Poincaré sections of Hamiltonian autonomous systems in Maxima

José Antonio Vallejo

Facultad de Ciencias
Universidad Autónoma de San Luis Potosí (México)
URL: `http://galia.fc.uaslp.mx/~jvallejo`
`email: josanv@gmail.com`

Version: February 15, 2017

ABSTRACT: This is my attempt to recreate the Poincare subpackge for computing Poincaré sections of autonomous Hamiltonian systems, present in the DEtools package of Maple™[1], and due to Cheb-Terrab and Oliveira (see Comp. Phys. Comm. **95** 2–3 (1996) 171–189).

IMPORTANT REMARK: If you are *not* using the WxMaxima frontend, omit the `wxplot_size` commands, and use `draw3d`, `draw2d` instead of `wxdraw3d`, `drawd2d` when replicating the commands given here.

1 Introduction

The package `poincare.mac` contains three functions: `hameqs`, `poincare3d`, and `poincare2d`, and calls a pre-compiled fourth function `rkfun.fasl`, obtained from `rkfun.lisp`, a modification of the vanilla `rk` function of Maxima with optimized code due to Richard Fateman. The original file is available at the URL `https://people.eecs.berkeley.edu/~fateman/lisp/rkfun.lisp`, and a copy of the lisp and the fasl files is distributed with `poincare.mac`. The files can be downloaded from:
`http://galia.fc.uaslp.mx/~jvallejo/poincare.mac`
`http://galia.fc.uaslp.mx/~jvallejo/rkfun.lisp`
`http://galia.fc.uaslp.mx/~jvallejo/rkfun.fasl`
`http://galia.fc.uaslp.mx/~jvallejo/PoincareDocumentation.pdf`
This document describes the syntax for these functions and gives some examples of use, with applications to some well-known and documented physical systems (to ease comparison with other programs).

The code performs far better than the corresponding command in Maple™. Nevertheless, the main goal here is usability and user-friendliness, leaving aside

[1] Used here: Maple 2016:1a (build 1133417). Maplesoft, a division of Waterloo Maple Inc., Waterloo, Ontario.

technical questions regarding numerical analysis. All the timings shown have been obtained in a Slackware 14.2 box running an Intel™ processor 4x Intel(R) Core(TM) i5-4590T CPU @ 2.00GHz, with 12199MB of memory. The Maxima version was 5.38.0, using SBCL-LISP 1.3.5, and the wxMaxima one 16.4.2.

To test the package, the easiest way is to put a copy of files `poincare.mac` and `rkfun.fasl` in your working directory and do `batch("poincare.mac")`. That's all. Then, you can proceed directly to the examples in section 2.
If desired, the package can be installed system-wide by putting a copy of the files `poincare.mac` and `rkfun.lisp` in a folder contained in the environment variable `file_search_maxima`, such as `/usr/share/maxima/5.38.0/share/contrib/` (you may need administrator rights in order to do that), and loading them with `batch("poincare.mac")`.

Suggestions and corrections are more than welcome. I take the opportunity to thank Prof. R. Fateman for the highly optimized `rkfun.lisp` code.

Start a Maxima session (I will use wxMaxima in what follows, please read the "Important remark" following the abstract) and load the package:

```
(%i1) batch("poincare.mac");
```

The functions defined in the package will show up, let us briefly discuss them. The first function constructs the Hamiltonian equations for a given Hamiltonian $H(q1, p1, \ldots, q2, p2)$. Any names can be used for the variables, but they must be given in pairs "coordinate, conjugate momentum". A good choice (used internally) is $(q1, p1, ..., qn, pn)$. A name must be provided for the components of the Hamiltonian vector field

$$X_H(q_1, p_1, ..., q_n, p_n) = \sum_{i=1}^{n} \left(\frac{\partial H}{\partial p_i} \frac{\partial}{\partial q_i} - \frac{\partial H}{\partial q_i} \frac{\partial}{\partial p_i} \right) .$$

Once a name, say XH, is chosen, the components of the Hamiltonian vector field will be globally defined functions XHj with $1 \leq j \leq 2n$, where n is the number of degrees of freedom, and will be available to Maxima. Notice that, for instance,

$$XH1(t, q_1, p_1, ..., q_n, p_n) = \frac{\partial H}{\partial p_1}$$

and

$$XH2(t, q_1, p_1, ..., q_n, p_n) = -\frac{\partial H}{\partial q_1} .$$

Here is the code:

```
(%i2) hameqs(H,name):=block([vv,t,tvv,n,Q,P,eqq,eqp,eqs],
      vv:args(lhs(apply(fundef,[H]))),tvv:cons(t,vv),
      n:length(vv)/2,
      Q:makelist(vv[2*j-1],j,1,n),P:makelist(vv[2*k],k,1,n),
      eqq:makelist(float(diff(apply(H,vv),P[j])),j,1,n),
      eqp:makelist(float(-diff(apply(H,vv),Q[j])),j,1,n),
      eqs:join(eqq,eqp),
      for j thru 2*n do
          define(funmake(concat(name,j),tvv),
                 block([],mode_identity(float,vv),eqs[j])),
      apply(compile,makelist(concat(name,j),j,1,2*n)),
      [makelist(apply(concat(name,j),tvv),j,1,2*n),vv,
       makelist(concat(name,j),j,1,2*n)]
)
```

