



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»

**Отчет по лабораторной работе №2
по дисциплине «Моделирование»**

Тема Метод Рунге-Кутта 4-го порядка при решении системы ОДУ

Студент Романов А.В.

Группа ИУ7-63Б

Оценка (баллы) _____

Преподаватель Градов В. М.

Тема работы

Программно-алгоритмическая реализация метода Рунге-Кутта 4-го порядка точности при решении системы ОДУ в задаче Коши.

Цель работы

Получение навыков разработки алгоритмов решения задачи Коши при реализации моделей, построенных на системе ОДУ, с использованием метода Рунге-Кутта 4-го порядка точности.

Теоретические сведения

Опишем колебательный контур с помощью системы уравнений:

$$\begin{cases} L_k \frac{dI}{dt} + (R_k + R_p(I)) \cdot I - U_C = 0 \\ \frac{dU_C}{dt} = -\frac{I}{C_k} \end{cases}$$

Значение $R_p(I)$ можно вычислить по формуле:

$$Rp = \frac{l_e}{2\pi \cdot \int_0^R \sigma(T(r)) r dr} = \frac{l_e}{2\pi R^2 \cdot \int_0^1 \sigma(T(z)) dz}$$

т. к. $z = r/R$.

Значение $T(z)$ вычисляется по формуле:

$$T(z) = T_0 + (T_w - T_0) \cdot Z^m$$

Заданы начальные параметры:

$R = 0.35$ см (Радиус трубы)

$l_e = 12$ см (Расстояние между электродами лампы)

$L_k = 187e-6$ Гн (Индуктивность)

$C_k = 268e-6$ Ф (Емкость конденсатора)

$R_k = 0.25$ Ом (Сопротивление)

$U_{c0} = 1400$ В (Напряжение на конденсаторе в начальный момент времени)

$I_0 = 0..3$ А (Сила тока в цепи в начальный момент времени $t = 0$)

$T_w = 2000$ К

Метод Рунге-Кутта четвертого порядка точности

Имеем систему уравнений вида:

$$\begin{cases} u'(x) = f(x, u(x)) \\ u(\xi) = \eta \end{cases}$$

Тогда:

$$\begin{aligned}y_{n+1} &= y_n + \frac{k_1+2k_2+2k_3+k_4}{6} \\k_1 &= h_n f(x_n, y_n) \\k_2 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}\right) \\k_3 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + \frac{k_2}{2}\right) \\k_4 &= h_n f(x_n + h_n, y_n + k_3)\end{aligned}$$

Рассмотрим обобщение формулы на случай двух переменных. Пусть дана система:

$$\begin{cases} u'(x) = f(x, u, v) \\ v'(x) = \varphi(x, u, v) \\ v(\xi) = v_0 \\ u(\xi) = u_0 \end{cases}$$

Тогда:

$$\begin{aligned}y_{n+1} &= y_n + \frac{k_1+2k_2+2k_3+k_4}{6} \\z_{n+1} &= z_n + \frac{q_1+2q_2+2q_3+q_4}{6}\end{aligned}$$

$$\begin{aligned}k_1 &= h_n f(x_n, y_n, z_n) \\k_2 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}, z_n + \frac{q_1}{2}\right) \\k_3 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + \frac{k_2}{2}, z_n + \frac{q_2}{2}\right) \\k_4 &= h_n f(x_n + h_n, y_n + k_3, z_n + q_3) \\q_1 &= h_n \varphi(x_n, y_n, z_n) \\q_2 &= h_n \varphi\left(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}, z_n + \frac{q_1}{2}\right) \\q_3 &= h_n \varphi\left(x_n + \frac{h_n}{2}, y_n + \frac{k_2}{2}, z_n + \frac{q_2}{2}\right) \\q_4 &= h_n \varphi(x_n + h_n, y_n + k_3, z_n + q_3)\end{aligned}$$

