PHY321: Two-body problems and Gravitational Forces

Morten Hjorth-Jensen[1,2]

[1]Department of Physics and Astronomy and Facility for Rare Ion Beams (FRIB), Michigan State University, USA
[2]Department of Physics, University of Oslo, Norway

Mar 27, 2022

Aims and Overarching Motivation

Monday.

1. Computational topics: functions and classes and how to use these in our course.
2. Discussion of the two-dimensional harmonic oscillator as a warm-up to the gravitational force problem and Kepler's laws. This is relevant for homework 8 and exercises 2 and 3.

Reading suggestion: Taylor section 8.4 and Lecture notes

Wednesday.

1. Elliptical orbits and Kepler's laws

Reading suggestion: Taylor sections 8.5-8.8

Friday.

1. Physical interpretation of various orbit types and discussion of homework 8. We will use parts of Friday for a regular lecture and thereafter work on homework 8.

Reading suggestion: Taylor section 8.5-8.8

Harmonic Oscillator in two dimensions, homework 8, exercises 2 and 3

The material here is relevant for the discussion of exercises 2 and 3 for homework 8. It serves also as a warm-up for the more interesting gravitational force problem.

Consider a particle of mass m in a 2-dimensional harmonic oscillator with potential

$$V = \frac{1}{2}kr^2 = \frac{1}{2}k(x^2+y^2).$$

If the orbit has angular momentum L, we can find the radius and angular velocity of the circular orbit as well as the b) the angular frequency of small radial perturbations.

We consider the effective potential. The radius of a circular orbit is at the minimum of the potential (where the effective force is zero). The potential is plotted here with the parameters $k = m = 0.1$ and $L = 1.0$.

```python
# Common imports
import numpy as np
from math import *
import matplotlib.pyplot as plt

Deltax = 0.01
#set up arrays
xinitial = 0.5
xfinal = 3.0
k = 1.0   # spring constant
m = 1.0   # mass, you can change these
AngMom = 1.0  #  The angular momentum
n = ceil((xfinal-xinitial)/Deltax)
x = np.zeros(n)
for i in range(n):
    x[i] = xinitial+i*Deltax
V = np.zeros(n)
V = 0.5*k*x*x+0.5*AngMom*AngMom/(m*x*x)
# Plot potential
fig, ax = plt.subplots()
ax.set_xlabel('r[m]')
ax.set_ylabel('V[J]')
ax.plot(x, V)
fig.tight_layout()
plt.show()
```

$$V_{\rm eff} = \frac{1}{2}kr^2 + \frac{L^2}{2mr^2}$$

Harmonic oscillator in two dimensions and effective potential

The effective potential looks like that of a harmonic oscillator for large r, but for small r, the centrifugal potential repels the particle from the origin. The combination of the two potentials has a minimum for at some radius $r_{\rm min}$.

$$\begin{aligned} 0 &= kr_{\rm min} - \frac{L^2}{mr_{\rm min}^3}, \\ r_{\rm min} &= \left(\frac{L^2}{mk}\right)^{1/4}, \\ \dot{\theta} &= \frac{L}{mr_{\rm min}^2} = \sqrt{k/m}. \end{aligned}$$

For particles at $r_{\rm min}$ with $\dot{r} = 0$, the particle does not accelerate and r stays constant, i.e. a circular orbit. The radius of the circular orbit can be adjusted by changing the angular momentum L.

For the above parameters this minimum is at $r_{\rm min} = 1$.

Now consider small vibrations about $r_{\rm min}$. The effective spring constant is the curvature of the effective potential.

$$\begin{aligned} k_{\rm eff} &= \left.\frac{d^2}{dr^2} V_{\rm eff}(r)\right|_{r=r_{\rm min}} = k + \frac{3L^2}{mr_{\rm min}^4} \\ &= 4k, \\ \omega &= \sqrt{k_{\rm eff}/m} = 2\sqrt{k/m} = 2\dot{\theta}. \end{aligned}$$

Because the radius oscillates with twice the angular frequency, the orbit has two places where r reaches a minimum in one cycle. This differs from the inverse-square force where there is one minimum in an orbit. One can show that the orbit for the harmonic oscillator is also elliptical, but in this case the center of the potential is at the center of the ellipse, not at one of the foci.

The solution is also simple to write down exactly in Cartesian coordinates. The x and y equations of motion separate,

$$\begin{aligned} \ddot{x} &= -kx, \\ \ddot{y} &= -ky. \end{aligned}$$

The general solution can be expressed as

$$\begin{aligned} x &= A\cos\omega_0 t + B\sin\omega_0 t, \\ y &= C\cos\omega_0 t + D\sin\omega_0 t. \end{aligned}$$

The code here finds the solution for x and y using the code we developed in homework 5 and 6 and the midterm. Note that this code is tailored to run in Cartesian coordinates. There is thus no angular momentum dependent term.

