FYS-MEK1110 Oblig 3

Samuel Bigirimana

February 28, 2021

task a) the position $s(t)$ given $l_0 = 0.5m$

By: $t = 0$, $L_0 = 0.5m$ and $h = 0.3$:

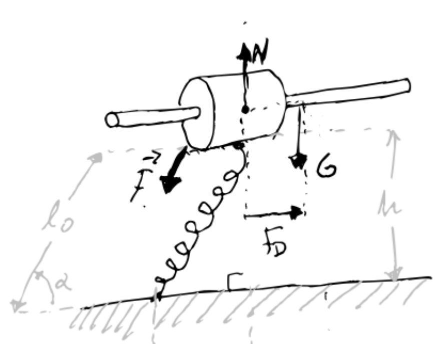
$$\begin{aligned} 0.5m &= \sqrt{(0.3m)^2 + (x_0)^2} \\ x_0 &= \sqrt{(0.5m)^2 - (0.3m)^2} \\ &= \underline{\underline{0.4m}} \end{aligned}$$

task b) deciding the spring length given any position

The block will only move on the x-axis, this means that the spring can´t be shorter than the hight difference between the block and the attachment point. So the only movement the block makes is alongside the x-axises given a given time:

$$l = \sqrt{(0.3)^2 + (x(t))^2}$$

task c) drawing the forces on the block:



Normal
$N = G$ gravitation
$\vec{F} = \vec{F}_D$
Spring Friction

task d) proving the force in x-direction

As the springforce is given as $F = -k(r - L_0)\frac{r}{r}$ and if we decomponise it in x-direction, we get:

$$\begin{aligned} F_x &= -k(r - L_0)\frac{x}{r} \\ &= -k(\frac{rx}{x} - \frac{L_0 x}{r}) \\ &= -k(x - \frac{L_0 x}{r}) \\ &= -kx(1 - \frac{L_0}{r}) \end{aligned}$$

$$\begin{aligned} r &= \sqrt{x^2 + y^2} \\ \Rightarrow y &= h \end{aligned}$$

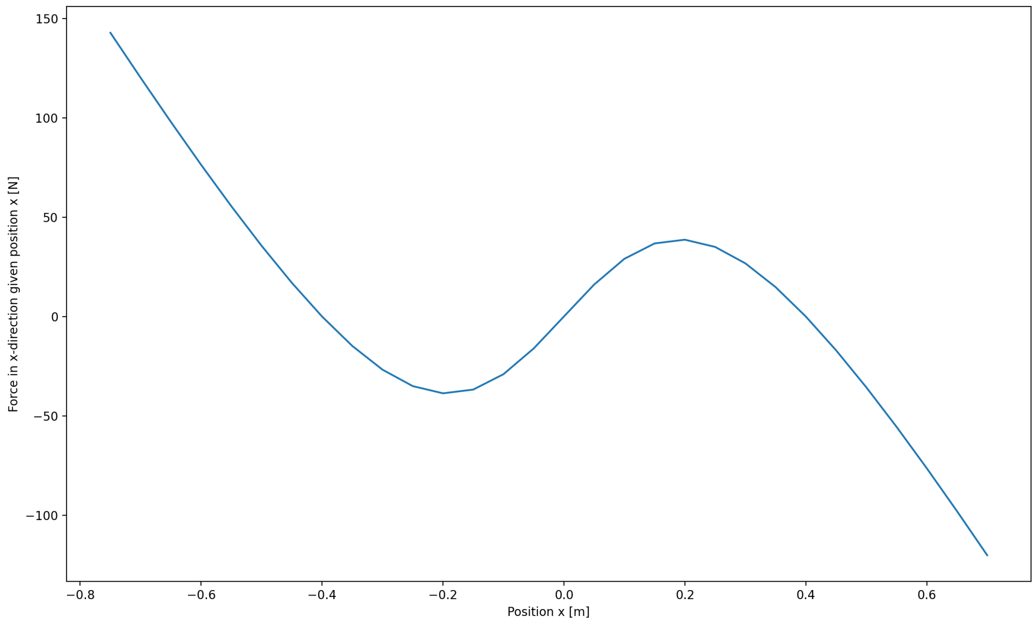
$$\Rightarrow F_x = -kx(1 - \frac{L_0}{\sqrt{x^2 + h^2}})$$

task e) plotting F_x given position in x-direction

› The program I made to build the plot:

```
import numpy as np
import matplotlib.pyplot as plt

k = 500       # spring constant in N/m
L0 = 0.5      # spring length in m
h = 0.3       # height in m

# Created a linspace that goes from the bounds defined by the task
x = np.arange(-0.75, 0.75, 0.05)

# Defined F_x function
F_x = -k*x*(1-(L0/np.sqrt(x**2+h**2)))

print(F_x)

plt.plot(x, F_x)
plt.xlabel("Position x [m]")
plt.ylabel("Force in x-direction given position x [N]")
plt.show()
```

