Отсчет по лабораторной работе №4

Модель гармонических колебаний

Динькиев Валерий

Содержание

[Цель работы 1](#_Toc65954493)

[Задание 1](#_Toc65954494)

[Выполнение лабораторной работы 1](#_Toc65954495)

[Выводы 8](#_Toc65954496)

[Ответы на вопросы к лабораторной работе 8](#_Toc65954497)

# Цель работы

Построить модель гармонических колебаний с помощью Python.

# Задание

**Вариант 16**

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

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы
2. Колебания гармонического осциллятора c затуханием и без действий внешней силы
3. Колебания гармонического осциллятора c затуханием и под действием внешней силы

На интервале (шаг 0,05) с начальными условиями

# Выполнение лабораторной работы

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

1.1. Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

Начальные условия:

Потери энергии отсутсвуют в системе

Собственная частота колебаний .

.

Правая часть уравнения .

1.2. Оформил начальные условия на Python и записал в виде функции решение правой части уравнения:

x\_0 = np.array([1.5, 1.1])  
w\_1 = 2 #частота колебаний  
g\_1 = 0.0 #затухание  
  
def F\_1(t):  
 f = 0  
 return f

1.3. Записал условия времени в программу на Python.

Решение ищем на интервале (шаг 0,05):

– начальный момент времени,

– предельный момент времени,

– шаг изменения времени.

Код на Python:

t\_0 = 0  
t\_max = 44  
dt = 0.05  
t = np.arange(t\_0, t\_max, dt)

1.4. Представил заданное уравнение второго порядка в виде системы двух уравнений первого порядка и записал в виде функции на Python:

def Y\_1(x, t):  
 dx1\_1 = x[1]  
 dx1\_2 = - w\_1\*x[0] - g\_1\*x[1] - F\_1(t)  
 return dx1\_1, dx1\_2

1.5. Запрограммировал решение системы уравнений:

x\_1 = odeint(Y\_1, x\_0, t)

1.6. Переписал отдельно в , а в :

y1\_1 = x\_1[:, 0]  
y1\_2 = x\_1[:, 1]

1.7. Описал построение фазового портрета гармонических колебаний:

plt.plot(y1\_1, y1\_2)

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

2.1. Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

Начальные условия:

Потери энергии в системе

Собственная частота колебаний .

.

Правая часть уравнения .

2.2. Оформил начальные условия на Python и записал в виде функции решение правой части уравнения:

x\_0 = np.array([1.5, 1.1])  
w\_2 = 3 #частота колебаний  
g\_2 = 3 #затухание  
  
def F\_2(t):  
 f = 0  
 return f

2.3. Условия интервала мы уже задали в начале.

2.4. Представил заданное уравнение второго порядка в виде системы двух уравнений первого порядка и записал в виде функции на Python:

def Y\_2(x, t):  
 dx2\_1 = x[1]  
 dx2\_2 = - w\_2\*x[0] - g\_2\*x[1] - F\_2(t)  
 return dx2\_1, dx2\_2

2.5. Запрограммировал решение системы уравнений:

x\_2 = odeint(Y\_2, x\_0, t)

2.6. Переписал отдельно в , а в :

y2\_1 = x\_2[:, 0]  
y2\_2 = x\_2[:, 1]

2.7. Описал построение фазового портрета гармонических колебаний с затуханием:

plt.plot(y2\_1, y2\_2)

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

3.1. Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

Начальные условия:

Потери энергии в системе

Собственная частота колебаний .

.

Правая часть уравнения .

3.2. Оформил начальные условия на Python и записал в виде функции решение правой части уравнения:

x\_0 = np.array([1.5, 1.1])  
w\_3 = 4 #частота колебаний  
g\_3 = 4 #затухание  
  
def F\_3(t):  
 f = math.sin(4\*t)  
 return f

3.3. Условия интервала мы уже задали в начале.

3.4. Представил заданное уравнение второго порядка в виде системы двух уравнений первого порядка и записал в виде функции на Python:

def Y\_3(x, t):  
 dx3\_1 = x[1]  
 dx3\_2 = - w\_3\*x[0] - g\_3\*x[1] - F\_3(t)  
 return dx3\_1, dx3\_2

3.5. Запрограммировал решение системы уравнений:

x\_3 = odeint(Y\_3, x\_0, t)

3.6. Переписал отдельно в , а в :

y3\_1 = x\_3[:, 0]  
y3\_2 = x\_3[:, 1]

3.7. Описал построение фазового портрета гармонических колебаний с затуханием и под действием внешней силы:

plt.plot(y3\_1, y3\_x2)

**4. Код на Python**

import math  
import numpy as np  
from scipy.integrate import odeint  
import matplotlib.pyplot as plt  
  