Here we have chose initial conditions that correspond to the minimum of the effective potential $r_{\rm min}$. We have chosen $x_0 = r_{\rm min}$ and $y_0 = 0$. Similarly, we use the centripetal acceleration to determine the initial velocity so that we have a circular motion (see back to the last question of the midterm). This means that we set the centripetal acceleration v^2/r equal to the force from the harmonic oscillator $-k\boldsymbol{r}$. Taking the magnitude of $\boldsymbol{r}$ we have then $v^2/r = k/mr$, which gives $v = \pm\omega_0 r$.

Since the code here solves the equations of motion in cartesian coordinates and the harmonic oscillator potential leads to forces in the x- and y-directions that are decoupled, we have to select the initial velocities and positions so that we don't get that for example $y(t) = 0$.

We set x_0 to be different from zero and v_{y0} to be different from zero.

```
DeltaT = 0.001
#set up arrays
tfinal = 10.0
n = ceil(tfinal/DeltaT)
# set up arrays
t = np.zeros(n)
v = np.zeros((n,2))
r = np.zeros((n,2))
radius = np.zeros(n)
# Constants of the model
k = 1.0   # spring constant
m = 1.0   # mass, you can change these
omega02 = k/m  # Frequency
AngMom = 1.0  #  The angular momentum
# Potential minimum
rmin = (AngMom*AngMom/k/m)**0.25
# Initial conditions as compact 2-dimensional arrays, x0=rmin and y0 = 0
x0 = rmin; y0= 0.0
r0 = np.array([x0,y0])
vy0 = sqrt(omega02)*rmin; vx0 = 0.0
v0 = np.array([vx0,vy0])
r[0] = r0
v[0] = v0
# Start integrating using the Velocity-Verlet  method
for i in range(n-1):
    # Set up the acceleration
    a =  -r[i]*omega02
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    anew = -r[i+1]*omega02
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    t[i+1] = t[i] + DeltaT
# Plot position as function of time
radius = np.sqrt(r[:,0]**2+r[:,1]**2)
fig, ax = plt.subplots(3,1)
ax[0].set_xlabel('time')
ax[0].set_ylabel('radius squared')
ax[0].plot(t,r[:,0]**2+r[:,1]**2)
ax[1].set_xlabel('time')
ax[1].set_ylabel('x position')
ax[1].plot(t,r[:,0])
ax[2].set_xlabel('time')
ax[2].set_ylabel('y position')
ax[2].plot(t,r[:,1])

fig.tight_layout()
save_fig("2DimHOVV")
plt.show()
```

We see that the radius (to within a given error), we obtain a constant radius.

The following code shows first how we can solve this problem using the radial degrees of freedom only. Here we need to add the explicit centrifugal barrier. Note that the variable r depends only on time. There is no x and y directions since we have transformed the equations to polar coordinates.

```
DeltaT = 0.01
#set up arrays
tfinal = 10.0
n = ceil(tfinal/DeltaT)
# set up arrays for t, v and r
t = np.zeros(n)
v = np.zeros(n)
r = np.zeros(n)
E = np.zeros(n)
# Constants of the model
AngMom = 1.0  #  The angular momentum
m = 1.0
k = 1.0
omega02 = k/m
c1 = AngMom*AngMom/(m*m)
c2 = AngMom*AngMom/m
rmin = (AngMom*AngMom/k/m)**0.25
# Initial conditions
r0 = rmin
v0 = 0.0
r[0] = r0
v[0] = v0
E[0] = 0.5*m*v0*v0+0.5*k*r0*r0+0.5*c2/(r0*r0)
# Start integrating using the Velocity-Verlet  method
for i in range(n-1):
    # Set up acceleration
    a = -r[i]*omega02+c1/(r[i]**3)
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    anew = -r[i+1]*omega02+c1/(r[i+1]**3)
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    t[i+1] = t[i] + DeltaT
    E[i+1] = 0.5*m*v[i+1]*v[i+1]+0.5*k*r[i+1]*r[i+1]+0.5*c2/(r[i+1]*r[i+1])
    # Plot position as function of time
fig, ax = plt.subplots(2,1)
ax[0].set_xlabel('time')
ax[0].set_ylabel('radius')
ax[0].plot(t,r)
ax[1].set_xlabel('time')
ax[1].set_ylabel('Energy')
ax[1].plot(t,E)
save_fig("RadialHOVV")
plt.show()
```

With some work using double angle formulas, one can calculate

$$\begin{aligned}
r^2 &= x^2 + y^2 \\
&= (A^2 + C^2)\cos^2(\omega_0 t) + (B^2 + D^2)\sin^2\omega_0 t + (AB + CD)\cos(\omega_0 t)\sin(\omega_0 t) \\
&= \alpha + \beta\cos 2\omega_0 t + \gamma\sin 2\omega_0 t, \\
\alpha &= \frac{A^2 + B^2 + C^2 + D^2}{2}, \quad \beta = \frac{A^2 - B^2 + C^2 - D^2}{2}, \quad \gamma = AB + CD, \\
r^2 &= \alpha + (\beta^2 + \gamma^2)^{1/2}\cos(2\omega_0 t - \delta), \quad \delta = \arctan(\gamma/\beta),
\end{aligned}$$

and see that radius oscillates with frequency $2\omega_0$. The factor of two comes because the oscillation $x = A\cos\omega_0 t$ has two maxima for x^2, one at $t = 0$ and one a half period later.

