МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6
по дисциплине «**Математическое моделирование сложных систем**»

на тему: **«Анализ переходных процессов при исследовании динамических моделей технических систем»**

Выполнил: студент гр. ИП-32
Коваленко А.И
Принял: доцент
Трохова Т.А

Гомель 2023

<u>Цель работы</u>: получить навыки выполнения анализа переходных процессов в динамических моделях с графической интерпретацией полученных результатов.

<u>Задача 4</u> **Исследование математической модели груза на жестком стержне**

Исходными данными для задачи являются:

m – масса груза
l – длина стержня
a – расстояние до демпфера
D – диаметр пружины
d – диаметр проволоки пружины
i – число витков пружины
G – модуль упругости
α - коэффициент вязкого сопротивления движения демпфера

Таблица 4.1 - Таблица исходных данных

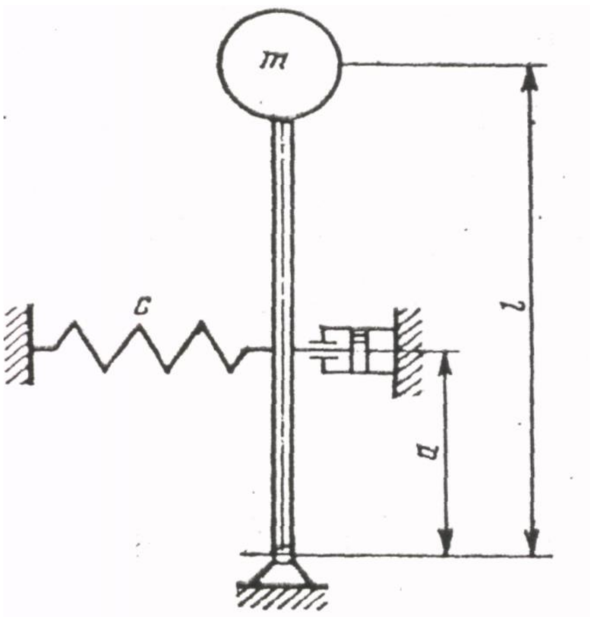
a(м)	l (м)	D(мм)	d (мм)	i	m (кг)	α	φ_0	$t_к$ (с)	Варьируемый параметр	N варианта
0,2	0,5	50	5	5	5	300	0,05	1	m	1
0,22	0,55	60	6	6	6	210	0,06	1,6	l	2
0,23	0,53	65	6,2	5	4	212	0,051	0,5	α	3
0,05	0,6	55	6,1	6	8	310	0,061	1,1	a	4

Для всех вариантов заданий $G=80*10^9$

Таблица 4.2 - Таблица значений варьируемых параметров

m	1,1	1,4	2,0	2,3	2,9	3,3	3,8	4,1	4,5
l	0,5	0,65	0,78	0,89	1,0	1,15	1,29	1,35	1,5
α	210	250	290	325	360	385	400	420	450
a	0,05	0,09	0,12	0,15	0,2	0,25	0,29	0,32	0,35

Описание математической модели



Груз массой m укреплен на абсолютно жестком безынерционном стержне длиной l, который удерживается в равновесии пружиной и демпфером. Демпфер имеет линейную характеристику трения $f = \alpha \cdot \dot{x}$.

В соответствии с принципом Даламбера составим дифференциальное уравнение движения груза, как уравнение равновесия при отклонении стержня на некоторый малый угол φ

