**Вопросы:**

1 – Как исследовать методы на сходимость?

2 – Аналитическое решение - ?

3 – Точное решение - ?

4 – Формулы для сходимости

5 -

% Тема 6

% Для всех методов вычислить абсолютную погрешность численного решения eps.

% Для М1 М2 М3 дать оценку на минимальное число отрезков для eps = 0.001

% Запрограммировать численные методы, исследовать на сходимость.

clc

close all;

clear all;

format long;

% Инициализация переменных

tochnoe = 3.2; % результат точного решения

eps = 0.001;

a = 0; % интервалы интегриродания

b = 2;

h = zeros(10, 1);

average\_error = zeros(10, 1);

norms = zeros(10, 5);

% Исследование сходимости

for j = 1:10

n = 2^(j+1); % число отрезков

h(j) = (b-a) / n; % шаг интегрирования

average\_error(j) = 1 / sqrt(n); % оценка средней ошибки для текущего числа отрезков

x = linspace(0, 2, n + 1); % вектор, который содержит равномерно распределенные значения от 0 до 2, с количеством эл n + 1

% Вычисление интеграла с помощью метода от 1 до 5

for k = 1:5

if k == 1

result = sum((-2 \* (x(1:end-1) + h(j)/2 - 1).^4 + 2) \* h(j)); % результат численного интегрирования метода

elseif k == 2

result = sum(((-2 \* (x(1:end-1) - 1).^4 + 2) + (-2 \* (x(2:end) - 1).^4 + 2)) / 2 \* h(j));

elseif k == 3

result = sum(((-2 \* (x(1:end-1) - 1).^4 + 2) + 4 \* (-2 \* ((x(1:end-1) + x(2:end)) / 2 - 1).^4 + 2) + (-2 \* (x(2:end) - 1).^4 + 2)) / 6 \* h(j));

elseif k == 4

num\_trials = 10; % количество повторных расчетов

results = zeros(1, num\_trials);

for i = 1:num\_trials

x\_rand = 4 \* rand(1, n);

results(i) = mean((-2 \* (x\_rand(1:end-1) + h(j)/2 - 1).^4 + 2) \* h(j));

end

result = mean(results);

elseif k == 5

num\_trials = 60;

results = zeros(1, num\_trials);

for i = 1:num\_trials

x\_y = 2 \* rand(2, n);

results(i) = 4 \* sum(x\_y(1,:) < (-2 \* (x\_y(1,:) - 1).^4 + 2)) / n;

end

result = mean(results);

end

% Вычисление абсолютной погрешности

norms(j, k) = abs(tochnoe - result);

end

end

% Вычисление минимального числа отрезков

syms x

second\_pr = diff((-2 \* (x - 1).^4 + 2), x, 2); % вторая производная функции

second\_pr\_func = matlabFunction(second\_pr);

fourth\_pr = diff((-2 \* (x - 1).^4 + 2), x, 4);

fourth\_pr\_func = double(subs(fourth\_pr, x, a));

kol\_otrez1 = ceil(sqrt((b - a)^3 \* abs(second\_pr\_func(a)) / (24 \* eps))); % для метода прямоугольников

kol\_otrez2 = ceil(sqrt((b - a)^3 \* abs(second\_pr\_func(a)) / (12 \* eps))); % для метода трапеций

kol\_otrez3 = ceil(sqrt(sqrt((b - a)^5 \* abs(fourth\_pr\_func) / (180 \* 16 \* eps)))); % для метода Симпсона

display(kol\_otrez1); display(kol\_otrez2); display(kol\_otrez3);

% Вывод результатов

for k = 1:3

figure;

loglog(h(1:9), norms(1:9, k), '-o', 'Color', [0.6, 0.4, 0.7]);

xlabel('шаг интегрирования (h)');

ylabel('абсолютная погрешность');

title(['сходимость метода ' num2str(k)], 'Color', 'red', 'FontSize', 14);

hold on;

