МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ (ШКОЛА)

Департамент математического и компьютерного моделирования

ОТЧЁТ

к лабораторной работе №3 по дисциплине

«Математическое и копмьютерное моделирование»

Направление подготовки
01.03.02 «Прикладная математика и информатика»

Выполнил студент гр.

Б9121-01.03.02сп

Держапольский Ю.В. ________
(Ф.И.О.) *(подпись)*

Профессор д.ф.-м. н.

Пермяков М. С. ________
(Ф.И.О.) *(подпись)*

« 4 » мая 2024 г.

г. Владивосток

2024

Содержание

1. Введение

2. Математическая модель

Движение математического маятника во времени можно описать на Декартовой плоскости (x, y) в зависимости от времени t. Предположим, что сам маятник является материальной точкой, длина невесомой нити L (м) и ускорение свободного падения $\left(g \approx 9.8\frac{\text{м}}{\text{с}^2}\right)$ постоянны. Тогда движение маятника вместо пары (x, y) можно описать углом отклонения от вертикальной оси α.

Воспользуемся уравнением моментов для материальной точки:

$$J\frac{d^2\alpha}{dt^2} = M,$$

где J – момент инерции относительно оси, M – момент сил. Для материальной точки: $J = mL^2$, где m – масса маятника.

На маятник влияет сила тяжести, равная $F = mg$, но на движение будет влиять только её составляющая, касательная к движению, поэтому $F = -mg\sin\alpha$. Эта сила перпендикулярна к нити, поэтому момент силы равен:

$$M = FL = -mgL\sin\alpha.$$

Подставляя в уравнение моментов, и приводя слагаемые:

$$mL^2\frac{d^2\alpha}{dt^2} + mgL\sin\alpha = 0 \Rightarrow \frac{d^2\alpha}{dt^2} + \frac{g}{L}\sin\alpha = 0.$$

Получили дифференциальное уравнение второго порядка, которое описывает угол отклонения маятника в зависимости от времени.

Обозначим $w^2 = \dfrac{g}{L}$ и добавим начальные условия для получения единственного решения:

$$\begin{cases} \ddot{\alpha} + w^2\sin\alpha = 0, \\ \dot{\alpha}(0) = \alpha_1, \quad \alpha(0) = \alpha_0. \end{cases}$$

Известно, что при небольших значениях углов $\sin x \approx x$, поэтому из нелинейной модели сделаем линейную: $\ddot{\alpha} + w^2\alpha = 0$.

2.1. Модели с внешними силами

Одной из внешних сил является трение. Оно зависит от скорости с некоторым коэффициентом k. Получим модель с трением: $\ddot{\alpha} + k\dot{\alpha} + w^2\alpha = 0$.

Также внешними силами могут быть вынужденные колебания: $A_f \sin\big(w_f t\big)$ с амплитудой A_f и частотой w_f. Модель с вынужденными колебаниями:
$\ddot{\alpha} + w^2\alpha = A_f \sin\big(w_f t\big)$.

Если на маятник будут действовать сразу обе предыдущие силы, то уравнение будет выглядеть так: $\ddot{\alpha} + k\dot{\alpha} + w^2\alpha = A_f \sin\big(w_f t\big)$.

3. Анализ модели

Проанализируем линейную модель:

$$\begin{cases} \ddot{\alpha} + w^2\alpha = 0, \\ \dot{\alpha}(0) = \alpha_1, \quad \alpha(0) = \alpha_0. \end{cases}$$

Известен вид аналитического решения: $\alpha(t) = C_1 \cos(wt) + C_2 \sin(wt)$. Находим частное решение: $\alpha(t) = \alpha_0 \cos(wt) + \dfrac{a_1}{w} \sin(wt)$. Можно найти такой угол φ и константу ρ, что $\rho \sin\varphi = \alpha_0$ и $\rho\cos\varphi = \dfrac{a_1}{w}$, поэтому по формуле синуса суммы: $\alpha(t) = \rho\sin(wt + \varphi)$.

Значит результатом решения будет синусоида в зависимости от начальных параметров.

4. Вычислительные эксперименты

4.1. Алгоритм

Для компьютерного вычисления будем использовать метод Рунге-Кутты, с помощью которого получим численное решение системы дифференциальных уравнений с заданными параметрами. После чего построим графики решений на координатной и фазовой плоскостях.

Для модели с силой трения и вынужденными колебаниями для каждой частоты колебаний найдём амплитуду

4.2. Программа

Для расчётов и визуализации был использован язык Python с библиотеками numpy и matplotlib.

```python
import numpy as np
import math


def runge_kutta(function, y0: float, a: float, b: float, h: float):
    num = math.ceil((b - a) / h)
    x_a = np.linspace(a, b, num=num, endpoint=False)
    y_a = [y0] * num

    for i in range(num - 1):
        k0 = function(x_a[i], y_a[i])
        k1 = function(x_a[i] + h / 2, y_a[i] + h * k0 / 2)
        k2 = function(x_a[i] + h / 2, y_a[i] + h * k1 / 2)
        k3 = function(x_a[i] + h, y_a[i] + h * k2)
        y_a[i + 1] = y_a[i] + h / 6 * (k0 + 2 * k1 + 2 * k2 + k3)

    return x_a, np.array(y_a)

def right(t, ab):
    return np.array([
        ab[1],
```

```python
            -w**2 * ab[0]
    ])

def model(y0, right):
    x_, y_ = runge_kutta(right, y0, t0, tn, (tn-t0)/n)
    y_ = y_.T

    return x_, y_

t0, tn = 0, 15
n = 10000

g = 9.8
L = 1
w = np.sqrt( g / L )

k = 0.01
Af = 1
wf = 0.5

init = [np.pi/100, 0]
x_, y_ = model(init, right)
```

4.3. Результаты

5. Заключение