$$mgl\varphi = ml^2\ddot{\varphi} - ca^2\varphi - a^2\alpha\dot{g} = 0$$

Обозначив

$$2n = a^2\alpha/\left(ml^2\right) \qquad p^2 = \left(ca^2 - mgl\right)/ml^2$$

запишем дифференциальное уравнение в виде

$$\ddot{\varphi} + 2n\dot{\varphi} + p^2\varphi = 0$$

$c = Gd^4/8D^3i$ - жесткость пружины

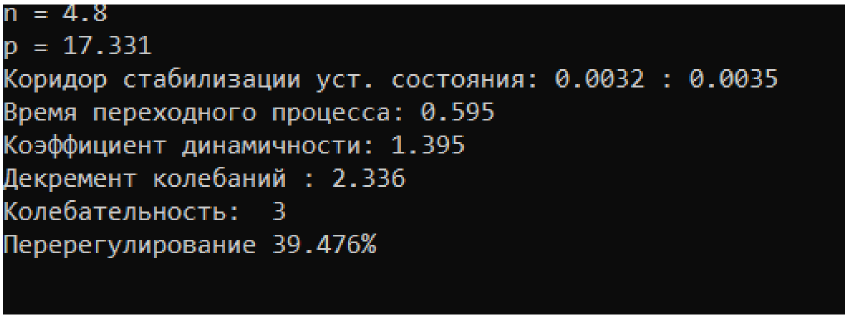
$p = \sqrt{\left(ca^2 - mgl\right)/ml^2}$ -частота собственных колебаний

$n = a^2\alpha/\left(2ml^2\right)$ - приведенный коэффициент сопротивления демпфера

$F(t) = F_0\sin(wt)$ – возмущающая сила, действующая на систему. Все параметры функции подобрать самостоятельно.

1. Рассчитать значение функции перемещения динамической системы без воздействия начальных значений перемещения и скорости с учетом ступенчатого воздействия (функция Хевисайда). Построить график этой функций.
2. Для функции перемещения п.1 рассчитать следующие параметры переходного процесса:
   - коридор стабилизации установившегося состояния;
   - время переходного процесса;
   - коэффициент динамичности;
   - декремент колебаний;
   - колебательность;
   - перерегулирование.

   Выполнить графическую интерпретацию первых двух результатов.



```
n = 4.8
p = 17.331
Коридор стабилизации уст. состояния: 0.0032 : 0.0035
Время переходного процесса: 0.595
Коэффициент динамичности: 1.395
Декремент колебаний : 2.336
Колебательность:  3
Перерегулирование 39.476%
```