Important remark: Although we will work with *autonomous* Hamiltonian systems, the components XHj returned by this command will have the set $(t, q_1, p_1, ..., q_n, p_n)$ as arguments. This is necessary to maintain consistency with the `rkfun` function, which can work with both, autonomous and non-autonomous systems. As we will see in the examples, the output of `hameqs` is a list composed of three sublists. The first sublist gives the expressions for the components XHj, which will be always free of the t coordinate, so there is little chance of making a mistake when working with them. But if at some point you get a message talking about "wrong number of arguments" or something similar, this could be the cause. The other sublists contained in the output of `hameqs` are a list of the phase-space variables as defined by the user, and a list with the names of the compothenents of the Hamiltonian vector field (in the notational example we are using, it would be `[XH1,...,XHn]`).

Next, it follows an auxilliary function to remove exactly one element at a given position, from a given list:

```
(%i3) lextract(ll,n):=block([l,a,b],
      l:length(ll),a:rest(ll,n),b:rest(ll,n-l-1),
      append(b,a)
      )
```

`poincare3d` gives the projection of the Hamiltonian orbits along a certain coordinate which is given as an argument `coord`. Other arguments are: a list of initial conditions `inicond` $= [q_1(0), p_1(0), \ldots, q_n(0), p_n(0)]$, and a list characterizing the time domain `timestep` $= [t, t_{ini}, t_{fin}, step]$. These arguments are also present in the `poincare2d` function below:

```
(%i4) poincare3d(H,name,inicond,timestep,coord):=block(
        [heq,vars,tvars,hfuns,c,sol],
        heq:hameqs(H,name),
        vars:heq[2],
        tvars:cons(t,vars),
        hfuns:heq[3],
        c:first(sublist_indices(vars,lambda([x],x = coord))),
        sol:rkfun(hfuns,tvars,float(inicond),timestep),
        map(lambda([x],lextract(x,c)),map(lambda([x],rest(x)),sol))
      )
```

The function `poincare2d` gives the surface of section selected by a list of arguments of the form `scene` $= [q0, c, qi, qj]$, that is, the surface $q0 = c$ in which coordinates $[qi, qj]$ are shown. It is the main command in the package, and its syntax tries to imitate the one used in Maple™. The method used in the computation of the Poincaré surface is that described in the paper by Cheb-Terrab and Oliveira: we select a set of initial conditions, follow the corresponding orbit numerically (at this point the `rkfun` is used in all its glory), and detect where we have crossing the $q0 = c$ surface by looking at changes of sign in the list of values for this coordinate minus c.

```
(%i5) poincare2d(H,name,inicond,timestep,scene):=block(
        [heq,vars,tvars,hfuns,solu,c,soluc,sola,solb,subind,sol,e,f],
        heq:hameqs(H,name),
        vars:heq[2],
        tvars:cons(t,vars),
        hfuns:heq[3],
        solu:rkfun(hfuns,tvars,float(inicond),timestep),
        c:first(sublist_indices(vars,lambda([x],x=first(scene)))),
        soluc:map(lambda([x],x[c+1]),solu),
        sola:rest(soluc,-1)-second(scene),
        solb:rest(soluc)-second(scene),
        subind:sublist_indices(sola*solb,lambda([x],is(x<0))),
        sol:makelist(solu[k],k,subind),
        e:first(sublist_indices(vars,lambda([x],x=third(scene))))+1,
        f:first(sublist_indices(vars,lambda([x],x=fourth(scene))))+1,
        makelist([j[e],j[f]],j,sol)
      )
```

2 Example: Two uncoupled oscillators (rational frequencies)

We follow here S. Lynch: *Dynamical Systems with Applications using Maple*, Birkhauser (2001), Chapter 9.
The Hamiltonian of two uncoupled oscillators is (in (x, v, q, p) coordinates, just to show that any naming convention can be used. But, of course, here v is the canonical momentum conjugate to x, and p that conjugate to q):

(%i6)
```
H(x,v,q,p):=w1*(x^2+v^2)/2+w2*(q^2+p^2)/2$
```

The corresponding Hamiltonian equations are:

(%i7)
```
hameqs(H,XH);
```

$$[v\,w1\,, -w1\,x, p\,w2, -q\,w2] \qquad (\%o7)$$

Let us fix the values of the frequencies $w1$, $w2$:

