МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

ИСПОЛНИТЕЛЬНЫЙ ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

По курсу: «Теория автоматического управления и системы автоматического управления»

Тема: «Устойчивость стационарных систем автоматического управления»

Выполнил:	Мархель Д.Г
Группа:	Э-13м-23
Проверил:	Дегтярев Д.А.

Москва, 2023 г.

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное
учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

ПРЕДВАРИТЕЛЬНЫЙ ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

По курсу: «Теория автоматического управления и системы автоматического управления»

Тема: «Устойчивость стационарных систем автоматического управления»

Выполнил:	Мархель Д.Г
Группа:	Э-13м-23
Проверил:	Дегтярев Д.А.

Москва, 2023 г.

9 Вариант

Исходные данные

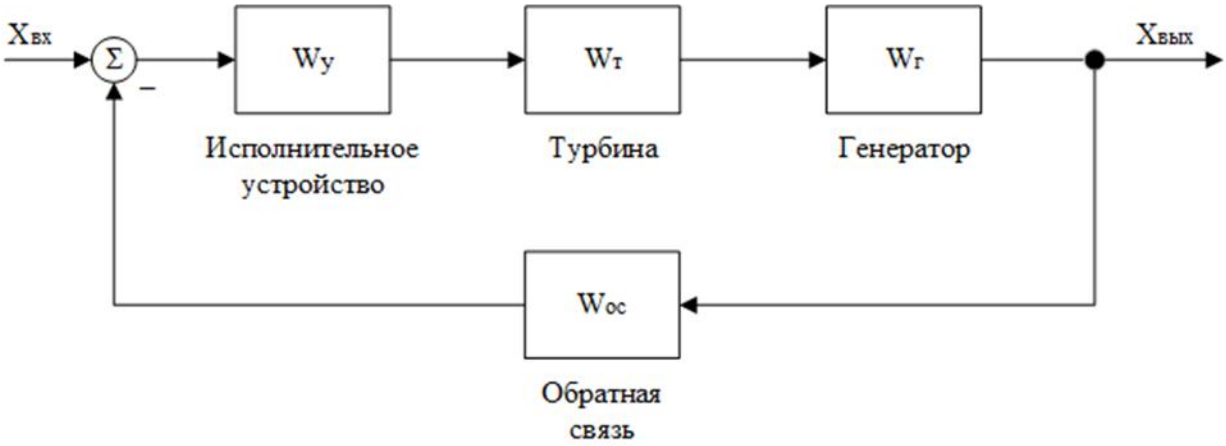


Рисунок 1 – Структура САУ

Таблица 1 – Перечень используемых элементов

Наименование элемента		Условное обозначение	Передаточная функция
обратная связь:	апериодическая жесткая	$W_{ос}$	$\dfrac{k_{ос}}{T_{ос} \cdot p + 1}$
генератор		$W_г$	$\dfrac{1}{T_г \cdot p + 1}$
турбина	гидравлическая	$W_т$	$\dfrac{0{,}01 \cdot T_{гт} \cdot p + 1}{0{,}05 \cdot T_г \cdot p + 1}$
исполнительное устройство		$W_у$	$\dfrac{k_у}{T_у \cdot p + 1}$

Таблица 2 – Параметры звеньев

№ Варианта	$k_у$	$T_у$, с	$T_г$, с	Турбина	$T_{гт}$, с	$T_{пт}$, с	$k_{пт}$	Обратная связь	$k_{ос}$	$T_{ос}$, с
9	22	20.0	5.0	Гидро-	2.0	-	-	АЖ	22	-

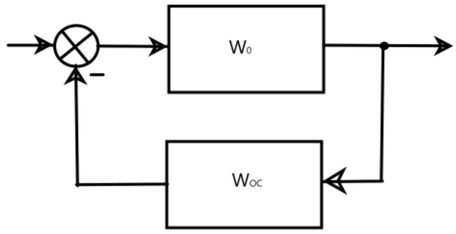


Рисунок 2 -Упрощенная схема САУ с обратной связью

Расчет передаточной характеристики для разомкнутой системы:

$$W_{\text{эквнез}} = W_0 = W_{\text{у}} * W_{\text{г}} * W_{\text{т}} * (W_{\text{ос}}) = \frac{k_{\text{у}}}{T_{\text{у}} \cdot p + 1} * \frac{1}{T_{\text{г}} \cdot p + 1} * \frac{0{,}01 \cdot T_{\text{гт}} \cdot p + 1}{0{,}05 \cdot T_{\text{г}} \cdot p + 1} * (\frac{k_{\text{ос}}}{T_{\text{ос}} \cdot p + 1}) =$$

$$= \frac{(9{,}68p + 484)}{25p^3 + 106{,}25p^2 + 25{,}25p + 1}$$

Расчет передаточной характеристики для замкнутой системы:

$$W_{\text{экв}} = \frac{W_0}{1 + W_{\text{ос}} * W_0} = \frac{W_{\text{у}} * W_{\text{г}} * W_{\text{т}}}{1 + W_{\text{ос}} * W_{\text{у}} * W_{\text{г}} * W_{\text{т}}} =$$

$$= \frac{\frac{k_{\text{у}}}{T_{\text{у}} \cdot p + 1} * \frac{1}{T_{\text{г}} \cdot p + 1} * \frac{0{,}01 \cdot T_{\text{гт}} \cdot p + 1}{0{,}05 \cdot T_{\text{г}} \cdot p + 1}}{1 + \frac{k_{\text{ос}}}{T_{\text{ос}} \cdot p + 1} * \frac{k_{\text{у}}}{T_{\text{у}} \cdot p + 1} * \frac{1}{T_{\text{г}} \cdot p + 1} * \frac{0{,}01 \cdot T_{\text{гт}} \cdot p + 1}{0{,}05 \cdot T_{\text{г}} \cdot p + 1}} =$$

$$= \frac{\frac{0{,}44p + 22}{25p^3 + 106{,}25p^2 + 25{,}25p + 1}}{1 + \frac{9{,}68p + 484}{25p^3 + 106{,}25p^2 + 25{,}25p + 1}} = \frac{0{,}44p + 22}{25p^3 + 106{,}25p^2 + 34{,}93p + 485}$$

Нахождение переходной характеристики:

$$h(t) = L^{-1}\left\{\frac{W_{\text{экв}}(p)}{p}\right\} = 0$$

Невозможно построить график переходной характеристики для данной функции.

Выявление значений полюсов передаточной функции замкнутой системы

$\mathrm{D(p) + K(p)} = 25p^3 + 106{,}25p^2 + 34{,}93p + 485 = 0$, где корни уравнения принимают как λ, тогда полюса будут иметь следующий вид:

$\lambda_1 = -4{,}8$ – вещественный корень, левый

$\lambda_2 = 0{,}275 + 1{,}991i$ – комплексный корень

$\lambda_3 = 0{,}275 - 1{,}991i$ – комплексный корень

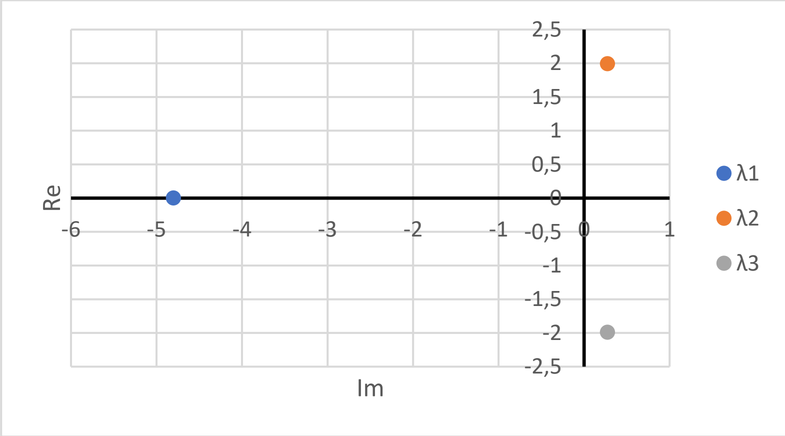


Рисунок 4 – Комплексная плоскость

Исходя из полученных значений мы не можем с точностью определить устойчива система или нет, так как комплексные корни - правые.

