$\longrightarrow$ fpprintprec:8$

1 Simple Harmonic Oscillator

Periodic phenomena abound in nature. Modeling them lead us to study differential equations that have oscillating solutions to them. The simplest of them is that of a simple harmonic oscillator provided by the set of equations,

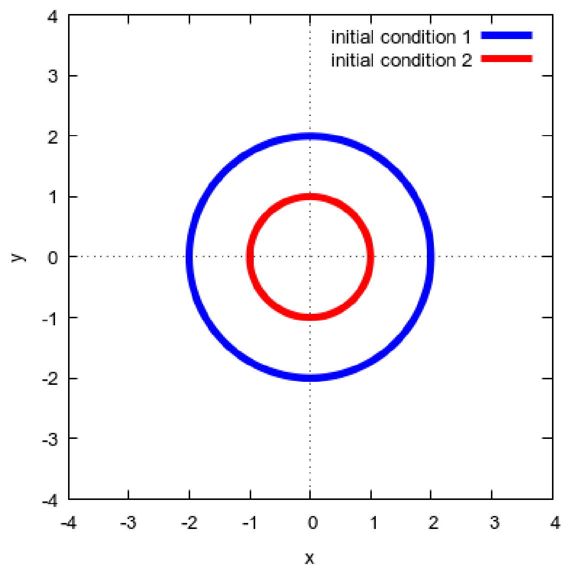
$$\begin{aligned} x' &= y \\ y' &= -x \end{aligned} \tag{1}$$

```
lin_ic2 : rk([y,-x],[x,y],[-1,0],[t,0,35,0.1])$
lin_ic1 : rk([y,-x],[x,y],[2,0],[t,0,35,0.1])$
lu_ic1 : makelist([lin_ic1[i][1],lin_ic1[i][2]],i,1,length(lin_ic1))$
lu_ic2 : makelist([lin_ic2[i][1],lin_ic2[i][2]],i,1,length(lin_ic2))$
uv_ic1 : makelist([lin_ic1[i][2],lin_ic1[i][3]],i,1,length(lin_ic1))$
uv_ic2 : makelist([lin_ic2[i][2],lin_ic2[i][3]],i,1,length(lin_ic2))$
```

The solutions form closed orbits in the phase space with solutions corresponding to different initial conditions forming concentric circles.

→ wxplot2d([[discrete,uv_ic1],[discrete,uv_ic2]],[x,-4,4],[y,-4,4],
[style,[lines,5]],[ylabel,"y"],[yx_ratio,1], [xlabel,"x"],[legend,"initial condition 1","initial condition 2"]);

(% t83)

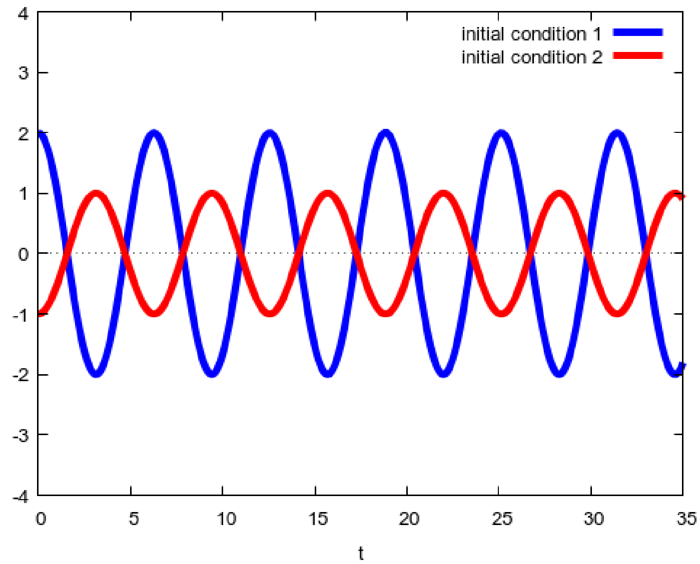


(% o83)

Even though the oscillations differ in their amplitude, the frequency of the oscillatory solutions starting at different initial conditions remain the same. Which is shown both in the graph of the solution as well as the frequency spectrum calculated from the time series.

