time ',h,tmax
c         write(*,*)'refining  t point ',t,yi(1),yi(2)
      CALL taylor_f77_eq_os_(t,yi,idir,istep,pabs,prel,
     &  tmax,ht,iordre,ifl)
      CALL SECCIO(YI,GG,DGG)
      IF(DABS(GG).GT.1.D-13)GO TO 22
      DO 4 I=1,N
4     YF(I)=YI(I)
      tfinal=t+tfinal
c check first integral: to be done
      write(*,*)'tfinal point time ',tfinal
      write(*,*)(yf(ii),ii=1,n)
      write(10,*)tfinal,(yf(ii),ii=1,2)
      return
      end


C*****************************************************************************
C                                                                            *
```

```
C     THE SURFACE g OF SECTION,IN THIS CASE
C          INPUT PARAMETERS:
C     Y(*)       POINT
C          OUTPUT PARAMETERS:
C     GG          FUNCTION THAT EQUATED TO 0 GIVES THE SURFACE OF
C                 SECTION
C     DGG(*)      GRADIENT OF FUNCTION GG
C                                                                      *
C***********************************************************************
        SUBROUTINE SECCIO(Y,GG,DGG)
        IMPLICIT REAL*8(A-H,O-Z)
        DIMENSION Y(2),DGG(2)
        GG=Y(2)
        DO 1 I=1,2
 1      DGG(I)=0.D0
        DGG(2)=1.d0
        RETURN
        END


C
C FIELD.F
C
C***********************************************************************
C
C     EQS OF MOTION IN synodical VARIABLES
C     X          TIME
C     Y(*)       POINT (Y(1),Y(2),....Y(n))
C     NEQ        NUMBER OF EQUATIONS
C          OUTPUT PARAMETERS:
C     F(*)       VECTOR FIELD
C
C***********************************************************************
      subroutine field(t,x,neq,f)
      implicit real*8 (a-h,o-z)
      dimension x(neq),f(neq)
c
      f(1) =x(2)
      f(2)= -x(1)
      return
```

```
      end
```

I basically changed a few lines of the main_os_sec1.f program. It is now done in order to ask for how many crossings do we want, and do a DO-WHILE loop since we arrive to the number of crossings desired.

In order to retain the previous points and time, we compute the value of x to be yf after the poincare call, and during the poincare function we do tmax=t+tmax.

We do four simulations, as asked in the assay description. The following images correspond to the plots and the outputs for them, doing 2 crossings.

The first column corresponds to time, and the second and third ones to x and y respectively.

1: $(x, y) = (1, 0)$, $idir = +1$

```
   3.1415926535898913       -1.0000000000000000        9.8059608393333275E-014
   6.2831853071796848        1.0000000000000002       -9.8226141847027049E-014
```

We note in the figure that the y-axis moves in a really small range. It is almost zero, so we are moving from $(-1, 0)$ to $(1, 0)$. The same happens for the other plots.

2: $(x, y) = (1, 0)$, $idir = -1$

```
  -3.1415926535898913       -1.0000000000000000       -9.8059608393333275E-014
  -6.2831853071796848        1.0000000000000002        9.8226141847027049E-014
```

Now the time is negative, since we are moving backward in time.

3: $(x, y) = (0, 1)$, $idir = +1$

