Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Российский химико-технологический университет имени Д.И. Менделеева»

Факультет цифровых технологий и химического инжиниринга

Кафедра информационных компьютерных технологий

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 13**

**ПО КУРСУ**

**«ЧИСЛЕННЫЕ МЕТОДЫ В СРЕДЕ MATLAB»:**

**«Системы дифф. уравнений»**

Ведущий преподаватель

Доцент кафедры ИКТ Филиппова Е.Б.

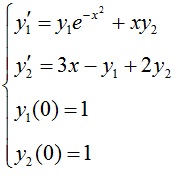
**СТУДЕНТ группы КС-20** Мелехин А.А.

**Москва**

**2024**

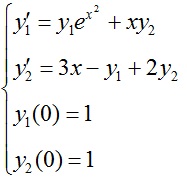
# **Задание**

**1.** Методами Эйлера, модифицированным Эйлера и Рунге-Кутта 4 порядка c шагом h=0.1 до Xкон= 1 решить следующую задачу Коши для нормальной системы второго порядка:



Оценку жёсткости системы производить на каждом шаге h. Вывести на один график результаты всех трёх методов.

**2**. Решить следующую задачу Коши для нормальной системы второго порядка методами явным Эйлера и неявным Эйлера c шагом h=0.1 до Хкон= 3:



Оценку жёсткости системы производить на каждом шаге h. Вывести на один график результаты обоих методов.

**Код (программа lab13.m)**

clc; clear;

% Начальные условия для первой системы

y0\_system1 = [1; 1];

x\_span\_system1 = [0, 2];

% Начальные условия для второй системы

y0\_system2 = [1; 1];

x\_span\_system2 = [0, 2];

% Шаг

h = 0.1;

% Вычисление решения первой системы стандартным методом MATLAB

ode\_options = odeset('RelTol', 1e-6, 'AbsTol', 1e-6);

[x\_system1, y\_system1] = ode45(@ode\_func\_system1, x\_span\_system1, y0\_system1, ode\_options);

y1\_standard\_system1 = y\_system1(:, 1)';

y2\_standard\_system1 = y\_system1(:, 2)';

% Решение первой системы методом Эйлера, Модифицированным методом Эйлера и методом Рунге-Кутты 4-го порядка

[x\_euler\_system1, y1\_euler\_system1, y2\_euler\_system1] = euler\_method(y0\_system1, x\_span\_system1, h);

[x\_mod\_euler\_system1, y1\_mod\_euler\_system1, y2\_mod\_euler\_system1] = modified\_euler\_method(y0\_system1, x\_span\_system1, h);

[x\_rk4\_system1, y1\_rk4\_system1, y2\_rk4\_system1] = rk4\_method(y0\_system1, x\_span\_system1, h);

% Вычисление числа жесткости для первой системы

stiffness\_euler\_system1 = compute\_stiffness(x\_euler\_system1, y1\_euler\_system1, y2\_euler\_system1);

stiffness\_mod\_euler\_system1 = compute\_stiffness(x\_mod\_euler\_system1, y1\_mod\_euler\_system1, y2\_mod\_euler\_system1);

stiffness\_rk4\_system1 = compute\_stiffness(x\_rk4\_system1, y1\_rk4\_system1, y2\_rk4\_system1);

stiffness\_standard\_system1 = compute\_stiffness(x\_system1, y1\_standard\_system1, y2\_standard\_system1);

% Построение графика числа жесткости для первой системы

figure;

hold on;

plot(x\_euler\_system1, stiffness\_euler\_system1, 'b', 'LineWidth', 2);

plot(x\_mod\_euler\_system1, stiffness\_mod\_euler\_system1, 'r', 'LineWidth', 2);

plot(x\_rk4\_system1, stiffness\_rk4\_system1, 'g', 'LineWidth', 2);

plot(x\_system1, stiffness\_standard\_system1, 'k', 'LineWidth', 2);

xlabel('x');

ylabel('Stiffness');

title('Число жесткости для первой системы');

legend('Эйлер', 'Модифицированный Эйлер', 'Рунге-Кутты', 'MATLAB');

grid on;

% Вычисление решения второй системы стандартным методом MATLAB

[x\_system2, y\_system2] = ode45(@ode\_func\_system2, x\_span\_system2, y0\_system2, ode\_options);

y1\_standard\_system2 = y\_system2(:, 1)';

y2\_standard\_system2 = y\_system2(:, 2)';