Ellipse reminder

Let us remind ourselves about what an ellipse is before we proceed.

```
import numpy as np
from matplotlib import pyplot as plt
from math import pi

u=1.      #x-position of the center
v=0.5     #y-position of the center
a=2.      #radius on the x-axis
b=1.5     #radius on the y-axis

t = np.linspace(0, 2*pi, 100)
plt.plot( u+a*np.cos(t) , v+b*np.sin(t) )
plt.grid(color='lightgray',linestyle='--')
plt.show()
```

Deriving Elliptical Orbits

Kepler's laws state that a gravitational orbit should be an ellipse with the source of the gravitational field at one focus. Deriving this is surprisingly messy. To do this, we first use angular momentum conservation to transform the equations of motion so that it is in terms of r and θ instead of r and t. The overall strategy is to

1. Find equations of motion for r and t with no angle (θ) mentioned, i.e. $d^2r/dt^2 = \cdots$. Angular momentum conservation will be used, and the equation will involve the angular momentum L.

2. Use angular momentum conservation to find an expression for $\dot{\theta}$ in terms of r.

3. Use the chain rule to convert the equations of motions for r, an expression involving $r, \dot{r}$ and $\ddot{r}$, to one involving $r, dr/d\theta$ and $d^2r/d\theta^2$. This is quitecomplicated because the expressions will also involve a substitution $u = 1/r$ so that one finds an expression in terms of u and θ.

4. Once $u(\theta)$ is found, you need to show that this can be converted to the familiar form for an ellipse.

The equations of motion give

$$\begin{aligned}
\frac{d}{dt}r^2 &= \frac{d}{dt}(x^2+y^2) = 2x\dot{x} + 2y\dot{y} = 2r\dot{r}, \\
\dot{r} &= \frac{x}{r}\dot{x} + \frac{y}{r}\dot{y}, \\
\ddot{r} &= \frac{x}{r}\ddot{x} + \frac{y}{r}\ddot{y} + \frac{\dot{x}^2+\dot{y}^2}{r} - \frac{\dot{r}^2}{r}.
\end{aligned} \tag{1}$$

Recognizing that the numerator of the third term is the velocity squared, and that it can be written in polar coordinates,

$$v^2 = \dot{x}^2 + \dot{y}^2 = \dot{r}^2 + r^2\dot{\theta}^2, \tag{2}$$

one can write $\ddot{r}$ as

$$\begin{aligned}
\ddot{r} &= \frac{F_x\cos\theta + F_y\sin\theta}{m} + \frac{\dot{r}^2 + r^2\dot{\theta}^2}{r} - \frac{\dot{r}^2}{r} \\
&= \frac{F}{m} + \frac{r^2\dot{\theta}^2}{r} \\
m\ddot{r} &= F + \frac{L^2}{mr^3}.
\end{aligned} \tag{3}$$

This derivation used the fact that the force was radial, $F = F_r = F_x\cos\theta + F_y\sin\theta$, and that angular momentum is $L = mrv_\theta = mr^2\dot{\theta}$. The term $L^2/mr^3 = mv^2/r$ behaves like an additional force. Sometimes this is referred to as a centrifugal force, but it is not a force. Instead, it is the consequence of considering the motion in a rotating (and therefore accelerating) frame.

Now, we switch to the particular case of an attractive inverse square force, $F = -\alpha/r^2$, and show that the trajectory, $r(\theta)$, is an ellipse. To do this we transform derivatives w.r.t. time to derivatives w.r.t. θ using the chain rule combined with angular momentum conservation, $\dot{\theta} = L/mr^2$.

$$\begin{aligned}
\dot{r} &= \frac{dr}{d\theta}\dot{\theta} = \frac{dr}{d\theta}\frac{L}{mr^2}, \\
\ddot{r} &= \frac{d^2r}{d\theta^2}\dot{\theta}^2 + \frac{dr}{d\theta}\left(\frac{d}{dr}\frac{L}{mr^2}\right)\dot{r} \\
&= \frac{d^2r}{d\theta^2}\left(\frac{L}{mr^2}\right)^2 - 2\frac{dr}{d\theta}\frac{L}{mr^3}\dot{r} \\
&= \frac{d^2r}{d\theta^2}\left(\frac{L}{mr^2}\right)^2 - \frac{2}{r}\left(\frac{dr}{d\theta}\right)^2\left(\frac{L}{mr^2}\right)^2
\end{aligned} \tag{4}$$

Equating the two expressions for $\ddot{r}$ in Eq.s (3) and (4) eliminates all the derivatives w.r.t. time, and provides a differential equation with only derivatives w.r.t. θ,

$$\frac{d^2r}{d\theta^2}\left(\frac{L}{mr^2}\right)^2 - \frac{2}{r}\left(\frac{dr}{d\theta}\right)^2\left(\frac{L}{mr^2}\right)^2 = \frac{F}{m} + \frac{L^2}{m^2r^3}, \tag{5}$$

that when solved yields the trajectory, i.e. $r(\theta)$. Up to this point the expressions work for any radial force, not just forces that fall as $1/r^2$.

