Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа №2

Дисциплина Моделирование

Тема Программно-алгоритмическая реализаци методов Рунге-Кутта 2-го и 4-го порядков точности при решении системы ОДУ в задаче Коши

Студент Набиев Ф.М.
Группа ИУ7–63Б
Оценка (баллы)
Преподаватель Градов В.М.

Москва, 2020 г.

ВВЕДЕНИЕ

Цель работы: Получение навыков разработки алгоритмов решения задачи Коши при реализации моделей, построенных на системе ОДУ, с использованием методов Рунге-Кутта 2-го и 4-го порядков точности.

1 Теоретическая честь

1.1 Исходные данные

Задана система электротехнических уравнений, описывающих разрядный контур, включающий постоянное активное сопротивление R_k, нелинейное сопротивление $R_p(I)$, зависящее от тока I, индуктивность L_k и емкость C_k.

$$\begin{cases} \dfrac{dI}{dT} = \dfrac{U - (R_k + R_p(I))I}{L_k}, \\ \dfrac{dU}{dt} = -\dfrac{I}{C_k}. \end{cases}$$

Начальные условия $t = 0$, $I = I_0$, $U = U_0$. Здесь I, U — ток и напряжение на конденсаторе.

Сопротивление $R_p(I)$ рассчитать по формуле:

$$R_p = \frac{I_p}{2\pi R^2 \int_0^1 \sigma(T(z))zdz}$$

Для функции $T(z)$ применить выражение $T(z) = T_0 + (T_w - T_0)z^m$.

Параметры T_0, m находятся интерполяцией из таблицы 1.1 при известном токе.

Коэффициент электропроводимости $\sigma(T)$ зависит от T и рассчитывается интерполяцией из таблицы 1.2.

Таблица 1.1 – значения I, $Tc(I)$, $m(I)$

I, A	T_0, k	m
0.5	6700	0.50
1	6790	0.55
5	7150	1.70
10	7270	3.0
50	8010	11.0
200	9185	32.0
400	1001	40.0
800	1114	41.0
1200	120	39.0

Таблица 1.2 – значения $T(z)$, $\sigma(T)$

T, K	σ, 1/Ом см
4000	0.031
5000	0.27
6000	2.05
7000	6.06
8000	12.0
9000	19.9
10000	29.6
11000	41.1
12000	54.1
13000	67.7
14000	81.5

1.2 Метод Рунге-Кутта второго порядка точности

Для системы вида

$$\begin{cases} u'(x) = f(x, u(x)) \\ u(x_0) = y_0 \end{cases}$$

метод Рунге-Кутта второго порядка точности записывается следующим образом:

$$y_{n+1} = y_n + h_n \cdot \left[(1-\alpha) \cdot f(x_n, y_n) + \alpha \cdot f\left(x_n + \frac{h_n}{2\alpha}, y_n + \frac{h_n}{2\alpha} \cdot f(x_n, y_n)\right)\right]$$

где α — произвольный параметр, $\alpha \in [0, 1]$.

1.3 Метод Рунге-Кутта четвёртого порядка точности

Для системы вида

$$\begin{cases} u'(x) = f_1(x, u, v)) \\ v'(x) = f_2(x, u, v) \\ u(\xi) = \eta_1 \\ v(\xi) = \eta_2 \end{cases}$$

метод Рунге-Кутта четвёртого порядка точности записывается следующим образом:

$$y_{n+1} = y_n + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

$$z_{n+1} = z_n + \frac{g_1 + 2g_2 + 2g_3 + g_4}{6}$$

где

$$\begin{array}{ll} k_1 = h_n \cdot f_1\big(x_n, y_n, z_n\big) & g_1 = h_n \cdot f_2\big(x_n, y_n, z_n\big) \\ k_2 = h_n \cdot f_1\big(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}, z_n + \frac{g_1}{2}\big) & g_2 = h_n \cdot f_2\big(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}, z_n + \frac{g_1}{2}\big) \\ k_3 = h_n \cdot f_1\big(x_n + \frac{h_n}{2}, y_n + \frac{k_2}{2}, z_n + \frac{g_2}{2}\big) & g_3 = h_n \cdot f_2\big(x_n + \frac{h_n}{2}, y_n + \frac{k_2}{2}, z_n + \frac{g_2}{2}\big) \\ k_4 = h_n \cdot f_1\big(x_n + h_n, y_n + k_3, z_n + g_3\big) & g_4 = h_n \cdot f_2\big(x_n + h_n, y_n + k_3, z_n + g_3\big) \end{array}$$

