**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 № 8

по дисциплине «Математическое моделирование сложных систем»

на тему: «**Моделирование динамических систем методами САУ**»

Выполнил: студент гр. ИП-31

Коваленко А.И.

Принял: ст. преподаватель

Трохова Т.А

Гомель 2023

**Цель работы:** Получить навыки моделирования САУ с использованием Python и в пакете Xcos, научиться применять функции Python для анализа моделей САУ, выполнять графическую интерпретацию полученных результатов.

**Практическая часть**

1. Создать модели типовых звеньев САУ:

- интегрирующего,

- дифференцирующего (реального),

- апериодического второго порядка,

- колебательного.

Параметры типовых звеньев задаются самостоятельно.

from control.matlab import \*

import matplotlib.pyplot as plt

# Интегрир

nom = [1]

den = [1,0]

W = tf(nom, den)

y,t = step(W)

plt.figure(0)

plt.plot(t,y)

print(W)

# Реальн фиг

plt.figure(1)

nom = [1,0]

den = [1,1]

W = tf(nom, den)

y,t = step(W)

plt.plot(t,y)

print(W)

# Апериодич

plt.figure(2)

nom = [-1,0]

den = [1,1]

W = tf(nom, den)

y,t = step(W)

plt.plot(t,y)

print(W)

# Колебательные

plt.figure(3)

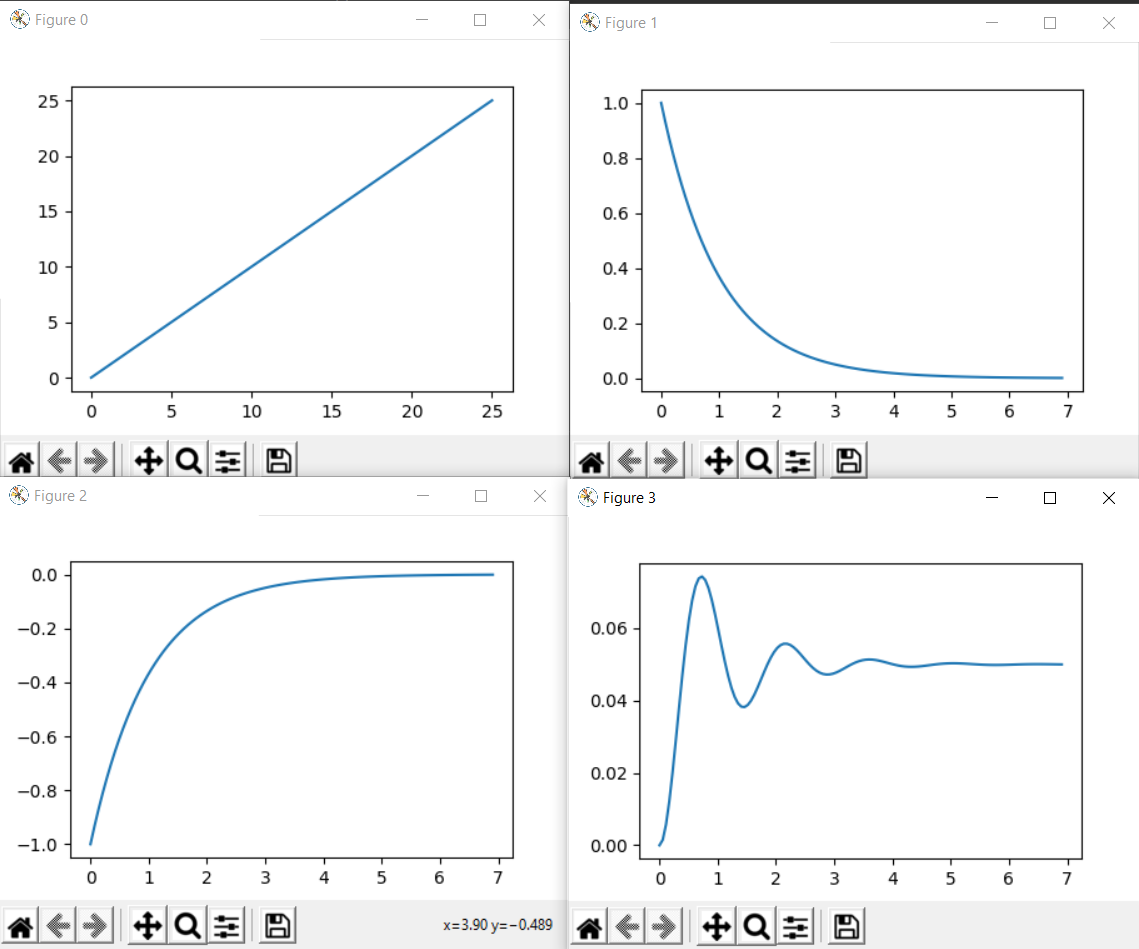
W = tf([1], [1, 2, 20])