The trick to simplifying this differential equation for the inverse square problems is to make a substitution, $u \equiv 1/r$, and rewrite the differential equation for $u(\theta)$.

$$\begin{aligned} r &= 1/u, \\ \frac{dr}{d\theta} &= -\frac{1}{u^2}\frac{du}{d\theta}, \\ \frac{d^2r}{d\theta^2} &= \frac{2}{u^3}\left(\frac{du}{d\theta}\right)^2 - \frac{1}{u^2}\frac{d^2u}{d\theta^2}. \end{aligned} \tag{6}$$

Plugging these expressions into Eq. (5) gives an expression in terms of u, $du/d\theta$, and $d^2u/d\theta^2$. After some tedious algebra,

$$\frac{d^2u}{d\theta^2} = -u - \frac{Fm}{L^2u^2}. \tag{7}$$

For the attractive inverse square law force, $F = -\alpha u^2$,

$$\frac{d^2u}{d\theta^2} = -u + \frac{m\alpha}{L^2}. \tag{8}$$

The solution has two arbitrary constants, A and θ_0,

$$\begin{aligned} u &= \frac{m\alpha}{L^2} + A\cos(\theta - \theta_0), \\ r &= \frac{1}{(m\alpha/L^2) + A\cos(\theta - \theta_0)}. \end{aligned} \tag{9}$$

The radius will be at a minimum when $\theta = \theta_0$ and at a maximum when $\theta = \theta_0 + \pi$. The constant A is related to the eccentricity of the orbit. When $A = 0$ the radius is a constant $r = L^2/(m\alpha)$, and the motion is circular. If one solved the expression $mv^2/r = -\alpha/r^2$ for a circular orbit, using the substitution $v = L/(mr)$, one would reproduce the expression $r = L^2/(m\alpha)$.

The form describing the elliptical trajectory in Eq. (9) can be identified as an ellipse with one focus being the center of the ellipse by considering the definition of an ellipse as being the points such that the sum of the two distances between the two foci are a constant. Making that distance $2D$, the distance between the two foci as $2a$, and putting one focus at the origin,

$$
\begin{aligned}
2D &= r + \sqrt{(r\cos\theta - 2a)^2 + r^2\sin^2\theta}, \qquad (10)\\
4D^2 + r^2 - 4Dr &= r^2 + 4a^2 - 4ar\cos\theta,\\
r &= \frac{D^2 - a^2}{D + a\cos\theta} = \frac{1}{D/(D^2-a^2) - a\cos\theta/(D^2-a^2)}.
\end{aligned}
$$

By inspection, this is the same form as Eq. (9) with $D/(D^2 - a^2) = m\alpha/L^2$ and $a/(D^2 - a^2) = A$.

Let us remind ourselves about what an ellipse is before we proceed.

```
import numpy as np
from matplotlib import pyplot as plt
from math import pi

u=1.     #x-position of the center
v=0.5    #y-position of the center
a=2.     #radius on the x-axis
b=1.5    #radius on the y-axis

t = np.linspace(0, 2*pi, 100)
plt.plot( u+a*np.cos(t) , v+b*np.sin(t) )
plt.grid(color='lightgray',linestyle='--')
plt.show()
```

Effective or Centrifugal Potential

The total energy of a particle is

$$
\begin{aligned}
E &= V(r) + \frac{1}{2}mv_\theta^2 + \frac{1}{2}m\dot{r}^2 \qquad (11)\\
&= V(r) + \frac{1}{2}mr^2\dot{\theta}^2 + \frac{1}{2}m\dot{r}^2\\
&= V(r) + \frac{L^2}{2mr^2} + \frac{1}{2}m\dot{r}^2.
\end{aligned}
$$

The second term then contributes to the energy like an additional repulsive potential. The term is sometimes referred to as the "centrifugal" potential, even though it is actually the kinetic energy of the angular motion. Combined with $V(r)$, it is sometimes referred to as the "effective" potential,

$$
V_{\rm eff}(r) = V(r) + \frac{L^2}{2mr^2}. \qquad (12)
$$

Note that if one treats the effective potential like a real potential, one would expect to be able to generate an effective force,

$$
\begin{aligned}
F_{\rm eff} &= -\frac{d}{dr}V(r) - \frac{d}{dr}\frac{L^2}{2mr^2} \qquad (13)\\
&= F(r) + \frac{L^2}{mr^3} = F(r) + m\frac{v_\perp^2}{r},
\end{aligned}
$$

which is indeed matches the form for $m\ddot{r}$ in Eq. (3), which included the **centrifugal** force.

