Control Theory HW 1

Alina Bogdanova BS18-04

February 2020

1 Introduction

The goal of this assignment is to solve the systems that represent oscillations: write their state-space model and simulate.

2 Task 1

- Write state-space model for a spring-damper system
- Make simulation

2.1 State-space model

The given system:

$$m\ddot{x} + b\dot{x} + kx = F(1.1)$$

This system can be represented as a *state-space model*:

$$\begin{cases} \dot{X} = \begin{bmatrix} -b/m & -k/m \\ 1 & 0 \end{bmatrix} X + \begin{bmatrix} F/m \\ 0 \end{bmatrix} where X = \begin{bmatrix} \dot{x} \\ x \end{bmatrix} \\ y = X \end{cases}$$

The system could be solved analytically with the use of eigendecomposition:
Let A = $\begin{bmatrix} -b/m & -k/m \\ 1 & 0 \end{bmatrix}$, b = $\begin{bmatrix} F/m \\ 0 \end{bmatrix}$

$$A = WDW^{-1}$$

$$W^{-1}\dot{X} = DW^{-1}X + W^{-1}b$$

let B = $W^{-1}b$
let R = $W^{-1}X$

$$\dot{R} = DR + B$$

- can be solved by usual methods

$$X = WR$$

Also the system may be solved numerically using such tools as Python's library scipy.

2.2 Simulation

The simulation of this system was made with the initial parameters: $F = 0.5N, m = 0.6kg, b = 0.5sN/m, k = 0.8N/m, x_0 = 0m, \dot{x_0} = 0.1m/s$

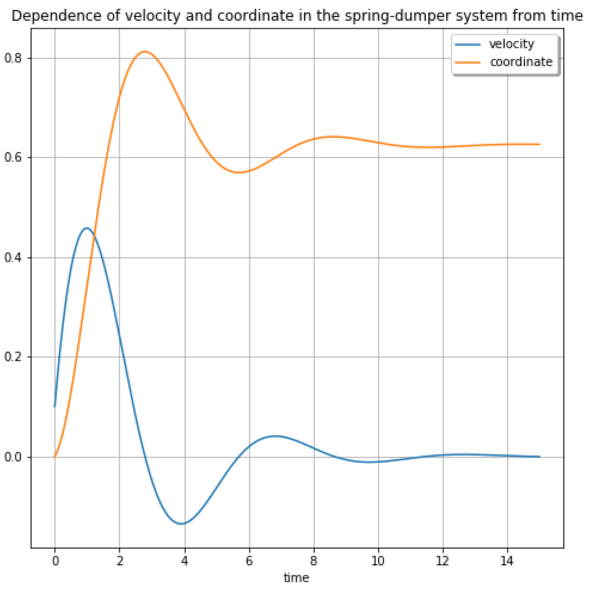

Figure 1: The simulation of the dump-spring system

On the figure 1 we can see the graphs of damped oscillations. Which is physically correct.

3 Task 2

System from Ogata B-3-4 (Same task) The system of the given task should be derived from physical task represented on the figure 2. The first step is to find out the differential equation, which should be solved.

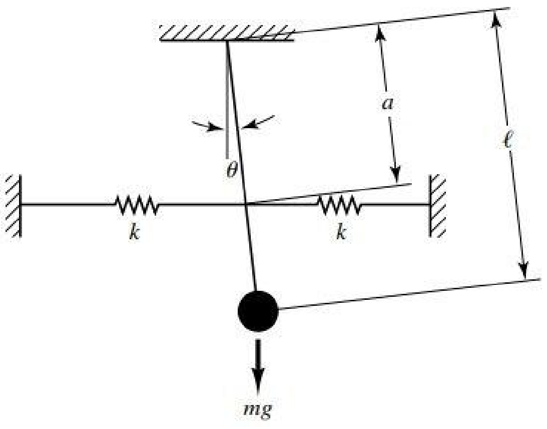

Figure 2: Task 2

3.1 Physical representation

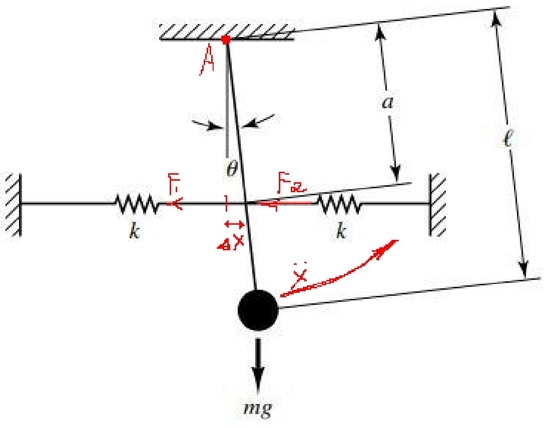

Figure 3: Task 2. Forces

First of all, lets find out forces, which are applied to the rod (see figure 3):
1. F_1, F_2 - forces of stretching, their value is $k\Delta x$
2. mg
3. $m\ddot{x}$ - force of the overall movement by the Newton's law

