Numerics of Dynamical Systems

Assignment 10

Conny Schweigert

Barcelona
08.05.2015

1 Programme

main_rtbp_flow10.f was used for the plots of (xmu,x') or (xmu,x).
Since vebrep from package_alg calculates as eigenvalues almost only zeros I used always the first eigenvector.

Listing 1: main_rtbp_flow10.f

```
c*************************************************************
c
c  MAIN_RTBP_FLOW9.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=4,m=4)
        dimension xi(n),x(n),O(m,m),A(n,n),RR(n),RI(n),VR(n,n),VI(n,n)
        dimension v(n),p(2),yf(n)
        common/param/xmu
        open(10,file='orbit.d',status='unknown')
        write(*,*)'sign'
        read(*,*) iregion
        write(*,*) 'idir?'
        read(*,*) idir
        write(*,*) 'ncrossing?'
        read(*,*) ncrossing
        ti=0.d0
        tmax = 6.28
        np=30
        write(*,*) 'xmu?'
        read(*,*) xmu

         call peq(xmu,xl1,xl2,xl3,cl1,cl2,cl3)
         write(*,*) 'xl3', xl3
         write(*,*) 'cl3', cl3
```

```
C=cl3
x(1)=xl3
x(2)=0
x(3)=0
x(4)=0
call JacobiMatrix(n,x,xmu,A)
write(*,*) 'A'
      do i=1,n
         write(*,*) (A(i,j),j=1,n)
      enddo
call vapvep(A,n,RR,RI,VR,VI)

if(idir.gt.0) then
   do i=1,n
      if(RI(i).gt.0) then
      k=i
      endif
    enddo
else
   do i=1,n
      if(RI(i).lt.0) then
      k=i
      endif
    enddo
endif
k=1

do i=1,n
v(i) = VR(i,k)
enddo
p(1) = RR(k)
write(*,*) 'Eigenvalue', p(1)
write(*,*) 'v', (v(i),i=1,n)
s = 1.d-6
 if(iregion.lt.0) then
s = -s
endif
x = x + s * v
write(*,*) 'initial point', (x(i),i=1,n)
```

```
          call jacobi(x,C,xmu,n)

         ti=0
         do j = 1,ncrossing
         t=0.d0
           write(10,*)t,(x(i),i=1,n)
           call poinc1(j,xmu,n,m,x,yf,tfinal,idir,ti)
           ti = ti + tfinal
         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(j,xmu,n,m,YI,YF,tfinal,idirorig,ti)
         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
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_var_(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
         write(10,*)t,(yi(ii),ii=1,n)
         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(xmu,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_var_(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
```

```
c␣check␣first␣integral:␣to␣be␣done
␣␣␣␣␣␣␣␣write(*,*)'tfinal point time ',tfinal
␣␣␣␣␣␣␣␣write(*,*)(yf(ii),ii=1,n)
␣␣␣␣␣␣␣␣write(10,*)t,(yf(ii),ii=1,n)
␣␣␣␣␣␣␣␣return
␣␣␣␣␣␣␣␣t␣=␣tfinal
␣␣␣␣␣␣␣␣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(xmu,t,x,neq,f)
␣␣␣␣␣␣implicit␣real*8␣(a-h,o-z)
␣␣␣␣␣␣dimension␣x(neq),f(neq)
c
␣␣␣␣␣␣umu=1.d0-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
␣␣␣␣␣␣c1=-umu/r032-xmu/r132
␣␣␣␣␣␣c2=3.d0*umu/r052
␣␣␣␣␣␣c3=3.d0*xmu/r152
␣␣␣␣␣␣␣␣omex=x(1)-(umu*(-xmu+x(1))/r032)-(xmu*(x(1)+umu)/r132)
␣␣␣␣␣␣␣␣omey=x(2)*(1.d0-(umu/r032)-(xmu/r132))
␣␣␣␣␣␣␣␣omexx=c1+c2*d1*d1+c3*d2*d2+1.d0
␣␣␣␣␣␣␣␣omexy=c2*d1*x(2)+c3*d2*x(2)
␣␣␣␣␣␣␣␣omeyy=c1+(c2+c3)*x(2)*x(2)+1.d0
␣␣␣␣␣␣f(1)=x(3)
␣␣␣␣␣␣f(2)=x(4)
␣␣␣␣␣␣f(3)=2.*x(4)+omex
␣␣␣␣␣␣f(4)=-2.*x(3)+omey
␣␣␣␣␣␣return
␣␣␣␣␣␣end
c**********Jacobi-Matrix␣of␣x
␣␣␣␣␣␣subroutine␣JacobiMatrix(n,x,xmu,A)
␣␣␣␣␣␣implicit␣real*8␣(a-h,o-z)
␣␣␣␣␣␣dimension␣x(n),A(n,n)
c
␣␣␣␣␣␣umu=1.d0-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
␣␣␣␣␣␣c1=-umu/r032-xmu/r132
␣␣␣␣␣␣c2=3.d0*umu/r052
␣␣␣␣␣␣c3=3.d0*xmu/r152
␣␣␣␣␣␣␣␣omex=x(1)-(umu*(-xmu+x(1))/r032)-(xmu*(x(1)+umu)/r132)
␣␣␣␣␣␣␣␣omey=x(2)*(1.d0-(umu/r032)-(xmu/r132))
␣␣␣␣␣␣␣␣omexx=c1+c2*d1*d1+c3*d2*d2+1.d0
␣␣␣␣␣␣␣␣omexy=c2*d1*x(2)+c3*d2*x(2)
␣␣␣␣␣␣␣␣omeyy=c1+(c2+c3)*x(2)*x(2)+1.d0
␣␣␣␣␣␣A(1,1)=0
␣␣␣␣␣␣A(1,2)=0
␣␣␣␣␣␣A(1,3)=1
␣␣␣␣␣␣A(1,4)=0
␣␣␣␣␣␣A(2,1)=0
␣␣␣␣␣␣A(2,2)=0
␣␣␣␣␣␣A(2,3)=0
␣␣␣␣␣␣A(2,4)=1
␣␣␣␣␣␣A(3,1)=omexx
␣␣␣␣␣␣A(3,2)=omexy
␣␣␣␣␣␣A(3,3)=0
␣␣␣␣␣␣A(3,4)=2
␣␣␣␣␣␣A(4,1)=omexy
␣␣␣␣␣␣A(4,2)=omeyy
␣␣␣␣␣␣A(4,3)=-2
␣␣␣␣␣␣A(4,4)=0
␣␣␣␣␣␣return
␣␣␣␣␣␣end
```

