ASSIGNMENT 10 Computation of Homoclinic Orbits of Equilibrium Points

Ignacio Coca

In this assay we do the same as in assay 9, but changing the value of μ, ie: we are going to calculate the unstable manifold up to first crossing with the Poincare section for each $\mu \in (0.001, 0.5)$. We save the points in a file and then we plot them.

The program is as follow:

```
      implicit real*8 (a-h,o-z)
      common/param/xmu

      parameter (n=20)
      dimension oM(4,4)
      dimension rr(4),ri(4),vr(4,4),vi(4,4)
      dimension x(n), yf(n), xi(n)

      dimension vep(4)
      dimension vap(2)


      open(10,file='orbit.d',status='unknown')


C      write(*,*) 'iregion'
C      read(*,*) iregion
C      write(*,*) 'n_crossing'
C      read(*,*) ncross

      ncross=1
      iregion=-1

      idirorig=1
      s=iregion*(1.d-6)

      xmu=0.001d0

      xincmu=0.00001d0
C               write(*,*) xincmu
      DO ik=1,5000
```

```
      xmu=xmu+xincmu

      if (xmu.gt.0.015d0)then
      xincmu=0.0001
      endif

      if(dabs(xmu-0.02d0).lt.0.001d0)then
      xmu=0.021
      endif

      if (xmu.gt.0.05d0)then
      xincmu=0.001
      endif

      if (xmu.gt.0.5d0)then
      goto 50
      endif


C      write(*,*) xmu, ik
      call peq(xmu,xl1,xl2,xl3,cl1,cl2,cl3)

C      write(12,*) xl3, cl3
      call omega(xl3, omexx, omexy, omeyy)

      oM(1,1)=0.d0
      oM(1,2)=0.d0
      oM(1,3)=1.d0
      oM(1,4)=0.d0

      oM(2,1)=0.d0
      oM(2,2)=0.d0
      oM(2,3)=0.d0
      oM(2,4)=1.d0

      oM(3,1)=omexx
      oM(3,2)=omexy
      oM(3,3)=0.d0
      oM(3,4)=2.d0
```

```
        oM(4,1)=omexy
        oM(4,2)=omeyy
        oM(4,3)=-2.d0
        oM(4,4)=0.d0

C        write(11,*) oM



        call vapvep(oM,4,rr,ri,vr,vi)


C   The vap with lambda>0 is the 3rd one
C   The corresponding vep is the 3rd one
C        write(*,*) rr

        if(idirorig.ge.(0.d0))then
        vap(1)=rr(3)
        vap(2)=ri(3)

        vep(1)=vr(1,3)
        vep(2)=vr(2,3)
        vep(3)=vr(3,3)
        vep(4)=vr(4,3)



        vep=vep
C        write(*,*) vep
C        write(*,*) vap

        x(1)=xl3+(s*vep(1))
        x(2)=0.d0+(s*vep(2))
        x(3)=0.d0+(s*vep(3))
        x(4)=0.d0+(s*vep(4))
        endif

C   The vap with lambda<0 is the 4rd one
C   The corresponding vep is the 4rd one
C        write(*,*) rr
```

```
        if(idirorig.le.(0.d0))then
        vap(1)=rr(4)
        vap(2)=ri(4)

        vep(1)=vr(1,4)
        vep(2)=vr(2,4)
        vep(3)=vr(3,4)
        vep(4)=vr(4,4)


        vep=vep
C        write(*,*) vep
C        write(*,*) vap

        x(1)=xl3+(s*vep(1))
        x(2)=0.d0+(s*vep(2))
        x(3)=0.d0+(s*vep(3))
        x(4)=0.d0+(s*vep(4))
        endif

                t=0.d0
C        write(10,*)t,(x(i),i=1,2)


        x(5)=1.
x(6)=0.
x(7)=0.
x(8)=0.
x(9)=0.
x(10)=1.
x(11)=0.
x(12)=0.
x(13)=0.
x(14)=0.
x(15)=1.
x(16)=0.