Критерий устойчивости Гурвица

$$\Delta_j = \begin{vmatrix} b_1 & b_3 & b_5 & & b_{2i-1} \\ b_0 & b_2 & b_4 & & b_{2i-2} \\ 0 & b_1 & b_3 & \cdots & b_{2i-3} \\ \cdots & \cdots & \cdots & & \cdots \\ 0 & \cdots & b_{i-2} & & b_i \end{vmatrix},\ i = 1,2,3$$

$$\Delta_1 = 106{,}25 > 0$$

$$\Delta_2 = 106{,}25 * 34{,}93 - 25 * 485 = -8413 < 0$$

$$\Delta_3 = \begin{vmatrix} 106{,}25 & 485 & 0 \\ 25 & 34{,}93 & 0 \\ 0 & 106{,}25 & 485 \end{vmatrix} = 106{,}25 * 34{,}93 * 485 - 25 * 485 * 485 = -4080638 < 0$$

Система неустойчива.

Критерий устойчивости Рауса

Таблица 3 – Таблица Рауса

Вспомога-тельные коэффи-циенты	Но-мер стро-ки	Номер столбца		
		1	2	3
	1	$c_{11} = 25$	$c_{12} = 34{,}93$	0
	2	$c_{21} = 106{,}25$	$c_{22} = 485$	0
$r_3 = 0{,}235$	3	$c_{31} = -79{,}18$	0	0
$r_4 = -1{,}34$	4	$c_{41} = 485$	0	0

r_4 – отрицательное. Следовательно система неустойчива.

Критерий устойчивости Михайлова

$$D(j\omega) = 25(j\omega)^3 + 106{,}25(j\omega)^2 + 34{,}93(j\omega) + 485.$$

$$D(j\omega) = U(\omega) + jV(\omega).$$

$$U(\omega) = 485 - 106{,}25\omega^2$$

$$jV(\omega) = 34{,}93\omega - 25\omega^3 = \omega(34{,}93 - 25\omega^2)$$

$$\omega_1 = 0$$

$$\omega_2 = 1{,}182$$

$$\omega_3 = 2{,}136$$

Таблица 4 – Таблица значений годографа по устойчивости Михайлова

ω	0	1,182	2,136	3	∞
U	485	336,55	0	-471,25	-∞
V	0	0	-169,02	-570,21	-∞

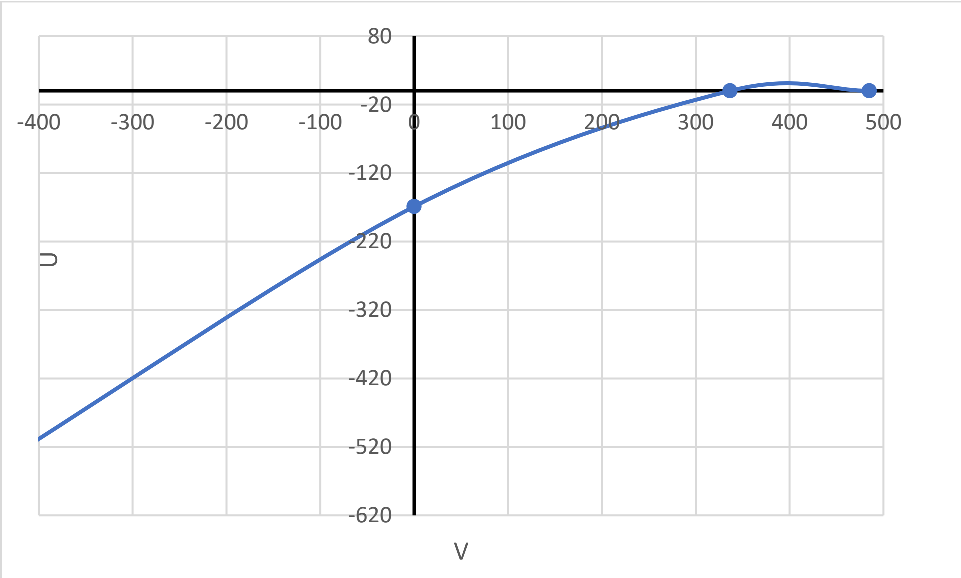


Рисунок 5 – годограф Михайлова

Наблюдая нарушение числа и последовательности пройденных кривой Михайлова квадрантов координатной плоскости, мы можем утверждать что система неустойчива.

Критерий устойчивости Найквиста

$$W_{\text{раз}}{}^{-1}(j\omega) = \frac{1}{W_{\text{раз}}(j\omega)} = \frac{b_0(j\omega)^{\mathrm{m}} + b_1(j\omega)^{\mathrm{m}-1} + \dots + b_{\mathrm{m}-1}j\omega + b_{\mathrm{m}}}{a_0p^{\mathrm{n}} + a_1p^{\mathrm{n}-1} + \dots + a_{\mathrm{n}-1}p + a_{\mathrm{n}}}$$

Характеристическое уравнение разомкнутой АСУ:

$$W_{\text{раз}}{}^{-1}(j\omega) = \frac{25(j\omega)^3 + 106.25(j\omega)^2 + 25{,}25(j\omega) + 1}{9{,}68p + 484}$$

$$U(\omega) = \frac{1 - 106{,}25(j\omega)^2}{9{,}68p + 484}$$

$$jV(\omega) = \frac{25{,}25(j\omega) - 25(j\omega)^3}{9{,}68p + 484}$$

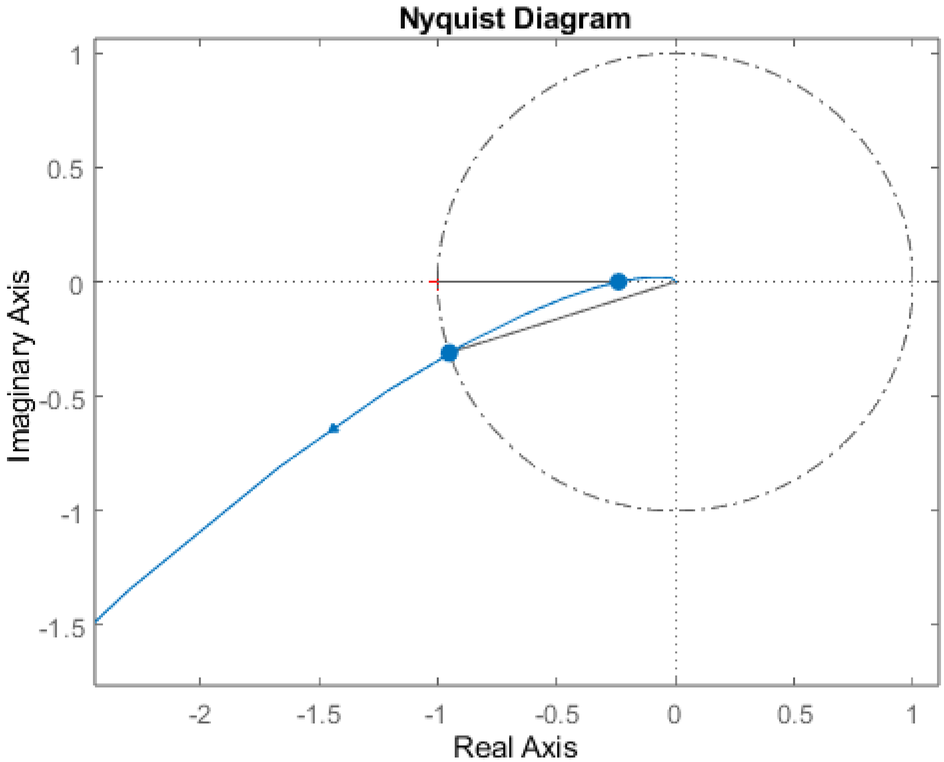


Рисунок 6 – Устойчивость по Найквисту

Ввиду того, что годограф не охватывает точку (-1;0) мы можем утверждать, что система неустойчива.

