|  |  |
| --- | --- |
| Федеральное агентство железнодорожного транспорта  Омский государственный университет путей сообщения  Кафедра «Автоматика и системы управления»  ЧИСЛЕННОЕ РЕШЕНИЕ ОДУ ПЕРВОГО ПОРЯДКА МОТОДОМ  ЭЙлЕРА-КОШИ  Лабораторная работа №8  по теме: «Теоретические основы аппаратно-программных средств» | |
|  | Студент гр. 21 м  \_\_\_\_\_\_\_\_\_\_\_\_ К.Н. Юрукина  «\_\_»\_\_\_\_\_\_\_\_\_ 2022 г.  Руководитель – доцент кафедры АиСУ  \_\_\_\_\_\_\_\_\_\_\_\_ А.Н. Смалев  «\_\_»\_\_\_\_\_\_\_\_\_ 2022 г. |
| Омск 2022 | |

Лабораторная работа №8

ЧИСЛЕННОЕ РЕШЕНИЕ ОДУ ПЕРВОГО ПОРЯДКА МОТОДОМ

ЭЙлЕРА-КОШИ

Цель работы:

Найти решение ОДУ первого порядка методом Эйлера-Коши и изобразить в виде графика.

Задание

Найдите методом Эйлера численное решение ОДУ первого порядка 8.1 от 2 до 4 с шагом 0,2 (вариант 27):

|  |  |
| --- | --- |
|  | 3(8.1) |

Определим массив *х*, а после методом Эйлера-Коши определим *у*. На основе полученных точек любым методом интерполяции, в нашем случае метод кубическим сплайном, найдем точки для построения графика (рисунок 1). Код основного скрипта представлен в листинге 1.

clc;

clear;

h = 0.2;

x = 2:h:4;

y = EL\_CSH(x);

disp(x);

disp(y);

M = progon (x, y);

x1 = min(x):0.01:max(x);

itr = interval(x, x1);

y1 = spline\_val(x,y,x1,itr,M);

a = min([x x1]);

b = max([x x1]);

figure(1);

plot(x,y, 'o' , x1,y1, 'r');

axis([a-0.5 b+0.5 min([y y1])-0.5 max([y y1])+0.5]);

xlabel('x');

ylabel('y');

grid on;

legend("Расчетные точки", "Интерполяция");

Листинг 1 – Основной скрипт программы

Метод Эйлера-Коши предполагает на предыдущем значении найти новое по формуле 7.2 (листинг 2).

|  |  |
| --- | --- |
|  | (7.2) |

function [y] = EL\_CSH (x)

n = length(x);

y = zeros(1,n);

y(1) = 3.045;

for i = 1:n-1

y1 = y(i)+ 0.2\*((9\*x(i)^2-1)\*exp(-x(i)) - y(i));

y(i+1) = y(i) + 0.1\*((9\*x(i)^2-1)\*exp(-x(i)) - y(i) + (9\*x(i+1)^2-1)\*exp(-x(i+1)) - y1);

end

end

Листинг 2 – Метод Эйлера-Коши

Таким образом, найдя промежуточные точки, рассчитываем новые методом кубического сплайна, алгоритм которого был расписан в предыдущей лабораторной работе (листинги 3-5).

function [ M ]= progon(x, y)

% входные аргументы: (x,y) - таблица заданных узлов

% выходные аргументы: M - массив вторых производных в узловых точках

n = length(x)-1; % число интервалов

h(1:n)=x(2:n+1)-x(1:n); % длины интервалов

% формирование трехдиагональной СЛАУ

A(1)=0; A(2:n-1)=h(2:n-1); % нижняя диагональ СЛАУ

B(1:n-1)=2\*(h(1:n-1)+h(2:n)); % главная диагональ СЛАУ

C(1:n-2)=h(2:n-1); C(n-1)=0; % верхняя диагональ СЛАУ

D=zeros(1:n-1); % правая часть СЛАУ

for i=1:n-1

D(i)=6\*((y(i+2)-y(i+1))/h(i+1)-(y(i+1)-y(i))/h(i));

end

% прямой ход метода прогонки

Q=zeros(1,n);

R=zeros(1,n);

for i=1:n-1

Q(i+1)=-(C(i)/(B(i)+A(i)\*Q(i)));

R(i+1)=(D(i)-A(i)\*R(i))/(B(i)+A(i)\*Q(i));

end

% обратный ход метода прогонки

M=zeros(1,n-1);

M(n-1)=R(n);

for i=n-2:-1:1

M(i)=Q(i+1)\*M(i+1)+R(i+1);

end

% дополняем массив произодных краевыми условиями M(a)=M(b)=0

M = [0, M, 0];

end

Листинг 3 – Прогонка

function [itr] = interval(x, x1)

n = length(x);

n1 = length(x1);

itr = zeros(n1, 1);

k = 1;

for i = 1:n1

if (x1(i) < x(1))

itr(i) = 0;

else

for j = k:n-1

if (x(j)<=x1(i) && x1(i)<=x(j+1))

itr(i)= j;

k = j;

end

end

if x1(i)>x(n)

itr(i)=n;

end

end

end

end

Листинг 4 – Код расстановки интервала

function [y1] = spline\_val(x,y,x1,itr,M)

n = length(x)-1;

n1 = length(x1);

y1 = zeros(1,n1);

h(1:n) = x(2:n+1) - x(1:n);

for i = 1:n1

j = itr(i);

if (j==0)

y1(i) = y(1) + ((x(1)-x(2))\*M(2)/6 + (y(2)-y(1))/(x(2)-x(1))) \* (x1(i)-x(1));

else

if (0 < j && j <= n)

y1(i) = (1/(6\*h(j))) \* ((M(j)\*(x(j+1)-x1(i))^3) + M(j+1)\*(x1(i)-x(j))^3) + (1/h(j)) \* ((y(j)-((M(j)\*h(j)^2)/6)) \* (x(j+1)-x1(i)) + (y(j+1)-((M(j+1)\*h(j)^2)/6)) \* (x1(i)-(x(j))));

else

if (j == n+1)

y1(i) = y(n+1) + ((x(n+1)-x(n))\*M(n)/6 + (y(n+1)-y(n))/(x(n+1)-x(n))) \* (x1(i)-x(n+1));

end

end

end

end

end

Листинг 5 – Кубический сплайн

Расчетные точки находятся в таблице 1.

Таблица 1 – Численное решение ОДУ

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 2,0 | 2,2 | 2,4 | 2,6 | 2,8 | 3,0 | 3,2 | 3,4 | 3,6 | 3,8 | 4,0 |
| 3,045 | 3,347 | 3,583 | 3,752 | 3,855 | 3,898 | 3,886 | 3,828 | 3,73 | 3,6 | 3,445 |

|  |
| --- |
|  |
| Рисунок 1 – Решение ОДУ методом Эйлера-Коши |

Вывод

Таким образом, было найдено решение ОДУ методом Эйлера-Коши. В код входила одна новая функция, которая реализовывала метод. Она рассчитывала новые точки по двум функциям, для текущей точки и для следующей. Значение второй функции считалось отдельно.

Второй составляющей программы являлась интерполяция кубическим сплайном, которая была в предыдущих лабораторных работах.