main_rtbp_flow10b.f was used for the plots of the manifolds (x,y).

Listing 2: main_rtbp_flow10b.f

```
c************************************************************
c
c    MAIN_RTBP_FLOW10b.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=4)
       dimension xi(n),x(n),A(n,n),RR(n),RI(n),VR(n,n),VI(n,n)
       dimension v(n),p(2),yf(n)
       common/param/xmu
       open(10,file='orbit.d',status='unknown')
       iregion = -1
       idir = 1
       ncrossing = 1
       ti=0.d0
       tmax = 6.28
       np=30

       write(*,*) 'End␣of␣intervall?'
       read(*,*) c
       if (c==0.5) then
       b = 0.499
       m=211
       else
       b = 0.099
       m=161
       endif

        do l=1,m
        if (l.lt.151) then
        xmu=l*0.0001
        else
         xmu = -1.192+l*0.008
        endif
c        +l*((c-0.001)/m)
        call peq(xmu,xl1,xl2,xl3,cl1,cl2,cl3)
       C=cl3
```

```
        x(1)=xl3
        x(2)=0
        x(3)=0
        x(4)=0
        call JacobiMatrix(n,x,xmu,A)
        write(*,*) 'A'
            do i=1,n
              write(*,*) (A(i,j),j=1,n)
            enddo
        call vapvep(A,n,RR,RI,VR,VI)
c******* Correction of eigenvalues
c        RR(1) = 0
c        RR(2) = 0
c        RR(3) = -0.5016
c        RR(4) = 0.5016
c        RR = SNGL(RR)
c        RI = SNGL(RI)
c        VR = SNGL(VR)
c        VI = SNGL(VI)
        k=1
c         do i=1,n
c           if(RR(i).gt.0) then
c           k=i
c           endif
c          enddo

c****** Correction of eigenvectors
c        VR(1,3) = -0.2894
c        VR(2,3) = -0.8457
c        VR(3,3) = 0.1452
c        VR(4,3) = 0.4242
c        VR(1,4) = -0.2894
c        VR(2,4) = 0.8457
c        VR(3,4) = -0.1452
c        VR(4,4) = 0.4242
        do i=1,n
        v(i) = VR(i,k)
        enddo
        p(1) = RR(k)
c        p(2) = RI(2)
        write(*,*) 'Eigenvalue', p(1)
```

```
c,      '+i*', p(2)
       write(*,*) 'v', (v(i),i=1,n)
       s = 1.d-6
        if(iregion.lt.0) then
       s = -s
       endif
       x = x + s * v
       write(*,*) 'initial␣point', (x(i),i=1,n)

c         call jacobi(x,C,xmu,n)

       ti=0
       do j = 1,ncrossing
       t=0.d0
c         write(10,*)t,(x(i),i=1,n)
c          call jacobi(x,C,xmu,n)
         call poinc1(j,xmu,n,m,x,yf,tfinal,idir,ti)
         xdfinal = yf(3)
         write(10,*) xmu, xdfinal
         ti = ti + tfinal
       end do
       end do
       end
c*********Computes Jacobi-Constant
      subroutine jacobi(x,C,xmu,n)
      implicit real*8 (a-h,o-z)
      dimension x(n)
      ro = dsqrt((x(1) - xmu)*(x(1) - xmu))
      rt = dsqrt((x(1) - xmu + 1.d0)*(x(1) - xmu + 1.d0))
      ome =0.5d0*(x(1)*x(1))+(1.d0-xmu)/ro+xmu/rt+0.5d0*xmu*(1.d0-xmu)
      Cnew = 2*ome
      Cdiff = dabs(C - Cnew)
      if (Cdiff.gt.1.d-3) then
      write(*,*) 'Jacobi␣constant␣not␣conserved'
      endif
      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(j,xmu,n,m,YI,YF,tfinal,idirorig,ti)
      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
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_var_(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,*)t,(yi(ii),ii=1,n)
      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(xmu,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_var_(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
        xdfinal = YF(3)
c check first integral: to be done
c        write(*,*)'tfinal point time ',tfinal
c        call checkperiod(j,tfinal)
c        write(*,*)(yf(3))
c        write(10,*) xdfinal
c        call matrix(yf,m,n)
        return
        t = tfinal
        end


C***********************************************************************
C                                                                      *
C     THE SURFACE g OF SECTION, IN THIS CASE
```