(%i8)
```
w1:8$
```

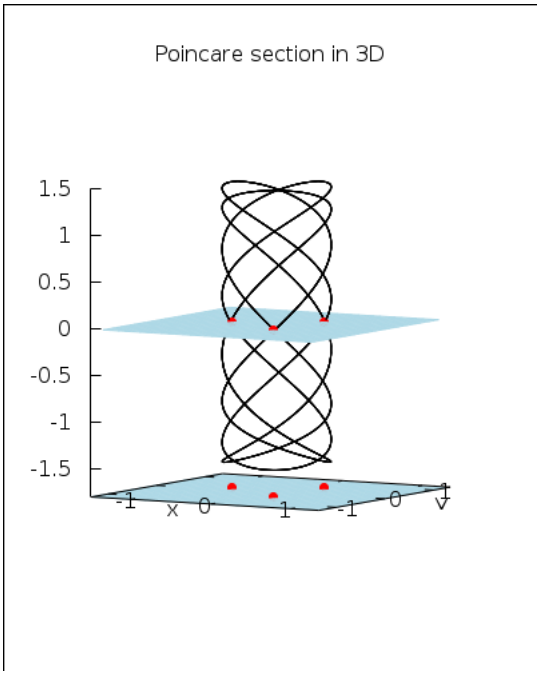
(%i9)
```
w2:3$
```

Next we plot the $3D$ Poincaré surface by projecting along the p coordinate (thus, the resulting graphics has (x, v, q) coordinates):

(%i10)
```
wxplot_size:[400,500]$
```

(%i11)
```
data1:poincare3d(H,XH,[0.3,0.5,0,1.5],[t,0,40,0.01],p)$
```

(%i12)
```
wxdraw3d(title="Poincare section in 3D",
dimensions=[350,500],view=[85,30],
xlabel="x",ylabel="v",zlabel="q",
xtics=1,ytics=1,
surface_hide=true,color="light-blue",
    explicit(0,x,-1.35,1.35,y,-1.35,1.35),
point_size=0,points_joined=true,color=black,line_width=1,
    points(data1),
user_preamble="set xyplane at -1.8",color="light-blue",
    explicit(-1.8,u,-1.5,1.35,v,-1.35,1.35),
point_size=1,point_type=filled_circle,color=red,points_joined=false,
    points([[-0.56,0,-1.78],[0.28,-0.52,-1.78],[0.3,0.48,-1.78],
            [-0.56,0,0],[0.28,-0.52,0],[0.3,0.48,0]]));
```



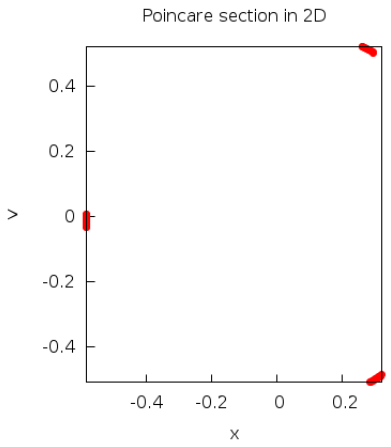
(%t12)

(%o12)

The graphics shows a $2D$ section at $q = 0$. Let's check this with the `poincare2d` command:

```
(%i13) data2:poincare2d(H,XH,[0.3,0.5,0,1.5],[t,0,40,0.01],[q,0,x,v])$

(%i14) wxdraw2d(title="Poincare section in 2D",
       xlabel="x",ylabel="v",
       xtics=0.2,
       point_size=1,point_type=7,color=red,
       points_joined=false,proportional_axes=xy,
           points(data2));
```



(%t14)

(%o14)

3 Example: Two uncoupled oscillators (irrational frequencies)

Again, we follow S. Lynch: *Dynamical Systems with Applications using Maple*, Birkhauser (2001), Chapter 9. The Hamiltonian is the same as above, but now the frequencies have irrational ratio:

(%i15) `remvalue(w1,w2)$`

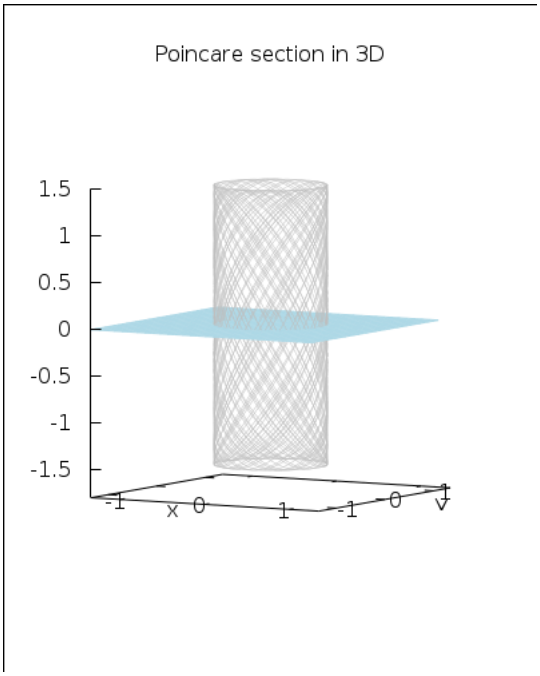
(%i16) `w1:sqrt(2)$`

(%i17) `w2:1$`

(%i18) `data3:poincare3d(H,XH,[0.3,0.5,0,1.5],[t,0,300,0.01],p)$`

Things look different now, the orbits are dense:

(%i19)
```
wxdraw3d(title="Poincare section in 3D",
dimensions=[350,500],view=[85,30],
xlabel="x",ylabel="v",zlabel="q",
xtics=1,ytics=1,
surface_hide=true,color="light-blue",
    explicit(0,x,-1.35,1.35,y,-1.35,1.35),
point_size=0,points_joined=true,color=gray,line_width=0.5,
    points(data3),
user_preamble="set xyplane at -1.8"
);
```