% Решение второй системы методом Эйлера, Модифицированным методом Эйлера и методом Рунге-Кутты 4-го порядка

[x\_euler\_system2, y1\_euler\_system2, y2\_euler\_system2] = euler\_method(y0\_system2, x\_span\_system2, h);

[x\_mod\_euler\_system2, y1\_mod\_euler\_system2, y2\_mod\_euler\_system2] = modified\_euler\_method(y0\_system2, x\_span\_system2, h);

[x\_rk4\_system2, y1\_rk4\_system2, y2\_rk4\_system2] = rk4\_method(y0\_system2, x\_span\_system2, h);

% Вычисление числа жесткости для второй системы

stiffness\_euler\_system2 = compute\_stiffness(x\_euler\_system2, y1\_euler\_system2, y2\_euler\_system2);

stiffness\_mod\_euler\_system2 = compute\_stiffness(x\_mod\_euler\_system2, y1\_mod\_euler\_system2, y2\_mod\_euler\_system2);

stiffness\_rk4\_system2 = compute\_stiffness(x\_rk4\_system2, y1\_rk4\_system2, y2\_rk4\_system2);

stiffness\_standard\_system2 = compute\_stiffness(x\_system2, y1\_standard\_system2, y2\_standard\_system2);

% Построение графика числа жесткости для второй системы

figure;

hold on;

plot(x\_euler\_system2, stiffness\_euler\_system2, 'b', 'LineWidth', 2);

plot(x\_mod\_euler\_system2, stiffness\_mod\_euler\_system2, 'r', 'LineWidth', 2);

plot(x\_rk4\_system2, stiffness\_rk4\_system2, 'g', 'LineWidth', 2);

plot(x\_system2, stiffness\_standard\_system2, 'k', 'LineWidth', 2);

xlabel('x');

ylabel('Stiffness');

title('Число жесткости для второй системы');

legend('Эйлер', 'Модифицированный Эйлер', 'Рунге-Кутты', 'MATLAB');

grid on;

% Построение графиков для первой системы

plot\_results(x\_system1, y1\_standard\_system1, y2\_standard\_system1, x\_euler\_system1, y1\_euler\_system1, y2\_euler\_system1, x\_mod\_euler\_system1, y1\_mod\_euler\_system1, y2\_mod\_euler\_system1, x\_rk4\_system1, y1\_rk4\_system1, y2\_rk4\_system1, 'Решение первой системы');

% Построение графиков для второй системы

plot\_results(x\_system2, y1\_standard\_system2, y2\_standard\_system2, x\_euler\_system2, y1\_euler\_system2, y2\_euler\_system2, x\_mod\_euler\_system2, y1\_mod\_euler\_system2, y2\_mod\_euler\_system2, x\_rk4\_system2, y1\_rk4\_system2, y2\_rk4\_system2, 'Решение второй системы');

function dydx = ode\_func\_system1(x, y)

dydx = zeros(2, 1);

dydx(1) = y(1)\*exp(-x^2) + x\*y(2);

dydx(2) = 3\*x - y(1) + 2\*y(2);

end

function dydx = ode\_func\_system2(x, y)

dydx = zeros(2, 1);

dydx(1) = y(1)\*exp(x^2) + x\*y(2);

dydx(2) = 3\*x - y(1) + 2\*y(2);

end

function plot\_results(x\_standard, y1\_standard, y2\_standard, x\_euler, y1\_euler, y2\_euler, x\_mod\_euler, y1\_mod\_euler, y2\_mod\_euler, x\_rk4, y1\_rk4, y2\_rk4, title\_str)

figure;

hold on;

plot(x\_standard, y1\_standard, 'k', 'LineWidth', 2);

plot(x\_standard, y2\_standard, 'k--', 'LineWidth', 2);

plot(x\_euler, y1\_euler, 'b', 'LineWidth', 2);

plot(x\_euler, y2\_euler, 'b--', 'LineWidth', 2);

plot(x\_mod\_euler, y1\_mod\_euler, 'r', 'LineWidth', 2);

plot(x\_mod\_euler, y2\_mod\_euler, 'r--', 'LineWidth', 2);