Исходный код алгоритма

Листинг 1: Реализация алгоритма Рунге-Кутта 4 порядка для решения системы ОДУ

```

1 def T(z, T0, m):
2     return T0 + (params.Tw - T0) * z ** m
3
4 def f(x, y, z, Rp):
5     return -((params.Rk + Rp) * y - z) / params.Lk
6
7 def phi(x, y, z):
8     return -y / params.Ck
9
10 def get_column(table, ind):
11     return list(map(lambda x: x[ind], table))
12
13 def sigma(T):
```

```

14     return interpolate(T, get_column(snd_table, 0), get_column(snd_table, 1))
15
16 def get_T0(l):
17     return interpolate(l, get_column(fst_table, 0), get_column(fst_table, 1))
18
19 def get_m(l):
20     return interpolate(l, get_column(fst_table, 0), get_column(fst_table, 2))
21
22 def get_Rp(l, T0, m):
23     integral = integrate.quad(lambda z: sigma(T(z, T0, m)) * z, 0, 1)
24     return params.Le / (2 * math.pi * params.R ** 2 * integral[0])
25
26 def interpolate(x, x_pts, y_pts, order=1):
27     return InterpolatedUnivariateSpline(x_pts, y_pts, k=order)(x)
28
29 def get_current_addition(h, coeffs, i, order):
30     if i == 0:
31         return 0, 0, 0
32     elif i == order - 1:
33         return h, coeffs.Kn, coeffs.Pn
34
35     return h / 2, coeffs.Kn / 2, coeffs.Pn / 2
36
37 def get_next_members(current_y, current_z, coeffs):
38     k_sum = 0, p_sum = 0
39
40     for i in range(len(coeffs)):
41         if i > 0 and i < len(coeffs) - 1:
42             k_sum += 2 * coeffs[i].Kn
43             p_sum += 2 * coeffs[i].Pn
44         else:
45             k_sum += coeffs[i].Kn
46             p_sum += coeffs[i].Pn
47
48     divider = 2 * (len(coeffs) - 2) + 2
49     return current_y + k_sum / divider, current_z + p_sum / divider
50
51
52 def get_runge_kutta(x, y, z, h, Rp, order=4):
53     coeffs = [RungeCoeffs(0, 0) for x in range(order)]
54
55     for i in range(order):
56         curr_h, y_add, z_add = get_current_addition(h, coeffs[i - 1], i, order)
57         coeffs[i] = RungeCoeffs(h * f(x + curr_h, y + y_add, z + z_add, Rp), h *
58             phi(x + curr_h, y + y_add, z + z_add))
59
60     return get_next_members(y, z, coeffs)

```

Результат работы программы

На рисунке 1 представлены графики зависимости от времени импульса t : $I(t)$, $U(t)$, $R_p(t)$, $I(t) * R_p(t)$, $T_0(t)$ при исходных данных. Интервал: $[0, 0.0008]$, шаг $h = 1e - 6$.