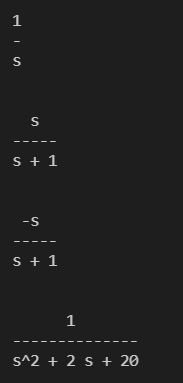
y,t = step(W)

plt.plot(t,y)

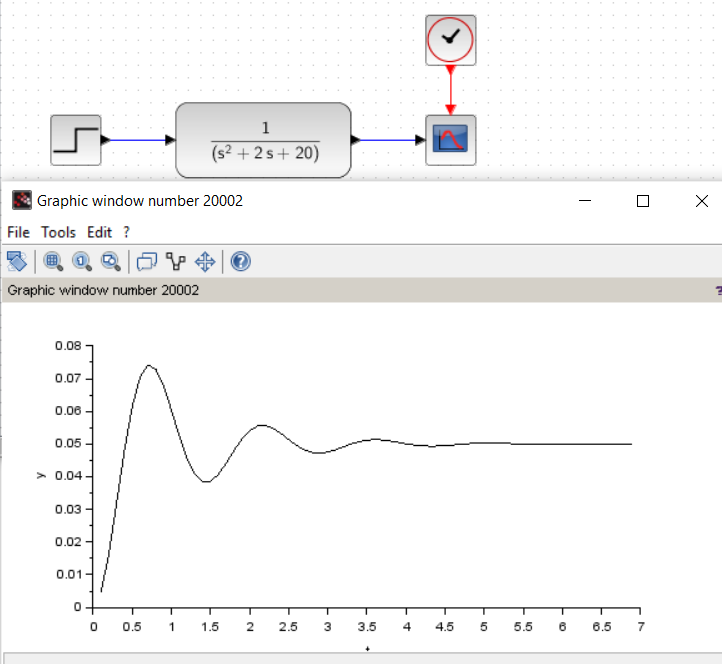
plt.show()

print(W)