› The plot results for F_x:

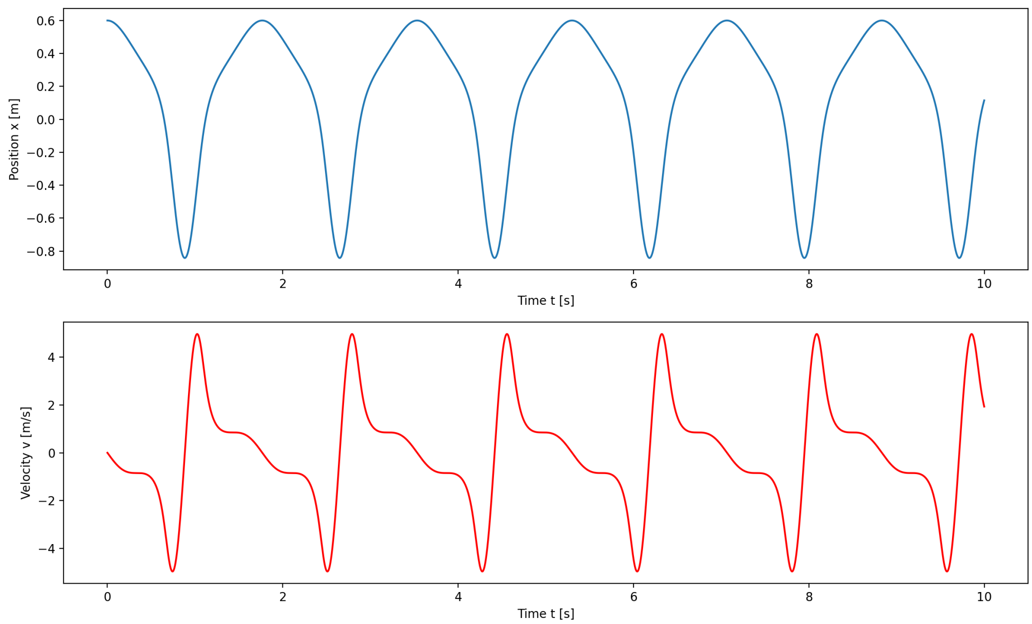


task f) solving the equations of motion numerically using the Euler-Cromer method given $x1 = 0.6m$

```
import numpy as np
import matplotlib.pyplot as plt

k = 500      # spring constant in N/m
h = 0.3      # height difference between block and "floor", in m
x0 = 0.4     # nautral position on x-axis, in m
x1 = 0.6     # draging the block to a start position on x-axis, in m
L0 = 0.5     # spring length in nautral position, in m
m = 5        # block weight, in kg

time = 10        # Time window
dt = 0.001       # Time intervals
```

```python
n = int(np.ceil(time/dt))

# Defining the arrays:
t = np.zeros(n,float)
r = np.zeros(n,float)
v = np.zeros(n,float)
a = np.zeros(n,float)

# Here we use the initial conditions for the first vector elements
r[0] = x1        # at the start, the person is at x = 0
t[0] = 0.0       # We start the clock at t = 0 s
v[0] = 0.0       # no velocity at the start, v0 = 0 m/s
a[0] = 0.0   # at the start, the speed = 0

# Now we calculate everything
for i in range(n-1):
        # calculating the force in x-direction
        Fx = -k*(r[i]-x0)*(1-(L0/np.sqrt(r[i]**2+h**2)))
        # calculating the acceleration:
        a[i] = Fx/m

        # Calculating with the Euler-Cromer method:
        v[i+1] = v[i] + a[i]*dt
        r[i+1] = r[i] + v[i+1]*dt
        t[i+1] = t[i] + dt

# In the first "window", we plot how the position x evolves with time
plt.subplot(2,1,1)
plt.plot(t,r)
plt.xlabel('Time t [s]')
plt.ylabel('Position x [m]')

# Now switch to the next "window", index = 2
# Here we want to plot how the velocity v varies with time t
plt.subplot(2,1,2)
plt.plot(t,v, 'r')
plt.xlabel('Time t [s]')
plt.ylabel('Velocity v [m/s]')

# Show everything:
plt.show()
```