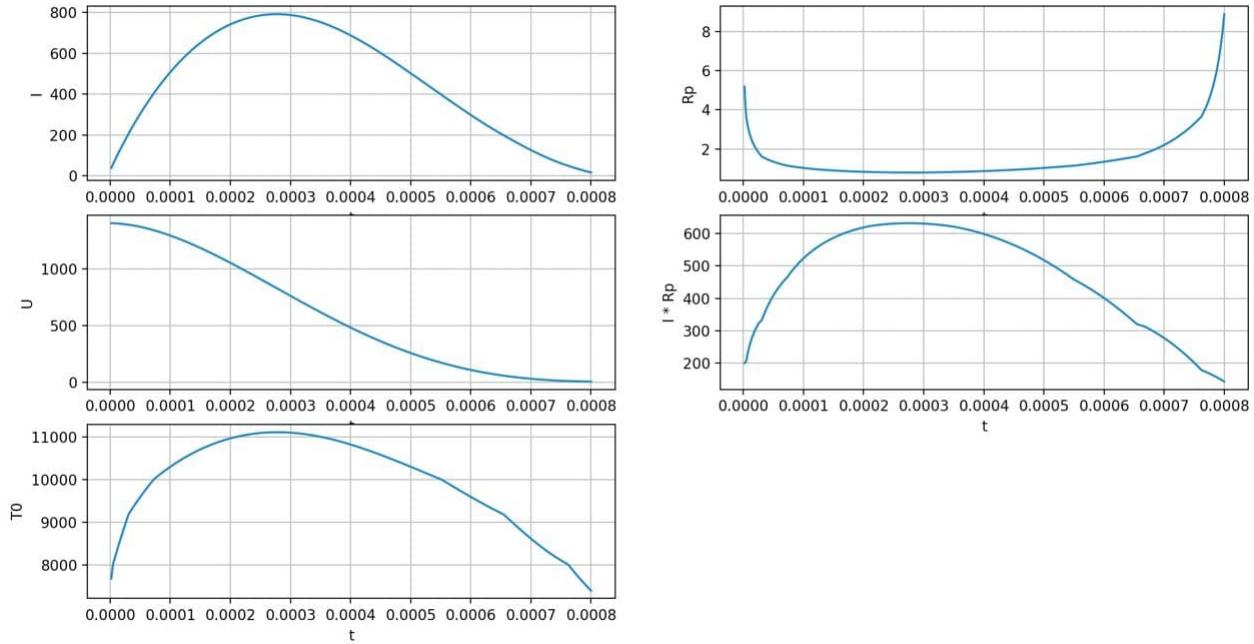


Рис. 1: Графики зависимости от времени импульса t

На рисунке 2 представлен график $I(t)$, при $R_k + R_p = 0$. Интервал: $[0, 0.0008]$, шаг $h = 1e - 6$.

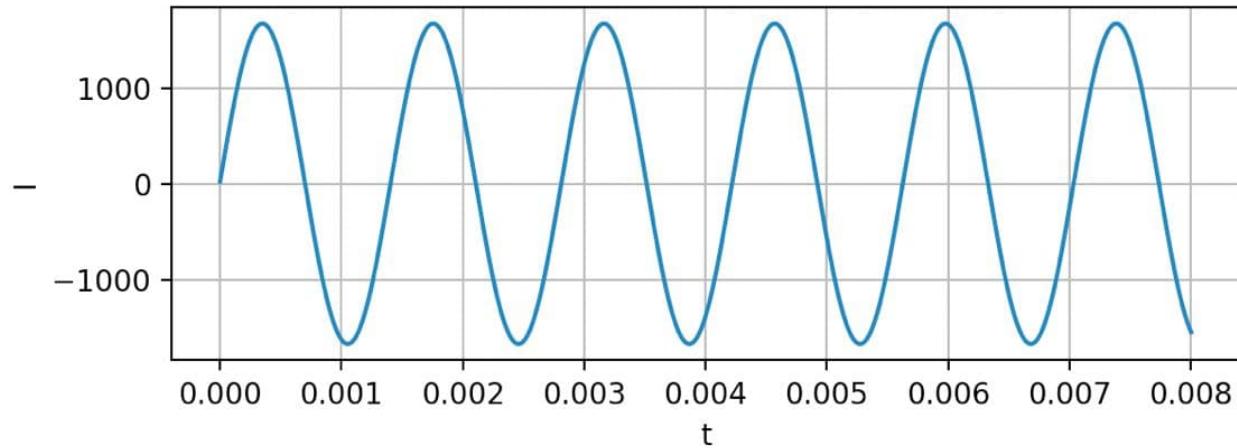


Рис. 2: График зависимости $I(t)$ при $R_k + R_p = 0$

На рисунке 3 представлен график $I(t)$, при $R_k + R_p = 200$. Интервал: $[0, 0.00002]$, шаг $h = 1e - 7$.