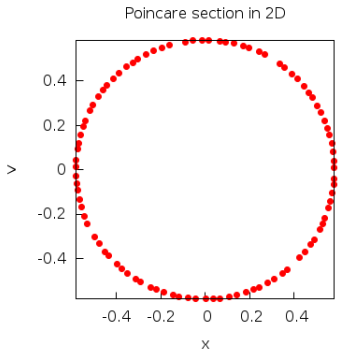
(%t19)

(%o19)

The $2D$ section reflects that change of behavior:

(%i20)
```
data4:poincare2d(H,XH,[0.3,0.5,0,1.5],[t,0,300,0.01],[q,0,x,v])$
```

```
(%i21) wxdraw2d(title="Poincare section in 2D",
       xlabel="x",ylabel="v",xtics=0.2,proportional_axes=xy,
       point_size=1,point_type=7,color=red,points_joined=false,
           points(data4));
```

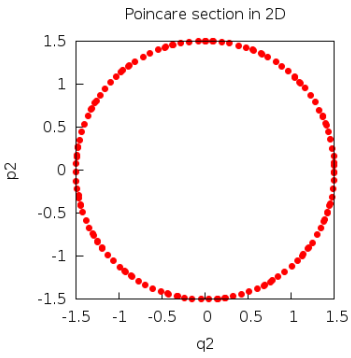


(%t21)

(%o21)

The same occurs in the $[q, p]$ plane (projection along v, $2D$ section at $x = 0$):

```
(%i22) data5:poincare2d(H,XH,[0.3,0.5,0,1.5],[t,0,300,0.01],[x,0,q,p])$
(%i23) wxdraw2d(title="Poincare section in 2D",
       xlabel="q2",ylabel="p2",xtics=0.5,proportional_axes=xy,
       point_size=1,point_type=7,color=red,points_joined=false,
           points(data5));
```



(%t23)

(%o23)

The conclusion is: the (quasiperiodic) motion takes place in tori! The KAM theorem tells us what happens with these tori, and the trajectories on them, when the Hamiltonian H contains a non-linear term (perturbation), and the energy increases. The kind of behavior described by KAM will be illustrated in the following examples (although the standard KAM theorem may not apply if the frequencies w_1, w_2 are commensurable!. Some additional reasoning is required to apply the theorem, as done in R. Cuerno et al.: *Deterministic chaos in the elastic pendulum: A simple laboratory for non linear dynamics*, Am. J. Phys. **60** 1 (1992) 73–79).

4 Example: The Hénon-Heiles model

Here, we intend to reproduce the graphics in Cheb-Terrab and Oliveira: *Poincaré sections of Hamiltonian systems*. Comp. Phys. Comm. **95** 2-3 (1996) 171–189.

```
(%i24) wxplot_size:[600,400]$
```

The Hamiltonian in this case is:

```
(%i25) K(q1,p1,q2,p2):=1/2*(q1^2+p1^2+q2^2+p2^2)+q2*q1^2-q2^3/3$
```

We fix some initial conditions for (q_1, q_2, p_2), the total energy H (in this case, $H = 1/24$), and then proceed to compute the initial values of p_1 corresponding to these:

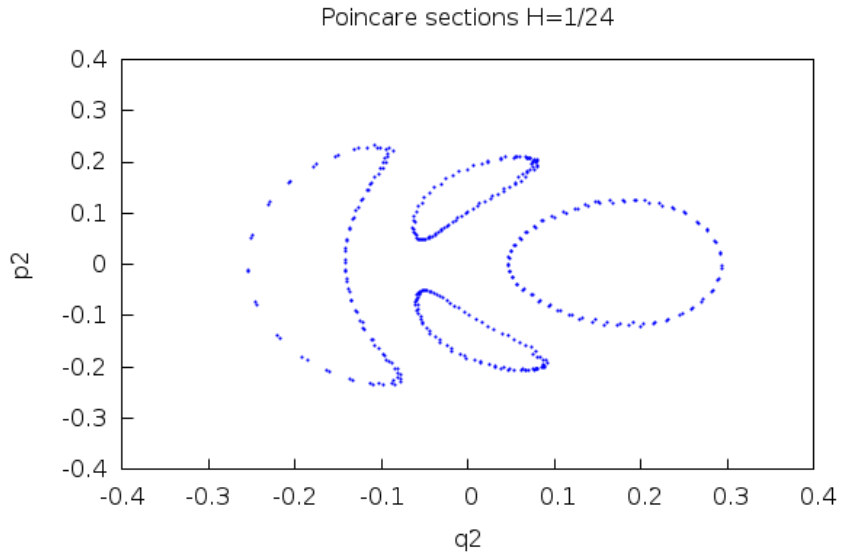
```
(%i26) series1:realroots(K(-0.2,p1,-0.2,0.1)=1/24)$
(%i27) data6:poincare2d(
       K,XK,[-0.2,rhs(first(series1)),-0.2,0.1],[t,-300,300,0.05],[q1,0,q2,p2]
       )$
(%i28) time(%);
```

