МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет прикладной математики, информатики и механики
Кафедра прикладной механики и информатики

Отчет по лабораторной работе №2

на тему:
**«Приближенное решение задачи Коши для системы обыкновенных дифференциальных уравнений исправленным методом Эйлера»**


Выполнил: студент 3 к. 1 гр. ПМИ в.о.

Бедарев Анатолий Андреевич

Проверил: к.ф-м.н, доц.

Гудович Николай Николаевич


Воронеж – 2018

**Содержание**

**Постановка задачи**

**1**. **Вар.2**. Написать программу приближенного решения задачи Коши с выводом графиков приближенных решений на экран. Использовать метод приближенного решения:

- исправленный метод Эйлера.

Сравнить результаты с явным методом Эйлера.

*Исходные данные*:

- отрезок [a, b];

- функции y'(x) и z'(x):

$$y' = \frac{y^2}{z-x},\ 0 \le x \le 1\ (1)$$

$$z' = y+1;\ (2)$$

- начальные условия:

$$y(0) = 1,\ (3)$$

$$z(0) = 1;\ (4)$$

- точное решение:

$$y = e^x,\ (5)$$

$$z = x + e^x\ (6)$$

**2**. На основании этого провести численный эксперимент и сделать выводы о сходимости при стремлении шага сетки к 0 (т.е. при стремлении N к бесконечности).

**Указания к выполнению лабораторной работы**

***Задача Коши*** – одна из основных задач теории дифференциальных уравнений (обыкновенных и с частными производными); состоит в нахождении решения (интеграла) дифференциального уравнения, удовлетворяющего так называемым начальным условиям (начальным данным).

Задача Коши обычно возникает при анализе процессов, определяемых дифференциальным законом эволюции и начальным состоянием (математическим выражением которых и являются уравнение и начальное условие).

Численные *методы интегрирования* обыкновенных *дифференциальных уравнений* и их систем вида y ' = f(x,y) можно построить только для случая известных начальных значений всех интегрируемых переменных (для задачи Коши). Общее решение дифференциальных уравнений содержит произвольные постоянные, которые недопустимы в математических моделях, поэтому в качестве искомой функции используется определенное начальными условиями частное решение.

Эти вычислительные методы основаны на замене дифференциальных уравнений алгебраическими. Операцию взятия производной невозможно представить в цифровых ЭВМ, поэтому производная заменяется разностным выражением того или иного вида. В зависимости от этого вида различаются *разностные схемы* численного представления дифференциальных уравнений и соответствующие им методы.

***Исправленный метод Эйлера*** – это численный метод получения решения дифференциального уравнения. Суть исправленного метода Эйлера в пошаговом вычислении значений решения y=y(x) дифференциального уравнения вида y ' = f(x,y) с начальным условием (x0; y0) по формуле (7). Исправленный метод Эйлера является методом 2-го порядка точности и называется методом «предиктор-корректор».

$$y_i = y_{i-1} + \frac{h}{2}\{f(\mathrm{x}_{i-1}, \mathrm{y}_{i-1}) + f(\mathrm{x}_i, \mathrm{y}_i + h \cdot f(\mathrm{x}_{i-1}, \mathrm{y}_{i-1}))\} \quad (7)$$

**Ход выполнения работы**

Выполнение лабораторной работы проводилось с использование языковых средств среды технического моделирования MatLab R2017a.
Для выполнения поставленных задач было разработано несколько программных модулей (см. Приложение), за основы был взят программный код модулей из первой лабораторной работы (явный и неявный методы Эйлера) с некоторыми доработками.

**На первом этапе** в среде MatLab реализован алгоритм приближения исправленным методом Эйлера в виде отдельных модулей:

- *EilerCorrectedMethod.m* – модуль, реализующий приближение исправленным методом Эйлера;
- *F1.m* – модуль, реализующий вычисление функции y'(x) по ф.(1);
- *F2.m* – модуль, реализующий вычисление функции z'(x) по ф.(2);
- *F4.m* – модуль, реализующий вычисление точного решения по ф.(5) и ф.(6).