The following code plots this effective potential for a simple choice of parameters, with a standard gravitational potential $-\alpha/r$. Here we have chosen $L = m = \alpha = 1$.

```
# Common imports
import numpy as np
from math import *
import matplotlib.pyplot as plt

Deltax = 0.01
#set up arrays
xinitial = 0.3
xfinal = 5.0
alpha = 1.0   # spring constant
m = 1.0   # mass, you can change these
AngMom = 1.0  #  The angular momentum
n = ceil((xfinal-xinitial)/Deltax)
x = np.zeros(n)
for i in range(n):
    x[i] = xinitial+i*Deltax
V = np.zeros(n)
V = -alpha/x+0.5*AngMom*AngMom/(m*x*x)
# Plot potential
fig, ax = plt.subplots()
ax.set_xlabel('r[m]')
ax.set_ylabel('V[J]')
ax.plot(x, V)
fig.tight_layout()
plt.show()
```

Gravitational force example. Using the above parameters, we can now study the evolution of the system using for example the velocity Verlet method. This is done in the code here for an initial radius equal to the minimum of the potential well. We seen then that the radius is always the same and corresponds to a circle (the radius is always constant).

```
# Common imports
import numpy as np
import pandas as pd
from math import *
import matplotlib.pyplot as plt
import os

# Where to save the figures and data files
PROJECT_ROOT_DIR = "Results"
FIGURE_ID = "Results/FigureFiles"
DATA_ID = "DataFiles/"

if not os.path.exists(PROJECT_ROOT_DIR):
    os.mkdir(PROJECT_ROOT_DIR)

if not os.path.exists(FIGURE_ID):
    os.makedirs(FIGURE_ID)

if not os.path.exists(DATA_ID):
    os.makedirs(DATA_ID)

def image_path(fig_id):
    return os.path.join(FIGURE_ID, fig_id)

def data_path(dat_id):
    return os.path.join(DATA_ID, dat_id)

def save_fig(fig_id):
    plt.savefig(image_path(fig_id) + ".png", format='png')


# Simple Gravitational Force   -alpha/r

DeltaT = 0.01
#set up arrays
tfinal = 100.0
n = ceil(tfinal/DeltaT)
# set up arrays for t, v and r
t = np.zeros(n)
v = np.zeros(n)
r = np.zeros(n)
# Constants of the model, setting all variables to one for simplicity
alpha = 1.0
AngMom = 1.0  #  The angular momentum
m = 1.0  # scale mass to one
c1 = AngMom*AngMom/(m*m)
c2 = AngMom*AngMom/m
rmin = (AngMom*AngMom/m/alpha)
# Initial conditions
r0 = rmin
v0 = 0.0
r[0] = r0
v[0] = v0
# Start integrating using the Velocity-Verlet  method
for i in range(n-1):
    # Set up acceleration
    a = -alpha/(r[i]**2)+c1/(r[i]**3)
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    anew = -alpha/(r[i+1]**2)+c1/(r[i+1]**3)
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    t[i+1] = t[i] + DeltaT
    # Plot position as function of time
fig, ax = plt.subplots(2,1)
ax[0].set_xlabel('time')
ax[0].set_ylabel('radius')
ax[0].plot(t,r)
ax[1].set_xlabel('time')
ax[1].set_ylabel('Velocity')
ax[1].plot(t,v)
save_fig("RadialGVV")
plt.show()
```

Changing the value of the initial position to a value where the energy is positive, leads to an increasing radius with time, a so-called unbound orbit. Choosing on the other hand an initial radius that corresponds to a negative energy and different from the minimum value leads to a radius that oscillates back and forth between two values.

Harmonic Oscillator in two dimensions. Consider a particle of mass m in a 2-dimensional harmonic oscillator with potential

$$V = \frac{1}{2}kr^2 = \frac{1}{2}k(x^2+y^2).$$

If the orbit has angular momentum L, we can find the radius and angular velocity of the circular orbit as well as the b) the angular frequency of small radial perturbations.

We consider the effective potential. The radius of a circular orbit is at the minimum of the potential (where the effective force is zero). The potential is plotted here with the parameters $k = m = 0.1$ and $L = 1.0$.

```
# Common imports
import numpy as np
from math import *
import matplotlib.pyplot as plt

Deltax = 0.01
#set up arrays
xinitial = 0.5
xfinal = 3.0
k = 1.0   # spring constant
m = 1.0   # mass, you can change these
AngMom = 1.0  #  The angular momentum
n = ceil((xfinal-xinitial)/Deltax)
x = np.zeros(n)
for i in range(n):
    x[i] = xinitial+i*Deltax
V = np.zeros(n)
V = 0.5*k*x*x+0.5*AngMom*AngMom/(m*x*x)
# Plot potential
fig, ax = plt.subplots()
ax.set_xlabel('r[m]')
ax.set_ylabel('V[J]')
ax.plot(x, V)
fig.tight_layout()
plt.show()
```

$$V_{\rm eff} = \frac{1}{2}kr^2 + \frac{L^2}{2mr^2}$$

The effective potential looks like that of a harmonic oscillator for large r, but for small r, the centrifugal potential repels the particle from the origin. The combination of the two potentials has a minimum for at some radius $r_{\rm min}$.

$$\begin{aligned}
0 &= kr_{\rm min} - \frac{L^2}{mr_{\rm min}^3},\\
r_{\rm min} &= \left(\frac{L^2}{mk}\right)^{1/4},\\
\dot{\theta} &= \frac{L}{mr_{\rm min}^2} = \sqrt{k/m}.
\end{aligned}$$

For particles at r_{min} with $\dot{r}=0$, the particle does not accelerate and r stays constant, i.e. a circular orbit. The radius of the circular orbit can be adjusted by changing the angular momentum L.