› The resulting plot:

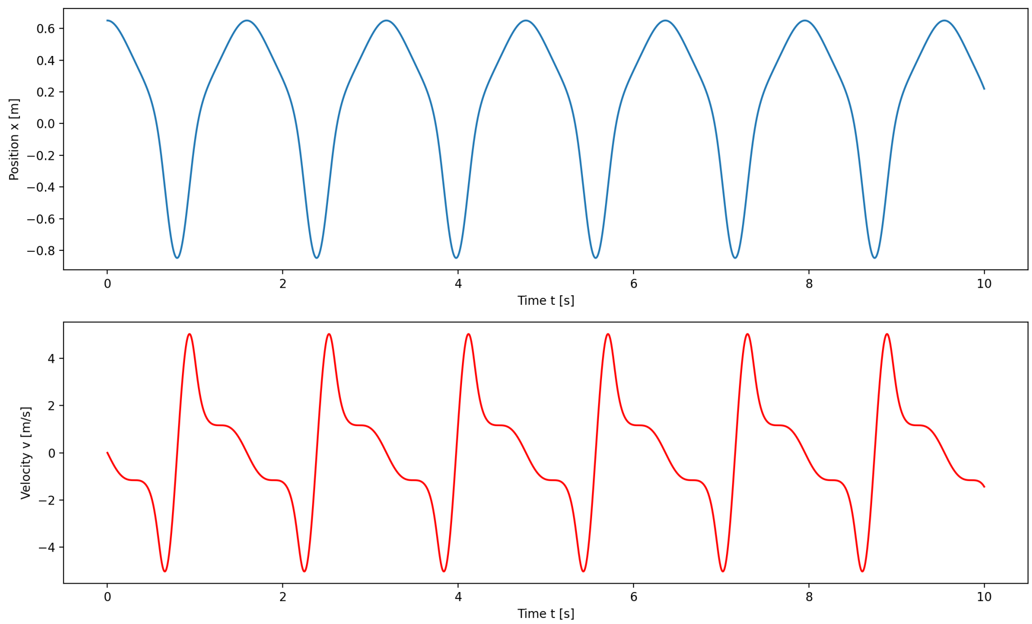


task g) solving the equations of motion numerically using the Euler-Cromer method given $x1 = 0.65m$

```
import numpy as np
import matplotlib.pyplot as plt

k = 500      # spring constant in N/m
h = 0.3      # height difference between block and "floor", in m
x0 = 0.4     # nautral position on x-axis, in m
x1 = 0.65     # draging the block to a start position on x-axis, in m
L0 = 0.5     # spring length in nautral position, in m
m = 5        # block weight, in kg

time = 10        # Time window
dt = 0.001       # Time intervals
```

```python
n = int(np.ceil(time/dt))

# Defining the arrays:
t = np.zeros(n,float)
r = np.zeros(n,float)
v = np.zeros(n,float)
a = np.zeros(n,float)

# Here we use the initial conditions for the first vector elements
r[0] = x1        # at the start, the person is at x = 0
t[0] = 0.0       # We start the clock at t = 0 s
v[0] = 0.0       # no velocity at the start, v0 = 0 m/s
a[0] = 0.0  # at the start, the speed = 0

# Now we calculate everything
for i in range(n-1):
        # calculating the force in x-direction
        Fx = -k*(r[i]-x0)*(1-(L0/np.sqrt(r[i]**2+h**2)))
        # calculating the acceleration:
        a[i] = Fx/m

        # Calculating with the Euler-Cromer method:
        v[i+1] = v[i] + a[i]*dt
        r[i+1] = r[i] + v[i+1]*dt
        t[i+1] = t[i] + dt

# In the first "window", we plot how the position x evolves with time
plt.subplot(2,1,1)
plt.plot(t,r)
plt.xlabel('Time t [s]')
plt.ylabel('Position x [m]')

# Now switch to the next "window", index = 2
# Here we want to plot how the velocity v varies with time t
plt.subplot(2,1,2)
plt.plot(t,v, 'r')
plt.xlabel('Time t [s]')
plt.ylabel('Velocity v [m/s]')

# Show everything:
plt.show()
```