**На втором этапе** в среде MatLab разработаны дополнительные модули для формирования графиков:

- *plotEiler.m* – выводит графики точного приближенного значений;
- *plotDeltaEiler.m* – выводит графики зависимости средней абсолютной ошибки приближения от числа частичных отрезков разбиения N;
- *deltaEiler.m* – впомогательный модуль для расчета средней абсолютной ошибки приближения.

**На третьем этапе** для выяснения вопроса о сходимости приближенного решения к точному значению функции в зависимости от количества отрезков разбиения выполнен численный эксперимент. Результаты представлены на рис. 1 – 4.

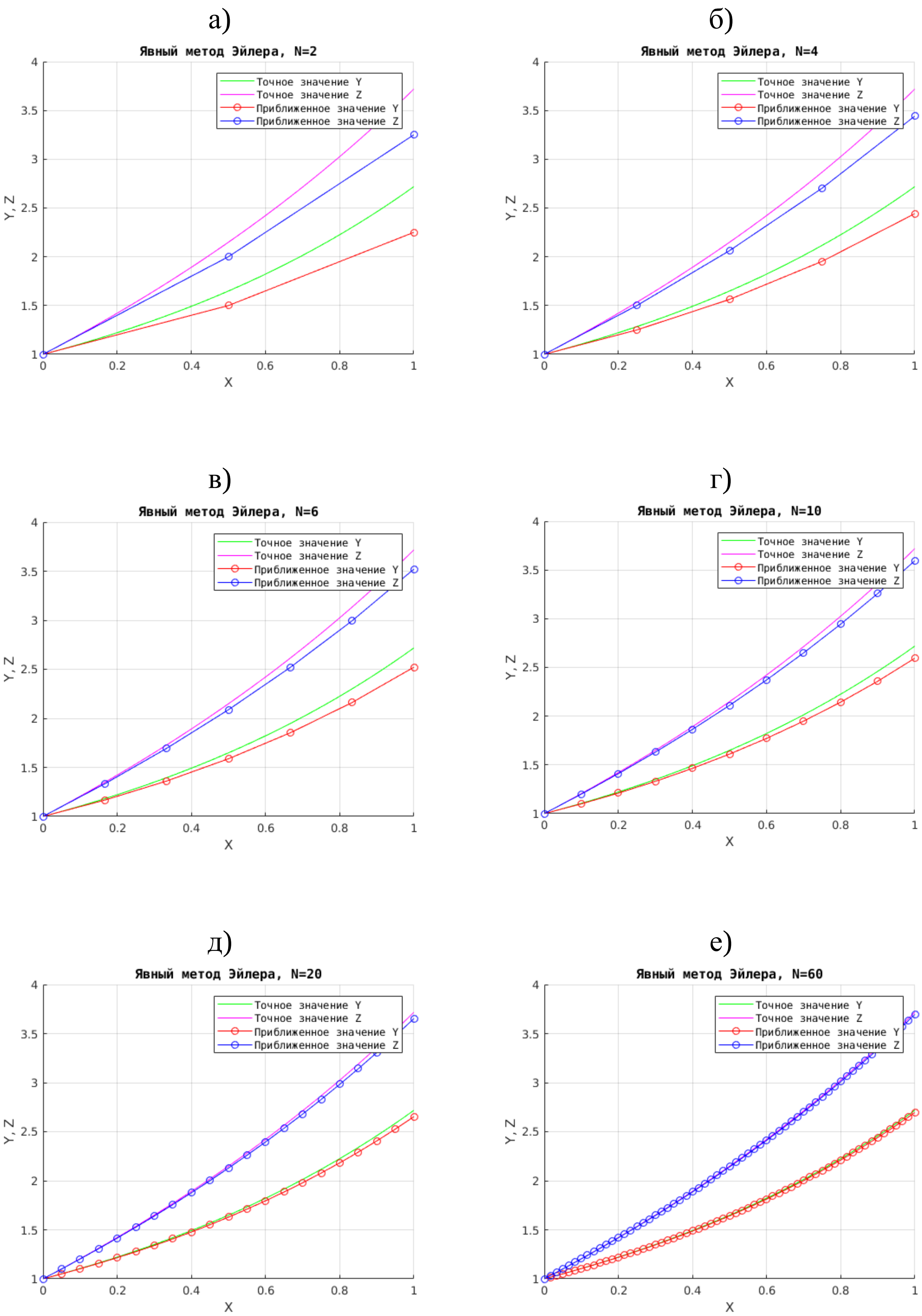


Рисунок 1 – Сравнение точности приближения при различных значениях числа частичных отрезков разбиения в явном методе Эйлера (метод 1-го порядка)