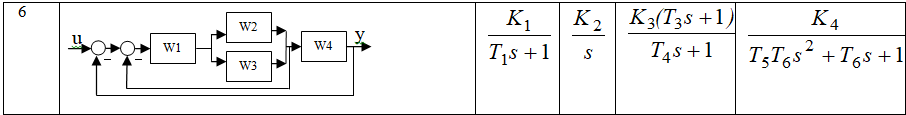


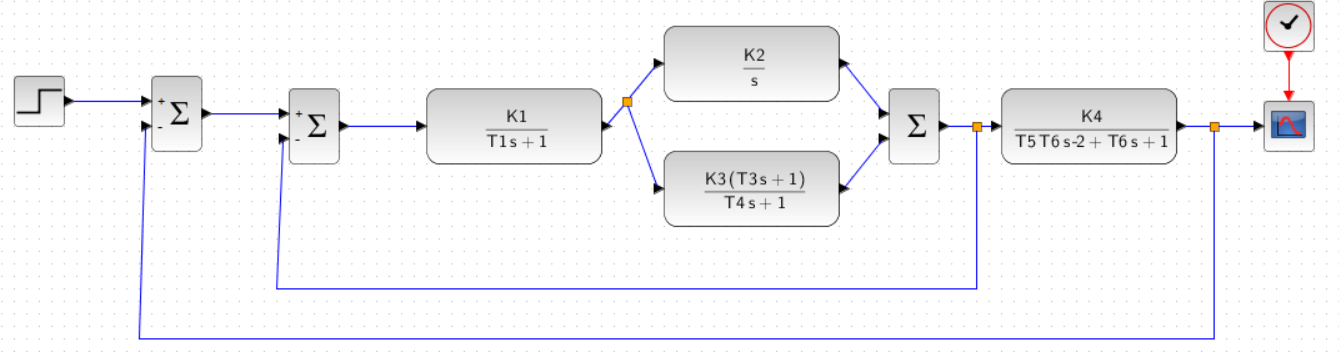


1. Для каждого звена определить переходную функцию (построить графики переходного процесса).
2. Для колебательного звена получить переходную характеристику с использованием Xcos. Сравнить переходную характеристику с полученной в п.2



1. С использованием Python и Xcos создать модель, вид которой приведен в приложении А. Получить временную характеристику модели в Python и Xcos, сравнить их. Параметры системы подобрать самостоятельно.





from control.matlab import \*

import matplotlib.pyplot as plt

K1=0.1

K2=10

K3=2

T1=3

T3=0.6

T4=2

K4=1

T5=4

T6=2

#W1 = K1/(T1\*s + 1)

#W2 = K2/s

#W3 = K3\*(T3\*s + 1)/(T4\*s + 1)

W1 = tf([K1], [T1, 1])

W2 = tf([K2], [1, 0])

W3 = tf([T3, 1], [T4, 1])

W4 = tf([K4], [T5\*T6, T6, 1])

W44 = W1\*(W2+W3)

W5 = 1

W6 = feedback(W44, W5)

W7 = W6 \* W4

W8 = 1

W9 = feedback(W7, W8)

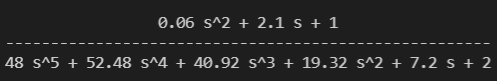
print(W9)

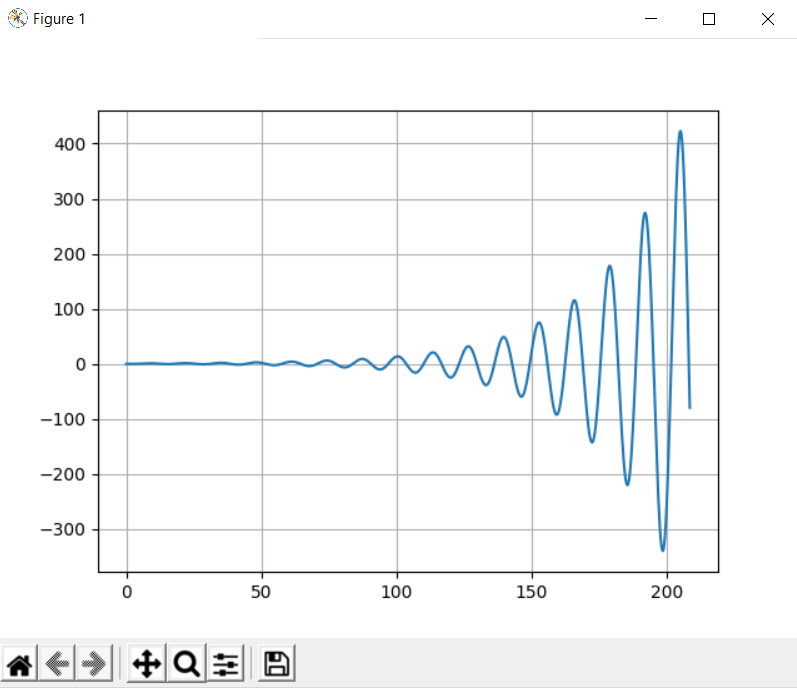
y,t = step(W9)