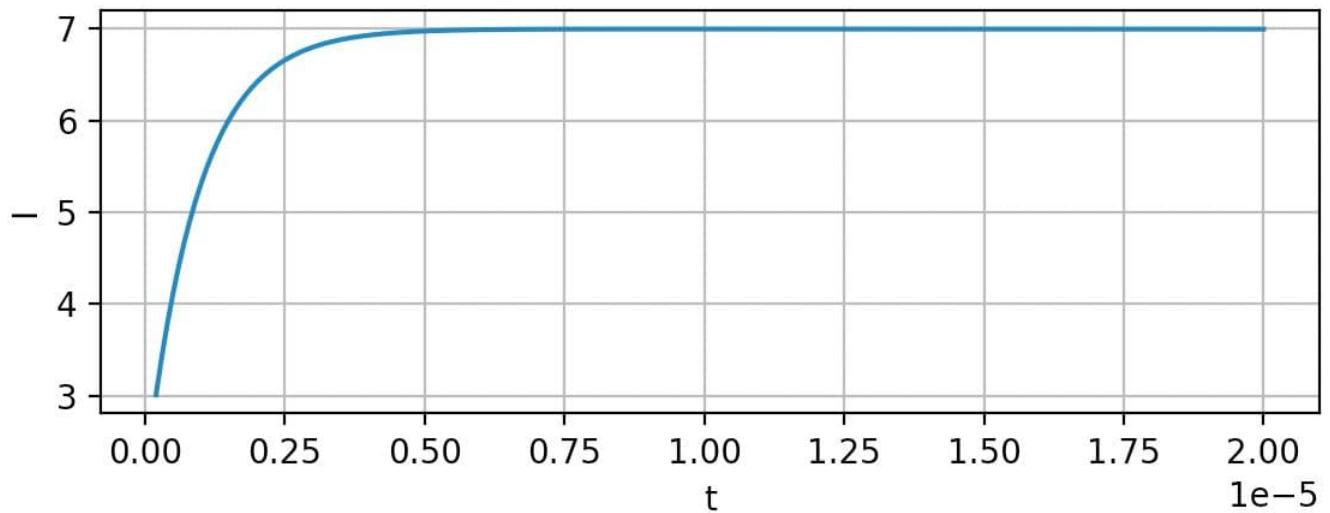


Рис. 3: График зависимости $I(t)$ при $R_k + R_p = const = 200$

На рисунках 4 - 10 представлены результаты исследования влияния параметров контура $C_k, L_k R_k$ на длительность импульса t .