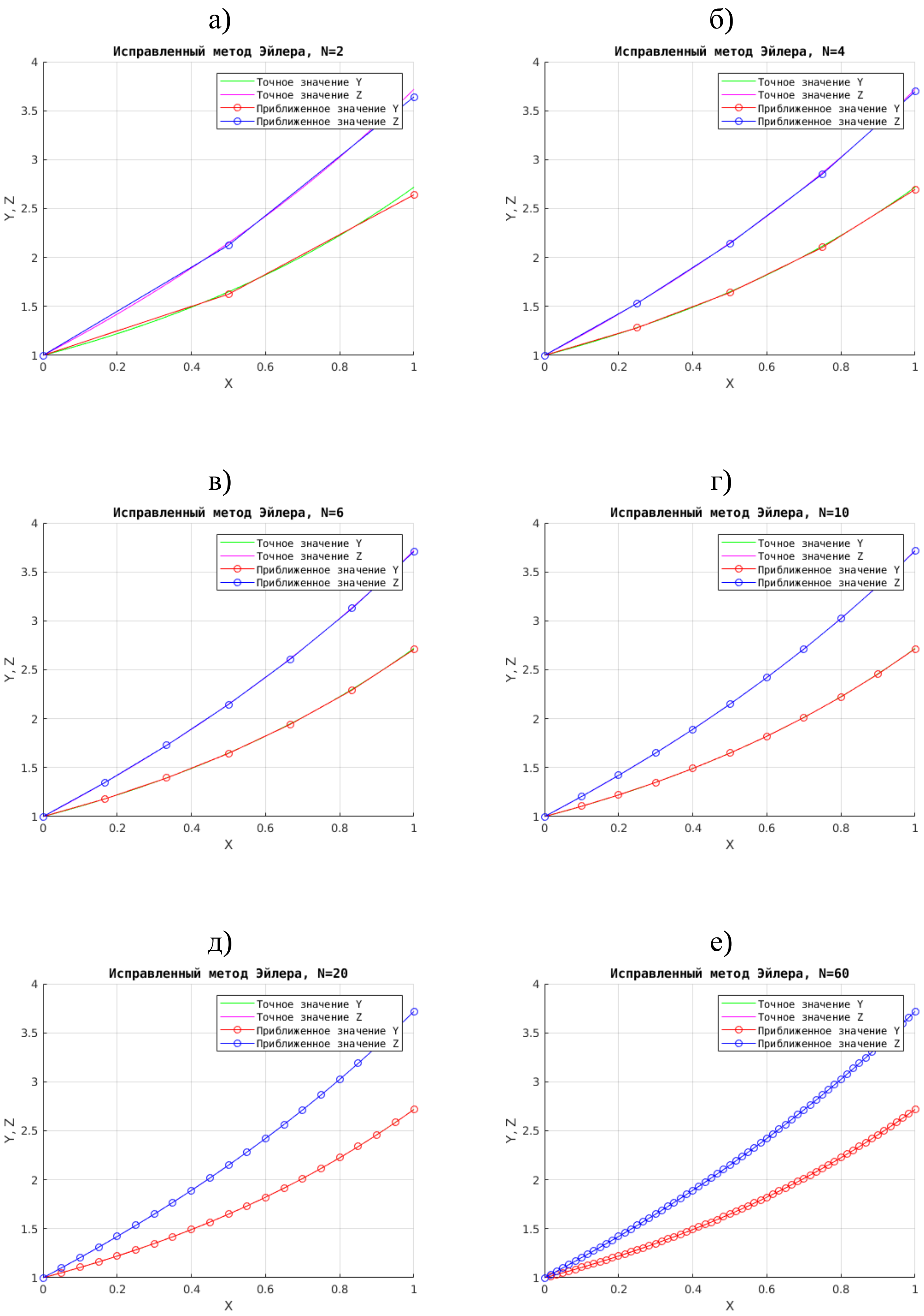


Рисунок 2 – Сравнение точности приближения при различных значениях числа частичных отрезков разбиения в исправленном методе Эйлера (2-го порядка)

Изменяя число частичных отрезков разбиения N 2 до 60, были получены сравнительные графики (рис. 1 и 2). Их анализ показал, прежде всего, что ***приближенное решение задачи Коши методами Эйлера обладает сходимостью, которая