x(17)=0.
x(18)=0.
x(19)=0.
```

```
x(20)=1.

        xi=x

       r1=dsqrt((xi(1)-xmu)*(xi(1)-xmu)+xi(2)*xi(2))
       r2=dsqrt((xi(1)-xmu+1.d0)*(xi(1)-xmu+1.d0)+xi(2)*xi(2))
       omeg=0.5d0*(xi(1)*xi(1)+xi(2)*xi(2))+(1.d0-xmu)/r1
     .  +xmu/r2+0.5d0*(1.d0-xmu)*xmu
       C_initial=2.d0*omeg-(xi(3)*xi(3)+xi(4)*xi(4))
C      write(*,*) 'C=', C_initial

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

write(10,*)xmu,yf(3)

C write(*,*) idirorig
C write(*,*) 'final x:'
C write(*,*) (x(ii), ii=1,4)
 50 end DO

        end



        subroutine omega(xl3, omexx, omexy, omeyy)
        implicit real*8(a-h,o-z)
        common/param/xmu
        dimension x(20)

        x(1)=xl3
        x(2)=0.d0
        x(3)=0.d0
        x(4)=0.d0

        umu=1.-xmu
        d1=x(1)-xmu
        d2=x(1)+umu
```

```
        r12=d1*d1+x(2)*x(2)
        r22=d2*d2+x(2)*x(2)
        r0=dsqrt(r12)
        r1=dsqrt(r22)

        r032=r12*r0
        r132=r22*r1
        r052=r12*r032
        r152=r22*r132

        omex=x(1)-(umu*(-xmu+x(1))/r032)-(xmu*(x(1)+umu)/r132)
        omey=x(2)*(1.-(umu/r032)-(xmu/r132))

omexx=1.-(umu*((r0*r0)-3.*d1)/(r0*r0*r0*r0*r0))
     .    -(xmu*( (r1*r1)-(3.*(umu+x(1))*(umu+x(1))) )/(r1*r1*r1*r1*r1))
omexy=x(2)*(((3.*umu*d1)/(r0*r0*r0*r0*r0))
     .    +(3.*xmu*(x(1)+umu))/ (r1*r1*r1*r1*r1))
omeyy=(1.-(umu/(r0*r0*r0))-(xmu/(r1*r1*r1))) + ( x(2)* ((3.
     .    *umu*x(2))/ (r0*r0*r0*r0*r0) )+ (xmu*3.*x(2))
     .      / (r1*r1*r1*r1*r1) )

        end

c
c routine to compute xl1,xl2,xl3,Cl2,Cl2,Cl3
c




        subroutine peq(xmu,xl1,xl2,xl3,cl1,cl2,cl3)
        implicit real*8(a-h,o-z)
        a=1.d0/3.d0

        i=0
c to compute L2 (on the left hand side of the small primary)
        x=xmu/(3.d0*(1.d0-xmu))
        x=x**a
  1        den=3.d0-2.d0*xmu+x*(3.d0-xmu+x)
```

```
        f=xmu*(1.d0+x)**2/den
        f=f**a
        x1=xmu-1.d0-x
        if (dabs(x-f).le.1.d-15)then
c CALL .... and compute C(L2)
         xl2=X1
             call c(xmu,xl2,cl2)
             go to 3
        endif
        i=i+1
        x=f
        go to 1
   2    format(e25.16,',',e25.16,',',e25.16)
   3    continue

c
c L1 (between the primaries)
c
        i=0
        x=xmu/(3.d0*(1.d0-xmu))
        x=x**a
  10        den=3.d0-2.d0*xmu-x*(3.d0-xmu-x)
        f=xmu*(1.d0-x)**2/den
        f=f**a
        x1=xmu-1.d0+x
        if (dabs(x-f).le.1.d-15)then
c CALL .... and compute C(L1)
         XL1=X1
             call c(xmu,xl1,cl1)
             go to 4
        endif
        i=i+1
        x=f
        go to 10
   4    continue
c
c L3 (on the right hand side of the big primary)
c
        i=0
        x=1.d0- xmu*7.d0/12.d0
```

```
C        x=x**a

  15        den=1.d0+2.d0*xmu + x*(2.d0+xmu+x)

        f=(1.d0-xmu)*(1.d0+x)**2/den
        f=f**a

        x1=xmu+x
        if (dabs(x-f).le.1.d-15)then
c CALL .... and compute C(L1)
         XL3=X1