→ wxplot2d([[discrete,lu_ic1],[discrete,lu_ic2]],[x,0,35],[y,-4,4], [style,[lines,5]],[ylabel," "], [xlabel,"t"],[legend,"initial condition 1","initial condition 2"]);

(% t84)



(% o84)

→ lin_ic1_shrunken : makelist(lin_ic1[j*10][2],j,1,32)$

→ lin_ic2_shrunken : makelist(lin_ic2[j*10][2],j,1,32)$

→

→ load(fft)$

→ load(qfft)$

→ (ns :∼32, fs : 1)$

→ dt : first(nyquist(ns,fs))$

sampling interval dt =

$$1.0$$

Nyquist integer knyq =

$$3$$

$$\frac{32}{2}$$

Nyquist freq fnyq =

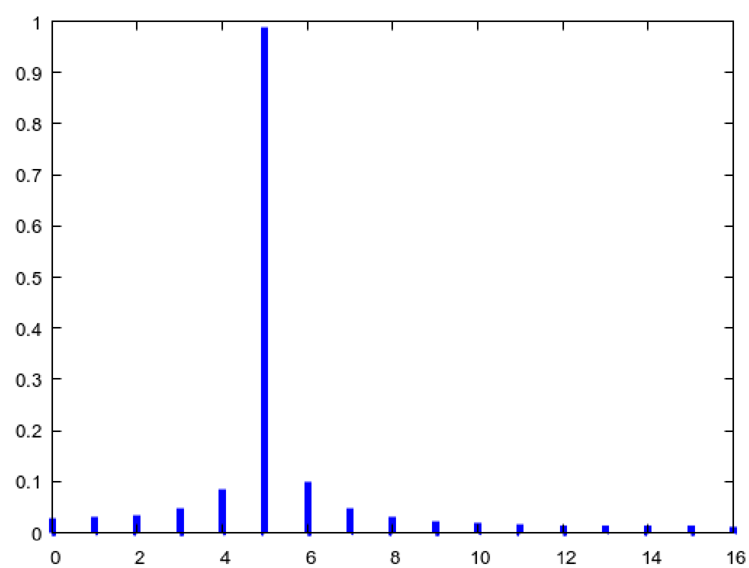
$$0.5$$

freq resolution df =

$$\frac{1}{32}$$

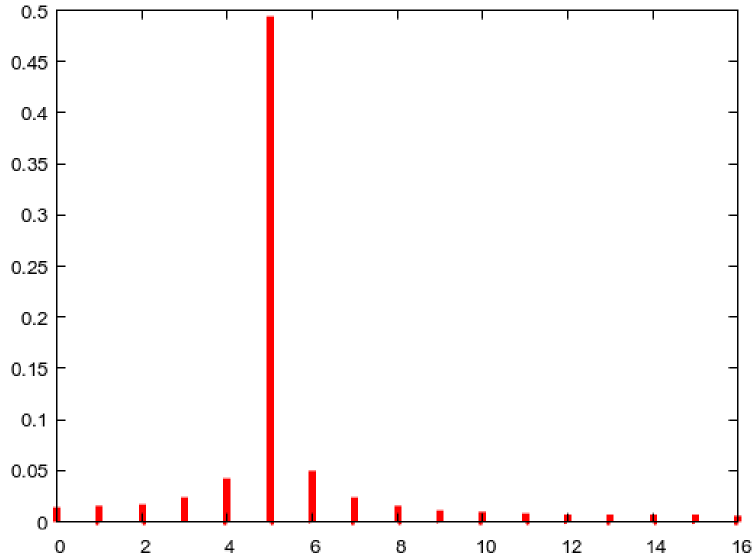
```
⟶ tflist_1 : vf(lin_ic1_shrunken,dt)$
⟶ tflist_2 : vf(lin_ic2_shrunken,dt)$
⟶ glist_1 : fft(lin_ic1_shrunken)$
⟶ glist_2 : fft(lin_ic2_shrunken)$
⟶ kglist_1 : kg(glist_1)$
⟶ kglist_2 : kg(glist_2)$
⟶ lvbars_1 : makelist([discrete,[[kglist_1[iter][1],0],[kglist_1[iter][1],kglist_1[iter][2]]]],iter,1,length(kglist_1))
⟶ lvbars_2 : makelist([discrete,[[kglist_2[iter][1],0],[kglist_2[iter][1],kglist_2[iter][2]]]],iter,1,length(kglist_2))
⟶ wxplot2d(lvbars_1,[x,0,16],[legend,false],[style,[lines,5,1]])$
```