2 Практическая часть

2.1 Реализация

Листинг 2.1 – Методы Рунге-Кутта второго и четвёртого порядков точности

```
def RungeKutta2Order(x0, y0, z0, h):
    alpha = 0.5
    nh = h / (2 * alpha)

    k1 = F1(x0, y0, z0)
    g1 = F2(x0, y0, z0)
    k2 = F1(x0 + nh, y0 + nh * k1, z0 + nh * g1)
    g2 = F2(x0 + nh, y0 + nh * k1, z0 + nh * g1)

    y1 = y0 + h * ((1 - alpha) * k1 + alpha * k2)
    z1 = z0 + h * ((1 - alpha) * g1 + alpha * g2)

    return y1, z1


def RungeKutta4Order(xn, yn, zn, hn):
    k1 = hn * F1(xn, yn, zn)
    g1 = hn * F2(xn, yn, zn)

    k2 = hn * F1(xn + hn / 2, yn + k1 / 2, zn + g1 / 2)
    g2 = hn * F2(xn + hn / 2, yn + k1 / 2, zn + g1 / 2)

    k3 = hn * F1(xn + hn / 2, yn + k2 / 2, zn + g2 / 2)
    g3 = hn * F2(xn + hn / 2, yn + k2 / 2, zn + g2 / 2)

    k4 = hn * F1(xn + hn, yn + k3, zn + g3)
    g4 = hn * F2(xn + hn, yn + k3, zn + g3)

    yn_1 = yn + (k1 + 2 * k2 + 2 * k3 + k4) / 6
    zn_1 = zn + (g1 + 2 * g2 + 2 * g3 + g4) / 6

    return yn_1, zn_1
```

Листинг 2.2 – Интерполяция

```
def interpolate(table, x):
    x1 = y1 = 0
    x2 = y2 = 0

    flag = False
```

```
    for i in range(len(table) - 1):
        if (table[i][0] <= x <= table[i + 1][0]):
            flag = True

            y1 = table[i][1]
            y2 = table[i + 1][1]
            x1 = table[i][0]
            x2 = table[i + 1][0]

    if (not flag):
        if (x < table[0][0]):
            return table[0][1]
        if (x> table[-1][0]):
            return table[-1][1]

    return y1 + ((x- x1) / (x2 - x1)) * (y2 - y1)
```

Листинг 2.3 – Метод Симпосона

```
def simpson(I, a=0, b=1, n=50):
    res = 0
    h = (b - a) / n

    for i in range(n):
        x1 = a + h * (i)
        x2 = a + h * (i + 1)

        res += (x2 - x1) / 6.0 * (Sigma(I, x1) +
               4.0 * Sigma(I, 0.5 * (x1 + x2)) + Sigma(I, x2))

    return res
```

Листинг 2.4 – Вычисление параметров

```
def Tz(T0, m, r):
    return (params['Tw'] - T0) * math.pow(r, m) + T0


def Sigma(I, z):
    T0 = interpolate(IT, I)
    m  = interpolate(IM, I)

    return z * interpolate(TS, Tz(T0, m, z))


def Rp(I):
    return params['Le'] / (2 * math.pi * params['R']**2 * simpson(I))

```

```
15
16 def F1(t, I, U):
17     return ((U - (params['Rk'] + Rp(I)) * I) / params['Lk'])
18
19
20 def F2(t, I, U):
21     return -1 / params['Ck'] * I
```

2.2 Примеры работы

Графики зависимости от времени импульса $I(t)$, $U(t)$, $R_p(t)$, $I(t) \cdot R_p(t)$, $T_0(t)$ при

- $R = 0.35$ см;
- $L_э = 12$ см;
- $L_k = 187$ 10–6 Гн;
- $C_k = 268$ 10–6 Ф;
- $R_k = 0.25$ Ом;
- $U_0 = 1400$ В;
- $I_0 = 0.3$ А;
- $T_w = 2000$ К.