             call c(xmu,xl3,cl3)
             go to 5
        endif
        i=i+1
        x=f
        go to 15
   5    continue

        end




c 2. A routine to compute the Jacobi integral 2*Omega(x,y)-(x'^2+y'^2)=C
c  BUT for a collinear equilibrium point, it is simply
c      C=2*Omega(x,0)
c
c
        subroutine c(xmu,xl,cl)
        implicit real*8(a-h,o-z)

        r1=dsqrt((xl-xmu)*(xl-xmu))
        r2=dsqrt((xl-xmu+1)*(xl-xmu+1))

        cl=2.*( 0.5*xl*xl + (1-xmu)/r1 + xmu/r2 + 0.5*xmu*(1-xmu) )
        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, C_initial)
        IMPLICIT REAL*8 (A-H,O-Z)
        common/param/xmu
        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_rtbp_(t,YI,idir,istep,pabs,prel,
     &  tmax,ht,iordre,ifl)

C
```

```
        CALL SECCIO(YI,GG,DGG)
        IF(GG*GA.LT.0.D0)go to 22
C        write(10,*)tfinal,(yi(ii),ii=1,4)

                r1=dsqrt((yi(1)-xmu)*(yi(1)-xmu)+yi(2)*yi(2))
        r2=dsqrt((yi(1)-xmu+1.d0)*(yi(1)-xmu+1.d0)+yi(2)*yi(2))
omeg=0.5d0*(yi(1)*yi(1)+yi(2)*yi(2))+(1.d0-xmu)/r1
     . +xmu/r2+0.5d0*xmu*(1.d0-xmu)
C=2.d0*omeg-(yi(3)*yi(3)+yi(4)*yi(4))

if (dabs(C-C_initial).ge. 1.d-8)then
   write(*,*) 'not same C'
   write(*,*) C
C    stop
endif

        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'
C            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_rtbp_(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
C        write(*,*)'tfinal point time ',tfinal
        write(*,*)(yf(ii),ii=1,1)
C        write(10,*)tfinal,(yf(ii),ii=1,4)
        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)
        common/param/xmu
        dimension x(20),f(20)
c

        umu=1.-xmu
        d1=x(1)-xmu
        d2=x(1)+umu

        r12=d1*d1+x(2)*x(2)
        r22=d2*d2+x(2)*x(2)
        r0=dsqrt(r12)
        r1=dsqrt(r22)

        r032=r12*r0
        r132=r22*r1
        r052=r12*r032
        r152=r22*r132

        omex=x(1)-(umu*(-xmu+x(1))/r032)-(xmu*(x(1)+umu)/r132)
        omey=x(2)*(1.-(umu/r032)-(xmu/r132))

omexx=1.-(umu*((r0*r0)-3.*d1)/(r0*r0*r0*r0*r0))
     .    -(xmu*( (r1*r1)-(3.*(umu+x(1))*(umu+x(1))) )/(r1*r1*r1*r1*r1))
omexy=x(2)*(((3.*umu*d1)/(r0*r0*r0*r0*r0))
     .    +(3.*xmu*(x(1)+umu))/ (r1*r1*r1*r1*r1))
omeyy=(1.-(umu/(r0*r0*r0))-(xmu/(r1*r1*r1))) + ( x(2)* ((3.
     .    *umu*x(2))/ (r0*r0*r0*r0*r0) )+ (xmu*3.*x(2))
     .      / (r1*r1*r1*r1*r1) )
```

```
f(1)=x(3)
f(2)= x(4)
f(3)=2.*x(4)+omex
f(4)=-2.*x(3)+omey

f(5)=x(13)
f(6)=x(14)
f(7)=x(15)
f(8)=x(16)

f(9)=x(17)
f(10)=x(18)
f(11)=x(19)
f(12)=x(20)

f(13)=x(5)*omexx+x(9)*omexy+2.*x(17)
f(14)=x(6)*omexx+x(10)*omexy+2.*x(18)
f(15)=x(7)*omexx+x(11)*omexy+2.*x(19)
f(16)=x(8)*omexx+x(12)*omexy+2.*x(20)

f(17)=x(5)*omexy+x(9)*omeyy-2.*x(13)
f(18)=x(6)*omexy+x(10)*omeyy-2.*x(14)
f(19)=x(7)*omexy+x(11)*omeyy-2.*x(15)
f(20)=x(8)*omexy+x(12)*omeyy-2.*x(16)

return
end
```