(% t96)



→ wxplot2d(lvbars_2,[x,0,16],[legend,false],[style,[lines,5,1]],[color,red])$

(% t97)



2 Weakly Nonlinear oscillator

A simple addition of nonlinear terms (modeling nonlinear damping, for example) to the equations modify the solutions in interesting ways. Let us explore this in a weakly nonlinear Duffing oscillator. The set of equations are,

$$x' = y$$
$$y' = -x + \epsilon x^3 \tag{2}$$

→ pts_ic2 : rk([v,-u-0.5*u^3],[u,v],[-1,0],[t,0,32,0.1])$

→ pts_ic1 : rk([v,-u-0.5*u^3],[u,v],[2,0],[t,0,32,0.1])$

→ nlu_ic2 : makelist([pts_ic2[i][1],pts_ic2[i][2]],i,1,length(pts_ic2))$

→ nlu_ic1 : makelist([pts_ic1[i][1],pts_ic1[i][2]],i,1,length(pts_ic1))$

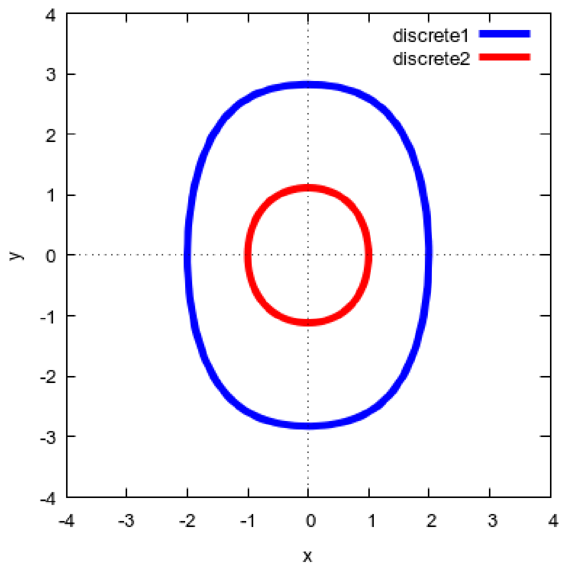
→ nluv_ic1 : makelist([pts_ic1[i][2],pts_ic1[i][3]],i,1,length(pts_ic1))$

→ nluv_ic2 : makelist([pts_ic2[i][2],pts_ic2[i][3]],i,1,length(pts_ic2))$

The solutions are no longer concentric circles.

$\longrightarrow$ wxplot2d([[discrete,nluv_ic1],[discrete,nluv_ic2]],[x,-4,4],[y,-4,4], [style,[lines,5]],[ylabel,"y"],[yx_ratio,1], [xlabel,"x"]);

(%t58)