Рис. 2.1 – Метод Рунге-Кутта второго порядка

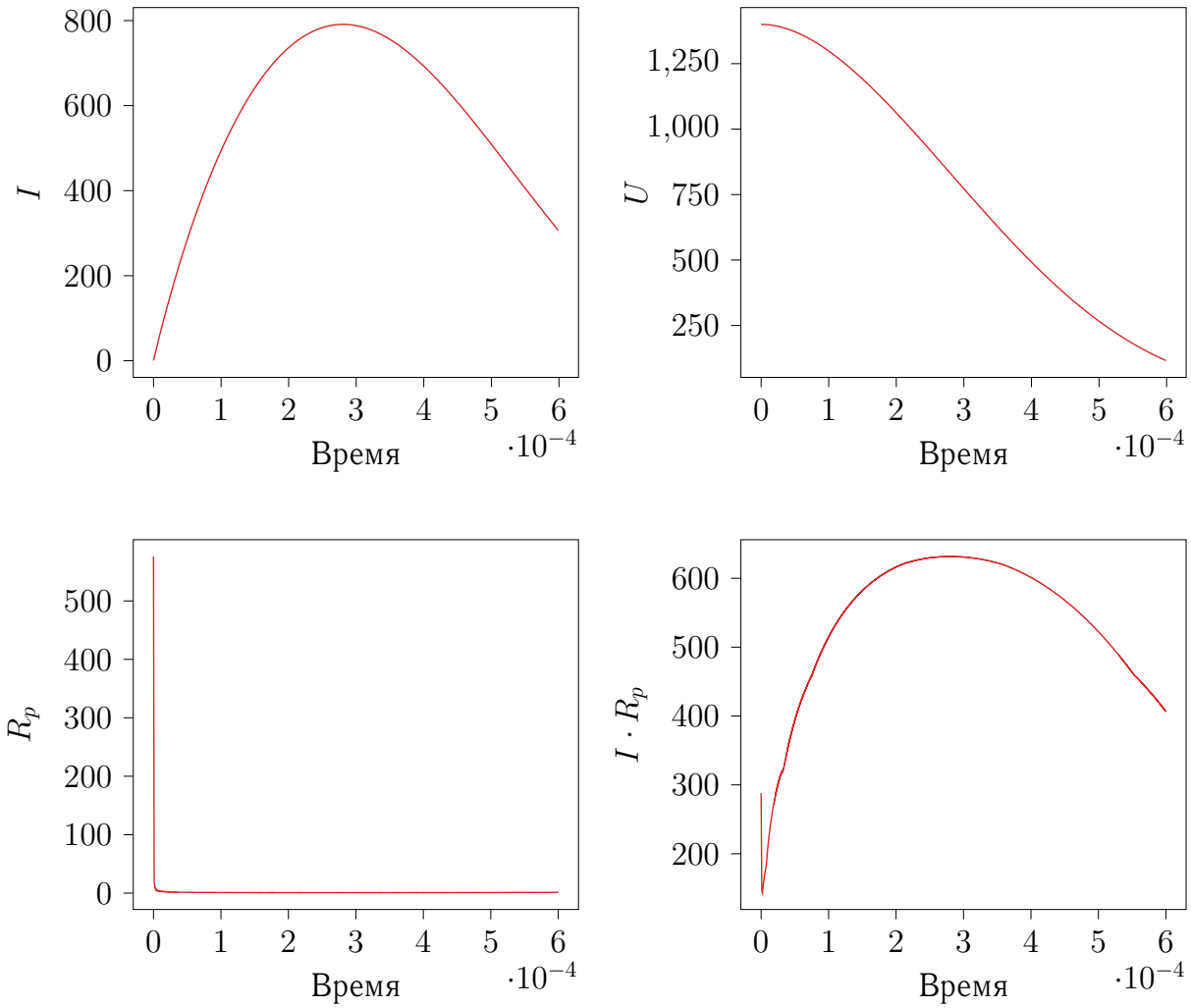
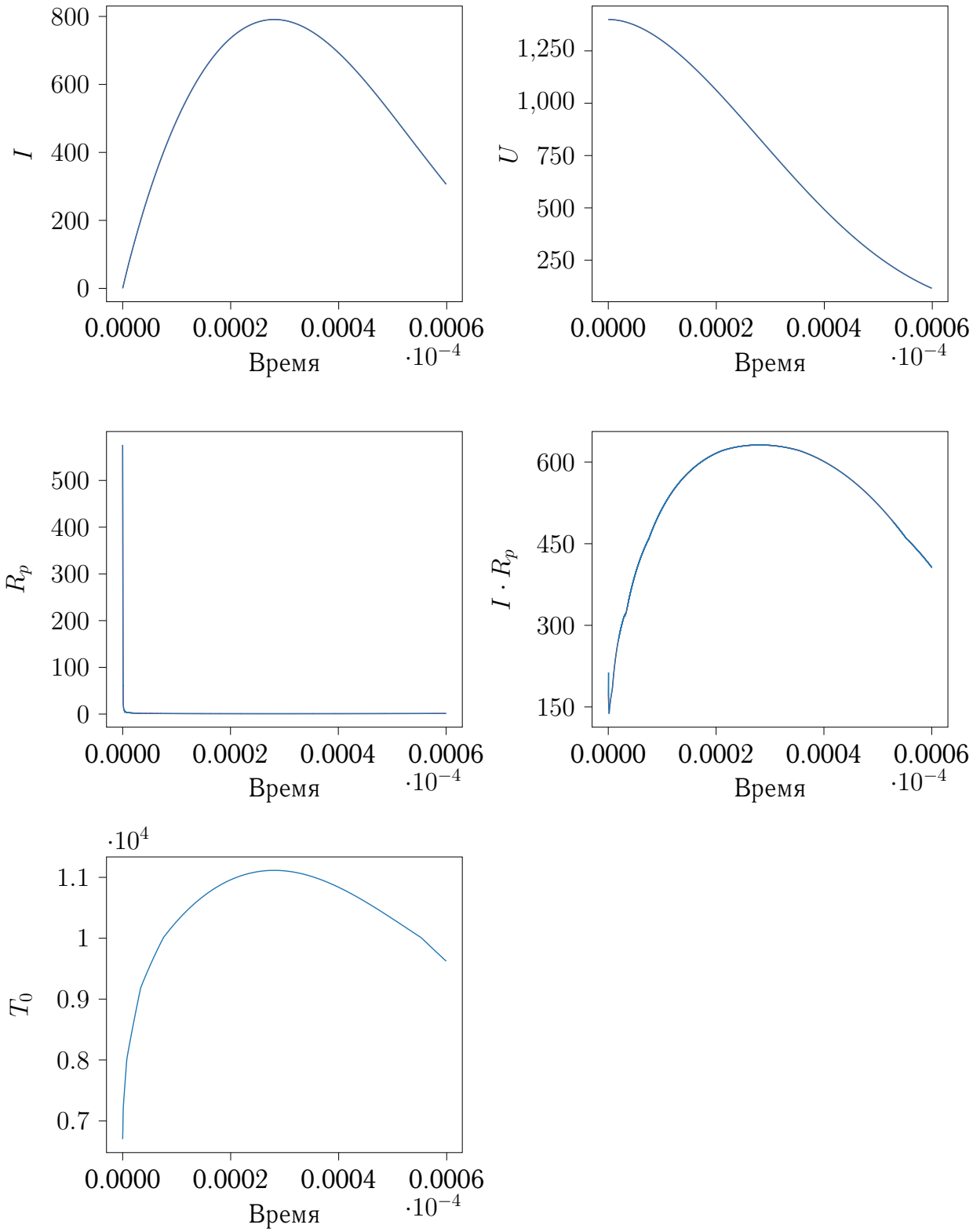