# начальные условия   
x\_0 = np.array([1.5, 1.1]) # вектор начальных условий  
w\_1 = 2 # частота колебаний  
g\_1 = 0.0 # затухание  
  
# интервал  
t\_0 = 0  
t\_max = 44  
dt = 0.05 # шаг изменения времени  
  
t = np.arange(t\_0, t\_max, dt)  
  
# функция для вычисления правой части уравнения  
def F\_1(t):  
 f = 0  
 return f  
  
# заданное уравнение в виде системы двух уравнений первого порядка  
def Y\_1(x, t):  
 dx1\_1 = x[1]  
 dx1\_2 = - w\_1\*x[0] - g\_1\*x[1] - F\_1(t)  
 return dx1\_1, dx1\_2  
  
# решение системы уравнений  
x\_1 = odeint(Y\_1, x\_0, t)  
  
y1\_1 = x\_1[:, 0]  
y1\_2 = x\_1[:, 1]  
  
# построение фазового портрета гармонических колебаний   
plt.plot(y1\_1, y1\_2)  
plt.grid(axis = 'both')  
  
w\_2 = 3.0 # частота колебаний  
g\_2 = 3.0 # затухание  
  
# функция для вычисления правой части уравнения  
def F\_2(t):  
 f = 0  
 return f  
  
# заданное уравнение в виде системы двух уравнений первого порядка  
def Y\_2(x, t):  
 dx2\_1 = x[1]  
 dx2\_2 = - w\_2\*x[0] - g\_2\*x[1] - F\_2(t)  
 return dx2\_1, dx2\_2  
  
# решение системы уравнений  
x\_2 = odeint(Y\_2, x\_0, t)  
y2\_1 = x\_2[:, 0]  
y2\_2 = x\_2[:, 1]  
  
# построение фазового портрета гармонических колебаний с затуханием  
plt.plot(y2\_1, y2\_2)  
plt.grid(axis = 'both')  
  
w\_3 = 4.0 # частота колебаний  
g\_3 = 4.0 # затухание  
  
# функция для вычисления правой части уравнения  
def F\_3(t):  
 f = math.sin(4\*t)  
 return f  
  
# заданное уравнение в виде системы двух уравнений первого порядка  
def Y\_3(x, t):  
 dx3\_1 = x[1]  
 dx3\_2 = - w\_3\*x[0] - g\_3\*x[1] - F\_3(t)  
 return dx3\_1, dx3\_2  
  
# решение системы уравнений  
x\_3 = odeint(Y\_3, x\_0, t)  
y3\_1 = x\_3[:, 0]  
y3\_2 = x\_3[:, 1]  
  
# построение фазового портрета гармонических колебаний с затуханием и под действием внешней силы  
plt.plot(y3\_1, y3\_2)  
plt.grid(axis = 'both')

4.1. Построил фазовый портрет гармонического осциллятора (см. рис. 1, 2 и 3):

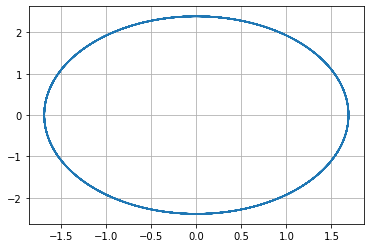


Figure 1: Колебания гармонического осциллятора без затуханий и без действий внешней силы

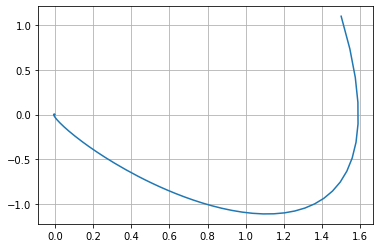


Figure 2: Колебания гармонического осциллятора c затуханием и без действий внешней силы

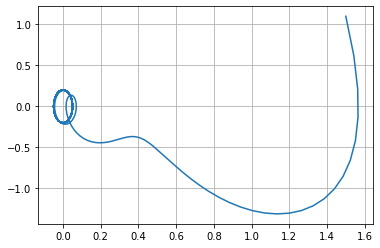


Figure 3: Колебания гармонического осциллятора c затуханием и под действием внешней силы

# Выводы

Построил модель гармонических колебаний с помощью Python.

# Ответы на вопросы к лабораторной работе

*1. Запишите простейшую модель гармонических колебаний*

.

*2. Дайте определение осциллятора*

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

*3. Запишите модель математического маятника*

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

4.1. Дифф. уравнение 2-го порядка

4.2. Сделаем замену.

4.3 Получаем систему уравнений:

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

Фазовый портрет — общая картина поведения системы, возникающая, если множество различных решений (соответствующих различным начальным условиям) изобразить на одной фазовой плоскости.

Фазовая траектория — гладкая кривая в фазовой плоскости, отвечающая решению уравнения движения как функции времени.