[0.247] (%o28)

```
(%i29) data7:poincare2d(
       K,XK,[-0.2,rhs(second(series1)),-0.2,0.1],[t,-300,300,0.05],[q1,0,q2,p2]
       )$
(%i30) time(%);
```

[0.222] (%o30)

```
(%i31) wxdraw2d(title="Poincare sections H=1/24",
       xlabel="q2",ylabel="p2",
       point_type=7,point_size=0.2,
           points(append(data6,data7)),
       xrange=[-0.4,0.4],yrange=[-0.4,0.4]);
```



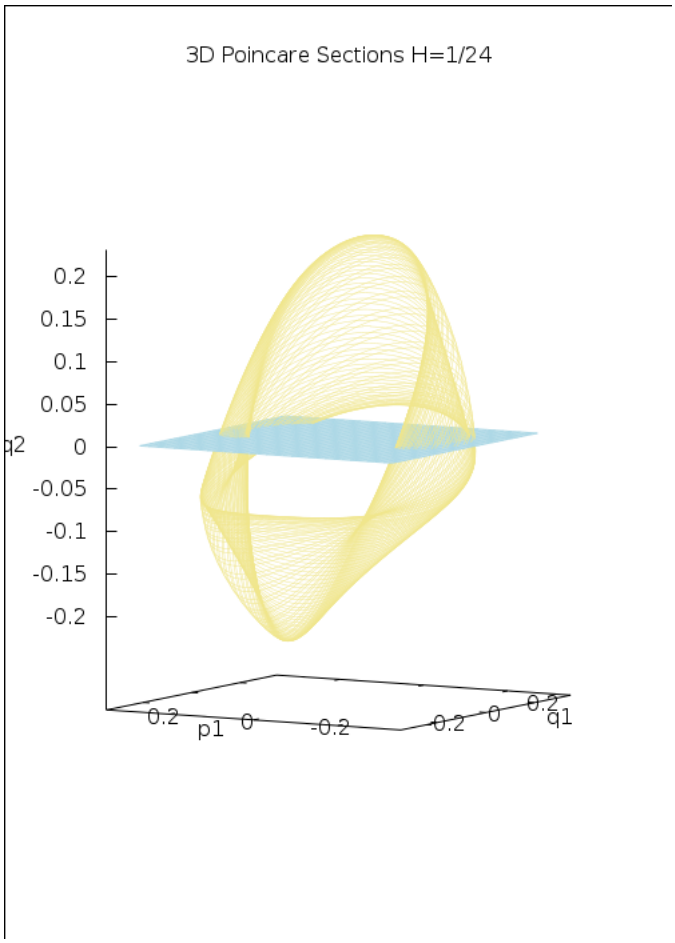
(%t31)

(%o31)

Let us see what happens from a $3D$ perspective. We clearly see the origin of the "wings" in the graphics:

```
(%i32) data8:poincare3d(
       K,XK,[-0.2,rhs(second(series1)),-0.2,0.1],[t,-300,300,0.05],p2
       )$
(%i33) wxplot_size:[500,700]$
```

```
(%i34) wxdraw3d(title="3D Poincare Sections H=1/24",
       dimensions = [500, 700],view = [85,300],
       surface_hide=true,color="light-blue",
        explicit(0,x,-0.3,0.3,y,-0.3,0.3),
       point_size=0,points_joined=true,color=khaki,line_width=0.5,
        points(data8),
       xlabel="q1",ylabel="p1",zlabel="q2",
       user_preamble="set xyplane at -0.31",
       xrange=[-0.35,0.35],yrange=[-0.35,0.35],xtics=0.2,ytics=0.2);
```



(%t34)

(%o34)

In the following commands, we increase the energy to $E = 1/18$, $E = 1/12$, and finally $E = 1/6$, the onset of chaos:

(%i35)
```
series2:realroots(K(-0.2,p1,-0.2,0.1)=1/18)$
```

(%i36)
```
data9:poincare2d(
K,XK,[-0.2,rhs(first(series2)),-0.2,0.1],[t,-500,500,0.01],[q1,0,q2,p2]
)$
```

(%i37)
```
time(%);
```

[1.947] (%o37)

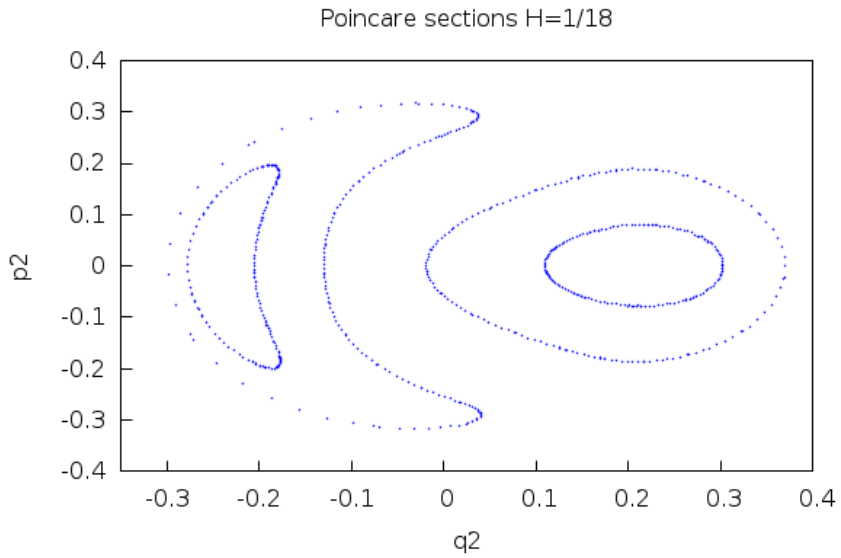
(%i38)
```
data10:poincare2d(
K,XK,[-0.2,rhs(second(series2)),-0.2,0.1],[t,-500,500,0.01],[q1,0,q2,p2]
)$
```