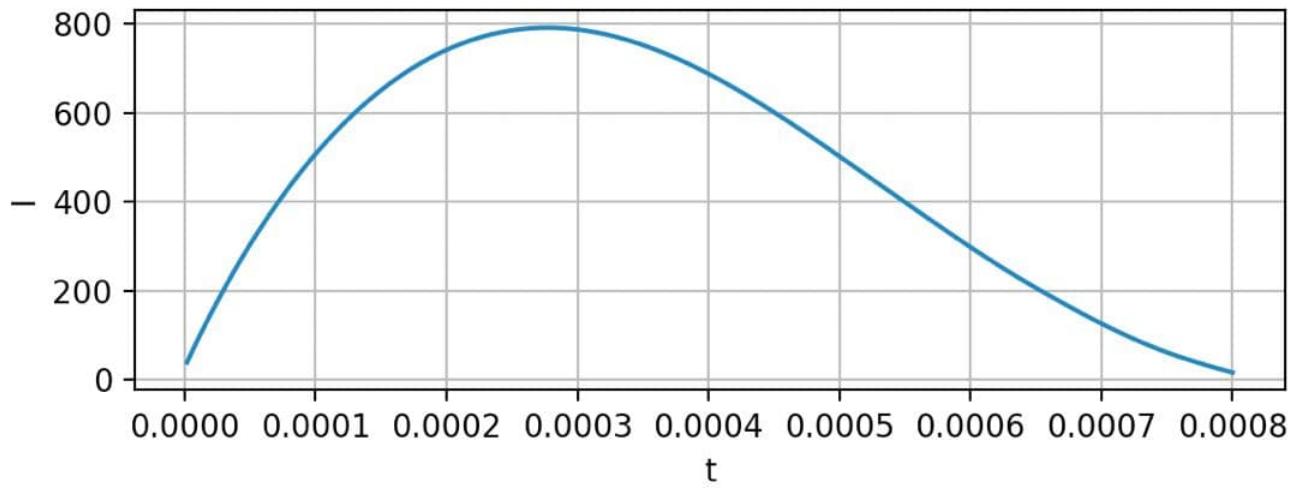


Рис. 4: График зависимости $I(t)$ при начальных параметрах

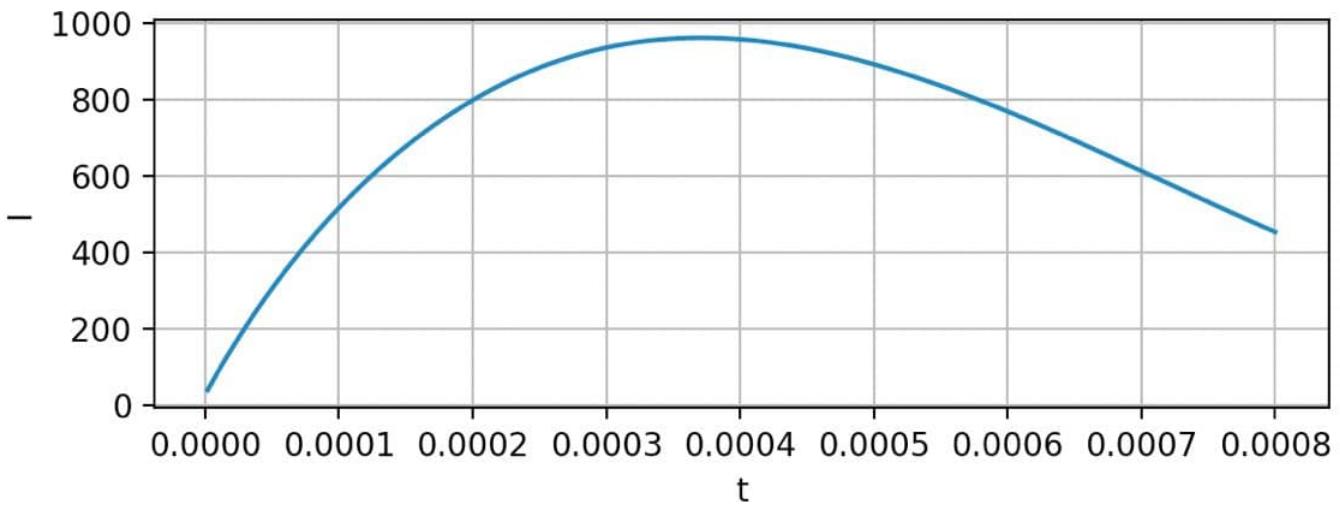


Рис. 5: График зависимости $I(t)$ при увеличении начального значения C_k в 2 раза.

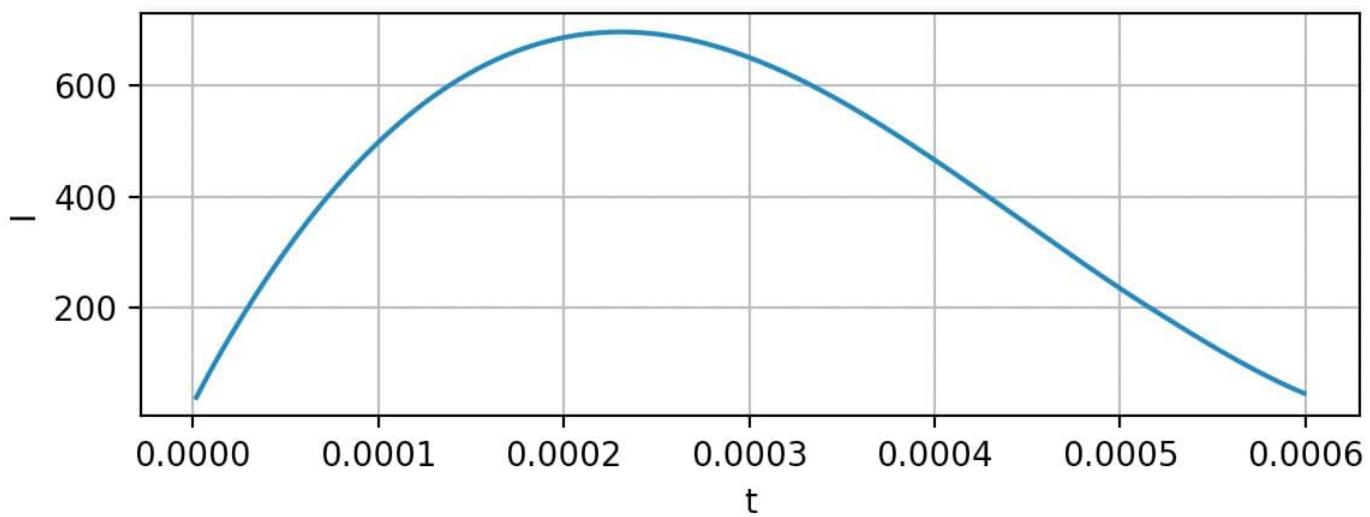


Рис. 6: График зависимости $I(t)$ при уменьшении начального значения C_k в 1.5 раза.

