

# **Отчёт по лабораторной работе 4**

**дисциплина: Математическое моделирование**

Василиса Михайловна Крючкова, НПИбд-02-18

# Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	15
5	Ответы на вопросы к лабораторной работе	16

# List of Tables

# List of Figures

3.1	Колебания без затуханий и без действий внешней силы . . . . .	13
3.2	Колебания с затуханием и без действий внешней силы . . . . .	14
3.3	Колебания с затуханием и под действием внешней силы . . . . .	14

# 1 Цель работы

Построить модель гармонических колебаний с помощью Python.

## 2 Задание

**Вариант 41** Постройте фазовый портрет гармонического осциллятора и решение уравнения гармонического осциллятора для следующих случаев:

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы  $\ddot{x} + 3,5x = 0$
2. Колебания гармонического осциллятора с затуханием и без действий внешней силы  $\ddot{x} + 7\dot{x} + 3x = 0$
3. Колебания гармонического осциллятора с затуханием и под действием внешней силы  $\ddot{x} + 5\dot{x} + 2x = 2 \sin(6t)$

На интервале  $t \in [0; 37]$  (шаг 0,05) с начальными условиями  $x_0 = 1, y_0 = 1, 2$

## 3 Выполнение лабораторной работы

### 1. Колебания без затуханий и без действий внешней силы

1.1. Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

$$\ddot{x} + 2\gamma\dot{x} + \omega_0^2 x = f(t)$$

Изучила начальные условия. Перед нами уравнение консервативного осциллятора, энергия колебания которого сохраняется во времени. Т. е. потери в системе отсутствуют, значит,  $\gamma = 0$ . Собственная частота колебаний  $\omega = 3,5$ .  $x_0 = 1, y_0 = 1, 2$ . Правая часть уравнения  $f(t) = 0$ .

1.2. Оформила начальные условия в код на Python:

```
x0 = np.array([1.0, 1.2])

w1 = 3.5 #частота, уже в квадрате
g1 = 0.0 #затухание, уже умноженное на 2

def F1(t):
    f = 0
    return f
```

1.3. Решение ищем на интервале  $t \in [0; 37]$  (шаг 0,05), значит,  $t_0 = 0$  – начальный момент времени,  $t_{max} = 37$  – предельный момент времени,  $dt = 0,05$  – шаг изменения времени.

1.4. Добавила в программу условия, описывающие время:

```

t0 = 0
tmax = 37
dt = 0.05
t = np.arange(t0, tmax, dt)

```

1.5. Представила заданное уравнение второго порядка в виде системы двух уравнений первого порядка и запрограммировала:

```

def Y1(x, t):
    dx1_1 = x[1]
    dx1_2 = - w1*x[0] - g1*x[1] - F1(t)
    return dx1_1, dx1_2

```

1.6. Запрограммировала решение системы уравнений:

```

x1 = odeint(Y1, x0, t)

```

1.7. Переписала отдельно  $x$  в  $y_1$ , а  $\dot{x}$  в  $y_2$ :

```

y1_1 = x1[:, 0]
y1_2 = x1[:, 1]

```

1.8. Описала построение фазового портрета:

```

plt.plot(y1_1, y1_2)

```

## 2. Колебания с затуханием и без действий внешней силы

2.1. Изучила начальные условия. Потери энергии в системе  $\gamma = 7$ . Собственная частота колебаний  $\omega = 3$ .  $x_0$  и  $y_0$  те же, что и в п. 1.1. Правая часть уравнения такая же, как и в п. 1.1.

2.2. Т. к. вектор начальных условий одинаков для всех пунктов задачи, задаю его один раз в начале. Остальные начальные условия оформила в код на Python (функцию F1 переименовала в F12, т. к. она подходит как для 1-ого, так и для 2-ого случаев):



```
w2 = 3.0
```

```
g2 = 7.0
```

```
def F12(t):
```

```
    f = 0
```

```
    return f
```

2.3. Т. к. интервал, на котором ищем решение, одинаков для всех пунктов задачи, задаю его один раз в начале.

2.4. Представила заданное уравнение второго порядка в виде системы двух уравнений первого порядка и запрограммировала:

```
def Y2(x, t):
```

```
    dx2_1 = x[1]
```

```
    dx2_2 = - w2*x[0] - g2*x[1] - F12(t)
```

```
    return dx2_1, dx2_2
```

2.5. Запрограммировала решение системы уравнений:

```
x2 = odeint(Y2, x0, t)
```

2.6. Переписала отдельно  $x$  в  $y_1$ , а  $\dot{x}$  в  $y_2$ :

```
y2_1 = x2[:, 0]
```

```
y2_2 = x2[:, 1]
```

2.7. Описала построение фазового портрета:

```
plt.plot(y2_1, y2_2)
```

### **3. Колебания с затуханием и под действием внешней силы**

3.1. Изучила начальные условия. Потери энергии в системе  $\gamma = 5$ . Собственная частота колебаний  $\omega = 2$ .  $x_0$  и  $y_0$  те же, что и в п. 1.1. Правая часть уравнения  $f(t) = 2 \sin(6t)$ .