Рис. 2.2 – Метод Рунге-Кутта четвёртого порядка



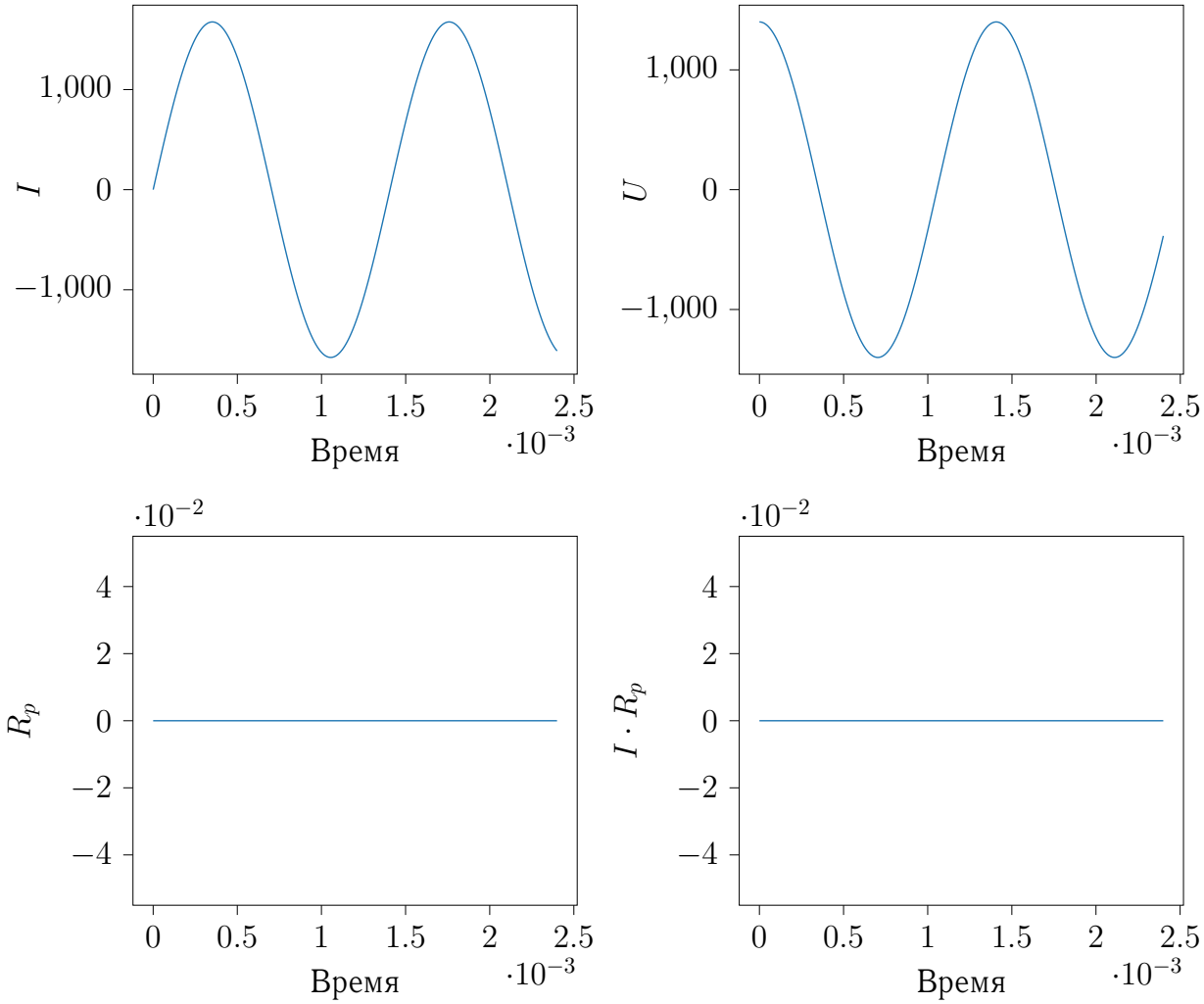
■

График зависимости $I(t)$ при $R_k + R_p = 0$. Обратить внимание на то, что в этом случае колебания тока будут не затухающими.

Изменим параметры:

- $L_э = 0$ см;
- $R_k = 0$ Ом;