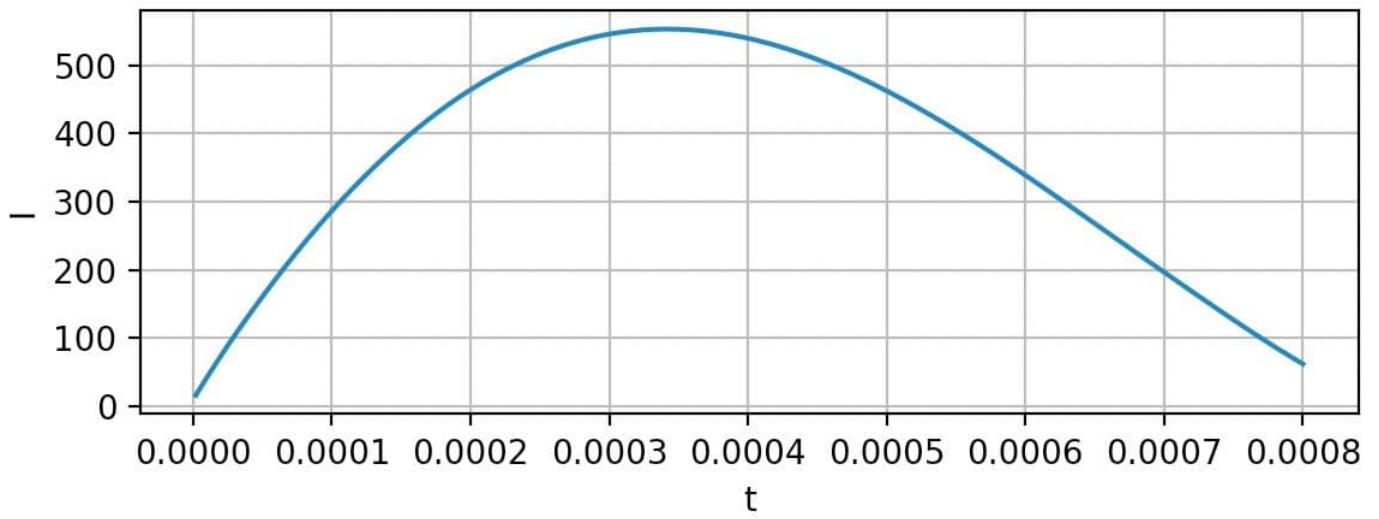


Рис. 7: График зависимости $I(t)$ при увеличении начального значения L_k в 2 раза