We note that we have used different increasing values of μ, and that we skipped $\mu = 0.02$ and the values around it. The reason why we do it is to avoid a singularity.

The (x, y) plot for $\mu = 0.008$ and for $\mu \in (0.001, 0.5)$ are:

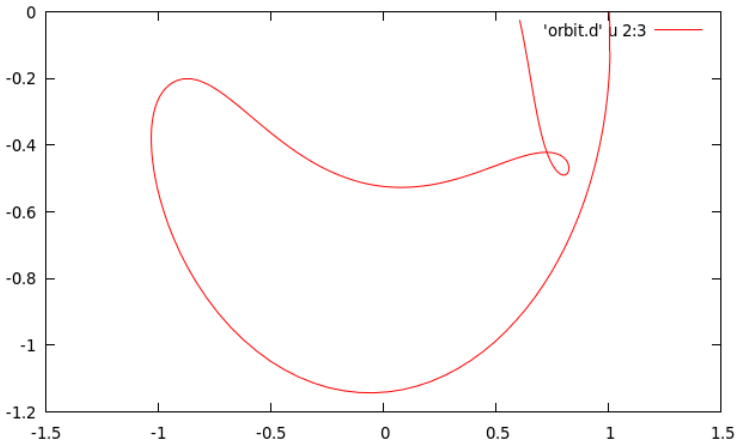

Figure 1: $\mu = 0.008$

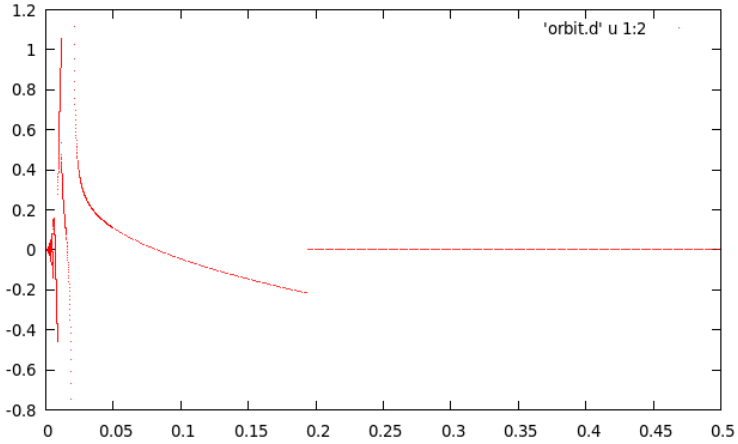

Figure 2: $\mu \in (0.001, 0.5)$

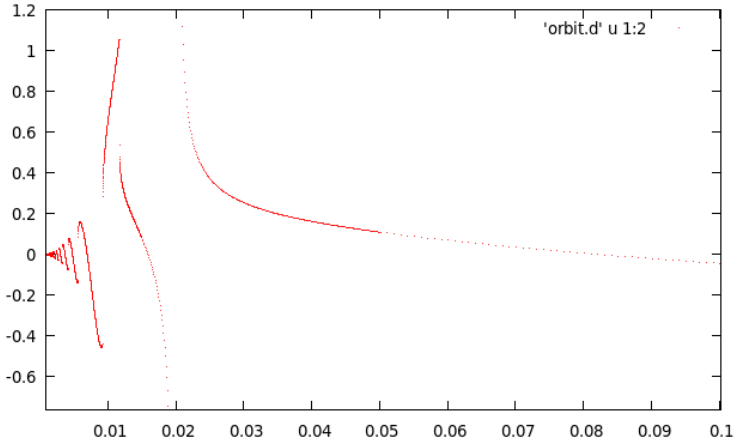

Figure 3: $\mu \in (0.001, 0.1)$