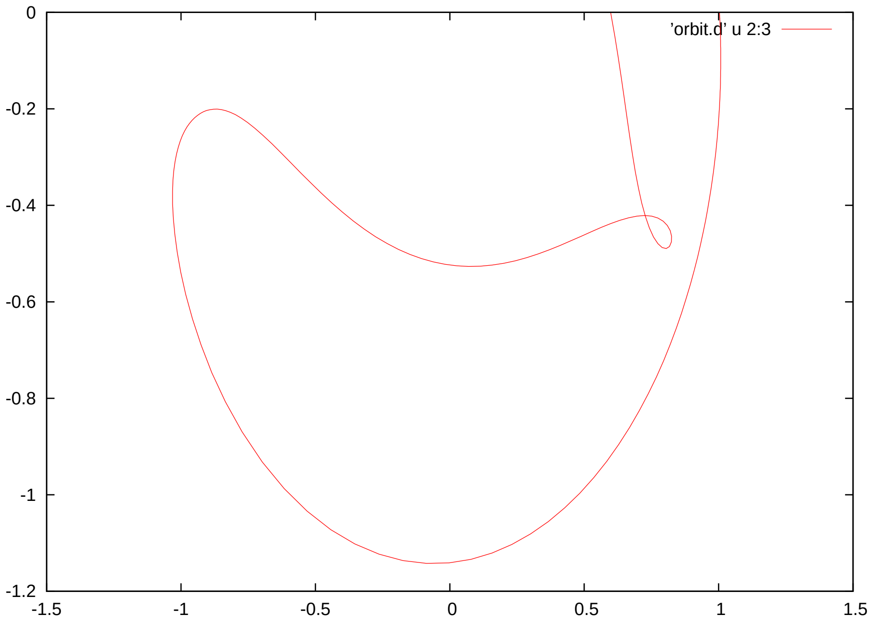
```
C␣␣␣␣␣␣␣␣␣␣INPUT␣PARAMETERS:
C␣␣␣␣␣Y(*)␣␣␣␣␣␣␣POINT
C␣␣␣␣␣␣␣␣␣␣OUTPUT␣PARAMETERS:
C␣␣␣␣␣GG␣␣␣␣␣␣␣␣␣␣FUNCTION␣THAT␣EQUATED␣TO␣0␣GIVES␣THE␣SURFACE␣OF
C␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣SECTION
C␣␣␣␣␣DGG(*)␣␣␣␣␣␣GRADIENT␣OF␣FUNCTION␣GG
C␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣*
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(xmu,t,x,neq,f)
␣␣␣␣␣␣implicit␣real*8␣(a-h,o-z)
␣␣␣␣␣␣dimension␣x(neq),f(neq)
c
␣␣␣␣␣␣umu=1.d0-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
␣␣␣␣␣␣c1=-umu/r032-xmu/r132
␣␣␣␣␣␣c2=3.d0*umu/r052
␣␣␣␣␣␣c3=3.d0*xmu/r152
␣␣␣␣␣␣␣␣omex=x(1)-(umu*(-xmu+x(1))/r032)-(xmu*(x(1)+umu)/r132)
␣␣␣␣␣␣␣␣omey=x(2)*(1.d0-(umu/r032)-(xmu/r132))
␣␣␣␣␣␣␣␣omexx=c1+c2*d1*d1+c3*d2*d2+1.d0
␣␣␣␣␣␣␣␣omexy=c2*d1*x(2)+c3*d2*x(2)
␣␣␣␣␣␣␣␣omeyy=c1+(c2+c3)*x(2)*x(2)+1.d0
␣␣␣␣␣␣f(1)=x(3)
␣␣␣␣␣␣f(2)=x(4)
␣␣␣␣␣␣f(3)=2.*x(4)+omex
␣␣␣␣␣␣f(4)=-2.*x(3)+omey
␣␣␣␣␣␣return
␣␣␣␣␣␣end
c**********Jacobi-Matrix␣of␣x
␣␣␣␣␣␣subroutine␣JacobiMatrix(n,x,xmu,A)
␣␣␣␣␣␣implicit␣real*8␣(a-h,o-z)
␣␣␣␣␣␣dimension␣x(n),A(n,n)
c
␣␣␣␣␣␣umu=1.d0-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
␣␣␣␣␣␣c1=-umu/r032-xmu/r132
␣␣␣␣␣␣c2=3.d0*umu/r052
␣␣␣␣␣␣c3=3.d0*xmu/r152
␣␣␣␣␣␣␣␣omex=x(1)-(umu*(-xmu+x(1))/r032)-(xmu*(x(1)+umu)/r132)
␣␣␣␣␣␣␣␣omey=x(2)*(1.d0-(umu/r032)-(xmu/r132))
```