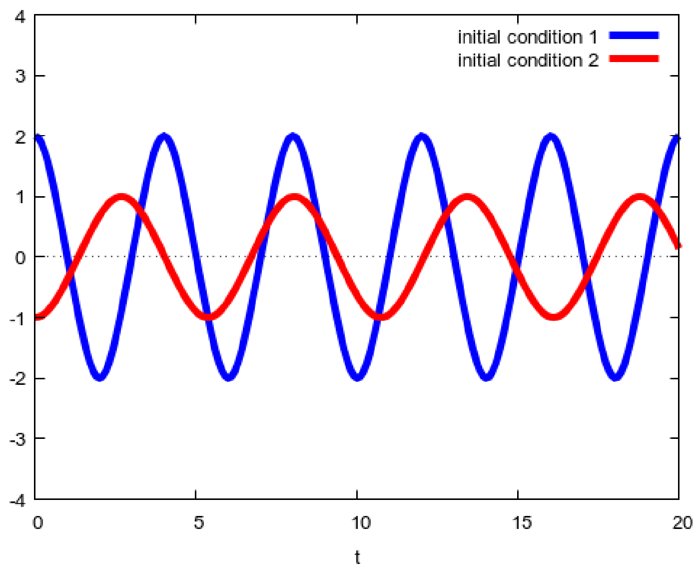


(%o58)

The oscillatory solutions starting from different initial conditions differ not only differ in their amplitude, but also in their frequencies which is shown in the subsequent graphs of solutions and the spectral plots. Here, The larger oscillations beat faster.

→ wxplot2d([[discrete,nlu_ic1],[discrete,nlu_ic2]],[x,0,20],[y,-4,4],
[style,[lines,5]],[ylabel," "], [xlabel,"t"],[legend,"initial condition 1","initial condition 2"])$

(% t59)