Определение запаса устойчивости при помощи критерия Найквиста

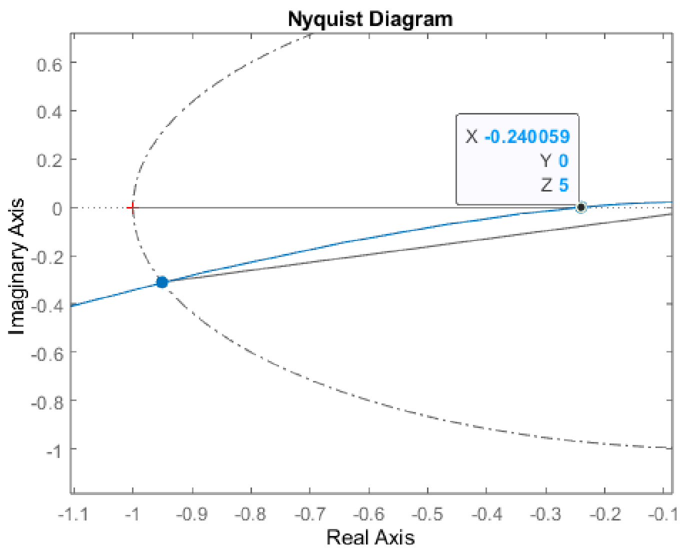


Рисунок 7 – Запас устойчивости по Найквисту

Запас устойчивости по фазе γ определяется углом от точки пересечения годографа с единичной окружностью, до отрицательной части вещественной оси, а запас устойчивости по амплитуде равен:

$$h = 1 - 0{,}24 = 0{,}76$$

Определение запаса устойчивости при помощи логарифмических частотных характеристик

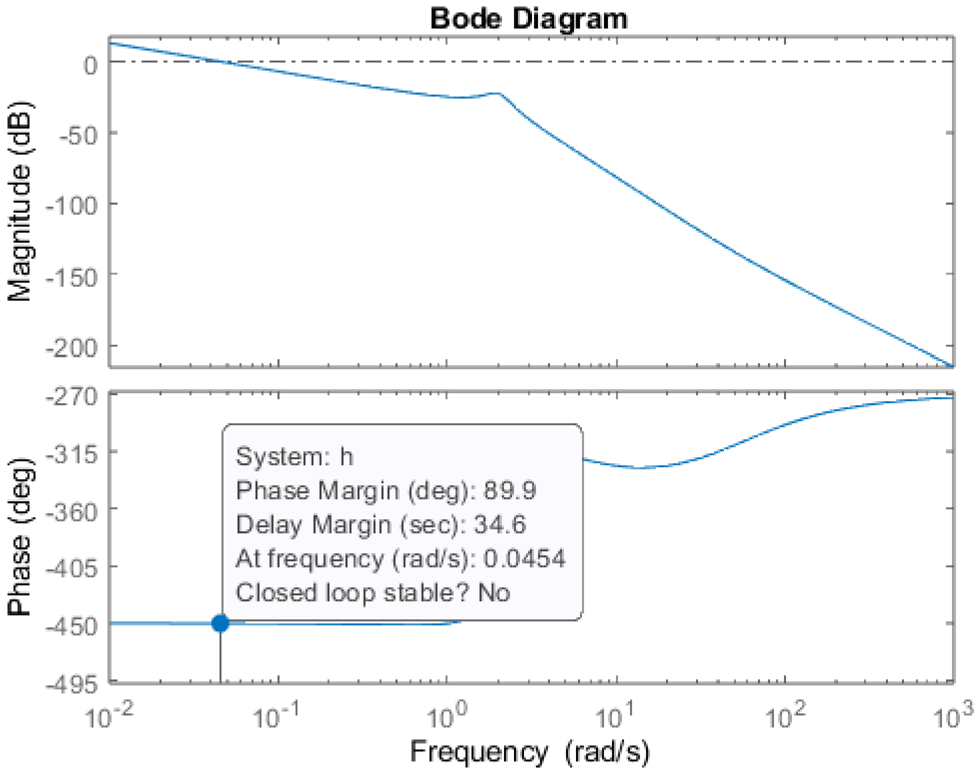


Рисунок 8 – Запас устойчивости по логарифмическим частотным характеристикам

Запас устойчивости по фазе отрицателен, а запас устойчивости по амплитуде отсутствует.

Определение диапазона устойчивости при помощи критерия устойчивости Рауса-Гурвица

$$\Delta = \begin{vmatrix} 106{,}25 & 1 \\ 25 & 25{,}25 + k \end{vmatrix} > 0$$

$$106{,}25 \cdot (25{,}25 + k) - 1 \cdot 25 > 0$$

$$k > \frac{1 \cdot 25}{106{,}25} - 25{,}25 > -25{,}014$$

Определение диапазона устойчивости методом D-разбиения

$$\mathrm{D}(\mathrm{j}\omega) = 106{,}25(\mathrm{j}\omega)^2 + 485$$

$$K(\mathrm{j}\omega) = 25(\mathrm{j}\omega)^3 + 34{,}93(\mathrm{j}\omega)$$

$$F(j\omega) = D_{\text{раз}}(j\omega) + K_{\text{раз}}(j\omega) - k$$

$$k = D_{\text{раз}}(j\omega) + K_{\text{раз}}(j\omega)$$

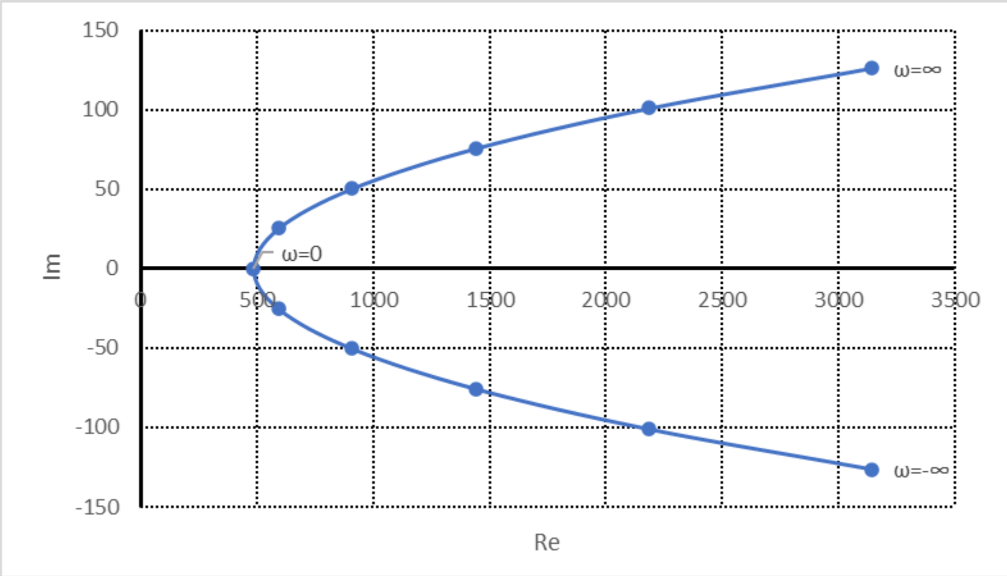


Рисунок 9 – График D-разбиения

Система неустойчива.

Вывод: Использовав различные методы проверки устойчивости системы, мы определили ее неустойчивость по всем параметрам.

Проверка системы при нахождении на границе устойчивости

Примем $k_{\text{ос}} = 5{,}29$, тогда:

$$W_{\text{экв}} = \frac{W_0}{1 + W_{\text{ос}} * W_0} = \frac{W_{\text{у}} * W_{\text{г}} * W_{\text{т}}}{1 + W_{\text{ос}} * W_{\text{у}} * W_{\text{г}} * W_{\text{т}}} =$$

$$= \frac{\frac{k_{\text{у}}}{T_{\text{у}} \cdot p + 1} * \frac{1}{T_{\text{г}} \cdot p + 1} * \frac{0{,}01 \cdot T_{\text{гт}} \cdot p + 1}{0{,}05 \cdot T_{\text{г}} \cdot p + 1}}{1 + \frac{k_{\text{ос}}}{T_{\text{ос}} \cdot p + 1} * \frac{k_{\text{у}}}{T_{\text{у}} \cdot p + 1} * \frac{1}{T_{\text{г}} \cdot p + 1} * \frac{0{,}01 \cdot T_{\text{гт}} \cdot p + 1}{0{,}05 \cdot T_{\text{г}} \cdot p + 1}} =$$

$$= \frac{0{,}44p + 22}{25p^3 + 106{,}25p^2 + 25{,}25p + 1 + (0{,}44p + 22) * 8}$$

Нахождение переходной функции:

$$h(t) = L^{-1}\left\{\frac{W_{\text{экв}}(p)}{p}\right\} = 0$$

Невозможно построить график переходной характеристики.

Выявление значений полюсов передаточной функции замкнутой системы

$\text{D(p)} = 25p^3 + 106{,}25p^2 + 27{,}57p + 117{,}38 = 0$, где корни уравнения принимают как λ, тогда полюса будут иметь следующий вид:

$\lambda_1 = -4{,}250$ – вещественный корень, левый

$\lambda_2 = 0{,}0001 + 1{,}05i$ – комплексный корень

$\lambda_3 = 0{,}0001 - 1{,}05i$ – комплексный корень

Судя по выбранному значению $k_{\text{ос}}$ система находится на границе устойчивости.