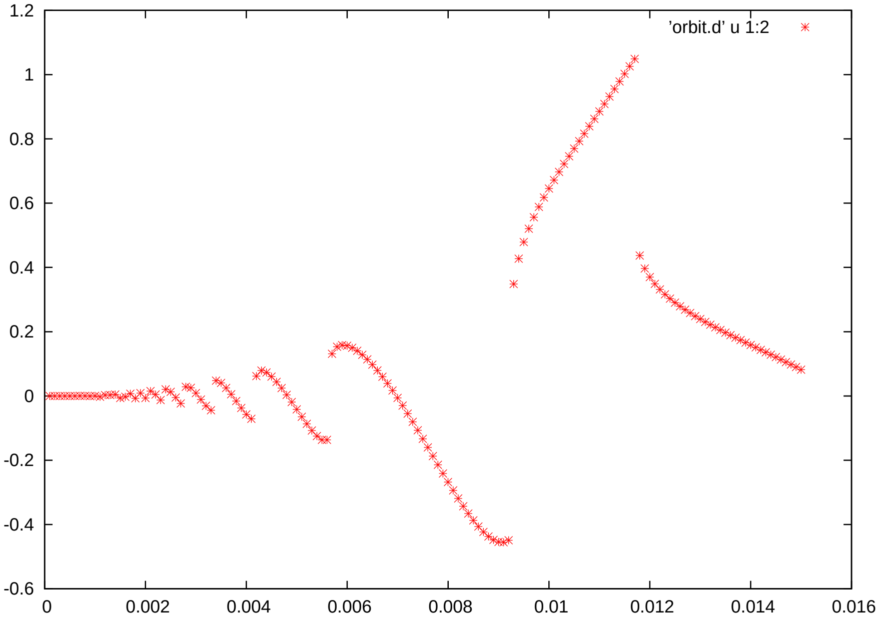
```
␣␣␣␣␣␣␣␣omexx=c1+c2*d1*d1+c3*d2*d2+1.d0
␣␣␣␣␣␣␣␣omexy=c2*d1*x(2)+c3*d2*x(2)
␣␣␣␣␣␣␣␣omeyy=c1+(c2+c3)*x(2)*x(2)+1.d0
␣␣␣␣␣␣A(1,1)=0
␣␣␣␣␣␣A(1,2)=0
␣␣␣␣␣␣A(1,3)=1
␣␣␣␣␣␣A(1,4)=0
␣␣␣␣␣␣A(2,1)=0
␣␣␣␣␣␣A(2,2)=0
␣␣␣␣␣␣A(2,3)=0
␣␣␣␣␣␣A(2,4)=1
␣␣␣␣␣␣A(3,1)=omexx
␣␣␣␣␣␣A(3,2)=omexy
␣␣␣␣␣␣A(3,3)=0
␣␣␣␣␣␣A(3,4)=2
␣␣␣␣␣␣A(4,1)=omexy
␣␣␣␣␣␣A(4,2)=omeyy
␣␣␣␣␣␣A(4,3)=-2
␣␣␣␣␣␣A(4,4)=0
␣␣␣␣␣␣return
␣␣␣␣␣␣end
```