Let's find the torque from the point A:

$$F_1 cos(\theta)a + F_2 cos(\theta)a + mgsin(\theta)l = m\ddot{x}l$$

$F_1 = k\Delta x = F_2$, $\Delta x = \delta sin(\theta) * a \approx a\theta$

$\ddot{x} = l\ddot{\theta}$

$$2a^2k\theta + mgl\theta = ml^2\ddot{\theta}$$

3.2 State-space model and its solution

The previous equation can be rewritten in the form:

$$\ddot{\theta} - \theta\frac{2a^2k + mgl}{ml^2} = 0$$

We can say, that this equation is similar to equation (1.1): $m = 1, b = 0, k = -\frac{2a^2k+mgl}{ml^2}, F = 0$ Our state-space model looks like:

$$\begin{cases} \dot{X} = \begin{bmatrix} 0 & -\frac{2a^2k+mgl}{ml^2} \\ 1 & 0 \end{bmatrix} X \\ y = X \end{cases}, whereX = \begin{bmatrix} \dot{\theta} \\ \theta \end{bmatrix}$$

3.3 Simulation

The simulation of this system was made with the initial parameters:
$\theta_0 = 0.1rad, \dot{\theta}_0 = 0.1m/s,$
$ml^2 = 0.1, 2a^2k + mgl = 0.2$

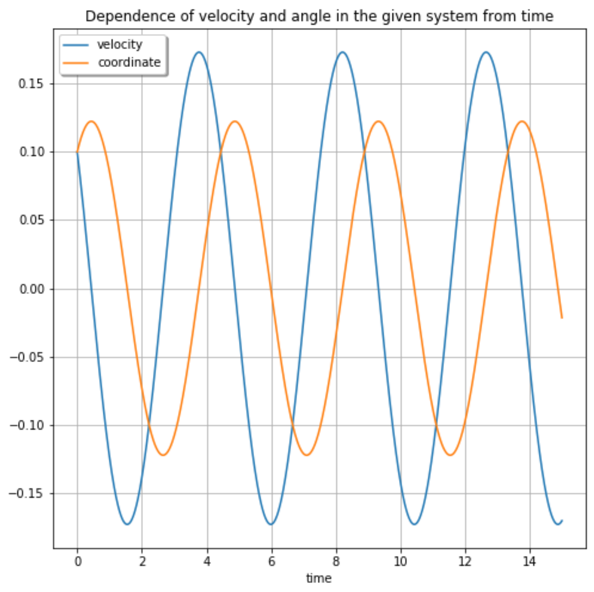

Figure 4: Plot of the oscillations of the system from the second task

4 Conclusion

To sum up, under certain circumstances the linear differential equations of the second order represent oscillations. One of the way to solve such equations is to represent them in state-space model and then solve the numerically, or using eigendecomposition.

Attachment

There is a code, used for simulation of the first task (with slight modifications it can be used for the second one)

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt

# initial values
F = 0.5
m = 0.6
b = 0.5
k = 0.8

# matrix A creation

n = 3
a = np.array([m, b, k])
a_norm = a[1:] / m
A = np.zeros((n-1, n-1))
A[0 , 0:] = -a_norm
A[1:, 0:(n-2)] = np.eye(n-2)

#matrix B creation
B = np.zeros(A.shape[0])
B[0] = F/m

def equationSimplie(x, t):
    return np.dot(A, x) + B

time = np.linspace(0, 15, 1000)
x0 = [0.1, 0]
res = odeint(equationSimplie, x0, time).T

# plotting
fig = plt.figure(figsize=(8, 8))
plt.title("Dependence␣of␣velocity␣and␣angle␣in␣the␣spring-dumper␣system␣from␣tim
plt.xlabel("time")
plt.plot(time, res[0], label="velocity")
plt.plot(time, res[1], label="coordinate")
plt.grid()
plt.legend(shadow=True)
```