For the above parameters this minimum is at $r_{\mathrm{min}}=1$.

Now consider small vibrations about r_{min}. The effective spring constant is the curvature of the effective potential.

$$\begin{aligned}
k_{\mathrm{eff}} &= \left.\frac{d^2}{dr^2}V_{\mathrm{eff}}(r)\right|_{r=r_{\mathrm{min}}} = k+\frac{3L^2}{mr_{\mathrm{min}}^4} \\
&= 4k, \\
\omega &= \sqrt{k_{\mathrm{eff}}/m} = 2\sqrt{k/m} = 2\dot{\theta}.
\end{aligned}$$

Because the radius oscillates with twice the angular frequency, the orbit has two places where r reaches a minimum in one cycle. This differs from the inverse-square force where there is one minimum in an orbit. One can show that the orbit for the harmonic oscillator is also elliptical, but in this case the center of the potential is at the center of the ellipse, not at one of the foci.

The solution is also simple to write down exactly in Cartesian coordinates. The x and y equations of motion separate,

$$\begin{aligned}
\ddot{x} &= -kx, \\
\ddot{y} &= -ky.
\end{aligned}$$

The general solution can be expressed as

$$\begin{aligned}
x &= A\cos\omega_0 t + B\sin\omega_0 t, \\
y &= C\cos\omega_0 t + D\sin\omega_0 t.
\end{aligned}$$

The code here finds the solution for x and y using the code we developed in homework 5 and 6 and the midterm. Note that this code is tailored to run in Cartesian coordinates. There is thus no angular momentum dependent term.

Here we have chose initial conditions that correspond to the minimum of the effective potential r_{min}. We have chosen $x_0=r_{\mathrm{min}}$ and $y_0=0$. Similarly, we use the centripetal acceleration to determine the initial velocity so that we have a circular motion (see back to the last question of the midterm). This means that we set the centripetal acceleration v^2/r equal to the force from the harmonic oscillator $-k\boldsymbol{r}$. Taking the magnitude of $\boldsymbol{r}$ we have then $v^2/r=k/mr$, which gives $v=\pm\omega_0 r$.

Since the code here solves the equations of motion in cartesian coordinates and the harmonic oscillator potential leads to forces in the x- and y-directions that are decoupled, we have to select the initial velocities and positions so that we don't get that for example $y(t)=0$.

We set x_0 to be different from zero and v_{y0} to be different from zero.

```
DeltaT = 0.001
#set up arrays
tfinal = 10.0
n = ceil(tfinal/DeltaT)
# set up arrays
t = np.zeros(n)
v = np.zeros((n,2))
r = np.zeros((n,2))
radius = np.zeros(n)
# Constants of the model
k = 1.0   # spring constant
m = 1.0   # mass, you can change these
omega02 = k/m  # Frequency
AngMom = 1.0  #  The angular momentum
# Potential minimum
rmin = (AngMom*AngMom/k/m)**0.25
# Initial conditions as compact 2-dimensional arrays, x0=rmin and y0 = 0
x0 = rmin; y0= 0.0
r0 = np.array([x0,y0])
vy0 = sqrt(omega02)*rmin; vx0 = 0.0
v0 = np.array([vx0,vy0])
r[0] = r0
v[0] = v0
# Start integrating using the Velocity-Verlet  method
for i in range(n-1):
    # Set up the acceleration
    a =  -r[i]*omega02
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    anew = -r[i+1]*omega02
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    t[i+1] = t[i] + DeltaT
# Plot position as function of time
radius = np.sqrt(r[:,0]**2+r[:,1]**2)
fig, ax = plt.subplots(3,1)
ax[0].set_xlabel('time')
ax[0].set_ylabel('radius squared')
ax[0].plot(t,r[:,0]**2+r[:,1]**2)
ax[1].set_xlabel('time')
ax[1].set_ylabel('x position')
ax[1].plot(t,r[:,0])
ax[2].set_xlabel('time')
ax[2].set_ylabel('y position')
ax[2].plot(t,r[:,1])

fig.tight_layout()
save_fig("2DimHOVV")
plt.show()
```

We see that the radius (to within a given error), we obtain a constant radius.

