University of Zurich - HS 2013

Computational Science I
Exercise notes: Driven Pendulum - Cyclotrons - Black Holes

Tobias Grubenmann

November 19, 2013

Exercise 1

The following code solves the problem of the driven pendulum with the built-in ODE solver from Python:

```
from scipy.integrate import odeint

omega = -2
epsilon = 0.3

def func(y, t):
    return [-sin(y[1])-epsilon*sin(y[1]-omega*t), y[0]]

t = arange(0, 1000, 0.1*2*pi/-omega)

for n in range(100) :

    pini = 0.01*n

    y0 = [pini, 0]

    y = odeint(func, y0, t)

    for i in range(100):
        plot(y[i*10][0], y[i*10][1], 'r,')
```

The first picture shows the surface of section for intial p between 0 and 1 with $\omega = -2$ and $\varepsilon = 0.3$. The simulation ran until $\omega t = 40\pi$.

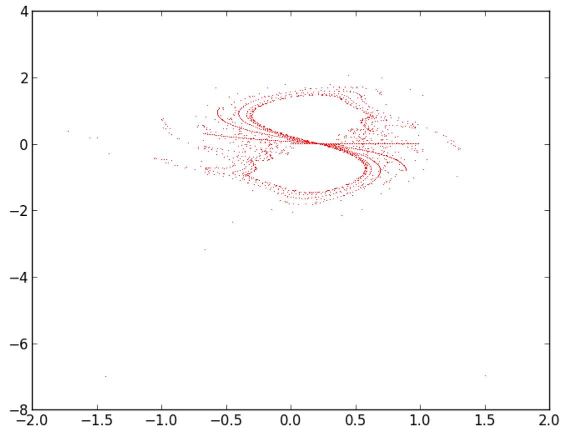

Figure 1: Driven pendulum for different intitial p between 0 and 1, until $\omega t = 40\pi$ with the Python solver

The second picture shows the same as the first but the simulation ran until $\omega t = 200\pi$.

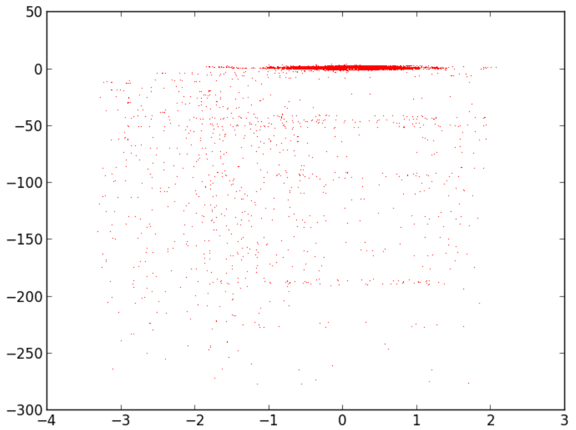

Figure 2: Driven pendulum for different intitial p between 0 and 1, until $\omega t = 200\pi$ with the Python solver

The following code solves the problem of the driven pendulum with the Leapfrog method:

```
omega = -2
```

```
epsilon = 0.3

dt = 0.1*2*pi/-omega

for n in range(100) :

    pini = 0.01*n

    p = [pini, 0]

    q = [0, 0]

    s = [0, 0]

    for i in range(200):

        q = [q[0] + p[0]*dt/2, q[1] + omega*dt/2]
        s = [sin(q[0]), epsilon*sin(q[0]-q[1])]
        p = [p[0] - (s[0]+s[1])*dt, p[1]+s[1]*dt]
        q = [q[0] + p[0]*dt/2, q[1] + omega*dt/2]

        if i%10==0:
            plot(p[0], q[0], 'r,')
```

The first picture shows the surface of section for intial p between 0 and 1 with $\omega = -2$ and $\varepsilon = 0.3$. The simulation ran until $\omega t = 40\pi$.

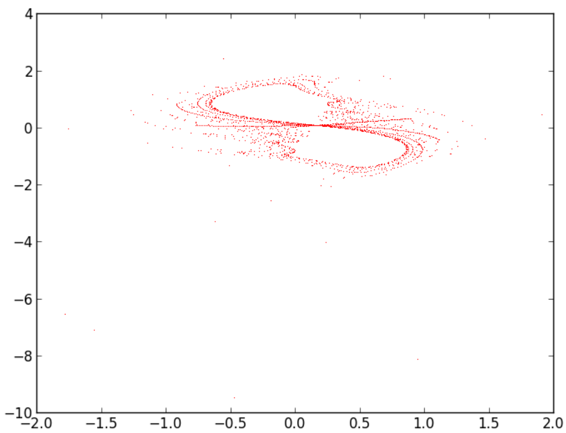

Figure 3: Driven pendulum for different intitial p between 0 and 1, until $\omega t = 40\pi$ with the Leapfrog solver

The second picture shows the same as the first but the simulation ran until $\omega t = 200\pi$.