тем выше, чем больше число частичных отрезков разбиения***. При этом приближение исправленным методом Эйлера (рис.4) сходится к точному решению быстрее, чем приближение явным методом (рис.3), что вполне логично, учитывая разный порядок точности этих методов.

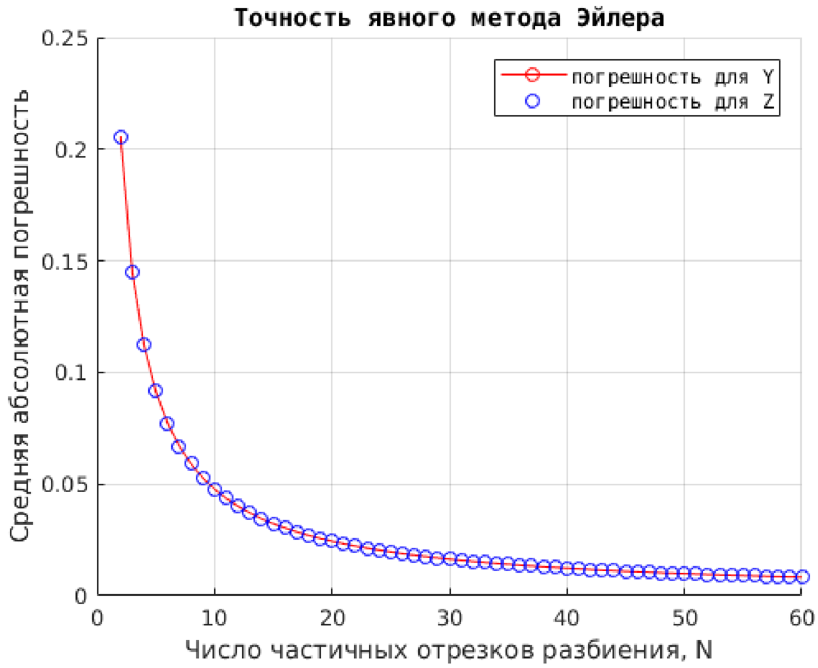


Рисунок 3 – Зависимость средней ошибки приближения от числа частичных отрезков разбиения, явный метод Эйлера