% идеальная сходимость по eps

if k==1

ideal\_convergence = sqrt((b - a).^3 .\* abs(second\_pr\_func(a)) ./ (24 \* eps)) ./ (2.^(1:9)); % теоретические оценки погрешности для каждого метода

end

if k==2

ideal\_convergence = sqrt((b - a).^3 .\* abs(second\_pr\_func(a)) ./ (12 \* eps)) ./ (2.^(1:9));

end

if k==3

ideal\_convergence = sqrt(sqrt((b - a).^5 .\* abs(fourth\_pr\_func) ./ (180 \* eps))) ./ (2.^(1:9));

end

% Построение линии идеальной сходимости

loglog(h(1:9), ideal\_convergence, '--', 'Color', 'green');

hold off;

end

for k = 4:5

figure;

loglog(h(1:9), norms(1:9, k), '-o', 'Color', [0.6, 0.4, 0.7]);

hold on;

% Расчет и добавление линии средней характерной погрешности

loglog(h(1:9), average\_error(1:9), '--', 'Color', 'green');

xlabel('шаг интегрирования (h)');

ylabel('абсолютная погрешность');

title(['сходимость метода ' num2str(k)], 'Color', 'red', 'FontSize', 14);

hold off;

end

% позволяет визуально оценить, как погрешность уменьшается с уменьшением шага интегрирования, и таким образом, оценить сходимость метода.

cls

clear all

close all

I = 3.2;

a = 0;

b = 2;

n = 4;

h(1) = (b - a) / n;

for k = 1:10

x(1) = h(k) / 2;

for i = 2:n

x(i) = x(i - 1) + h(k);

end

x

y = f(x)

S = 0;

for i = 1:n

S=S+f(x(i)) \* h(k);

end

S

err(k) = abs(S-I)

errideal (k) = (b-a).^3\*24/(24\*n^2)

h(k+1)=h(k)/2;

n=2\*n

end

h

err

loglog(h(1:size(err, 2)), err, 'b-', h(1:size(err, 2)), errideal, 'r-')

function y=f(x)

y = -2\*(x-1).^4+2;

end

% Тема7.2 Задание2

clc

close all

clear all

format long;

A = [1, 2, 3, -1; 2, 5, 8, 0; 3, 8, 14, -2; -1, 0, -2, 15];

R = [2; 8; 13; 10];

[str, stl] = size(A);

L = zeros(str, stl); % новая матрица с нулями

%

for j = 1:str % по столбцам

L(j, j) = sqrt(A(j, j) - sum(L(j, 1:j-1).^2)); % значения на главной диагонали

for i = j+1:str % по строкам

L(i, j) = (A(i, j) - L (i, 1:j-1) \* L(j,1:j-1)')/L(j, j);

end

end

Trans = zeros(str, stl); % транспонированнная матрица L

for i = 1:str

Trans(i, :) = L(:,i);

end

y = zeros(str, 1);

for i = 1:str

y(i) = (R(i) - L(i, 1:i-1)\*y(1:i-1))/L(i,i);

end

x = zeros(str, 1);

for i = str:-1:1

x(i) = (y(i) - Trans(i, 1+i:str) \* x(i+1:str));

end

disp('x = '); disp(x);

% Тема7 Часть 3 Задание 1 (Метод простых итераций)

clc

close all

clear all

format long;

A = [16, 8, 4, 0; 8, 13, -1, 3; 4, -1, 18, -5; 0, 3, -5, 11];

b = [4; 17; 12; 1];

x0 = zeros(size(b)); % начальное приближение

q = 0.9;

e = 0.001;

max\_iter = 1000;

x\_pred = x0;

x\_tec = x0;

iter = 0;

while iter < max\_iter

for i = 1:length(b)

sum = 0;

for j = 1:length(b)

if j ~= i

sum = sum + A(i, j) \* x\_pred(j);

end

end

x\_tec(i) = (b(i)-sum) / A(i, i);

end

if q/(1 - q) \* norm(x\_tec - x\_pred) < e

break;

end

x\_pred = x\_tec;

iter = iter + 1;

end

% iter

% x\_tec

% Тема7 Часть 3 Задание 2 (Метод Зейделя)