› The resulting plot:



task h) proving the force in y-direction

$$
\begin{aligned}
F_y &= -k(r - L_0)\frac{h}{r} \\
&= -k\left(\frac{rh}{h} - \frac{L_0 h}{r}\right) \\
&= -k\left(h - \frac{L_0 h}{r}\right) \\
&= -kh\left(1 - \frac{L_0}{r}\right)
\end{aligned}
$$

$$
\begin{aligned}
r &= \sqrt{x^2 + y^2} \\
\Rightarrow y &= h
\end{aligned}
$$

$$
\Rightarrow F_y = -kh\left(1 - \frac{L_0}{\sqrt{x^2 + h^2}}\right)
$$

task i) deciding the neutral force N:

Since the block is stand still by $t = 0$, N has to be the same size as the gravitational size (G):
$N = G = mass * gravitationalacceleration = 5kg * 9.81m/s^2 = \underline{\underline{49N}}$

task j) profing N(x):

If the sum of all forces on the block is equal to 0 (the block is standing still), we get:

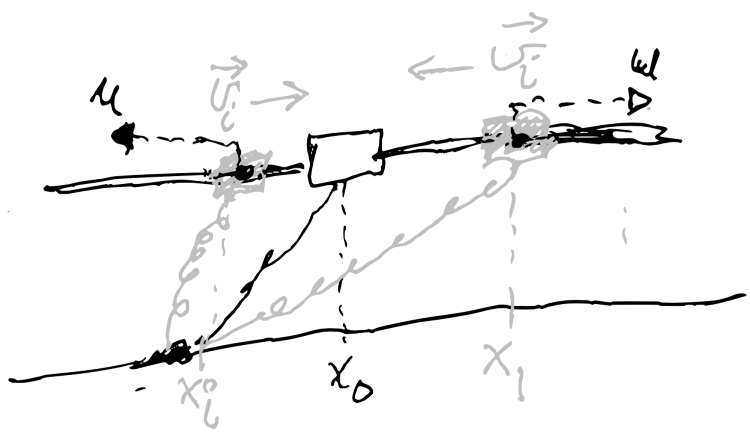
$$\sum F = 0$$

$$N + (-G) + (-kh(1 - (\frac{L_0}{\sqrt{x^2 + h^2}}))) = 0$$

$$\Rightarrow N(x) - m * g = kh(1 - (\frac{L_0}{\sqrt{x^2 + h^2}}))$$

$$N(x) = kh(1 - (\frac{L_0}{\sqrt{x^2 + h^2}})) + m * g$$

task k) drawing the forces on the block with friction:



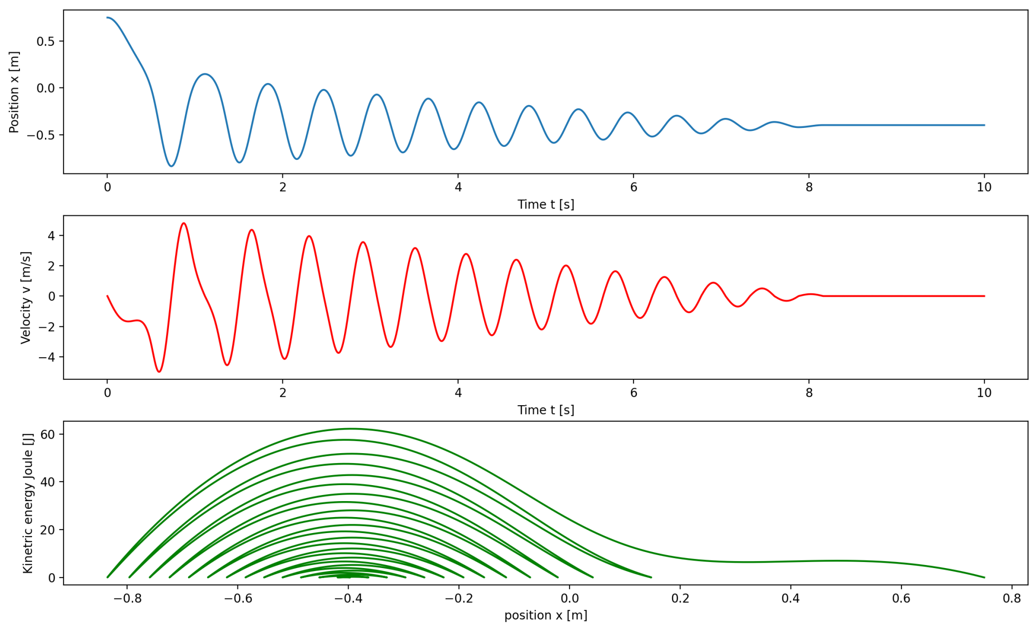
$u = N \cdot \mu_d$ wher $N = |N|$ and $\mu_d = 0.05$