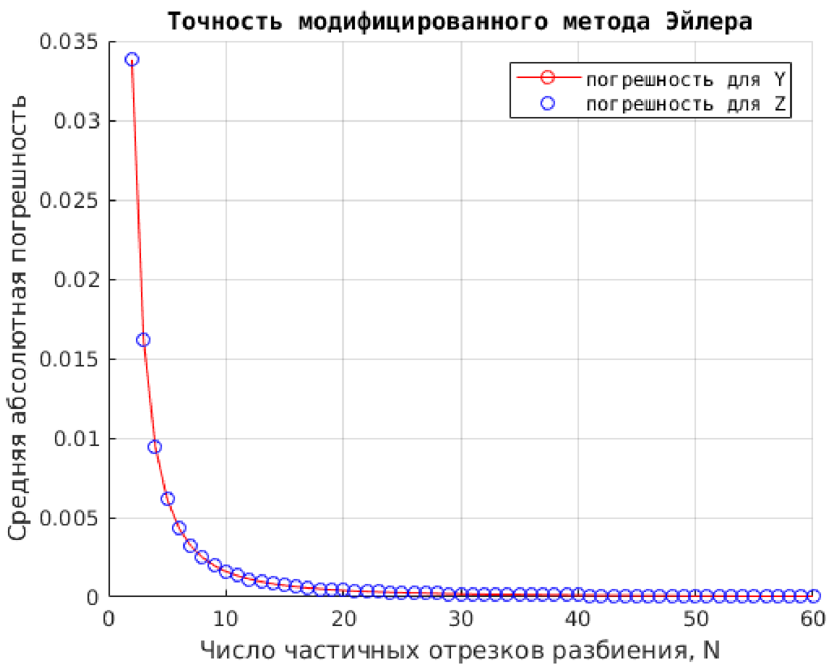


Рисунок 4 – Зависимость средней ошибки приближения от числа частичных отрезков разбиения, исправленный метод Эйлера:

Как видно из рис.4, точность явного метода Эйлера даже при достаточно малых значениях числа частичных отрезков разбиения на порядок выше точности неявного метода и обладает большей скоростью сходимости.

Явный метод Эйлера не обладает такой быстрой сходимостью (рис. 3), но, тем не менее, сходимость вполне логично прослеживается, что особенно заметно на больших значениях N (более 30). Здесь большую роль в формировании суммарной ошибки играет погрешность самого метода, нежели ошибки округления.

Таким образом, в случае использования исправленного метода в качестве достаточного числа частичных отрезков разбиения можно рекомендовать N = 2…5; в отношении явного метода Эйлера необходимо большее число частичных отрезков разбиения, от 20 и выше в зависимости от задачи и требований к погрешности.

**Выводы по работе**

1. В ходе лабораторной работе рассмотрены способы приближенного решения задачи Коши для системы обыкновенных дифференциальных уравнений методами Эйлера первого и второго порядка, а также разработаны функциональные модули для системы MatLab, с помощью которых можно проводить такое приближенное решение и строить графики.

2. В результате численного эксперимента установлено, что приближенное решение задачи Коши методами Эйлера обладает сходимостью, которая тем выше, чем больше число частичных отрезков разбиения.

3. Методы более высокого порядка точности, в частности, исправленный метод Эйлера, сходятся к точному решению гораздо быстрее, чем приближение методами первого порядка. Суммарная погрешность обусловлена как использованием приближенных методов расчета, так и ошибками округления.

4. Исправленный метод Эйлера в силу особенностей вычисления требует использования дополнительных ресурсов ЭВМ. Тем не менее, он гораздо эффективнее явного метода и менее ресурсоемок, чем неявный метод Эйлера, поэтому его использование имеет ряд преимуществ перед методами первого порядка точности.

5. В случае использования исправленного метода для поставленной в работе задачи Коши в качестве достаточного числа частичных отрезков разбиения можно рекомендовать N = 2…5; в отношении явного метода Эйлера необходимо большее число частичных отрезков разбиения, от 20 выше.

## **Список литературы**

1. *Кетков, Ю.Л. и др.* Matlab 7: программирование, численные методы / Ю.Л. Кетков, А.Ю. Кетков, М.М. Шульц, - СпБ.: БХВ-Петербург, 2005 г. – 752 с.