2 Plots

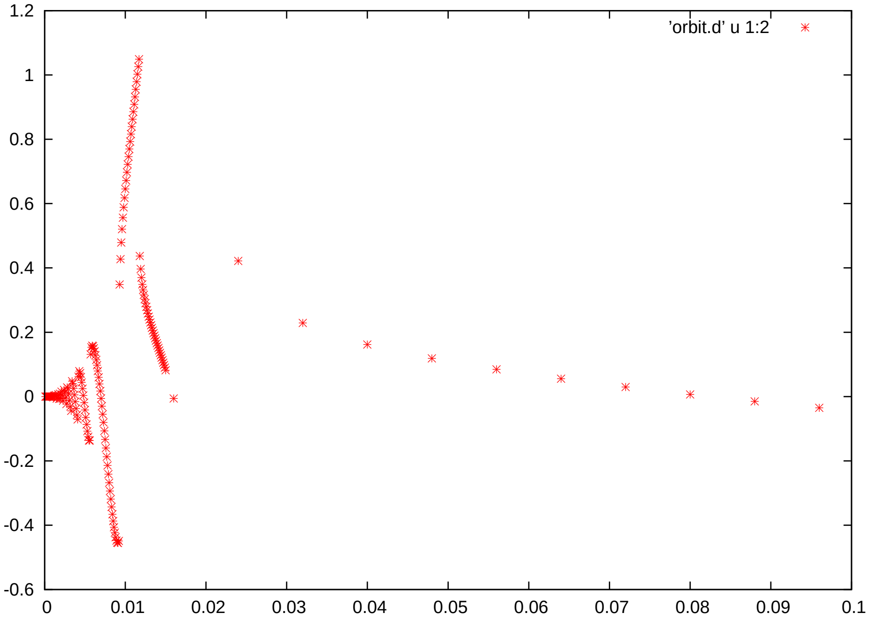
2.1 Orbit (x,y) of mu = 0.008



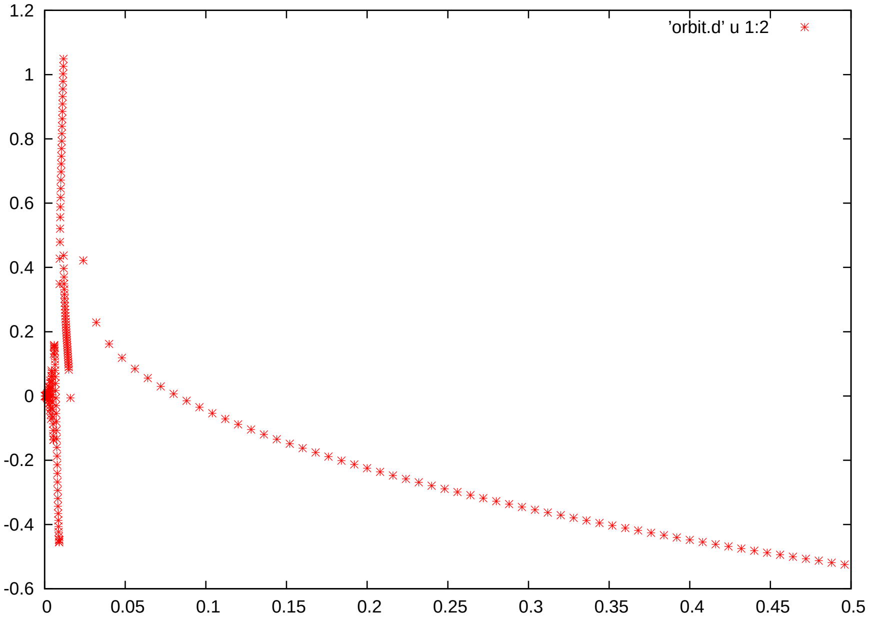
2.2 Graph (xmu,x') for mu in (0.001,0.015)