Рисунок 11 – Комплексная плоскость

По отображённым на плоскости значениям мы видим, что система находится на границе устойчивости.

Критерий устойчивости Гурвица

$$\Delta_j = \begin{vmatrix} b_1 & b_3 & b_5 & & b_{2i-1} \\ b_0 & b_2 & b_4 & & b_{2i-2} \\ 0 & b_1 & b_3 & \cdots & b_{2i-3} \\ \cdots & \cdots & \cdots & & \cdots \\ 0 & \cdots & b_{i-2} & & b_i \end{vmatrix},\ i = 1,2,3$$

$$\Delta_1 = 106{,}25 > 0$$

$$\Delta_2 = 106{,}25 * 27{,}57 - 25 * 117{,}38 = -4{,}38 < 0$$

$$\Delta_3 = \begin{vmatrix} 106{,}25 & 117{,}38 & 0 \\ 25 & 27{,}57 & 0 \\ 0 & 106{,}25 & 117{,}38 \end{vmatrix} = 410 * 43{,}77 * 177 - 100 * 177 * 177 = -514 < 0$$

Система неустойчива.

Критерий устойчивости Рауса

Таблица 6 – Измененная Таблица Рауса

Вспомогательные коэффициенты	Номер строки	Номер столбца		
		1	2	3
	1	$c_{11} = 25$	$c_{12} = 27{,}57$	0
	2	$c_{21} = 106{,}25$	$c_{22} = 117{,}38$	0
$r_3 = 0{,}24$	3	$c_{31} = -0{,}04$	0	0
$r_4 = -2656$	4	$c_{41} = 117$	0	0

r_4 – положительное. Следовательно система устойчива.

Критерий устойчивости Михайлова

$$D(j\omega) = 25(j\omega)^3 + 106{,}25(j\omega)^2 + 27{,}57(j\omega) + 117{,}38.$$

$$D(j\omega) = U(\omega) + jV(\omega).$$

$$U(\omega) = 117{,}38 - 106{,}25\omega^2$$

$$jV(\omega) = 27{,}57\omega - 25\omega^3 = \omega(27{,}57 - 25\omega^2)$$

$$\omega_1 = 0$$

$$\omega_2 = 1{,}05014$$

$$\omega_3 = 1{,}051$$

Таблица 7 – Измененная Таблица значений годографа по устойчивости Михайлова

ω	0	0,5	1,05014	1,051	∞
U	117,38	90,8175	0,208	0	-∞
V	0	10,66	0	-0,04	-∞

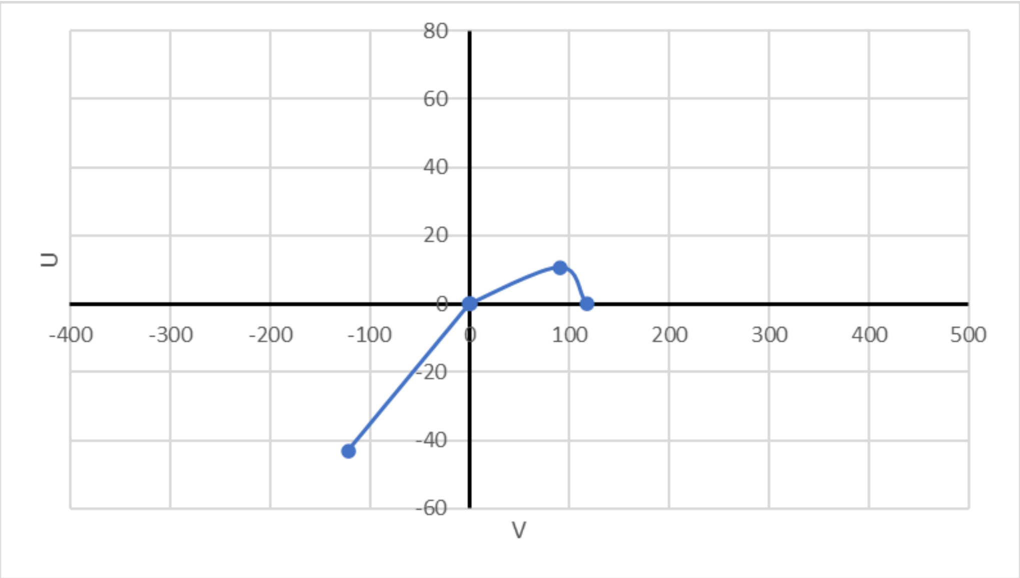


Рисунок 12 – годограф Михайлова

Наблюдая нарушение числа и последовательности пройденных кривой Михайлова квадрантов координатной плоскости, мы можем утверждать что система неустойчива.

Критерий устойчивости Найквиста

$$W_{\text{раз}}(j\omega) = \frac{a_0 p^{\text{n}} + a_1 p^{\text{n}-1} + \cdots + a_{\text{n}-1} p + a_{\text{n}}}{b_0 (j\omega)^{\text{m}} + b_1 (j\omega)^{\text{m}-1} + \cdots + b_{\text{m}-1} j\omega + b_{\text{m}}}$$

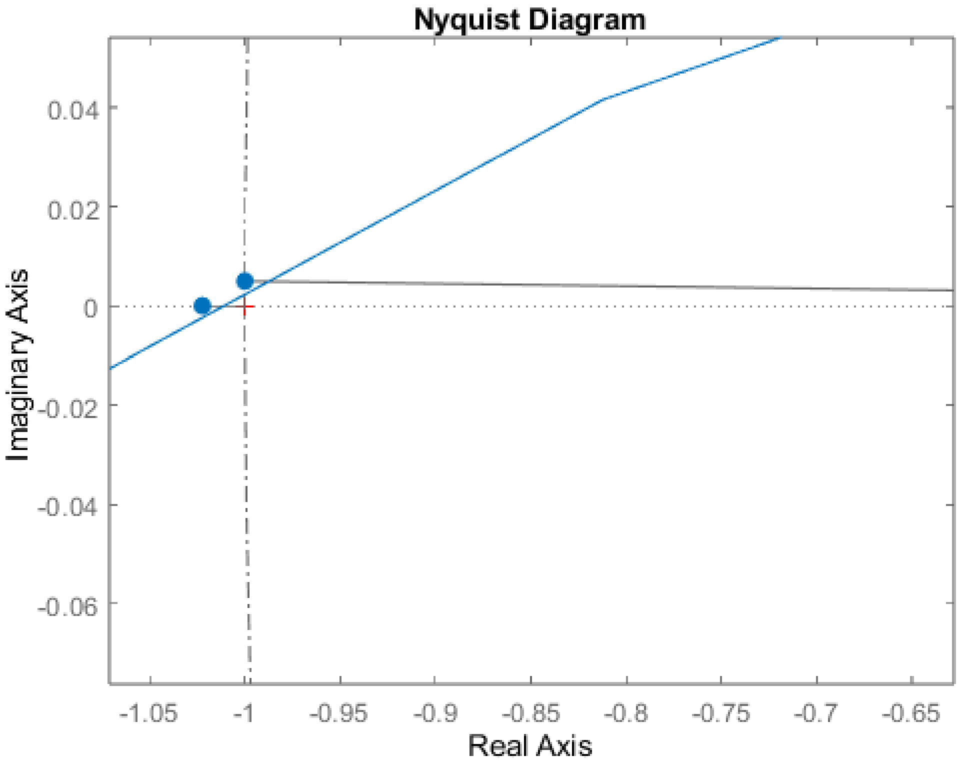


Рисунок 6 – Устойчивость по Найквисту

Ввиду того, что годограф охватывает точку (-1;0) мы можем утверждать, что система устойчива.