# Приложение (листинг)

| Модуль MatLab **F1.m** |
|---|

```
function F1=F1(X, Y, Z)

F1 = Y.^2./(Z-X);

end
```

| Модуль MatLab **F2.m** |
|---|

```
function F2=F2(X, Y, Z) % тут X и Z лишь в качестве заглушки

F2 = Y+1;

end
```

| Модуль MatLab **F4.m** |
|---|

```
function [Y, Z]=F4(X)

% Точное решение

Y = exp(X);

Z = X + exp(X);

end
```

| Модуль MatLab **EilerCorrectedMethod.m** |
|---|

```
function [X, Y, Z] = EilerCorrectedMethod(a, b, N, stPy, stPz)

% Функция для приближения исправленным методом Эйлера

% a - начало отрезка

% b - конец отрезка

% N - количество отрезков разбиения

% stPy, stPz - начальные условия, то есть значения Y и Z в точке X=a

h = abs(b-a)/N; % Расстояние между узлами

X = a:h:b; % Формируем вектор X узловых точек

Y = X * 0; % Формируем пока еще пустой вектор Y

Z = Y; % Формируем пока пустой вектор Z
```

```
% Подставляем начальные условия:
Y(1) = stPy;
Z(1) = stPz;
% Начинаем обходить последовательно все точки и вычисляем значения на
% текущем шаге исходя из данных предыдущего шага:
for i = 2 : N+1
Ya = Y(i-1) + h * F1(X(i-1), Y(i-1), Z(i-1)); % явное
Za = Z(i-1) + h * F2(X(i-1), Y(i-1), Z(i-1)); % явное
Y(i) = Y(i-1) + h/2 * (F1(X(i-1), Y(i-1), Z(i-1)) + F1(X(i), Ya, Za));
Z(i) = Z(i-1) + h/2 * (F2(X(i-1), Y(i-1), Z(i-1)) + F2(X(i), Ya, Za));
end
end
```

## Модуль MatLab **EilerExplicitMethod.m**

```
function [X, Y, Z] = EilerExplicitMethod(a, b, N, stPy, stPz)
% Функция для приближения явным методом Эйлера
% a - начало отрезка
% b - конец отрезка
% N - количество отрезков разбиения
% stPy, stPz - начальные условия, то есть значения Y и Z в точке X=a
h = abs(b-a)/N; % Расстояние между узлами
X = a:h:b; % Формируем вектор X узловых точек
Y = X * 0; % Формируем пока еще пустой вектор Y
Z = Y; % Формируем пока пустой вектор Z
% Подставляем начальные условия:
Y(1) = stPy;
Z(1) = stPz;
% Начинаем обходить последовательно все точки и вычисляем значения на
% текущем шаге исходя из данных предыдущего шага:
for i = 2 : N+1
```

```
Y(i) = Y(i-1) + h * F1(X(i-1), Y(i-1), Z(i-1));

Z(i) = Z(i-1) + h * F2(X(i-1), Y(i-1), Z(i-1));

end

end
```

## Модуль MatLab **deltaEiler.m**

```
function d = deltaEiler(Y1, Y2)

% Функция вычисления погрешности между значениями функции и значениями

% приближения

n = length(Y1); % Количество узловых точек

D = Y1 * 0; % Матрица разности

for i = 1 : n

D(i) = Y1(i) - Y2(i);

end

d = 0;

for i = 1 : n

d = d + abs(D(i));

end

d = d / n;

end
```