(%i39)
```
time(%);
```

[1.878] (%o39)

(%i40)
```
wxplot_size:[600,400]$
```

(%i41)
```
wxdraw2d(title="Poincare sections H=1/18",
xlabel="q2",ylabel="p2",
point_type=7,point_size=0.1,
    points(append(data9,data10)),
xrange=[-0.35,0.4],yrange=[-0.4,0.4]);
```



(%t41)

(%o41)

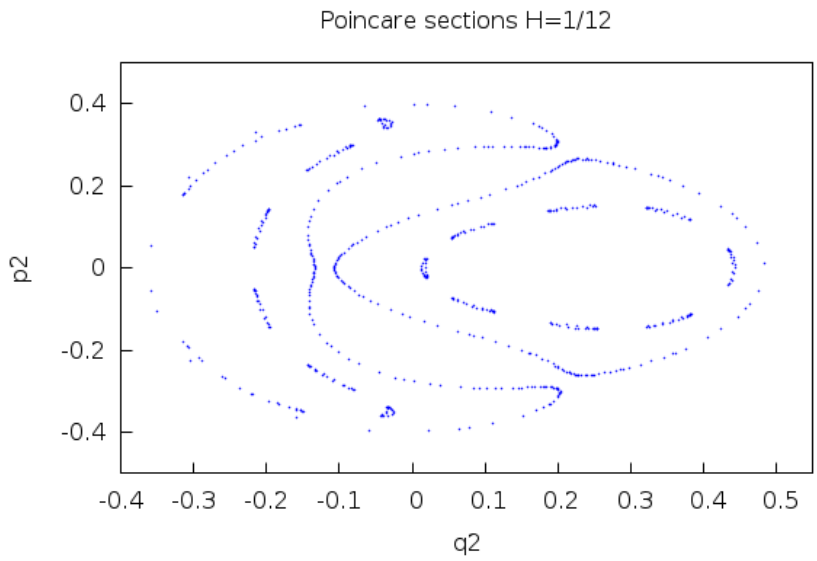
```
(%i42) series3:realroots(K(-0.2,p1,-0.2,0.1)=1/12)$
(%i43) data11:poincare2d(
       K,XK,[-0.2,rhs(first(series3)),-0.2,0.1],[t,-500,500,0.01],[q1,0,q2,p2]
       )$
(%i44) time(%);
```

[1.947] (%o44)

```
(%i45) data12:poincare2d(
       K,XK,[-0.2,rhs(second(series3)),-0.2,0.1],[t,-500,500,0.01],[q1,0,q2,p2]
       )$
(%i46) time(%);
```

[1.918] (%o46)

```
(%i47) wxdraw2d(title="Poincare sections H=1/12",
       xlabel="q2",ylabel="p2",
       point_type=7,point_size=0.1,
           points(append(data11,data12)),
       xrange=[-0.4,0.55],yrange=[-0.5,0.5]);
```



(%t47)

(%o47)

```
(%i48) series4:realroots(K(-0.2,p1,-0.2,0.1)=1/6)$
```

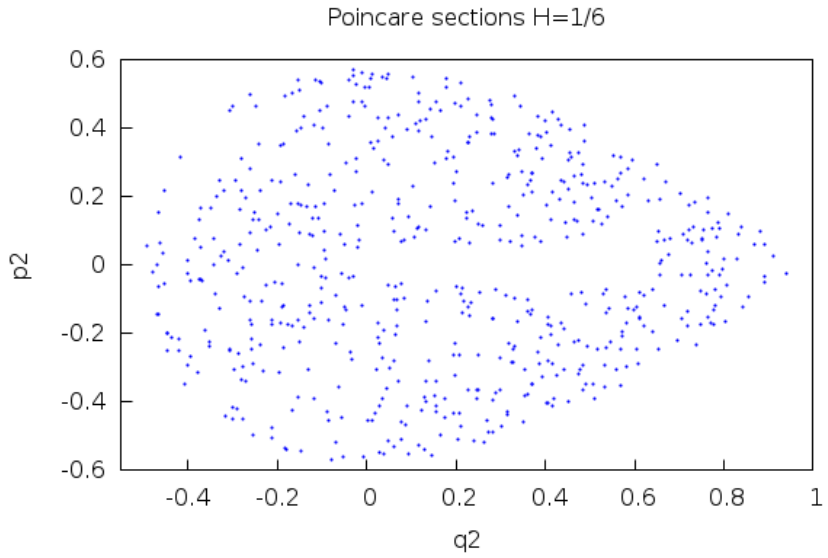
```
(%i49) data13:poincare2d(
       K,XK,[-0.2,rhs(first(series4)),-0.2,0.1],[t,-500,500,0.01],[q1,0,q2,p2]
       )$
(%i50) time(%);
```