Рисунок 1 – Результат выполнения программы

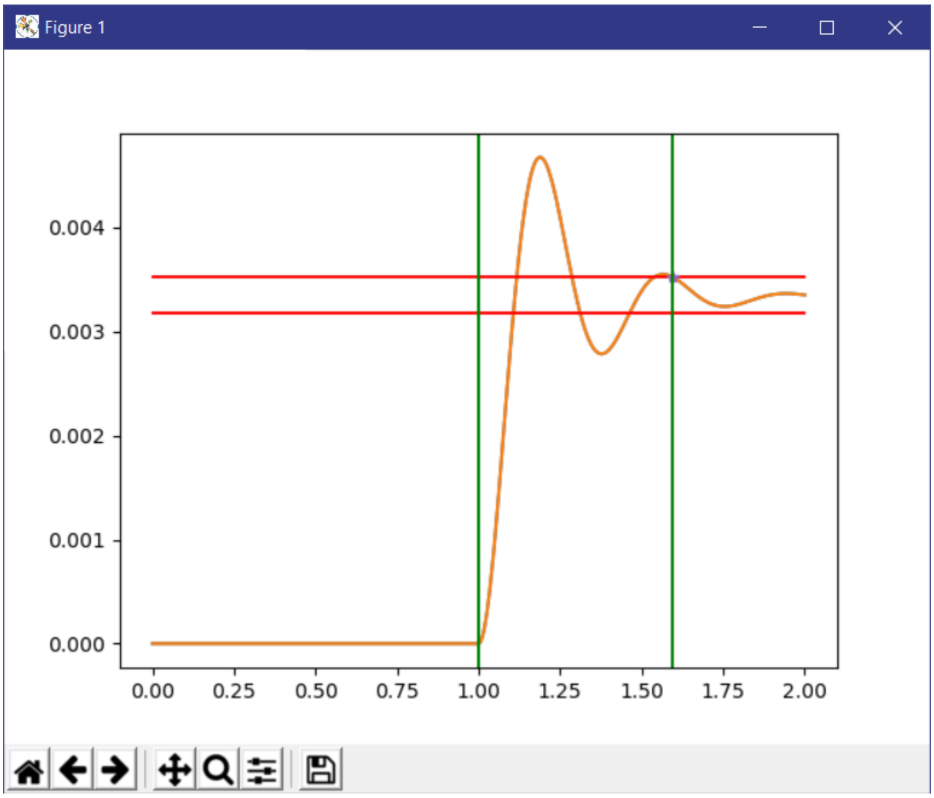


Рисунок 2 – График функции

Листинг программы:

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

tk = 2
s = 500

m = 5
a = 0.2
l = 0.5
D = 0.05
d = 0.005
i = 5

alpha = 300
G = 80 * 10 ** 9
g = 9.81
c = (G * d ** 4) / (8 * D ** 3 * i)
p = np.sqrt((c * (a**2) - m * g*l) / (m * (l**2)))
n = alpha*(a**2) / (2*m * (l**2))
t = np.linspace(0, tk, s)
y0 = [0, 0]

def function(y, t):
    if t > 1:
        F = 1
    else:
        F = 0
    return [y[1], -2*n * y[1] - p**2 * y[0] + F]

Y = odeint(function, y0, t)

print("n = " f"{n:.{1}f}")
print("p = " f"{p:.{3}f}")

moveArr = Y[:, 0]
n = np.linspace(0, tk, s)
value = moveArr[len(moveArr) - 1]
topLine = value + 0.05 * value
bottomLine = value - 0.05 * value

print("Коридор стабилизации уст. состояния: " f"{bottomLine:.{4}f}" " : " f"{topLine:.{4}f}")


tlast = 0
Ylast = 0
for i in range(1,s):
    tmp = Y[i, 0]
    if tmp <= bottomLine or tmp >= topLine :
        Ylast = Y[i, 0]
        tlast = n[i]

vibrancy = 0

for i in range(1, s):
    if (n[i] < tlast and Y[i, 0] < Y[i + 1, 0] and Y[i + 1, 0] > Y[i + 2, 0]) or (n[i] < tlast
and Y[i, 0] > Y[i + 1, 0] and Y[i + 1, 0] < Y[i + 2, 0]):
        t[vibrancy + 1] = np.abs(Y[i, 0] - value)
        vibrancy = vibrancy + 1

Amax = max(moveArr) - value
D = t[1] / t[2]
dynamicCoeff = 1 + Amax / value;
maxValY = Amax + value;
re_regulation = (max(moveArr) - value) / value * 100
```

```
print("Время переходного процесса: " f"{tlast - 1:.{3}f}")
print("Коэффициент динамичности: " f"{dynamicCoeff:.{3}f}")
print("Декремент колебаний : " f"{D:.{3}f}")
print("Колебательность: ", vibrancy)
print( "Перерегулирование " f"{re_regulation:.{3}f}%")

plt.plot(t, Y[:, 0])
plt.plot(t, Y[:, 0], [0, tk], [topLine, topLine], [0, tk], [bottomLine, bottomLine])
plt.plot([0, tk], [topLine, topLine], [0, tk], [bottomLine, bottomLine], color='r')
plt.plot(tlast,Ylast,'*')
plt.axvline(x = 1, color='g')
plt.axvline(x = tlast, color='g')
plt.show()
```

Вывод: Математическая модель предназначена предсказать поведение реального объекта, но всегда представляет собой ту или иную степень его идеализации. С помощью математических методов описывается, как правило, идеальный объект или процесс, построенный на этапе моделирования. Таким образом, в данной лабораторной работе была разобрана математическая модель и построена зависимость от воздействия функции Хевисайда.