3.2. Т. к. вектор начальных условий одинаков для всех пунктов задачи, задаю его один раз в начале. Остальные начальные условия оформила в код на Python:

```
w3 = 2.0
```

```
g3 = 5.0
```

```
def F3(t):
```

```
    f = 2*np.sin(6*t)
```

```
    return f
```

3.3. Т. к. интервал, на котором ищем решение, одинаков для всех пунктов задачи, задаю его один раз в начале.

3.4. Представила заданное уравнение второго порядка в виде системы двух уравнений первого порядка и запрограммировала:

```
def Y3(x, t):
```

```
    dx3_1 = x[1]
```

```
    dx3_2 = - w3*x[0] - g3*x[1] - F3(t)
```

```
    return dx3_1, dx3_2
```

3.5. Запрограммировала решение системы уравнений:

```
x3 = odeint(Y3, x0, t)
```

3.6. Переписала отдельно  $x$  в  $y_1$ , а  $\dot{x}$  в  $y_2$ :

```
y3_1 = x3[:, 0]
```

```
y3_2 = x3[:, 1]
```

3.7. Описала построение фазового портрета:

```
plt.plot(y3_1, y3_2)
```

## 4. Сборка программы

4.1. Собрала код программы воедино и получила следующее:

```

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

x0 = np.array([1.0, 1.2]) #вектор начальных условий

w1 = 3.5 #частота, уже в квадрате
g1 = 0.0 #затухание, уже умноженное на 2

w2 = 3.0
g2 = 7.0

w3 = 2.0
g3 = 5.0

def F12(t):
    f = 0
    return f

def F3(t):
    f = 2*np.sin(6*t)
    return f

t0 = 0
tmax = 37
dt = 0.05
t = np.arange(t0, tmax, dt)

```

```

def Y1(x, t):
    dx1_1 = x[1]
    dx1_2 = - w1*x[0] - g1*x[1] - F12(t)
    return dx1_1, dx1_2

def Y2(x, t):
    dx2_1 = x[1]
    dx2_2 = - w2*x[0] - g2*x[1] - F12(t)
    return dx2_1, dx2_2

def Y3(x, t):
    dx3_1 = x[1]
    dx3_2 = - w3*x[0] - g3*x[1] - F3(t)
    return dx3_1, dx3_2

x1 = odeint(Y1, x0, t)
x2 = odeint(Y2, x0, t)
x3 = odeint(Y3, x0, t)

y1_1 = x1[:, 0]
y1_2 = x1[:, 1]

y2_1 = x2[:, 0]
y2_2 = x2[:, 1]

y3_1 = x3[:, 0]
y3_2 = x3[:, 1]

plt.plot(y1_1, y1_2)

```

```
plt.grid(axis = 'both')
```

```
plt.plot(y2_1, y2_2)
```

```
plt.grid(axis = 'both')
```

```
plt.plot(y3_1, y3_2)
```

```
plt.grid(axis = 'both')
```

4.2. Получила фазовые портреты гармонического осциллятора (см. рис. 3.1, 3.2 и 3.3):