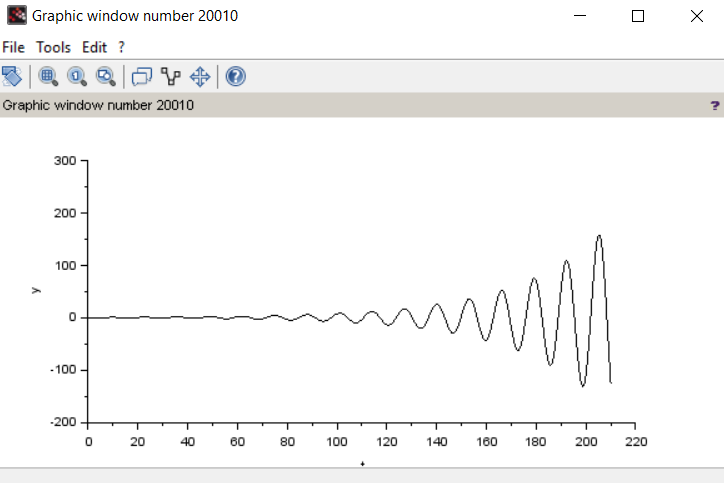
plt.plot(t, y)

plt.grid()

plt.show()







1. Для колебательного звена получить амплитудно-частотную характеристику и построить диаграмму Bode.
2. Для колебательного звена определить по АЧХ значение максимальной амплитуды и частоты, при которой она достигается, исследовать явление резонанса.

from control.matlab import \*

import matplotlib.pyplot as plt

import numpy as np

K1=0.1

K2=10

K3=2

T1=3

T3=0.6

T4=2

K4=1

T5=4

T6=2

W1 = tf([K1], [T1, 1])

W2 = tf([K2], [1, 0])

W3 = tf([T3, 1], [T4, 1])

W4 = tf([K4], [T5\*T6, T6, 1])

W44 = W1\*(W2+W3)

W5 = 1

W6 = feedback(W44, W5)

W7 = W6 \* W4

W8 = 1

W9 = feedback(W7, W8)

plt.figure(1)

tt = np.linspace(0, 210, 200)

y,t = step(W9,tt)

plt.plot(t, y)

plt.grid()

num = [1]

den = [1, 2, 20]

function = tf(num,den)

mag, phase, omega  = bode(function)

plt.plot()

plt.grid()

plt.figure(2)

maximum = np.argmax(mag)

for i in range(len(mag)):

    if(mag[i] == max(mag)):

        maxA = mag[i]

        maxW = omega[i]