The following code shows first how we can solve this problem using the radial degrees of freedom only. Here we need to add the explicit centrifugal barrier. Note that the variable r depends only on time. There is no x and y directions since we have transformed the equations to polar coordinates.

```
DeltaT = 0.01
#set up arrays
tfinal = 10.0
n = ceil(tfinal/DeltaT)
```

```
# set up arrays for t, v and r
t = np.zeros(n)
v = np.zeros(n)
r = np.zeros(n)
E = np.zeros(n)
# Constants of the model
AngMom = 1.0  #  The angular momentum
m = 1.0
k = 1.0
omega02 = k/m
c1 = AngMom*AngMom/(m*m)
c2 = AngMom*AngMom/m
rmin = (AngMom*AngMom/k/m)**0.25
# Initial conditions
r0 = rmin
v0 = 0.0
r[0] = r0
v[0] = v0
E[0] = 0.5*m*v0*v0+0.5*k*r0*r0+0.5*c2/(r0*r0)
# Start integrating using the Velocity-Verlet  method
for i in range(n-1):
    # Set up acceleration
    a = -r[i]*omega02+c1/(r[i]**3)
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    anew = -r[i+1]*omega02+c1/(r[i+1]**3)
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    t[i+1] = t[i] + DeltaT
    E[i+1] = 0.5*m*v[i+1]*v[i+1]+0.5*k*r[i+1]*r[i+1]+0.5*c2/(r[i+1]*r[i+1])
    # Plot position as function of time
fig, ax = plt.subplots(2,1)
ax[0].set_xlabel('time')
ax[0].set_ylabel('radius')
ax[0].plot(t,r)
ax[1].set_xlabel('time')
ax[1].set_ylabel('Energy')
ax[1].plot(t,E)
save_fig("RadialHOVV")
plt.show()
```

With some work using double angle formulas, one can calculate

$$\begin{aligned}
r^2 &= x^2+y^2\\
&= (A^2+C^2)\cos^2(\omega_0 t)+(B^2+D^2)\sin^2\omega_0 t+(AB+CD)\cos(\omega_0 t)\sin(\omega_0 t)\\
&= \alpha+\beta\cos 2\omega_0 t+\gamma\sin 2\omega_0 t,\\
\alpha &= \frac{A^2+B^2+C^2+D^2}{2},\quad \beta=\frac{A^2-B^2+C^2-D^2}{2},\quad \gamma=AB+CD,\\
r^2 &= \alpha+(\beta^2+\gamma^2)^{1/2}\cos(2\omega_0 t-\delta),\quad \delta=\arctan(\gamma/\beta),
\end{aligned}$$

and see that radius oscillates with frequency $2\omega_0$. The factor of two comes because the oscillation $x=A\cos\omega_0 t$ has two maxima for x^2, one at $t=0$ and one a half period later.

Stability of Orbits

The effective force can be extracted from the effective potential, $V_{\rm eff}$. Beginning from the equations of motion, Eq. (1), for r,

$$\begin{aligned} m\ddot{r} &= F + \frac{L^2}{mr^3} \\ &= F_{\rm eff} \\ &= -\partial_r V_{\rm eff}, \\ F_{\rm eff} &= -\partial_r \left[V(r) + (L^2/2mr^2)\right]. \end{aligned} \tag{14}$$

For a circular orbit, the radius must be fixed as a function of time, so one must be at a maximum or a minimum of the effective potential. However, if one is at a maximum of the effective potential the radius will be unstable. For the attractive Coulomb force the effective potential will be dominated by the $-\alpha/r$ term for large r because the centrifugal part falls off more quickly, $\sim 1/r^2$. At low r the centrifugal piece wins and the effective potential is repulsive. Thus, the potential must have a minimum somewhere with negative potential. The circular orbits are then stable to perturbation.

The effective potential is sketched for two cases, a $1/r$ attractive potential and a $1/r^3$ attractive potential. The $1/r$ case has a stable minimum, whereas the circular orbit in the $1/r^3$ case is unstable.

If one considers a potential that falls as $1/r^3$, the situation is reversed and the point where $\partial_r V$ disappears will be a local maximum rather than a local minimum. **Fig to come here with code**

The repulsive centrifugal piece dominates at large r and the attractive Coulomb piece wins out at small r. The circular orbit is then at a maximum of the effective potential and the orbits are unstable. It is the clear that for potentials that fall as r^n, that one must have $n > -2$ for the orbits to be stable.

Consider a potential $V(r) = \beta r$. For a particle of mass m with angular momentum L, find the angular frequency of a circular orbit. Then find the angular frequency for small radial perturbations.

For the circular orbit you search for the position $r_{\rm min}$ where the effective potential is minimized,

$$\begin{aligned} \partial_r \left\{\beta r + \frac{L^2}{2mr^2}\right\} &= 0, \\ \beta &= \frac{L^2}{mr_{\rm min}^3}, \\ r_{\rm min} &= \left(\frac{L^2}{\beta m}\right)^{1/3}, \\ \dot{\theta} &= \frac{L}{mr_{\rm min}^2} = \frac{\beta^{2/3}}{(mL)^{1/3}} \end{aligned}$$

Now, we can find the angular frequency of small perturbations about the circular orbit. To do this we find the effective spring constant for the effective potential,

$$\begin{aligned} k_{\rm eff} &= \partial_r^2 \, V_{\rm eff}|_{r_{\rm min}} \\ &= \frac{3L^2}{mr_{\rm min}^4}, \\ \omega &= \sqrt{\frac{k_{\rm eff}}{m}} \\ &= \frac{\beta^{2/3}}{(mL)^{1/3}}\sqrt{3}. \end{aligned}$$

If the two frequencies, $\dot{\theta}$ and ω, differ by an integer factor, the orbit's trajectory will repeat itself each time around. This is the case for the inverse-square force, $\omega = \dot{\theta}$, and for the harmonic oscillator, $\omega = 2\dot{\theta}$. In this case, $\omega = \sqrt{3}\dot{\theta}$, and the angles at which the maxima and minima occur change with each orbit.