[1.999] (%o50)

```
(%i51) data14:poincare2d(
       K,XK,[-0.2,rhs(second(series4)),-0.2,0.1],[t,-500,500,0.01],[q1,0,q2,p2]
       )$
(%i52) time(%);
```

[1.908] (%o52)

```
(%i53) wxdraw2d(title="Poincare sections H=1/6",
       xlabel="q2",ylabel="p2",
       point_type=7,point_size=0.2,
           points(append(data13,data14)),
       xrange=[-0.55,1],yrange=[-0.6,0.6]);
```



(%t53)

(%o53)

The graphics coincide with those of Cheb-Terrab and Oliveira. Now, armed with confidence, we can proceed to more sophisticate figures...

5 Example: The elastic pendulum

Here we reproduce some figures from Carretero et al.: *Regular and chaotic behaviour in an extensible pendulum.* Eur. J. Phys. **15** (1994) 139–148. First, we fix a parameter:

(%i54)
```
eps:3/4$
```

Now comes the Hamiltonian:

(%i55)
```
H(q1,p1,q2,p2):=(p1^2+p2^2)/2+(q1^2+q2^2)/2-eps*q1^2*(1+q2)/2$
```

We can obtain an analytic expression for p_2 once the energy E, and the initial values of (q_1, p_1, q_2) are known:

(%i56)
```
solve(H(q1,p1,q2,p2)=E,p2);
```

(%o56)

$$[p2 = -\frac{\sqrt{-4q2^2+3q1^2\,q2-q1^2-4p1^2+8E}}{2}, p2 = \frac{\sqrt{-4q2^2+3q1^2\,q2-q1^2-4p1^2+8E}}{2}]$$

(%i57)
```
subst(E=0.00875,%);
```

(%o57)

$$[p2 = -\frac{\sqrt{-4q2^2+3q1^2\,q2-q1^2-4p1^2+0.07}}{2}, p2 = \frac{\sqrt{-4q2^2+3q1^2\,q2-q1^2-4p1^2+0.07}}{2}]$$

(%i58)
```
define(f(q1,p1,q2),rhs(first(%)));
```

$$\mathrm{f}\,(q1, p1, q2) := -\frac{\sqrt{-4q2^2+3q1^2\,q2-q1^2-4p1^2+0.07}}{2} \qquad (\%o58)$$

(%i59)
```
define(g(q1,p1,q2),rhs(second(%th(2))));
```

$$\mathrm{g}\,(q1, p1, q2) := \frac{\sqrt{-4q2^2+3q1^2\,q2-q1^2-4p1^2+0.07}}{2} \qquad (\%o59)$$

Now, we compute the $q_2 = 0$ surface of section, for a big enough set of initial conditions (q_1, p_1, q_2):

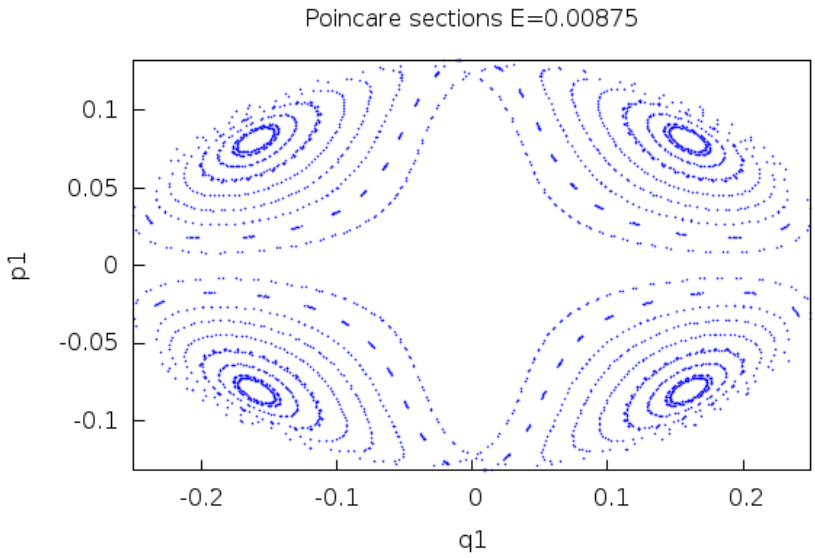
```
(%i60) for j:1 thru 10 do
       data1[j]:poincare2d(
       H,XH,[0.15,j/100,0.001,g(0.15,j/100,0.001)],[t,0,1000,0.01],[q2,0,q1,p1]
       )$

(%i61) time(%);
```

[17.222] (%o61)

```
(%i62) points1:xreduce(append,create_list(data1[j],j,makelist(k,k,1,10)))$

(%i63) wxdraw2d(title="Poincare sections E=0.00875",
       xlabel="q1",ylabel="p1",
       point_type=7,point_size=0.1,
           points(points1)
       );
```