Рис. 2.3 – Графики при $R_k + R_p = 0$

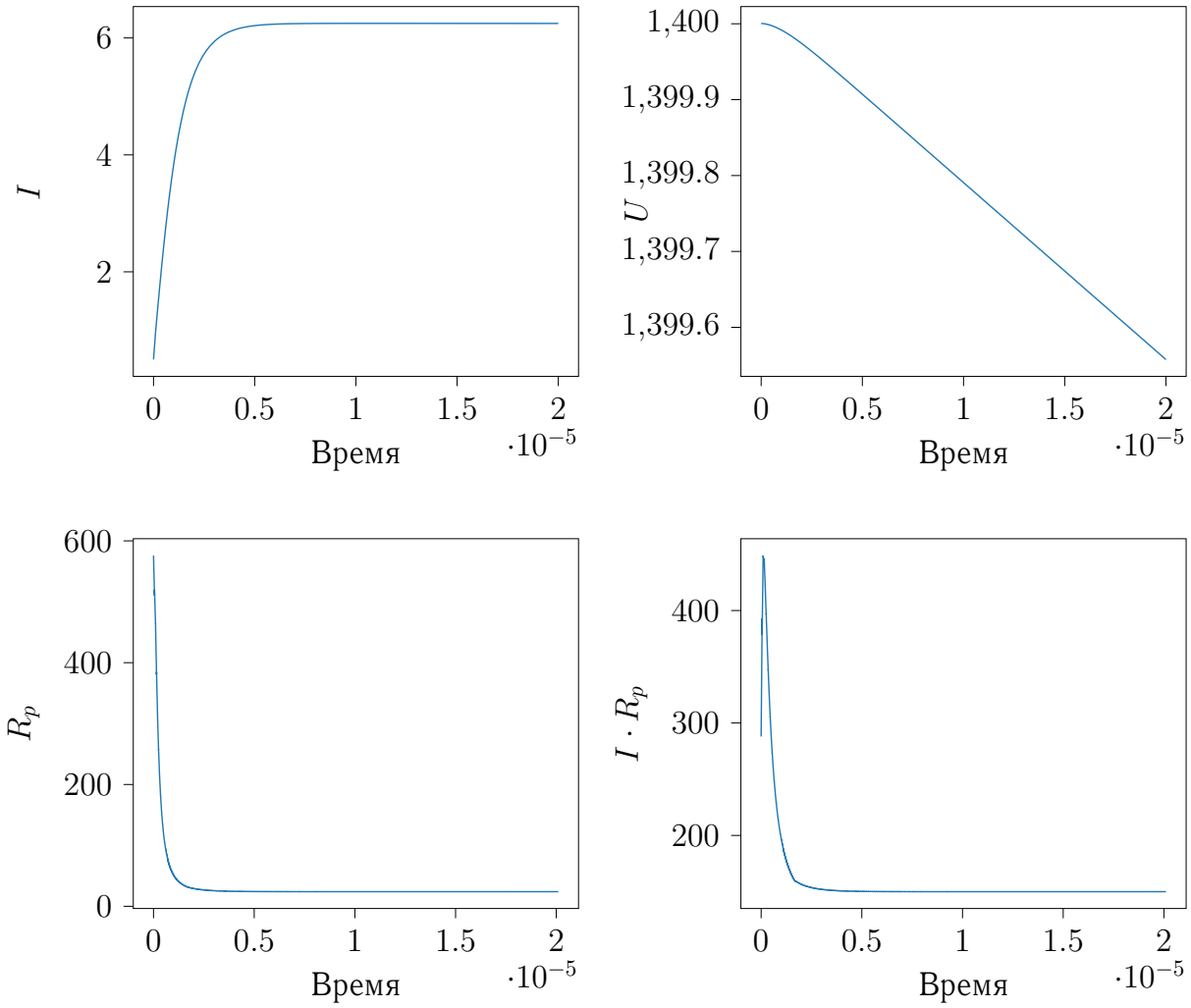


Ввиду отсутствия сопротивления колебания тока стали незатухающими.

■

График зависимости $I(t)$ при $R_k = 200$ Ом в интервале значений t 0–20 мкс.

Рис. 2.4 – Графики при $R_k = 200$ Ом



3 Ответы на контрольные вопросы

3.1 Какие способы тестирования программы можно предложить?

Программу можно тестировать на разных значениях входных параметров.

Например, на разных значениях шага. Уменьшая его, получится обнаружить определённое значение, дальнейшее уменьшение которого не будет менять результат. Это будет точный результат.

Так же можно проверять случай, когда $R_k + R_p = 0$. В этом случае колебания тока должны быть незатухающими.

3.2 Получите систему разностных уравнений для решения сформулированной задачи неявным методом трапеций. Опишите алгоритм реализации полученных уравнений.

Неявный метод трапеций записывается следующим образом:

$$y_{n+1} = y_n + \frac{h}{2} \cdot \Big(f(x_n, y_n) + \alpha \cdot f(x_{n+1}, y_{n+1}) \Big)$$

Тогда для системы

$$\begin{cases} \dfrac{dI}{dT} = \dfrac{U - (R_k + R_p(I))I}{L_k}, \\ \dfrac{dU}{dt} = -\dfrac{I}{C_k}. \end{cases}$$

имеем следующее:

$$I_{n+1} = I_n + \frac{h}{2L_k} \cdot \Big(U_n - (R_k + R_p(I_n))I_n + U_{n+1} - (R_k + R_p(I_{n+1}))I_{n+1} \Big)$$