Code example with gravitional force. The code example here is meant to illustrate how we can make a plot of the final orbit. We solve the equations in polar coordinates (the example here uses the minimum of the potential as initial value) and then we transform back to cartesian coordinates and plot x versus y. We see that we get a perfect circle when we place ourselves at the minimum of the potential energy, as expected.

```
# Simple Gravitational Force   -alpha/r

DeltaT = 0.01
#set up arrays
tfinal = 8.0
n = ceil(tfinal/DeltaT)
# set up arrays for t, v and r
t = np.zeros(n)
v = np.zeros(n)
r = np.zeros(n)
phi = np.zeros(n)
x = np.zeros(n)
y = np.zeros(n)
# Constants of the model, setting all variables to one for simplicity
alpha = 1.0
AngMom = 1.0  #  The angular momentum
m = 1.0  # scale mass to one
c1 = AngMom*AngMom/(m*m)
c2 = AngMom*AngMom/m
rmin = (AngMom*AngMom/m/alpha)
# Initial conditions, place yourself at the potential min
r0 = rmin
v0 = 0.0  # starts at rest
r[0] = r0
v[0] = v0
phi[0] = 0.0
# Start integrating using the Velocity-Verlet  method
```

```
for i in range(n-1):
    # Set up acceleration
    a = -alpha/(r[i]**2)+c1/(r[i]**3)
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    anew = -alpha/(r[i+1]**2)+c1/(r[i+1]**3)
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    t[i+1] = t[i] + DeltaT
    phi[i+1] = t[i+1]*c2/(r0**2)
# Find cartesian coordinates for easy plot
x = r*np.cos(phi)
y = r*np.sin(phi)
fig, ax = plt.subplots(3,1)
ax[0].set_xlabel('time')
ax[0].set_ylabel('radius')
ax[0].plot(t,r)
ax[1].set_xlabel('time')
ax[1].set_ylabel('Angle $\cos{\phi}$')
ax[1].plot(t,np.cos(phi))
ax[2].set_ylabel('y')
ax[2].set_xlabel('x')
ax[2].plot(x,y)

save_fig("Phasespace")
plt.show()
```

Try to change the initial value for r and see what kind of orbits you get. In order to test different energies, it can be useful to look at the plot of the effective potential discussed above.

However, for orbits different from a circle the above code would need modifications in order to allow us to display say an ellipse. For the latter, it is much easier to run our code in cartesian coordinates, as done here. In this code we test also energy conservation and see that it is conserved to numerical precision. The code here is a simple extension of the code we developed for homework 4.

```
# Common imports
import numpy as np
import pandas as pd
from math import *
import matplotlib.pyplot as plt

DeltaT = 0.01
#set up arrays
tfinal = 10.0
n = ceil(tfinal/DeltaT)
# set up arrays
t = np.zeros(n)
v = np.zeros((n,2))
r = np.zeros((n,2))
E = np.zeros(n)
# Constants of the model
m = 1.0   # mass, you can change these
alpha = 1.0
# Initial conditions as compact 2-dimensional arrays
x0 = 0.5; y0= 0.
r0 = np.array([x0,y0])
v0 = np.array([0.0,1.0])
r[0] = r0
```

```python
v[0] = v0
rabs = sqrt(sum(r[0]*r[0]))
E[0] = 0.5*m*(v[0,0]**2+v[0,1]**2)-alpha/rabs
# Start integrating using the Velocity-Verlet  method
for i in range(n-1):
    # Set up the acceleration
    rabs = sqrt(sum(r[i]*r[i]))
    a =  -alpha*r[i]/(rabs**3)
    # update velocity, time and position using the Velocity-Verlet method
    r[i+1] = r[i] + DeltaT*v[i]+0.5*(DeltaT**2)*a
    rabs = sqrt(sum(r[i+1]*r[i+1]))
    anew = -alpha*r[i+1]/(rabs**3)
    v[i+1] = v[i] + 0.5*DeltaT*(a+anew)
    E[i+1] = 0.5*m*(v[i+1,0]**2+v[i+1,1]**2)-alpha/rabs
    t[i+1] = t[i] + DeltaT
# Plot position as function of time
fig, ax = plt.subplots(3,1)
ax[0].set_ylabel('y')
ax[0].set_xlabel('x')
ax[0].plot(r[:,0],r[:,1])
ax[1].set_xlabel('time')
ax[1].set_ylabel('y position')
ax[1].plot(t,r[:,0])
ax[2].set_xlabel('time')
ax[2].set_ylabel('y position')
ax[2].plot(t,r[:,1])

fig.tight_layout()
save_fig("2DimGravity")
plt.show()
print(E)
```