Определение запаса устойчивости при помощи критерия Найквиста

Запас устойчивости по фазе γ определяется углом от точки пересечения годографа с единичной окружностью, до отрицательной части вещественной оси, а запас устойчивости по амплитуде равен:

$$h = 1{,}022 - 1 = 0{,}022$$

Определение запаса устойчивости при помощи логарифмических частотных характеристик

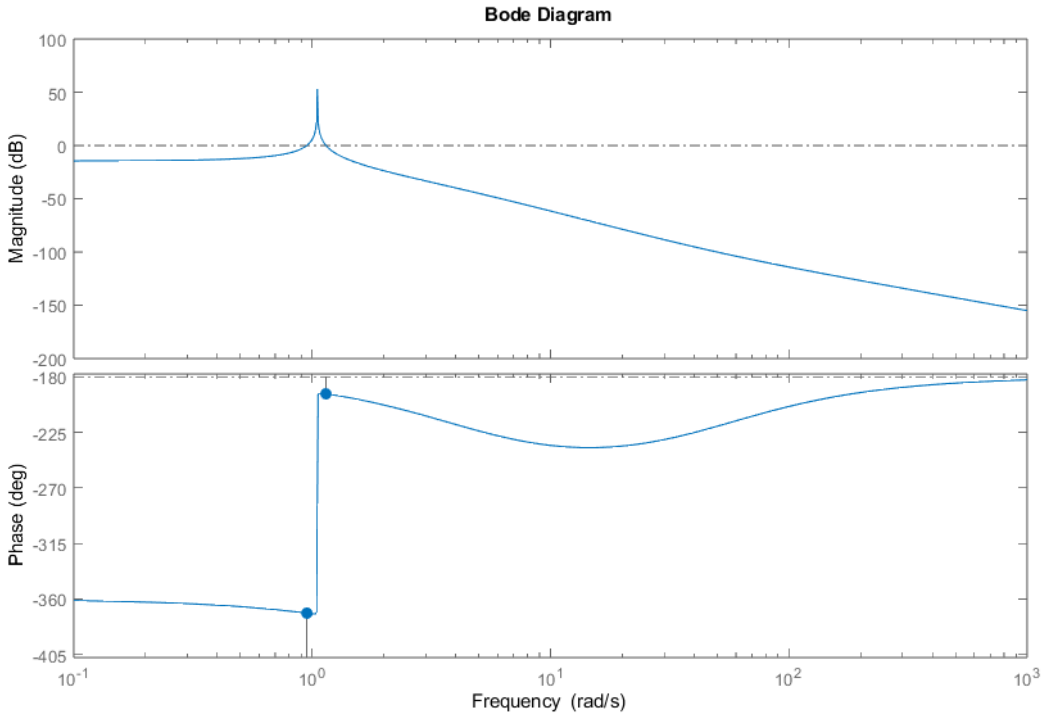


Рисунок 8 – Запас устойчивости по логарифмическим частотным характеристикам

Запас устойчивости по фазе и запас устойчивости по амплитуде не отрицательны.

Определение диапазона устойчивости при помощи критерия устойчивости Рауса-Гурвица