2.3 Graph (xmu,x') for mu in (0.001,0.1)

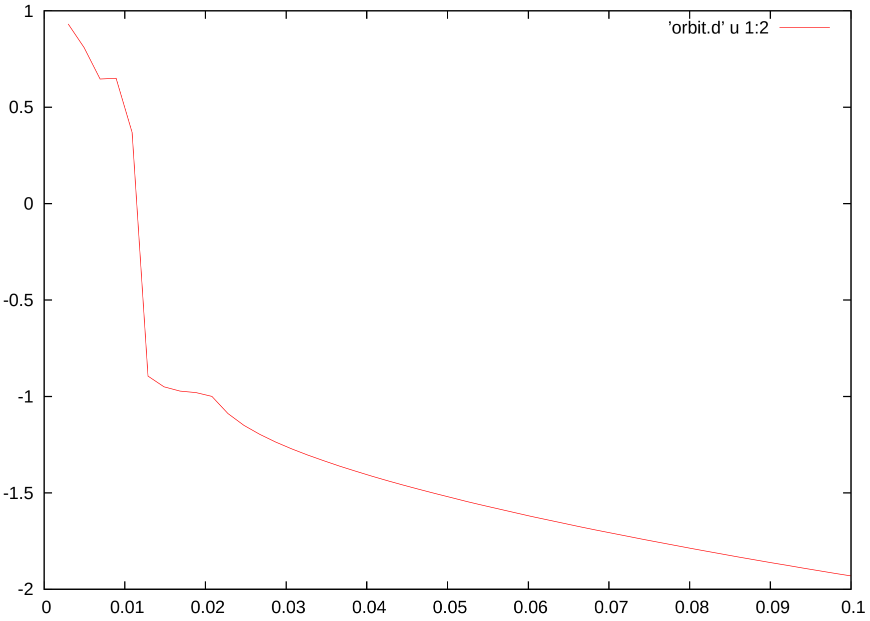


2.4 Graph (xmu,x') for mu in (0.001,0.5)



3 Optional Part

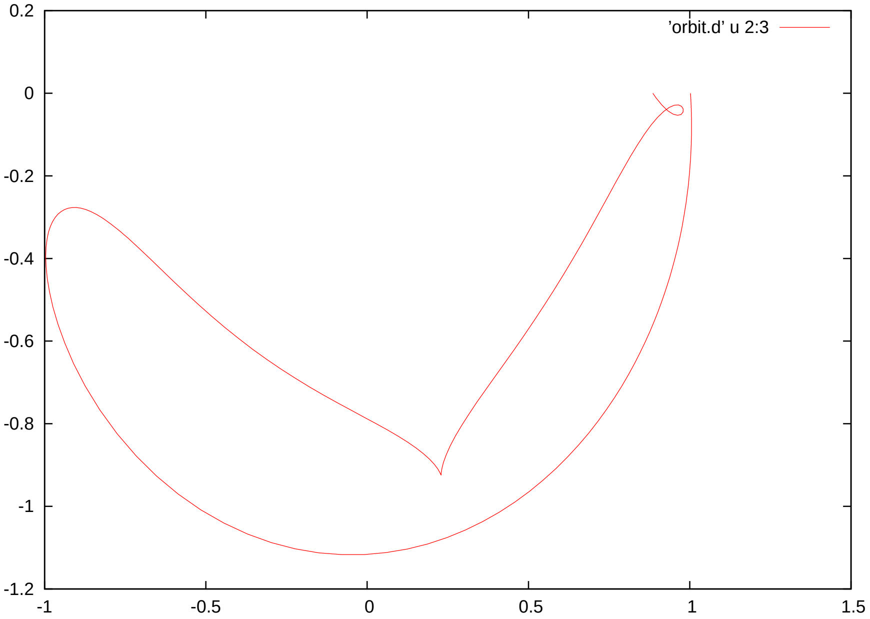
3.1 Curve (mu,x1)



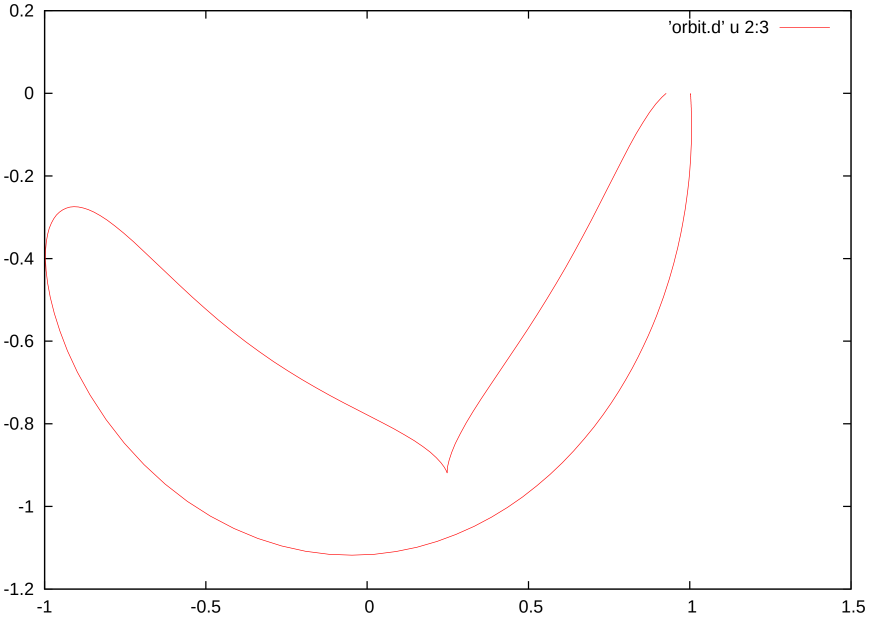
The value of x_1 at the first crossing is decreasing in μ. It starts with $x_1 = 1$, becomes negative approximately at $\mu = 0.015$ and decreases further until almost $x_1 = -2$ at $\mu = 0.1$. The curve is not always strictly decreasing. Approximately at $\mu = 0.1$ x_1 is slightly increasing as one can also see on the plots for $\mu = 0.0056$ to $\mu = 0.0057$. What the plot not shows is that x_1 is still decreasing in μ as one may see on the plots for $\mu = 0.01179$ to $\mu = 0.083$.