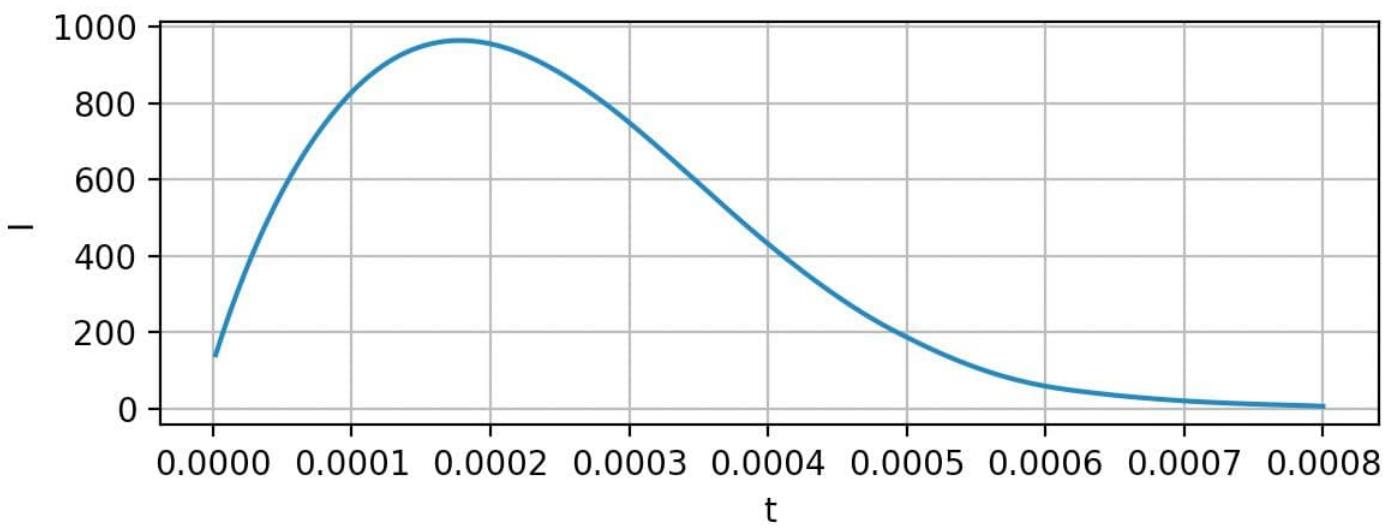


Рис. 8: График зависимости $I(t)$ при уменьшении начального значения L_k в 2 раза

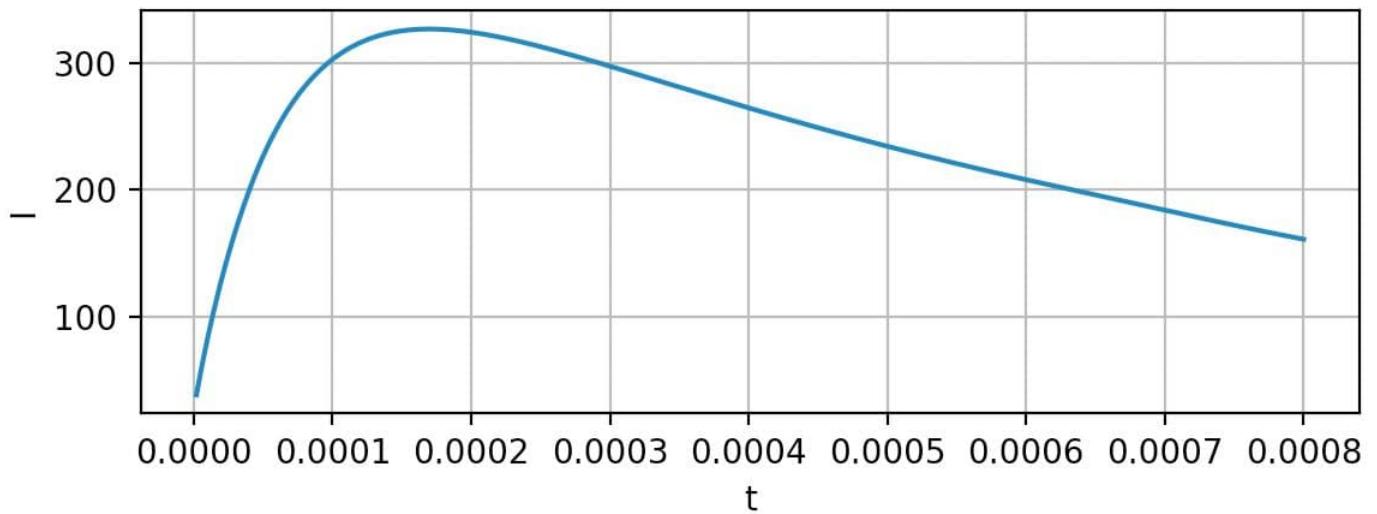


Рис. 9: График зависимости $I(t)$ при увеличении начального значения C_k в 10 раз