→ nlin_ic1_shrunken : makelist(pts_ic1[j*10][2],j,1,32)$

→ nlin_ic2_shrunken : makelist(pts_ic2[j*10][2],j,1,32)$

→ tflist_1 : vf(nlin_ic1_shrunken,dt)$

→ tflist_2 : vf(nlin_ic2_shrunken,dt)$

→ glist_1 : fft(nlin_ic1_shrunken)$

→ glist_2 : fft(nlin_ic2_shrunken)$

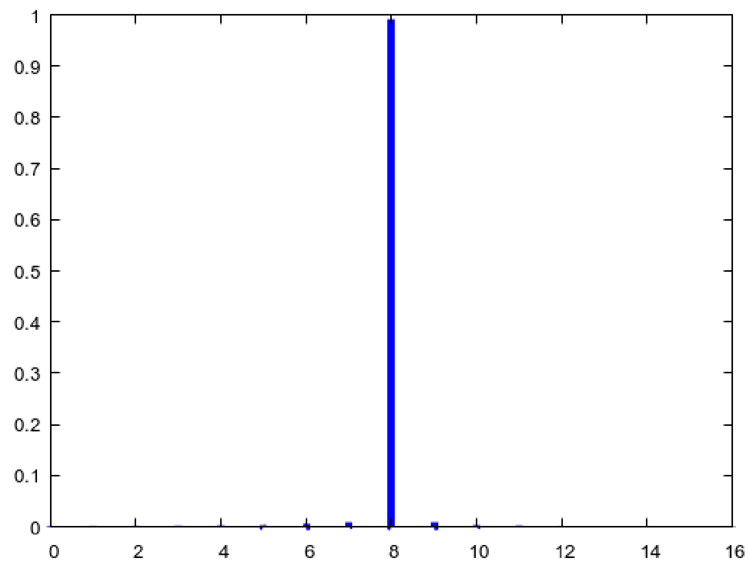
→ kglist_1 : kg(glist_1)$

→ kglist_2 : kg(glist_2)$

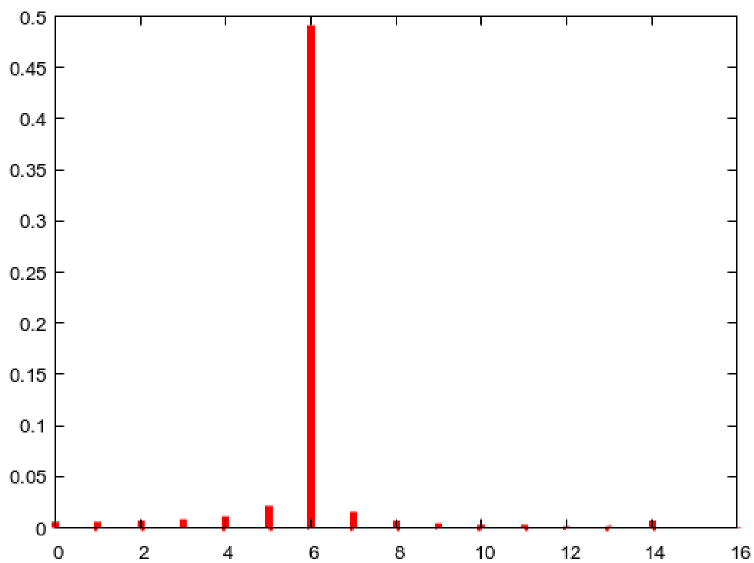
→ nlvbars_1 : makelist([discrete,[[kglist_1[iter][1],0],[kglist_1[iter][1],kglist_1[iter][2]]]],iter,1,length(kglist_1

→ nlvbars_2 : makelist([discrete,[[kglist_2[iter][1],0],[kglist_2[iter][1],kglist_2[iter][2]]]],iter,1,length(kglist_2

$\longrightarrow$ wxplot2d(nlvbars_1,[x,0,16],[legend,false],[style,[lines,5,1]])$
(%t108)



$\longrightarrow$ wxplot2d(nlvbars_2,[x,0,16],[legend,false],[style,[lines,5,1]],[color,red])$
(%t109)



3 Poincaré-Lindstedt Method

We can use Poincare-Lindsteadt method to find the analytic expression for the approximate solution and the frequency (in $T = \omega t$, wω is the frequency). Both the approximate solution and the approximate frequency are given as functions of the perturbation parameter ϵ the order of which can be manually set. As Maxima is a fully fleded programming language, we can code the implementation of Poincare-Lindstedt method ourselves if we wish to.

```
⟶ lindstedt_ic1 : Lindstedt('diff(x,t,2)+x+e*x^3,e,2,[-1,0]);
```

(lindstedt_ ic1)

$$[[[-\frac{(\cos(5T) - 24\cos(3T) + 23\cos(T))\ e^2}{1024} - \frac{(\cos(3T) - \cos(T))\, e}{32} - \cos(T)], T = \left(-\frac{21e^2}{256} + \frac{3e}{8} + 1\right) t]]$$

```
⟶ lindstedt_ic2 : Lindstedt('diff(x,t,2)+x+e*x^3,e,2,[2,0]);m
```

(lindstedt_ ic2)

$$[[[\frac{(\cos(5T) - 24\cos(3T) + 23\cos(T))\ e^2}{32} + \frac{(\cos(3T) - \cos(T))\, e}{4} + 2\cos(T)], T = \left(-\frac{21e^2}{16} + \frac{3e}{2} + 1\right) t]]$$

```
⟶ f : ev(lindstedt_ic1[1][1][1],lindstedt_ic1[1][2]);
```

(f)

$$-\frac{e^2\left(\cos\left(5\left(-\frac{21e^2}{256} + \frac{3e}{8} + 1\right)t\right) - 24\cos\left(3\left(-\frac{21e^2}{256} + \frac{3e}{8} + 1\right)t\right) + 23\cos\left(\left(-\frac{21e^2}{256} + \frac{3e}{8} + 1\right)t\right)\right)}{1024} - \frac{e\left(\cos\left(3\left(\right.\right.\right.}{}$$

```
⟶ f~: ev(ev(f,e=0.5));
```

(% o143)

$$-2.441406210^{-4}\ (\cos(5.8349609t) - 24\cos(3.5009765t) + 23\cos(1.1669921t)) - 0.015625\,(\cos(3.5009765t) - \text{co}$$

```
⟶ mylis : makelist([i/10.0],i,1,200)$
⟶ li_g : map(lambda([i],num(ev(ev(f,t=i),e=0.5))),mylis)$
⟶ length(li_g);
```

(% o156) 200

⟶