3.2 Part 3

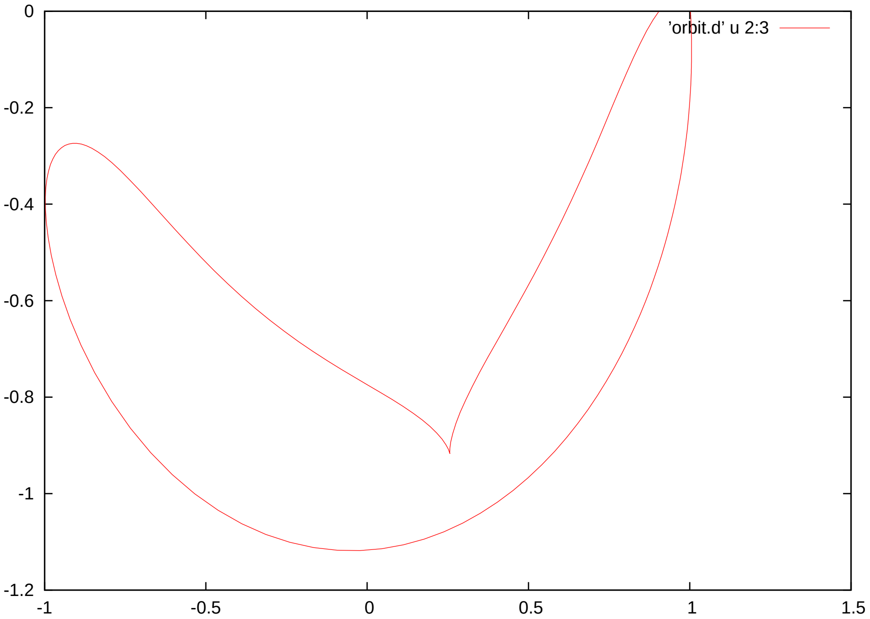
3.2.1 (x,y)-manifold for mu=0.0056



3.2.2 (x,y)-manifold for mu=0.00567



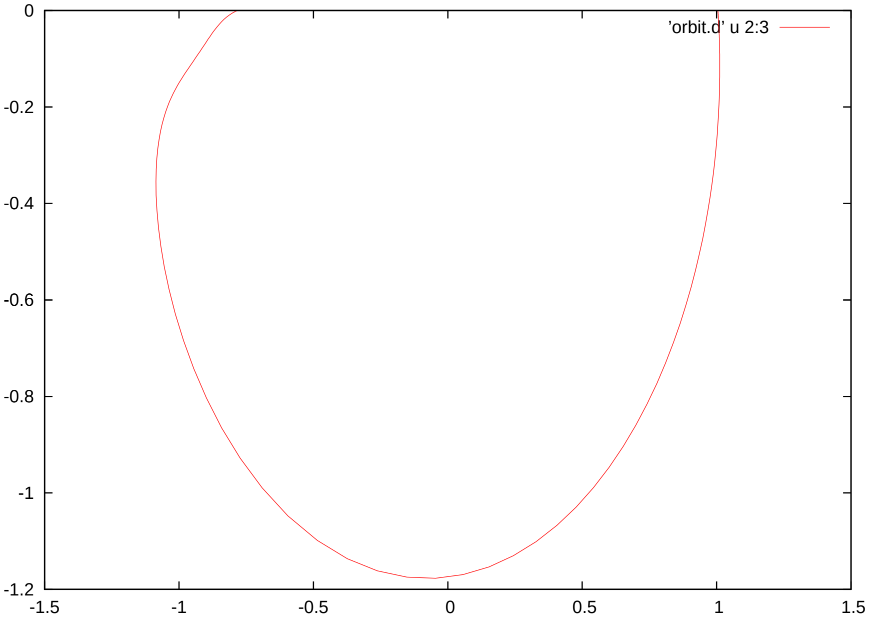
3.2.3 (x,y)-manifold for mu=0.0057



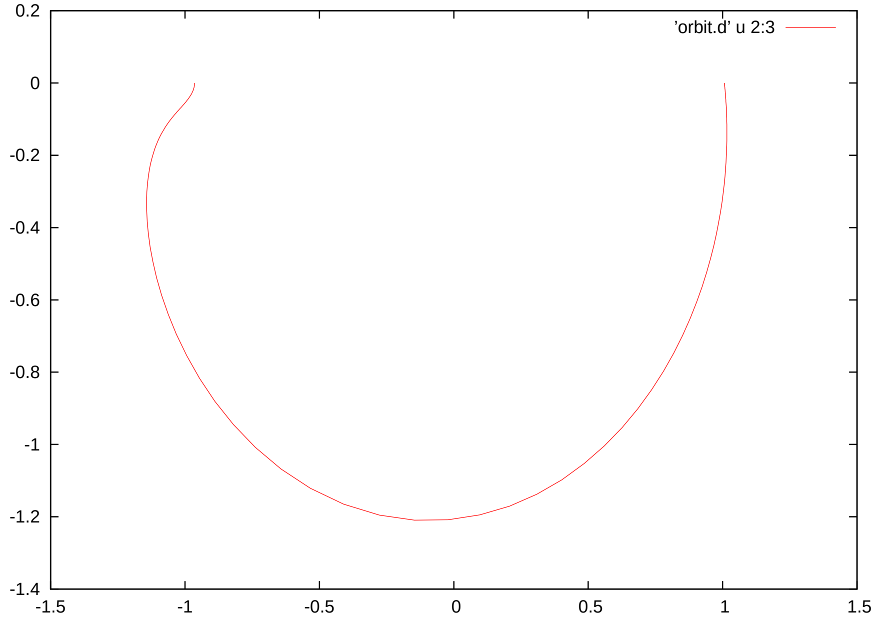
The graphs for $\mu = 0.0056$ to $\mu = 0.0057$ show the parts where x_1 at the first crossing is slightly increasing.