$$U_{n+1} = U_n - \frac{h}{2C_k} \cdot \Big(I_n + I_{n+1} \Big)$$

Таким образом:

$$I_{n+1} = I_n + \frac{h}{2L_k} \cdot \Big(U_n - (R_k + R_p(I_n))I_n + U_n - \\ - \frac{h}{2C_k} \cdot \Big(I_n + I_{n+1}\Big) - (R_k + R_p(I_{n+1}))I_{n+1}\Big)$$

Здесь I_{n+1} присутствует в обеих частях уравнения. При этом I_{n+1} нельзя выразить, так как $R_p(I)$ вычисляется посредством интерполяции и численного интегрирования. Тогда воспользуемся методом простых итераций:

1. Сначала в I_{n+1} в правой части уравнения подставим I_n.
2. Затем вычислим значение выражения, которое так же подставим в правую часть уравнения на место I_{n+1}.
3. Будем повторять шаг 2 пока $|\frac{I_{n+1}-I_n}{I_{n+1}}| > \epsilon$.

Зная I_{n+1} можем найти U_{n+1}. Так как нам известны I_0, U_0 и шаг, используя полученные формулы и метод простых итераций мы можем получить решение системы уравнений.

3.3 Из каких соображений проводится выбор того или иного метода, учитывая, что чем выше порядок точности метода, тем он более сложен?

Выбор того или иного метода обусловлен достижимостью теоретического порядка точности этого метода при решении конкретного ОДУ. В случае когда, менее точный и, соответственно, более простой метод не может обеспечить сходящийся к точному результат, следует выбрать более сложный метод.