$$\Delta = \begin{vmatrix} 106{,}25 & 1 \\ 25 & 25{,}25 + k \end{vmatrix} > 0$$

$$106{,}25 \cdot (25{,}25 + k) - 1 \cdot 25 > 0$$

$$k > \frac{1 \cdot 25}{106{,}25} - 25{,}25 > -25{,}014$$

Определение диапазона устойчивости методом D-разбиения

$$\mathrm{D}(\mathrm{j\omega}) = 106{,}25(\mathrm{j\omega})^2 - 117{,}38$$

$$K(\mathrm{j\omega}) = 25(\mathrm{j\omega})^3 - 27{,}57(\mathrm{j\omega})$$

$$F(j\omega) = D_{\text{раз}}(j\omega) + K_{\text{раз}}(j\omega) - k$$

$$k = D_{\text{раз}}(j\omega) + K_{\text{раз}}(j\omega)$$

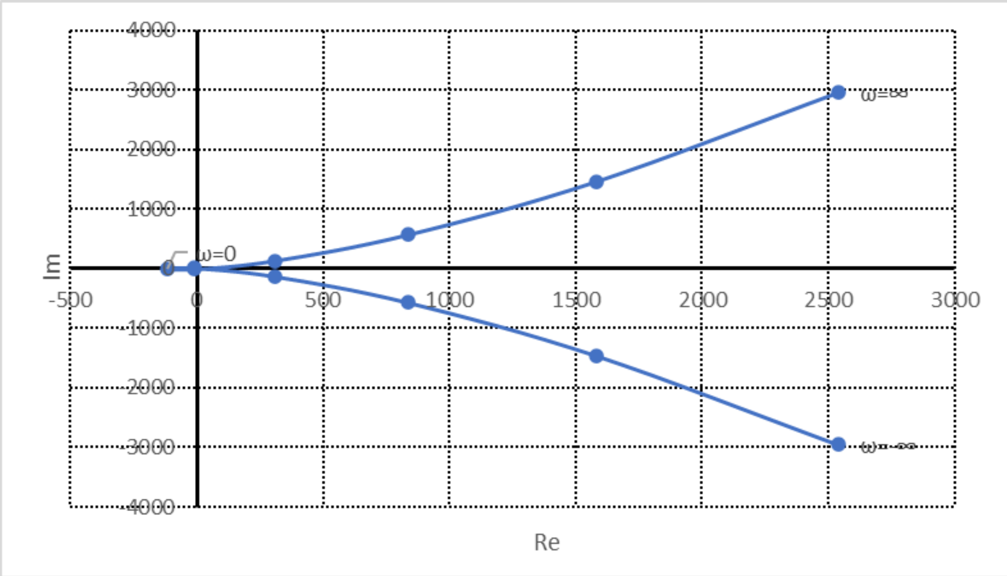


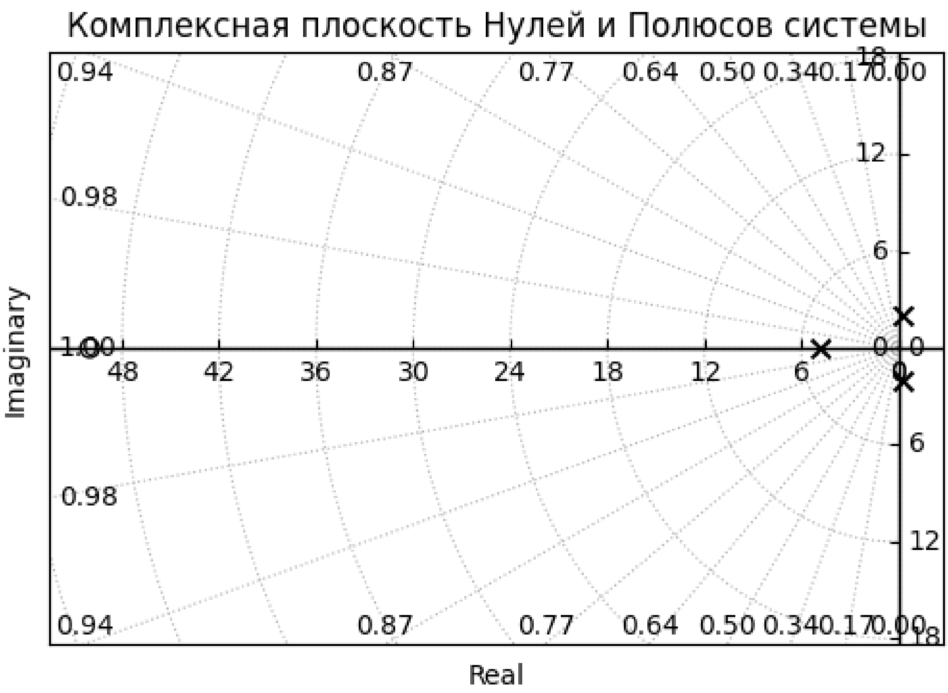
Рисунок 9 – График D-разбиения

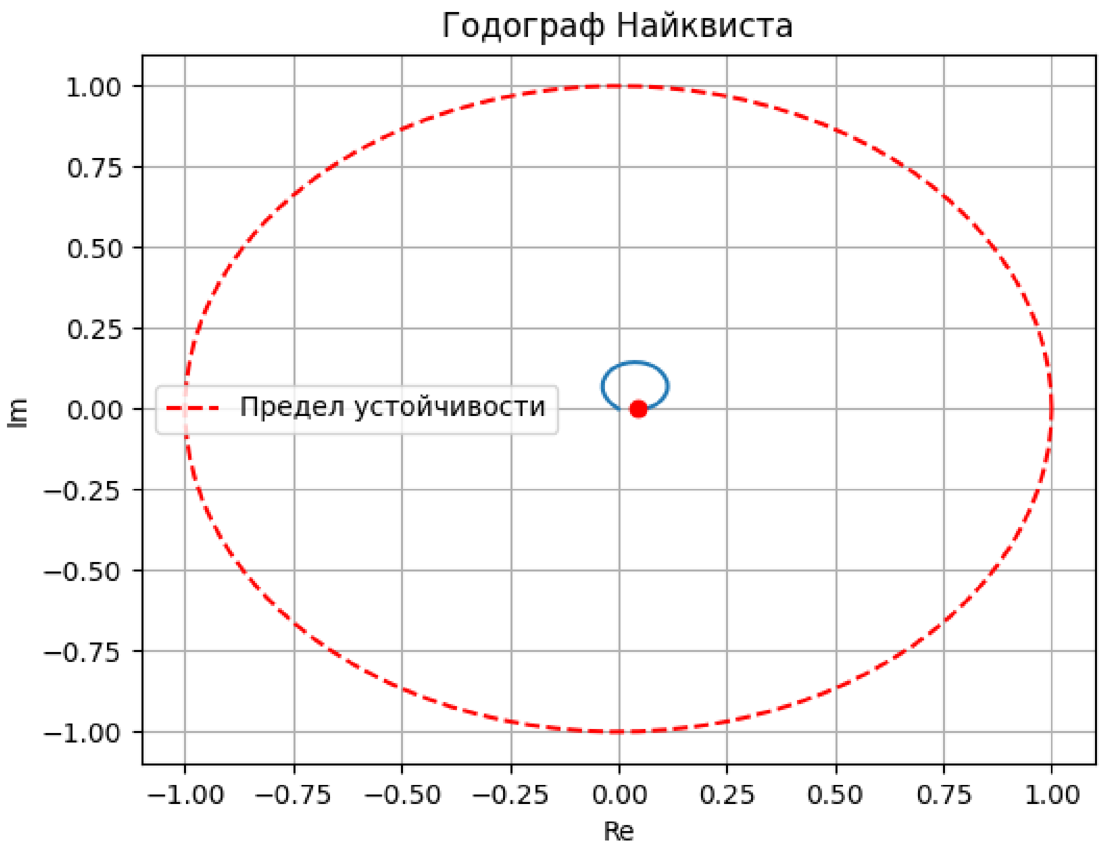
Система неустойчива.

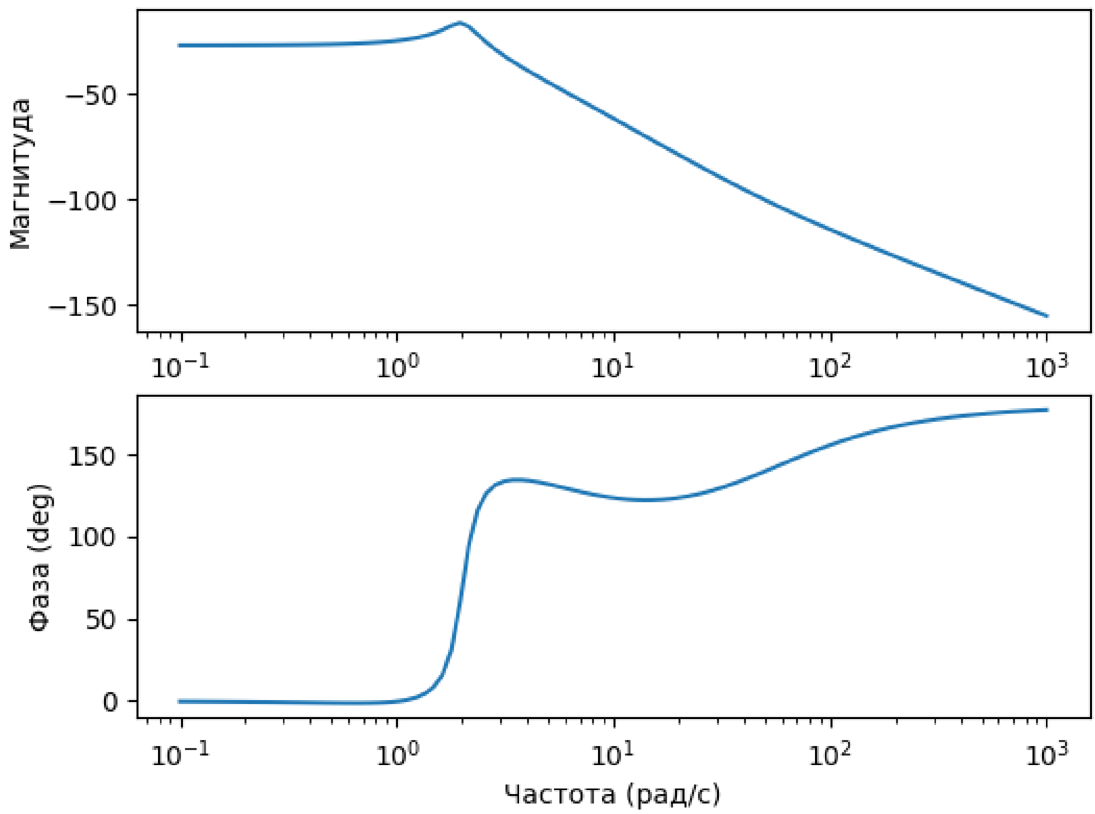
Вывод: Использовав различные методы проверки устойчивости системы, мы определили, что при изменении коэффициента обратной связи система оказалась на границе устойчивости.

Результаты полученные в ходе работы с Python.

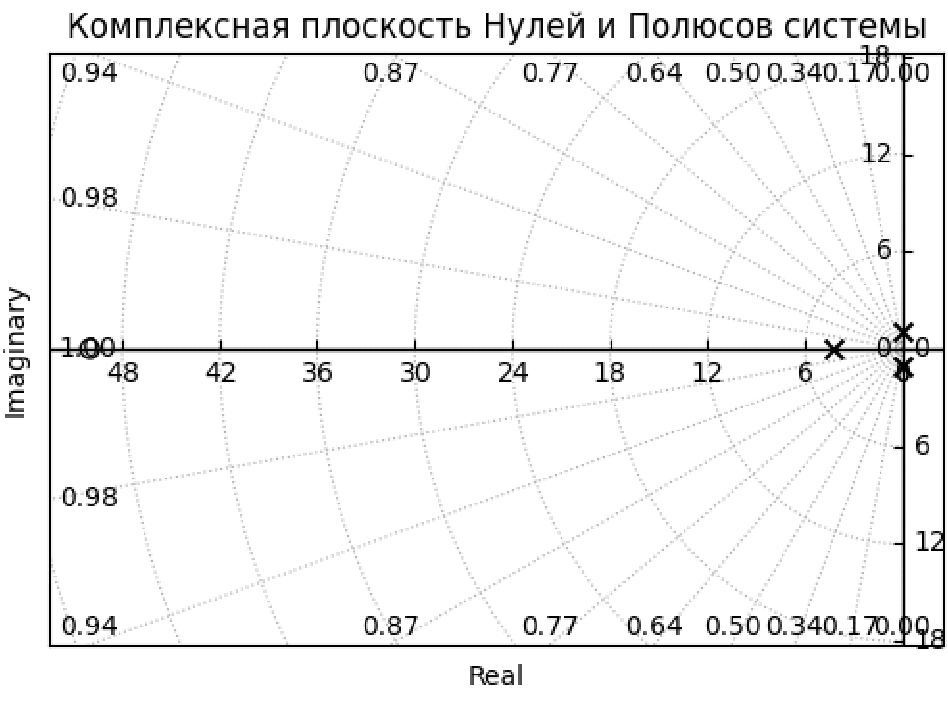
Для заданного коэффициента обратной связи (k=22):