3.3 Part 4

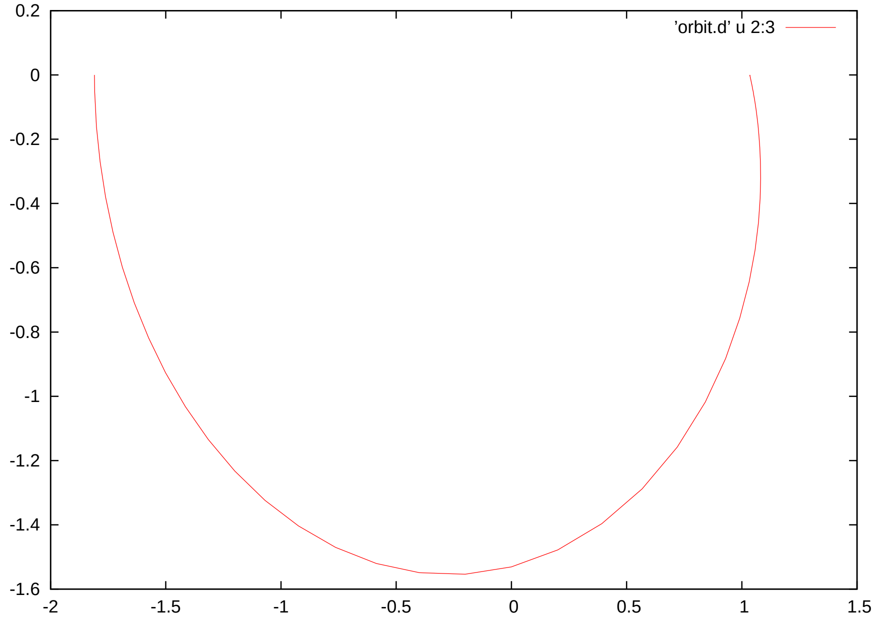
3.3.1 (x,y)-manifold for mu=0.01179



3.3.2 (x,y)-manifold for mu=0.0159375



3.3.3 (x,y)-manifold for mu=0.083



The graph shows a part of an orbit which opens more when μ increases, because the initial x increases and x_1 at the crossing decreases.