› The program for task k and i:

```
import numpy as np
import matplotlib.pyplot as plt

k = 500       # spring constant in N/m
h = 0.3       # height difference between block and "floor", in m
x0 = 0.4      # nautral position on x-axis, in m
x1 = 0.65      # draging the block to a start position on x-axis, in m
L0 = 0.5      # spring length in nautral position, in m
m = 5         # block weight, in kg

time = 10         # Time window
dt = 0.001        # Time intervals

n = int(np.ceil(time/dt))

# Defining the arrays:
t = np.zeros(n,float)
r = np.zeros(n,float)
v = np.zeros(n,float)
a = np.zeros(n,float)

# Here we use the initial conditions for the first vector elements
r[0] = x1         # at the start, the person is at x = 0
t[0] = 0.0        # We start the clock at t = 0 s
v[0] = 0.0        # no velocity at the start, v0 = 0 m/s
a[0] = 0.0   # at the start, the speed = 0

# Now we calculate everything
for i in range(n-1):
        # calculating the force in x-direction
        Fx = -k*(r[i]-x0)*(1-(L0/np.sqrt(r[i]**2+h**2)))
        # calculating the acceleration:
        a[i] = Fx/m

        # Calculating with the Euler-Cromer method:
        v[i+1] = v[i] + a[i]*dt
        r[i+1] = r[i] + v[i+1]*dt
        t[i+1] = t[i] + dt

# In the first "window", we plot how the position x evolves with time
plt.subplot(2,1,1)
plt.plot(t,r)
plt.xlabel('Time t [s]')
plt.ylabel('Position x [m]')
```

```
# Now switch to the next "window", index = 2
# Here we want to plot how the velocity v varies with time t
plt.subplot(2,1,2)
plt.plot(t,v, 'r')
plt.xlabel('Time t [s]')
plt.ylabel('Velocity v [m/s]')

# Show everything:
plt.show()
```

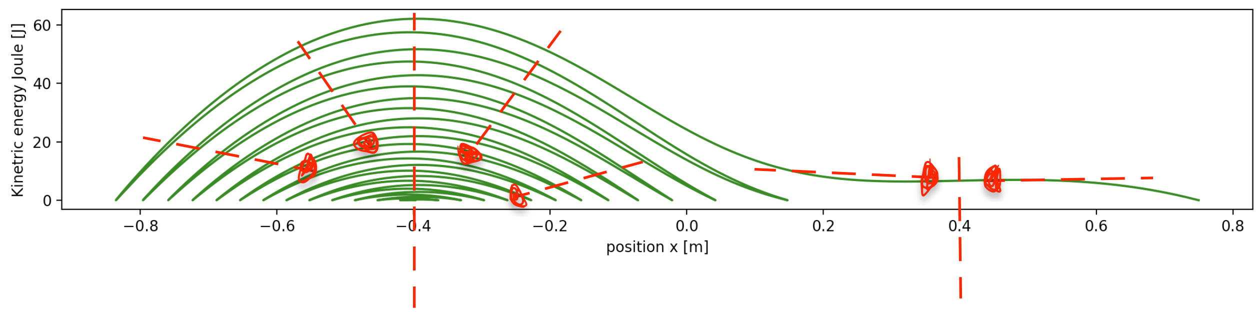
› The resulting plot:



› Task k; as we can see from the plot, the block´s movement is now getting braked by the friction. The force is not preserved and the movement stops.
› Task l: the last graph shows the kinetic energy given a position x. As we can see, the energy is at its´highest when the block is right above the neutral position X_0. This is becausen the movement has its highest velocity by the point.

task m) finding the equilibrium points

By looking at the plot for the kinetic energy, we can see that the equilibrium points for this equation are 0.4m and -0.4m on the x-axis.



Stable:
this is the position where the block stabilizes and the energy dies out.

Unstable:
The block passes straight beyond, but it flattens out before gaining speed the other way.