## Модуль MatLab **plotEiler.m**

```
function plotEiler(a, b, N, stPy, stPz, type)

% Основная функция, вычисляет приближение к функциям по одному из

% методов Эйлера (явного или неявного) и строит сравнительные графики для исходной функции и

% приближающей функции, с нанесением узловых точек

% a - начало отрезка

% b - конец отрезка

% N - количество отрезков разбиения

% stPy, stPz - начальные условия, то есть значения Y и Z в точке X=a,
```

```
% type - вариант метода Эйлера (0 - если явный, 1 - неявный)

% 1 Создаем новое окно для графика и подписываем оси

figure;

xlabel('X');

ylabel('Y, Z');

hold on;

grid on;

% 2 Находим приближающие значения метдом Эйлера:

X = zeros(1);

Y = X;

Z = X;

if (type == 0) % Если явный метод, то

[X, Y, Z] = EilerExplicitMethod(a, b, N, stPy, stPz);

titleValue = 'Явный';

end

if (type == 1) % Если НЕявный метод, то

[X, Y, Z] = EilerImplicitMethod(a, b, N, stPy, stPz);

titleValue = 'Неявный';

end

if (type == 2) % Если исправленный метод, то

[X, Y, Z] = EilerCorrectedMethod(a, b, N, stPy, stPz);

titleValue = 'Исправленный';

end

if (type == 3) % Если модифицированный метод, то

[X, Y, Z] = EilerModifiedMethod(a, b, N, stPy, stPz);

titleValue = 'Модифицированный';

end

titleValue = strcat(titleValue, ' метод Эйлера, N=', int2str(N));

% 3 Считаем и печатаем график точных функций:

X1 = a:0.001:b;

[Yt, Zt] = F4(X1);
```

```
plot(X1, Yt, 'green');

plot(X1, Zt, 'm');

% 4 Печатаем графики приближений:

plot (X, Y, 'r-o');

plot (X, Z, 'b-o');

% 5 Подписываем легенду

title(titleValue, 'FontName', 'Courier');

h1 = legend('Точное значение Y', 'Точное значение Z', 'Приближенное значение Y', 'Приближенное значение Z');

set(h1, 'FontName', 'Courier');

end
```

## Модуль MatLab **plotDeltaEiler.m**

```
function plotDeltaEiler(a, b, N_start, N_end, stPy, stPz, type)

% Функция печати графика зависимости погрешности приближения от числа

% узловых точек

% a - начало отрезка

% b - конец отрезка

% N_start - начальное количество отрезков разбиения

% N_end - конечное количество отрезков разбиения

% stPy, stPz - начальные условия, то есть значения Y и Z в точке X=a,

% type - вариант метода Эйлера (0 - если явный, 1 - неявный)

% 1 Создаем новое окно для графика и подписываем оси

figure;

xlabel('Число частичных отрезков разбиения, N');

ylabel('Средняя абсолютная погрешность');

grid on; hold on;

% 2 Выполняем приближение

j = 1;

deltaY = N_start : N_end;
```

```
deltaY = deltaY * 0;

deltaZ = deltaY;

for N = N_start : N_end

if (type == 0)

[X, Y, Z] = EilerExplicitMethod(a, b, N, stPy, stPz);

titleValue = ' явного';

end

if (type == 1)

[X, Y, Z] = EilerImplicitMethod(a, b, N, stPy, stPz);

titleValue = ' неявного';

end

if (type == 2)

[X, Y, Z] = EilerCorrectedMethod(a, b, N, stPy, stPz);

titleValue = ' исправленного';

end

if (type == 3)

[X, Y, Z] = EilerModifiedMethod(a, b, N, stPy, stPz);

titleValue = ' модифицированного';

end

[Yt, Zt] = F4(X);

deltaY(j) = deltaEiler(Yt, Y);

deltaZ(j) = deltaEiler(Zt, Z);

j = j + 1;

end

titleValue = strcat('Точность', titleValue, ' метода Эйлера');

X = N_start : N_end;

% 3 Печатаем графики погрешностей:

plot(X, deltaY, 'r-o');

plot(X, deltaZ, 'bo');

% 4 Подписываем легенду

title(titleValue, 'FontName', 'Courier');
```

```
h1 = legend('погрешность для Y', 'погрешность для Z');

set(h1, 'FontName', 'Courier');

end
```