```
   1.5707963267948968        1.0000000000000000       -2.6188395256619007E-016
   4.7123889803847874      -0.99999999999999978        9.7962450326151418E-014
```

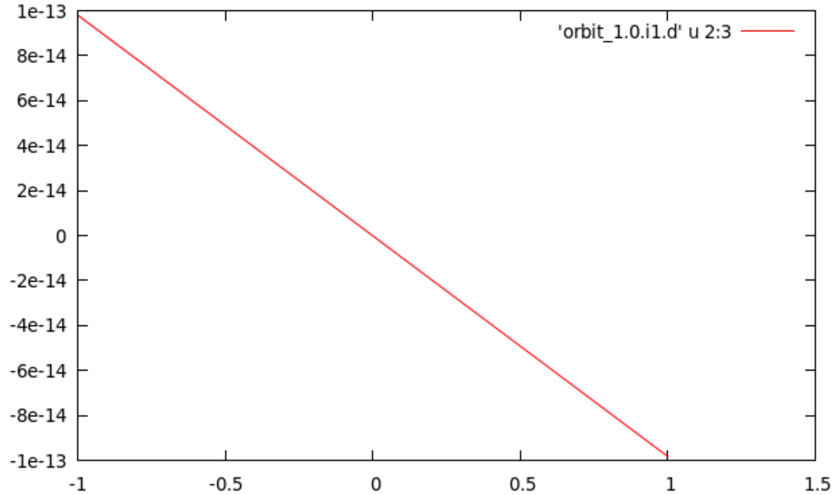

Figure 1: 1) (1,0), idir=+1

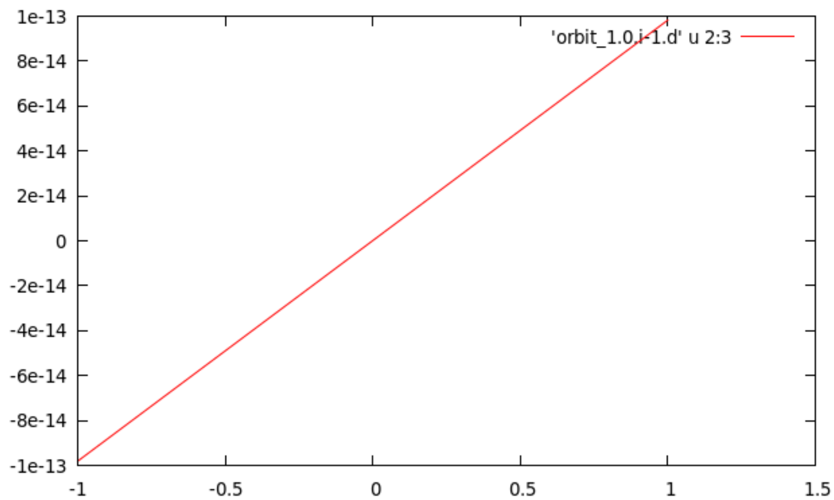

Figure 2: 2) (1,0), idir=-1

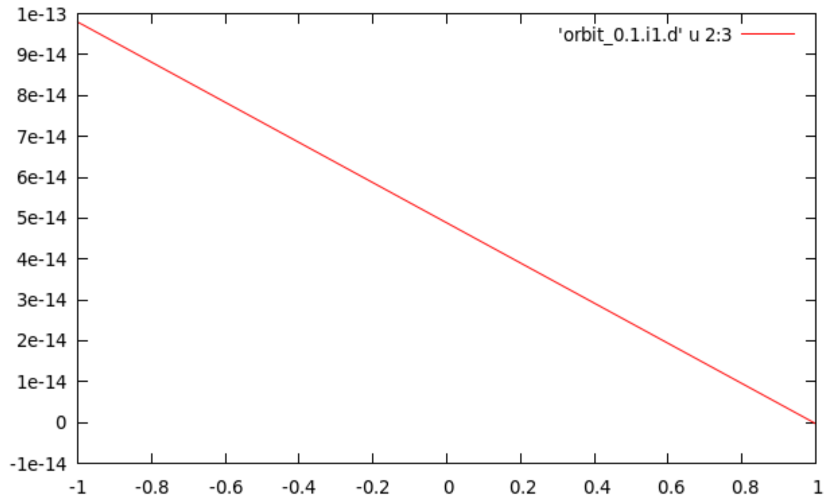

Figure 3: 3) (0,1), idir=+1

We note that for cases 3 and 4, the time to the first crossing is not π. This happens because we are not starting at the poincare section as we did in cases 1 and 2. This way we first have to move to the Poincare section and then after π time we find another crossing.

4: $(x, y) = (0, 1)$, $idir = -1$

```
-1.5707963267948968         -1.0000000000000000         -2.6188395256619007E-016
-4.7123889803847874         0.99999999999999978          9.7962450326151418E-014
```

Now the time is negative, since we are moving backward in time.

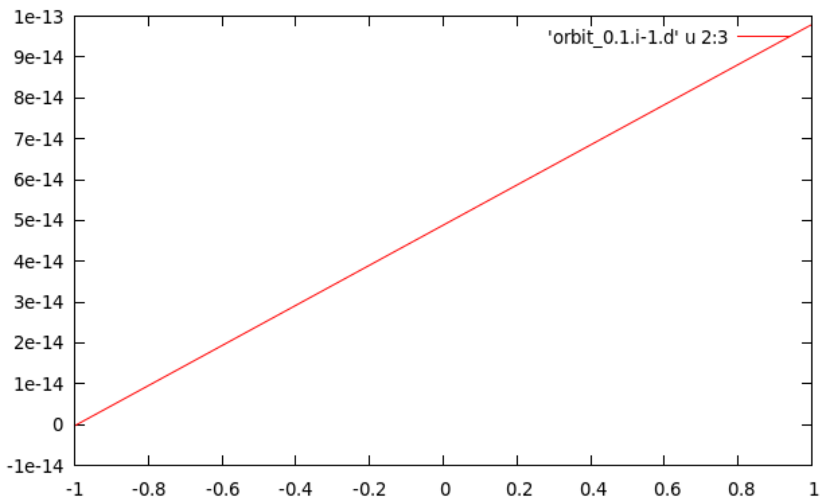

Figure 4: 4) (0,1), idir=-1