plot(x\_rk4, y1\_rk4, 'g', 'LineWidth', 2);

plot(x\_rk4, y2\_rk4, 'g--', 'LineWidth', 2);

xlabel('x');

ylabel('y');

title(title\_str);

legend('y1 - MATLAB', 'y2 - MATLAB', 'y1 - Эйлер', 'y2 - Эйлер', 'y1 - Модифицированный Эйлер', 'y2 - Модифицированный Эйлер', 'y1 - Рунге-Кутты', 'y2 - Рунге-Кутты');

grid on;

end

function [x, y1, y2] = euler\_method(y0, x\_span, h)

x = x\_span(1):h:x\_span(2);

y1 = zeros(size(x));

y2 = zeros(size(x));

y1(1) = y0(1);

y2(1) = y0(2);

for i = 1:(length(x)-1)

y1\_prime = y1(i)\*exp(-x(i)^2) + x(i)\*y2(i);

y2\_prime = 3\*x(i) - y1(i) + 2\*y2(i);

y1(i+1) = y1(i) + h \* y1\_prime;

y2(i+1) = y2(i) + h \* y2\_prime;

end

end

function [x, y1, y2] = modified\_euler\_method(y0, x\_span, h)

x = x\_span(1):h:x\_span(2);

y1 = zeros(size(x));

y2 = zeros(size(x));

y1(1) = y0(1);

y2(1) = y0(2);

for i = 1:(length(x)-1)

y1\_prime = y1(i)\*exp(-x(i)^2) + x(i)\*y2(i);

y2\_prime = 3\*x(i) - y1(i) + 2\*y2(i);

y1\_temp = y1(i) + h \* y1\_prime;

y2\_temp = y2(i) + h \* y2\_prime;

y1(i+1) = y1(i) + h/2 \* (y1\_prime + (y1\_temp\*exp(-(x(i)+h)^2) + (x(i)+h)\*y2\_temp));

y2(i+1) = y2(i) + h/2 \* (y2\_prime + (3\*(x(i)+h) - y1\_temp + 2\*y2\_temp));

end

end

function [x, y1, y2] = rk4\_method(y0, x\_span, h)

x = x\_span(1):h:x\_span(2);

y1 = zeros(size(x));

y2 = zeros(size(x));

y1(1) = y0(1);

y2(1) = y0(2);

for i = 1:(length(x)-1)

k1\_y1 = y1(i)\*exp(-x(i)^2) + x(i)\*y2(i);

k1\_y2 = 3\*x(i) - y1(i) + 2\*y2(i);

k2\_y1 = (y1(i) + h/2\*k1\_y1)\*exp(-(x(i)+h/2)^2) + (x(i)+h/2)\*(y2(i) + h/2\*k1\_y2);

k2\_y2 = 3\*(x(i)+h/2) - (y1(i) + h/2\*k1\_y1) + 2\*(y2(i) + h/2\*k1\_y2);

k3\_y1 = (y1(i) + h/2\*k2\_y1)\*exp(-(x(i)+h/2)^2) + (x(i)+h/2)\*(y2(i) + h/2\*k2\_y2);

k3\_y2 = 3\*(x(i)+h/2) - (y1(i) + h/2\*k2\_y1) + 2\*(y2(i) + h/2\*k2\_y2);

k4\_y1 = (y1(i) + h\*k3\_y1)\*exp(-(x(i)+h)^2) + (x(i)+h)\*(y2(i) + h\*k3\_y2);

k4\_y2 = 3\*(x(i)+h) - (y1(i) + h\*k3\_y1) + 2\*(y2(i) + h\*k3\_y2);

y1(i+1) = y1(i) + h/6 \* (k1\_y1 + 2\*k2\_y1 + 2\*k3\_y1 + k4\_y1);

y2(i+1) = y2(i) + h/6 \* (k1\_y2 + 2\*k2\_y2 + 2\*k3\_y2 + k4\_y2);

end

end

function stiffness = compute\_stiffness(x, y1, y2)

stiffness = zeros(size(x));

for i = 1:length(x)

y1\_prime = y1(i)\*exp(-x(i)^2) + x(i)\*y2(i);

y2\_prime = 3\*x(i) - y1(i) + 2\*y2(i);

stiffness(i) = max(abs([y1\_prime, y2\_prime]));

end

end

**Результаты расчётов**