A = [16, 8, 4, 0; 8, 13, -1, 3; 4, -1, 18, -5; 0, 3, -5, 11];

b = [4; 17; 12; 1];

n = length(b);

x = zeros(n, 1);

q = 0.9;

e = 0.001;

max\_iter = 1000;

x\_pred = x;

iter = 0;

while iter < max\_iter

for i = 1:n

x(i) = (b(i) - A(i, 1:i-1) \* x(1:i-1) - A(i, i+1:n) \* x\_pred(i+1:n)) / A(i, i);

end

iter

x

q / (1 - q) \* norm(x - x\_pred, Inf);

if q / (1 - q) \* norm(x - x\_pred, Inf) < e

break;

end

x\_pred = x;

iter = iter + 1;

end

% Погрешность

Exact = A\b;

for i = 1:4

prost\_iter(i) = abs(Exact(i) - x\_tec(i));

seidel(i) = abs(Exact(i) - x(i));

end

%prost\_iter

%seidel

% Тема 7 Часть 4 (Метод сопряженных градиентов)

clc

close all

clear all

format long;

L = [9, 0, 0; 4, 12, 0; 3, 2, 6];

L'

A = L\*L'

x = [3; 2; 1]

b = A\*x

X = zeros(size(b)); % начальное приближение

for k = 1:length(b)

k

g(:,k) = (A\*X - b)

if k~=1

d(:,k) = -g(:, k) + dot(g(:,k), g(:, k)) / dot(g(:, k-1), g(:, k-1)) \* d(:, k-1)

else

d(:,k) = -g(:,k)

end

s(k) = dot(g(:,k), g(:,k))/dot(A\*d(:,k), d(:,k))

X = X + s(k)\*d(:,k)

end

% Тема 8 (Численные методы решения обыкновенных дифференциальных уравнений)

clc;

clear all;

close all;

% Определение функций

f = @(x, y) y/(x^2 + x);

exact = @(x) 3\*x/(x + 1);

% Определение параметров

Nmax = 10;

a = 2; % Начальная точка

b = 6; % Конечная точка

% Цикл по степеням двойки

for n = 1:Nmax

N = 2^n;

h = (b-a)/N;

x = a:h:b;

y\_euler = zeros(1, N+1);

y\_rk = zeros(1, N+1);

y\_euler(1) = 2;

y\_rk(1) = 2;

norm\_euler = 0;

norm\_rk = 0;

% Метод Эйлера

for i = 2:N+1

y\_euler(i) = y\_euler(i-1) + h\*f(x(i-1), y\_euler(i-1)); % приближенное решение

norm\_euler = norm\_euler + abs(exact(x(i)) - y\_euler(i)); % норма

end

norm\_euler = norm\_euler \* h;

% Метод Рунге-Кутты второго порядка

for i = 2:N+1

k = y\_rk(i-1) + h\*f(x(i-1), y\_rk(i-1));

y\_rk(i) = y\_rk(i-1) + h\*(f(x(i-1), y\_rk(i-1)) + f(x(i), k))/2;

norm\_rk = norm\_rk + abs(exact(x(i)) - y\_rk(i));

end

norm\_rk = norm\_rk \* h;

% Построение графиков

figure(n);

plot(x, y\_euler, 'r', x, y\_rk, 'b', x, exact(x), 'g');

legend('Метод Эйлера', 'Метод Рунге-Кутты', 'Точное решение');

title(['N = ', num2str(N)]);

xlabel('x');

ylabel('y');

% Вычисление погрешности

if n > 1

pogr\_euler(n-1) = log2(norm\_euler\_prev/norm\_euler);

pogr\_rk(n-1) = log2(norm\_rk\_prev/norm\_rk);

end

norm\_euler\_prev = norm\_euler;

norm\_rk\_prev = norm\_rk;

end

% Построение графика погрешности

figure(Nmax+1);

plot(2:Nmax, pogr\_euler, 'r', 2:Nmax, pogr\_rk, 'b');

legend('Метод Эйлера', 'Метод Рунге-Кутты');

title('Погрешность');

xlabel('N');

ylabel('Pogr');