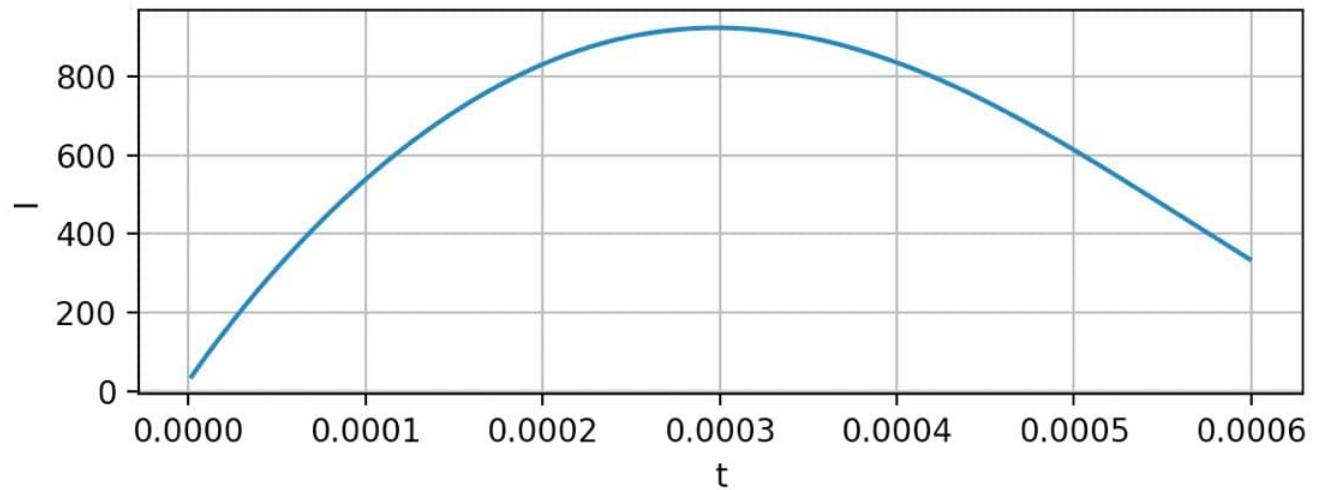


Рис. 10: График зависимости $I(t)$ при уменьшении начального значения C_k в 10 раз

- увеличение C_k приводит к увеличению длительности импульса t ;
- уменьшение C_k приводит к уменьшению длительности импульса t ;
- увеличение L_k приводит к увеличению длительности импульса t ;
- уменьшение L_k приводит к уменьшению длительности импульса t ;

- увеличение R_k приводит к увеличению длительности импульса t ;
- уменьшение R_k приводит к уменьшению длительности импульса t ;

Ответы на вопросы

1. Какие способы тестирования программы, кроме указанного в п. 2, можете предложить ещё?

Мы можем убрать лампу: при небольших значениях R_k получим затухающие колебания, при больших значения R_k получим апериодическое затухание.

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

$$U_{n+1} = U_n + \frac{h}{2} + f(x_n, u_n) + f(x_{n+1}, u_{n+1}) + O(h^2) \quad (1)$$

$$\begin{cases} \frac{dI}{dT} = \frac{U - (R_k + R_p(I))I}{L_k} \\ \frac{dU}{dt} = -\frac{I}{C_k} \end{cases} \quad (2)$$

$$I_{n+1} = I_n + \frac{h}{2} \left(\frac{U_n - (R_k + R_p(I_n))I_n}{L_k} + \frac{U_{n+1} - (R_k + R_p(I_{n+1}))I_{n+1}}{L_k} \right) \quad (3)$$

$$U_{n+1} = U_n + \frac{h}{2} \left(-\frac{I_n}{C_k} - \frac{I_{n+1}}{C_k} \right) = U_n - \frac{h}{2} \left(\frac{I_n + I_{n+1}}{C_k} \right) \quad (4)$$

Подставляя (4) в (3), имеем:

$$I_{n+1} = I_n + \frac{h}{2L_k} \left(2U_n - (R_k + R_p(I_n))I_n + \frac{h}{2C_k} (R_k + R_p(I_{n+1}))I_{n+1} \right) - \left(R_k + R_p(I_{n+1}) + \frac{h}{2C_k} \right) I_{n+1} \quad (5)$$

3. Из каких соображений проводится выбор численного метода того или иного порядка точности, учитывая, что чем выше порядок точности метода, тем он более сложен требует, как правило, больших ресурсов вычислительной системы?

Оценивается погрешность для частного случая вида правой части дифференциального уравнения: $\varphi(x, \nu) = \varphi(x)$

Так, если $\varphi(x, \nu)$ непрерывна и ограничена и непрерывны её четвертые производные, то наилучший результат достигаем при

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

$$\begin{aligned} k_1 &= h_n f(x_n, y_n) \\ k_2 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + \frac{k_1}{2}\right) \end{aligned}$$

$$\begin{aligned}k_3 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + \frac{k_2}{2}\right) \\k_4 &= h_n f\left(x_n + \frac{h_n}{2}, y_n + k_3\right)\end{aligned}$$

В случае если $\varphi(x, \nu)$ не имеет таких производных, то четвертый порядок схемы не может быть достигнут и стоит применять более простые схемы.