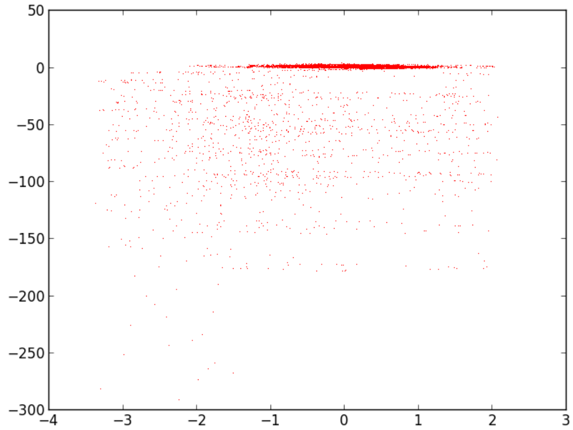

Figure 4: Driven pendulum for different intitial p between 0 and 1, until $\omega t = 200\pi$ with the Leapfrog solver

Exercise 2

To solve the cyclotron problem we need first to get the differential equations from the Hamiltonian. For the relativistic case this gives the following system:

$$
\begin{aligned}
\dot{x} &= \frac{p_x + y}{\sqrt{1 + (p_x + y)^2 + (p_y - x)^2}} \\
\dot{y} &= \frac{p_y - x}{\sqrt{1 + (p_x + y)^2 + (p_y - x)^2}} \\
\dot{p_x} &= \frac{p_y - x}{\sqrt{1 + (p_x + y)^2 + (p_y - x)^2}} \\
\dot{p_x} &= -\frac{p_x + y}{\sqrt{1 + (p_x + y)^2 + (p_y - x)^2}} + \alpha \cos(\omega t)
\end{aligned}
$$

For the non-relativistic case we have the following system:

$$
\begin{aligned}
\dot{x} &= p_x + y \\
\dot{y} &= p_y - x \\
\dot{p_x} &= p_y - x \\
\dot{p_x} &= -p_x - y + \cos(\omega t)
\end{aligned}
$$

The following code solves for the relativistic case:

```
from scipy.integrate import odeint

omega = 1
alpha = 1

def func(y, t):
    a = sqrt(1+(y[2]+y[1])**2+(y[3]-y[0])**2)
    return [(y[2]+y[1])/a, (y[3]-y[0])/a, (y[3]-y[0])/a, -(
        y[2]+y[1])/a+alpha*cos(omega*t)]

t = arange(0, 100, 0.1)

y0 = [0, 0, 0, 0]

y = odeint(func, y0, t)

plot(t, y)
```

The following pictures shows the plots of x, y, p_x, p_y for the relativistic case for different ω and α:

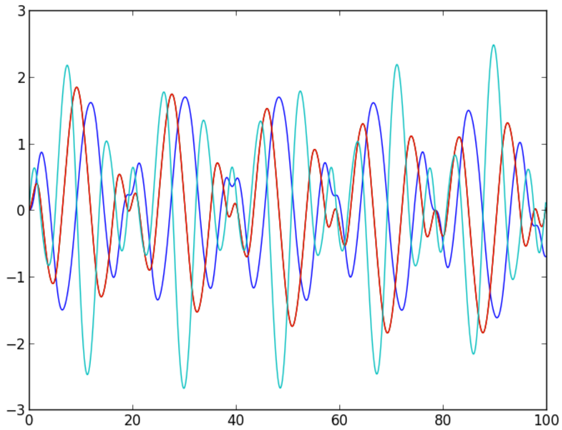

Figure 5: x, y, p_x, p_y for a cyclotron with $\omega = 1$, $\alpha = 1$

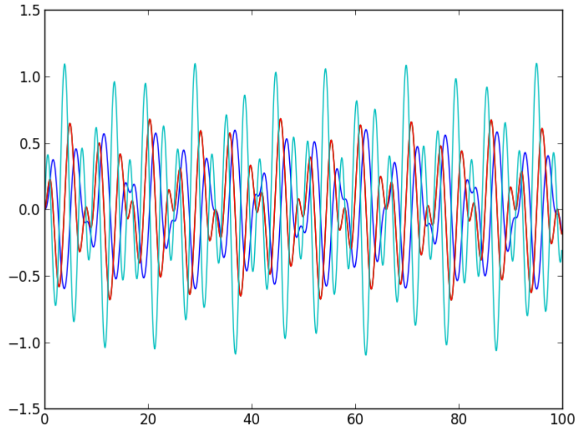

Figure 6: x, y, p_x, p_y for a cyclotron with $\omega = 2$, $\alpha = 1$

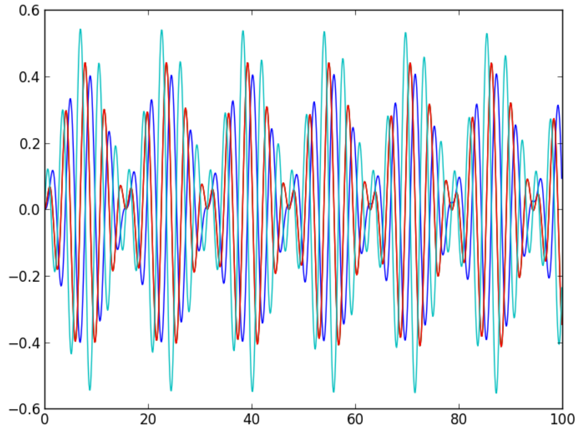

Figure 7: x, y, p_x, p_y for a cyclotron with $\omega = 2$, $\alpha = 0.3$

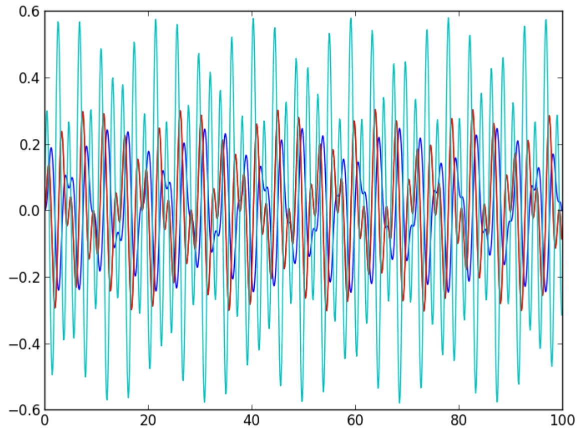

Figure 8: x, y, p_x, p_y for a cyclotron with $\omega = 3$, $\alpha = 1$

The following code solves for the non-relativistic case:

```
from scipy.integrate import odeint

omega = 1

def func(y, t):
    return [y[2]+y[1], y[3]-y[0], y[3]-y[0], -y[2]-y[1]+cos
       (omega*t)]

t = arange(0, 100, 0.1)

y0 = [0, 0, 0, 0]

y = odeint(func, y0, t)

plot(t, y)
```

The following pictures shows the plots of x, y, p_x, p_y for the non-relativistic case for different ω:

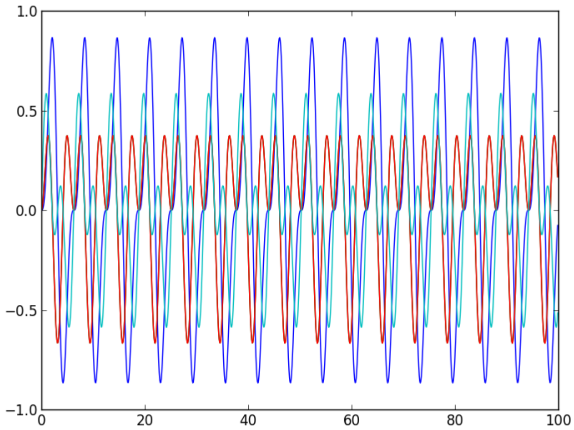

Figure 9: x, y, p_x, p_y for a cyclotron with $\omega = 1$

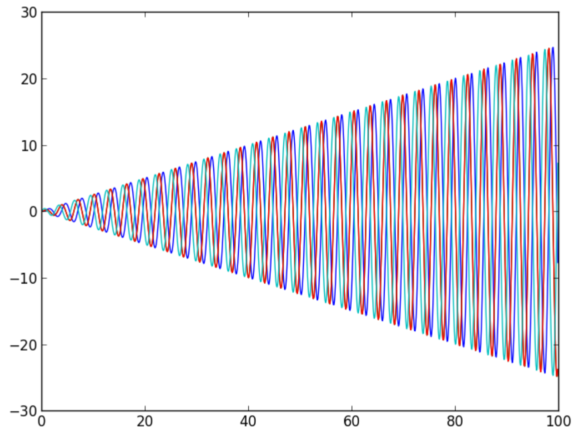

Figure 10: x, y, p_x, p_y for a cyclotron with $\omega = 2$

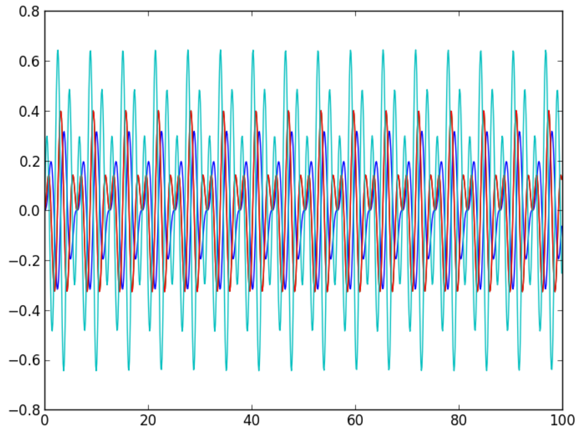

Figure 11: x, y, p_x, p_y for a cyclotron with $\omega = 3$

Exercise 3

To solve this problem we need first to get the differential equations from the Hamiltonian:

$$
\begin{aligned}
\dot{x} &= p_x - \frac{2(xp_x + yp_y)x}{r^3} \\
\dot{y} &= p_y - \frac{2(xp_x + yp_y)y}{r^3} \\
\dot{p_x} &= -\frac{1}{2}\left(1 - \frac{2}{r}\right)^{-2} \frac{2}{r^2} \cdot \frac{x}{r} + \frac{2(xp_x + yp_y)p_x \cdot r^3 - (xp_x + yp_y)^2 3r^2 \cdot \frac{x}{r}}{r^6} \\
\dot{p_x} &= -\frac{1}{2}\left(1 - \frac{2}{r}\right)^{-2} \frac{2}{r^2} \cdot \frac{y}{r} + \frac{2(xp_x + yp_y)p_y \cdot r^3 - (xp_x + yp_y)^2 3r^2 \cdot \frac{y}{r}}{r^6}
\end{aligned}
$$

The following code solves the black hole problem:

```
from scipy.integrate import odeint

def func(y, t):
    r = sqrt(y[0]**2+y[1]**2)
    return [
        y[2]-2*(y[0]*y[2]+y[1]*y[3])*y[0]/(r**3),
        y[3]-2*(y[0]*y[2]+y[1]*y[3])*y[1]/(r**3),
        -0.5*((1-2/r)**-2)*2/(r**2)*y[0]/r + (2*(y[0]*y[2]+
           y[1]*y[3])*y[2]*r**3-((y[0]*y[2]+y[1]*y[3])**2)
           *3*(r**2)*y[0]/r)/(r**6),
```

```
        -0.5*((1-2/r)**-2)*2/(r**2)*y[1]/r + (2*(y[0]*y[2]+
           y[1]*y[3])*y[3]*r**3-((y[0]*y[2]+y[1]*y[3])**2)
           *3*(r**2)*y[1]/r)/(r**6)
        ]

t = arange(0, 50, 0.01)

y0 = [8, 0, 0, 0.2]

y = odeint(func, y0, t)

for i in range(5000):
    plot(y[i][0], y[i][1], 'r,')
```

The following figures show the orbit for an initial velocity of $(0, 0.2)$ and different initial positions:

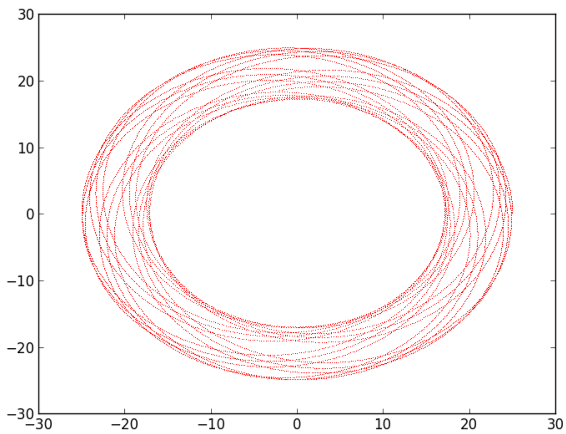

Figure 12: Orbit near a black hole with initial position $(25, 0)$

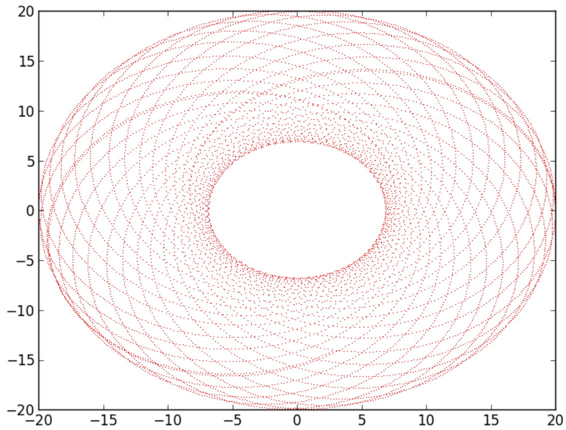

Figure 13: Orbit near a black hole with initial position $(20, 0)$

Figure 14: Orbit near a black hole with initial position $(8, 0)$