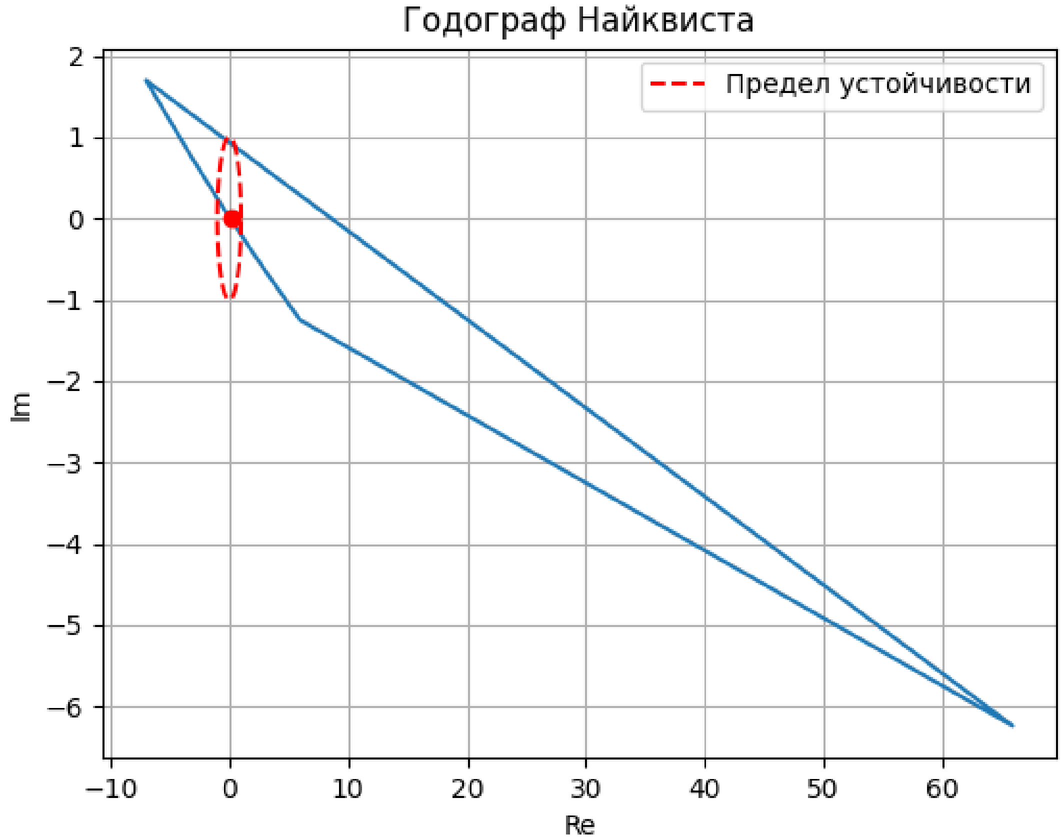


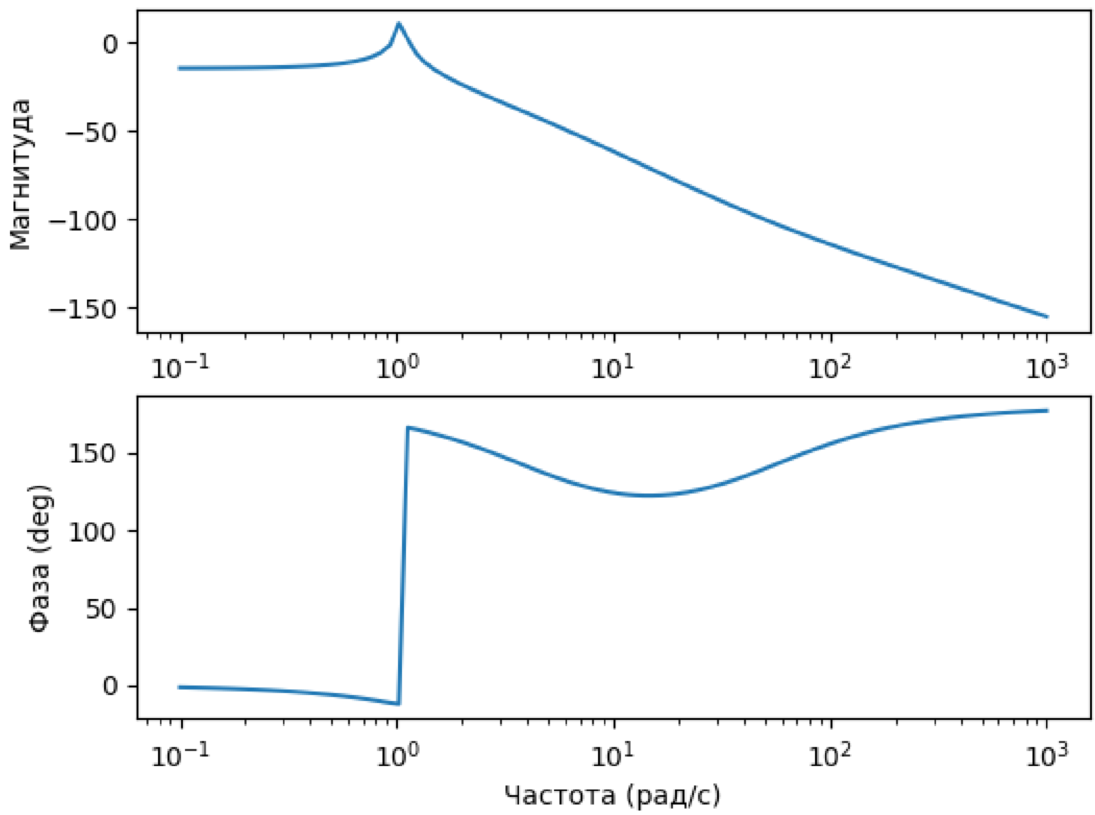




Для измененного коэффициента обратной связи (k=5,29):




Годограф Найквиста
Предел устойчивости
Im
Re
2
1
0
−1
−2
−3
−4
−5
−6
−10
0
10
20
30
40
50
60


Магнитуда
Фаза (deg)
Частота (рад/с)
0
−50
−100
−150
150
100
50
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}

Судя по полученным данным, можно утверждать, что расчеты были произведены верно.