(%t63)

(%o63)

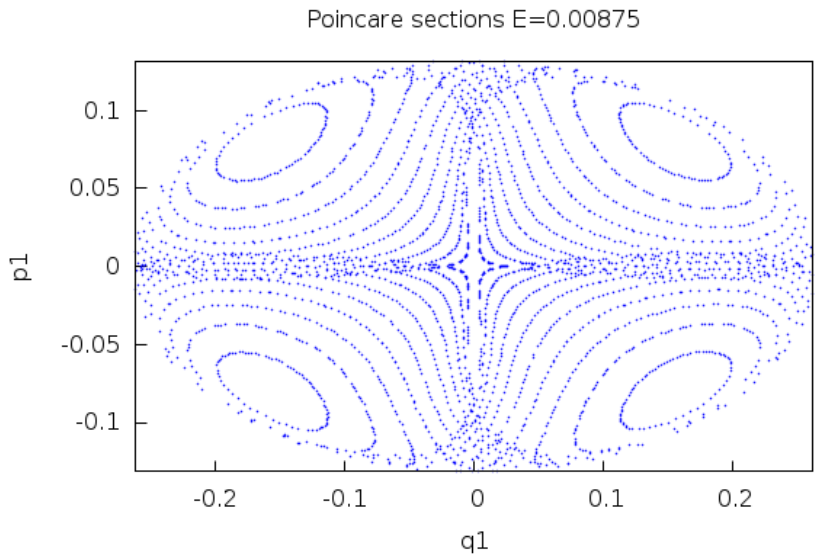
In order to fill the center of the figure, we must select another set of initial conditions, whose orbits pass through that region:

```
(%i64) for j:1 thru 10 do
       data2[j]:poincare2d(
       H,XH,[2*j/100,0,j/100+0.0025,f(2*j/100,0,j/100+0.0025)],
       [t,0,1000,0.01],[q2,0,q1,p1]
       )$
```

```
(%i65) time(%);
```

[17.369] (%o65)

```
(%i66) points2:xreduce(append,create_list(data2[j],j,makelist(k,k,1,10)))$
(%i67) wxdraw2d(title="Poincare sections E=0.00875",
       xlabel="q1",ylabel="p1",
       point_type=7,point_size=0.1,
           points(points2)
       );
```

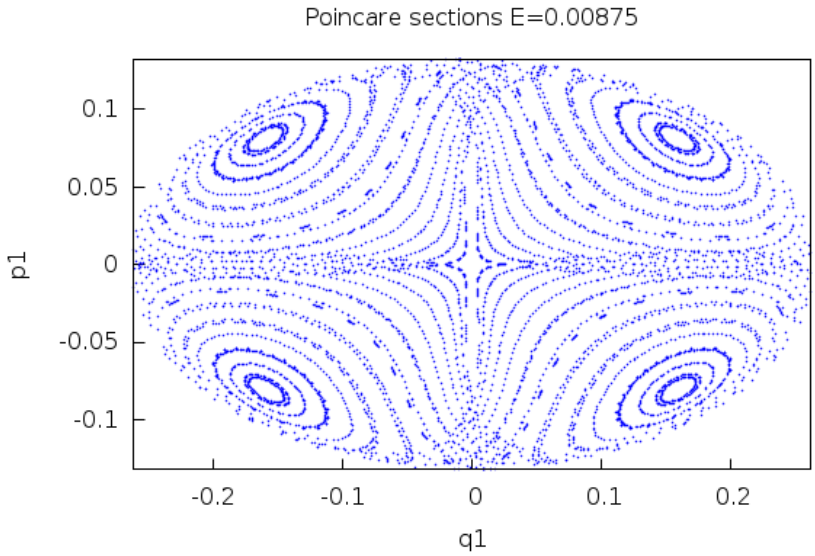


(%t67)

(%o67)

Joining both sets of orbits we get a nice figure...

```
(%i68) wxdraw2d(title="Poincare sections E=0.00875",
       xlabel="q1",ylabel="p1",
       point_type=7,point_size=0.1,
           points(append(points1,points2))
       );
```



(%t68)

(%o68)

The reader can have some fun by repeating these commands with higher energies, for instance, taking the values $E = 0.01875$, $E = 0.02875$, $E = 0.03875$, as it is done in the aforementioned paper by Carretero et al. The result is a diffusion process at the end of which the points are uniformly scattered, thus illustrating the fact that the dynamics has lost any regularity.

6 Conclusions

The Maxima package `poincare.mac` reproduces the results appearing in textbooks and research papers and obtained using the DEtools in Maple™. The graphical output quality is quite good, comparable (to say the least) to that of commercial software, but at no cost (for comparison, Maple™ in its student's version costs 1 000USD.) Regarding computation times, the Maxima version outperforms its Maple™ competitor: the heaviest computation in this paper is executed in `(%i64)`, taking 50 seconds in Maple™ (as can be seen in the worksheet

http://galia.fc.uaslp.mx/~jvallejo/ElasticPendulum-MapleSession.pdf), while Maxima only requires a third of this time.