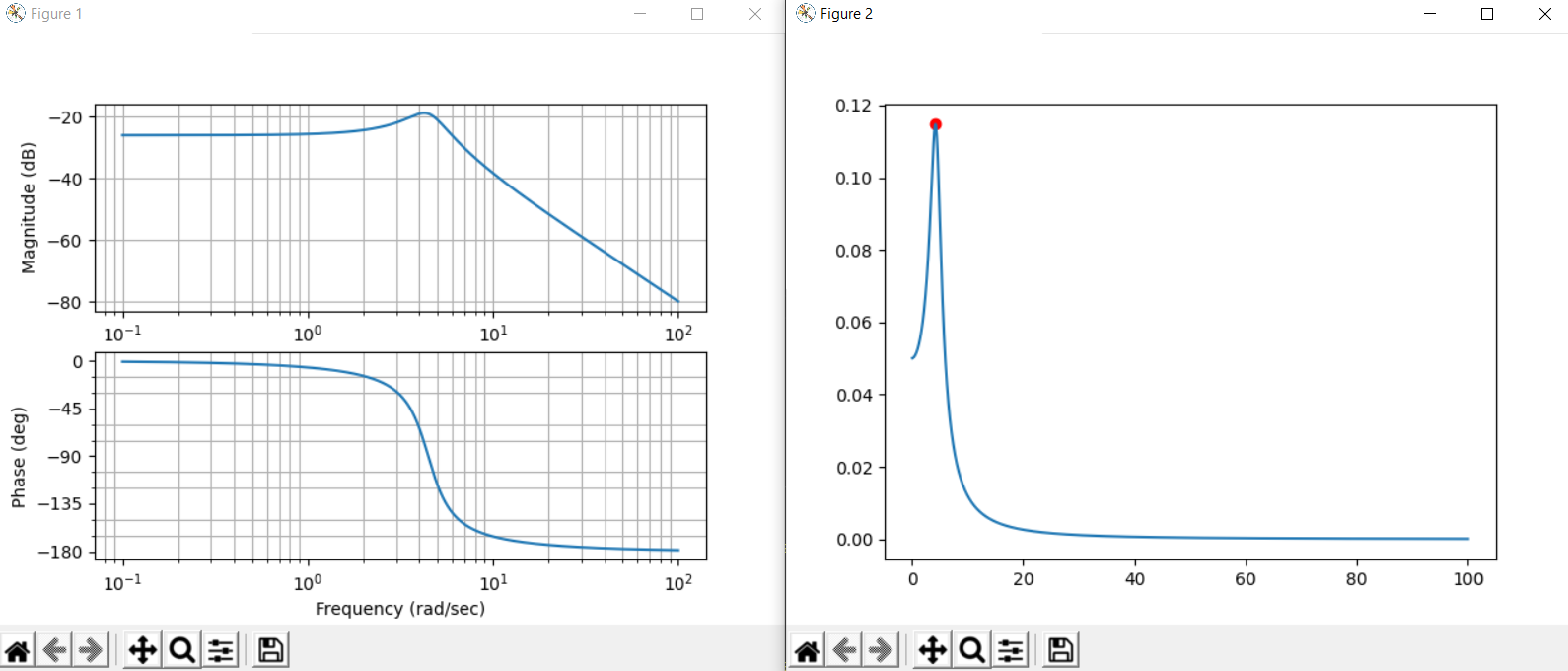
print("Амплитуда = ", maxA)

print("Частота = ", maxW)

plt.plot(omega,mag)

plt.scatter(omega[maximum], mag[maximum], c="red")

plt.show()





1. Система описывается передаточной функцией вида:



Необходимо подобрать такие значения A, B, C, чтобы система была

а) устойчива с колебаниями и без колебаний,

б) неустойчива с колебаниями и без колебаний.

Построить график переходной характеристики, подтверждающие правильность выбора коэффициентов.

Сделать вывод о том, как влияет параметр K на вид переходной характеристики системы.

from control.matlab import \*

from control import \*

import matplotlib.pyplot as plt

K=20

A=1

B=5

C=5

function = tf([K],[A,B,C])

coeff = pole(function)

print(coeff)

y,x = step(function)

string = "Устойчива без колебаний: "+ str(coeff)

plt.figure(0)

plt.title(string)

plt.plot(x,y)

#---------------------------------------------------------------

K=10

A=5

B=1

C=1

function = tf([K],[A,B,C])

coeff = pole(function)

print(coeff)

y,x = step(function)

string = "Устойчива c колебаниями: "+ str(coeff)

plt.figure(1)

plt.title(string)

plt.plot(x,y)

#---------------------------------------------------------------

K=10

A=1

B=-1

C=1

function = tf([K],[A,B,C])

coeff = pole(function)

print(coeff)

y,x = step(function)

string = "Неустойчива c колебаниями: "+ str(coeff)

plt.figure(2)

plt.title(string)

plt.plot(x,y)

 #--------------------------------------------------------------

K=5

A=1

B=-3

C=1

function = tf([K],[A,B,C])

coeff = pole(function)

print(coeff)

y,x = step(function)