Приложение

Код программы (https://github.com/TAULaboratory/laboratory2-Riserhool/blob/bb95b310d943cda8498dbd2fc83b2de0e43b940b/lab2.py)

```
s = sp.Symbol('s', rational=True)


def multiply_blocks(num_list, den_list):
    numerator = 1
    denominator = 1

    for num in num_list:
        numerator *= sp.Poly(num, s)

    for den in den_list:
        denominator *= sp.Poly(den, s)

    return numerator, denominator

def check_stability(num, den):
    H = sp.Poly(num, s) / sp.Poly(den, s)
    poles = sp.roots(den, s)
    zeros = sp.roots(num, s)

    is_stable = all(sp.re(p) < 0 for p in poles)

    return poles, zeros, is_stable

num_blocks = int(input("Введите количество блоков в системе управления: "))
num_list = []
den_list = []
```

```
for i in range(num_blocks):
    print(f"Блок № {i + 1}")
    num = input("Введите значения(коэффициенты) числителя через пробел: ").split()
    den = input("Введите значения(коэффициенты) знаменателя через пробел: ").split()
    num_list.append([float(n) for n in num])
    den_list.append([float(d) for d in den])

s = sp.symbols('s')
numerator, denominator = multiply_blocks(num_list, den_list)

forward_path = sp.Poly(numerator, s) / sp.Poly(denominator, s)

print("Передаточная функция прямого пути :", forward_path)



num_feedback_blocks = int(input("Введите количество блоков обратой связи: "))
num_feedback_list = []
den_feedback_list = []

for i in range(num_feedback_blocks):
    print(f"Блок обратной связи № {i + 1}")
    num_feedback = input("Введите значения(коэффициенты) числителя обратной связи через пробел: ").split()
    den_feedback = input("ЕВведите значения(коэффициенты) знаменателя обратной связи через пробел: ").split()
    num_feedback_list.append([float(n) for n in num_feedback])
    den_feedback_list.append([float(d) for d in den_feedback])
```

```
s = sp.symbols('s')
numerator_feedback, denominator_feedback =
multiply_blocks(num_feedback_list, den_feedback_list)
feedback_path = sp.Poly(numerator_feedback, s) / sp.Poly(denominator_feedback,
s)


feedback_type = input("Определите тип обратной связи:\n1. Отрицательная\n2.
Положительная\n")



# Расчет передаточной функции с обратной свзяью
if feedback_type == '1':
    transfer_function = forward_path / (1 + (forward_path * feedback_path))
elif feedback_type == '2':
    transfer_function = forward_path / (1 - (forward_path * feedback_path))
else:
    print("Неправильно указан тип обратной связи!")


G=transfer_function.simplify()
print("Результирующая передаточная функция:")
sp.pprint(G)
numerator_final, denominator_final = sp.fraction(G)
top, bot = [[float(i) for i in sp.Poly(i, s).all_coeffs()] for i in G.as_numer_denom()]
sys=co.TransferFunction(top, bot)
plt.figure(1)
plt.figure(figsize=(6, 4), dpi=100)
co.pzmap(sys, grid=True)
plt.title('Комплексная плоскость Нулей и Полюсов системы')
plt.show()


poles, zeros, is_stable = check_stability(numerator_final, denominator_final)
```

```python
print("Полюса:", poles)
print("Нули:", zeros)
print("Соответственно положению полюсов на комплексной плоскости система: ", "Стабильна" if is_stable else "Нестабильна")


def nyquist_criterion(num, den):
    # Определение границ частоты в бесконечность + и -
    w = np.logspace(-3, 3, num=1000)

    # Расчет передаточной функции для каждой частоты
    s = 1j * w
    G = np.polyval(num, s) / np.polyval(den, s)

    # Расчет магнитуды и фазы от G
    mag = np.abs(G)
    phase = np.angle(G)

    # Годограф Найквиста
    plt.figure(2)
    plt.plot(np.real(G), np.imag(G))
    plt.plot(np.real(G[0]), np.imag(G[0]), 'ro')  # Точка отчета
    plt.xlabel('Re')
    plt.ylabel('Im')
    plt.title('Годограф Найквиста')
    plt.grid(True)

    # Построение запаса устойчивости
    R = np.max(mag)
    theta = np.linspace(0, 2 * np.pi, num=100)
    re_limit = np.cos(theta)
```

```
    im_limit = np.sin(theta)
    plt.plot(re_limit, im_limit, 'r--', label='Предел устойчивости')
    plt.legend()

    plt.show()

    # Определение устойчивости по Найквистуу
    crosses_origin = np.sum(np.diff(np.sign(mag)))  # Сколько раз кривая
пересекла черту

    if crosses_origin == 0:
        print("Соответственно критерию Найквиста, система устойчива.")
    elif crosses_origin < 0:
        print("Соответственно критерию Найквиста, система неустойчива.")
    else:
        print("Соответственно критерию Найквиста, система на границе
устойчивости.")

nyquist_criterion(top, bot)



def create_hurwitz_matrix(coefficients):
    k = 0
    matrix = []
    for _ in range(0, len(coefficients)-1):
        column = []
        for d in range(0, len(coefficients)-1):
            if 2*d+1-k < 0:
                column.append(0)
            else:
                try:
```

```
                column.append(coefficients[2*d+1-k])
            except IndexError:
                column.append(0)
            d += 1
        matrix.append(column)
        k += 1
    return np.array(matrix)

#def check_matrix_determinant(matrix):
#   check if the matrix denominator renders the system unstable or marginally
#   stable

def create_minor(matrix, row, column):
    # Убираем i-столбец и j-столбец
    return matrix[
        np.array(list(range(row)) +
                 list(range(row+1, matrix.shape[0])))[:,np.newaxis],
        np.array(list(range(column)) +
                 list(range(column+1, matrix.shape[1])))]

def check_stability(hurwitz_matrix, coefficients):
    a=0
    print(hurwitz_matrix)
    print("Определитель матрицы: "+
          str(np.linalg.det(hurwitz_matrix)))
    if np.linalg.det(hurwitz_matrix) > 0:
        print("Определитель матрицы больше нуля")
    elif np.linalg.det(hurwitz_matrix) == 0:
        print("Определитель матрицы равен нулю, система на границе
устойчивости")
```

```
    else:
        return("Определитель матрицы меньше нуля, система неустойчива")


    for _ in range(0, len(coefficients)-2):
        x,y = hurwitz_matrix.shape
        hurwitz_matrix= create_minor(np.array(hurwitz_matrix), x-1, y-1)
        hurwitz_matrix= create_minor(hurwitz_matrix, x-1, y-1)
        print(hurwitz_matrix)
        print("Определитель матрицы: " +
              str(np.linalg.det(hurwitz_matrix)))
        if np.linalg.det(hurwitz_matrix) > 0:
            continue
        elif np.linalg.det(hurwitz_matrix) == 0:
            print("Определитель матрицы равен нулю, система на границе
устойчивости")
            continue
        else:
            print("Определитель матрицы меньше нуля, система неустойчива")


    print("Определители матрицы больше нуля, система устойчива")



def create_minor(matrix, row, column):


    return matrix[np.array(list(range(row))+list(range(row+1,
matrix.shape[0])))[:,np.newaxis],
               np.array(list(range(column))+list(range(column+1, matrix.shape[1])))]


coefficients = bot
```

```
print("Для устойчивости все определители по Гурвицу должны быть больше нуля")
matrix=create_hurwitz_matrix(coefficients)


a=0
newmatrix=np.array(matrix)
for _ in range(0, len(coefficients)-2):
   x,y = newmatrix.shape
   newmatrix= create_minor(np.array(newmatrix), x-1, y-1)
   print(newmatrix)
   print("Определитель данной матрицы равен:" +
        str(np.linalg.det(newmatrix)))
   if np.linalg.det(newmatrix) > 0:
      continue
   elif np.linalg.det(newmatrix) == 0:
      print("Система на границе устойчивости")
      continue
   else:
      print("Система неустойчива")
print("Система устойчива.")



print('Матрица Гурвица: \n')
newmatrix=matrix
print(check_stability(newmatrix, coefficients))




# Определение передаточной функции
```

```
tf = TransferFunction(top, bot)

# Перевод передаточой функции к графическому виду to state-space
ss = StateSpace(tf)

# Вычисление параметров функции
eig = np.linalg.eigvals(ss.A)

# Проверка устойчивости
if np.all(np.real(eig) < 0):
    print("Система устойчива")
else:
    print("Система неустойчива")

# Построение частотного графика характеристик
w, mag, phase = tf.bode()
plt.figure(4)
fig, (ax1, ax2) = plt.subplots(2, 1)
ax1.semilogx(w, mag)
ax1.set_ylabel("Магнитуда")
ax2.semilogx(w, phase)
ax2.set_xlabel("Частота (рад/с)")
ax2.set_ylabel("Фаза (deg)")
plt.show()



coeffVector = np.array(bot)
coeffLength = len(coeffVector)
rhTableColumn = int(np.ceil(coeffLength / 2))
```

```
# Запуск таблицы Раусса-Гурвица с нулями
rhTable = np.zeros((coeffLength, rhTableColumn))

# Расчет первой строки таблицы
rhTable[0, :] = coeffVector[::2]

# Проверка на четность длины вектора коэффициента
if coeffLength % 2 != 0:
    # Нечетная, вторая строка
    rhTable[1, :rhTableColumn - 1] = coeffVector[1::2]
else:
    # Четная, вторая строка
    rhTable[1, :] = coeffVector[1::2]

# Расчет строк
epss = 0.01

for i in range(2, coeffLength):
    # Особый случай: строка с нулями
    if np.all(rhTable[i - 1, :] == 0):
        order = coeffLength - i
        cnt1 = 0
        cnt2 = 1
        for j in range(rhTableColumn - 1):
            rhTable[i - 1, j] = (order - cnt1) * rhTable[i - 2, cnt2]
            cnt1 += 2
            cnt2 += 1

    for j in range(rhTableColumn - 1):
        # Расчет значений в таблице
```

```
        firstElemUpperRow = rhTable[i - 1, 0]
        rhTable[i, j] = ((rhTable[i - 1, 0] * rhTable[i - 2, j + 1]) - (
                rhTable[i - 2, 0] * rhTable[i - 1, j + 1])) / firstElemUpperRow

        # Особый случай: ноль в первом столбце
        if rhTable[i, 0] == 0:
            rhTable[i, 0] = epss

# Подсчет полюсов справа от оси
unstablePoles = 0

for i in range(coeffLength - 1):
    if np.sign(rhTable[i, 0]) * np.sign(rhTable[i + 1, 0]) == -1:
        unstablePoles += 1

# Вывод полученных значений
print("\nТаблица Рауса-Гурвица:")
print(rhTable)

# Вывод результата об устойчивости
if unstablePoles == 0:
    print("~~~~~> Система устойчива! <~~~~~")
else:
    print("~~~~~> Система неустойчива! <~~~~~")

print("\nКоличество полюсов справа: %d" % unstablePoles)

sysRoots = np.roots(coeffVector)
print("\nПолученные корни уравнения:")
# Полученные корни уравнения
```

```
print(sysRoots)

def mikhailova_stability(numerator_coeffs, denominator_coeffs):
    # Получение полюсов передаточной функции
    poles = np.roots(denominator_coeffs)

    # Подсчет кол-ва полюсов с правой стороны плоскости
    rhp_poles_count = sum(np.real(poles) > 0)

    # Проверка устойчивости
    if rhp_poles_count == 0:
        return "Устойчива"
    elif rhp_poles_count == 1:
        return "На границе устойчивости"
    else:
        return "Неустойчива"

stability_result = mikhailova_stability(top, bot)
print("По критерию Михайлова система: ", stability_result)
```