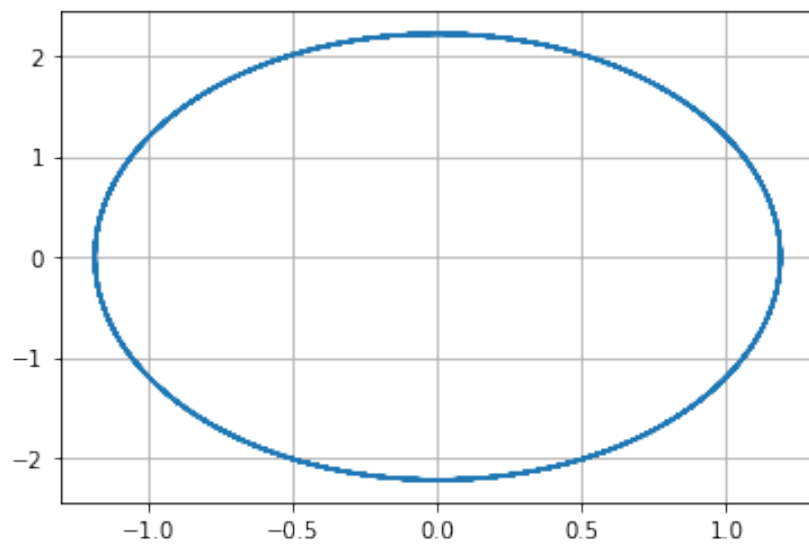


Figure 3.1: Колебания без затуханий и без действий внешней силы

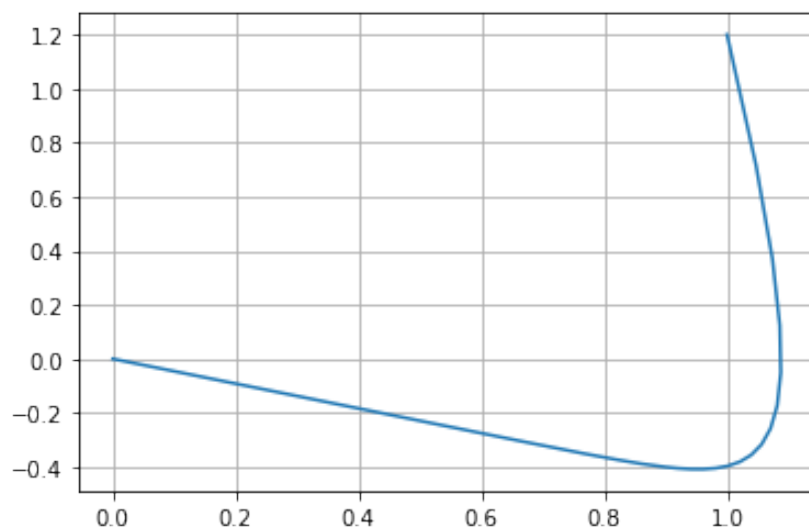


Figure 3.2: Колебания с затуханием и без действий внешней силы

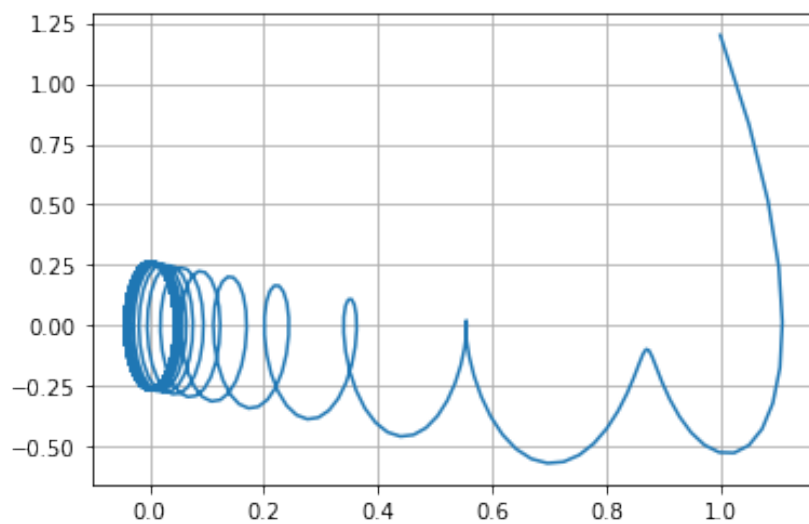


Figure 3.3: Колебания с затуханием и под действием внешней силы

## 4 Выводы

Построила модель гармонических колебаний с помощью Python.

## 5 Ответы на вопросы к лабораторной работе

### 1. Запишите простейшую модель гармонических колебаний

Простейшим видом колебательного процесса являются простые гармонические колебания, которые описываются уравнением  $x = x_m \cos(\omega t + \varphi_0)$ .

### 2. Дайте определение осциллятора

Осциллятор — система, совершающая колебания, то есть показатели которой периодически повторяются во времени.

### 3. Запишите модель математического маятника

Уравнение динамики принимает вид:

$$\frac{\partial^2 \alpha}{\partial t^2} + \frac{g}{L} \sin \alpha = 0$$

В случае малых колебаний полагают  $\sin(\alpha) \approx \alpha$ . В результате возникает линейное дифференциальное уравнение

$$\frac{\partial^2 \alpha}{\partial t^2} + \frac{g}{L} \alpha = 0$$

или

$$\frac{\partial^2 \alpha}{\partial t^2} + \omega^2 \alpha = 0$$

### 4. Запишите алгоритм перехода от дифференциального уравнения второго порядка к двум дифференциальным уравнениям первого порядка



Пусть у нас есть дифференциальное уравнение 2-го порядка:

$$\ddot{x} + \omega_0^2 x = f(t)$$

Для перехода к системе уравнений первого порядка сделаем замену (это метод Ранге-Кутты):

$$y = \dot{x}$$

Тогда получим систему уравнений:

$$\begin{cases} y = \dot{x} \\ \dot{y} = -\omega_0^2 x \end{cases}$$

##### *5. Что такое фазовый портрет и фазовая траектория?*

Фазовый портрет — это то, как величины, описывающие состояние системы (= динамические переменные), зависят друг от друга.

Фазовая траектория — кривая в фазовом пространстве, составленная из точек, представляющих состояние динамической системы в последовательные моменты времени в течение всего времени эволюции.