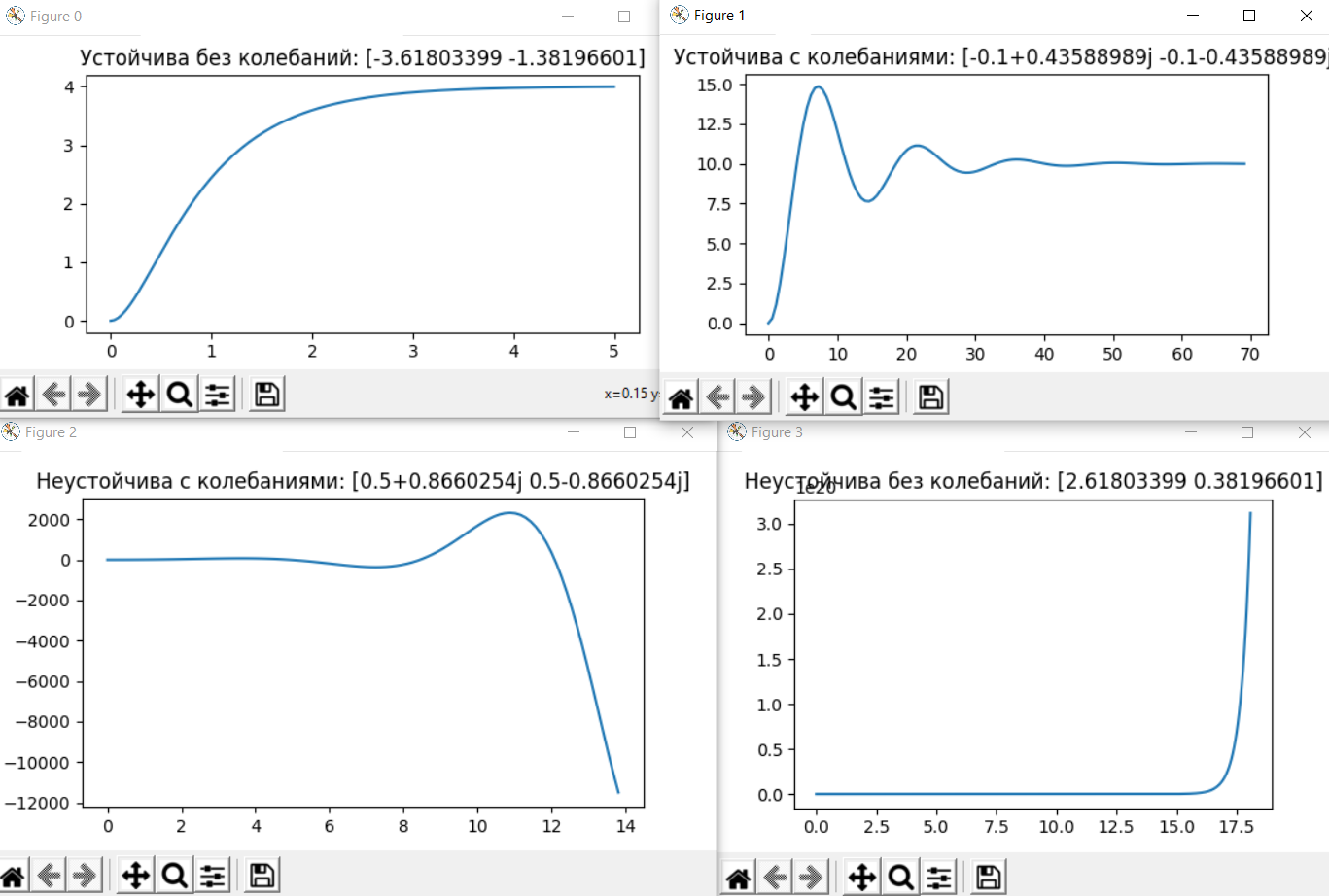
string = "Неустойчива без колебаний: "+ str(coeff)

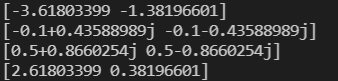
plt.figure(3)

plt.title(string)

plt.plot(x,y)

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 разобраны схемы реализации математической модели в пакете Xcos программы Scilab, результат выполнения которых были сравнены с аналогичными решениями в Python.