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 №1

на тему:

«Приближенное решение задачи Коши для системы обыкновенных дифференциальных уравнений методами 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 методы приближенного решения:

- явный метод Эйлера;

- неявный метод 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.

Явный метод Эйлера основан на аппроксимации производной простейшей разностной схемой вида:

$$y' = \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} \approx \frac{y_{k+1} - y_k}{x_{k+1} - x_k} \qquad (7)$$

Отсюда в силу решаемого дифференциального уравнения y ' = f(x,y) выводится разностное уравнение метода:

$$y_{k+1} = y_k + f(x_k, y_k) \cdot \Delta x \qquad (8)$$

Небольшую неизбежную погрешность при такой аппроксимации можно обеспечить только малым шагом интегрирования Δx. Поэтому численное решение задачи Коши на достаточно большом промежутке изменения аргумента – очень кропотливая процедура, немыслимая без вычислительной техники.

Неявный метод Эйлера является, с идеологической точки зрения, совсем небольшим усложнением явного метода, в котором разностное уравнение имеет вид:

$$y_{k+1} = y_k + f(x_{k+1}, y_{k+1}) \cdot \Delta x \qquad (9)$$

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).

На первом этапе в среде MatLab реализованы алгоритмы приближения явным и неявным методами Эйлера; алгоритмы оформлены отдельными программными модулями в следующих файлах:

- *EilerExplicitMethod.m* – модуль, реализующий приближение явным методом Эйлера;
- *EilerImplicitMethod.m* – модуль, реализующий приближение неявным методом Эйлера;
- *F1.m* – модуль, реализующий вычисление функции y'(x) по ф.(1);
- *F2.m* – модуль, реализующий вычисление функции z'(x) по ф.(2);
- *F3.m* – модуль, необходимый для работы неявного метода Эйлера, возвращает значения функций ф.(1) и 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.

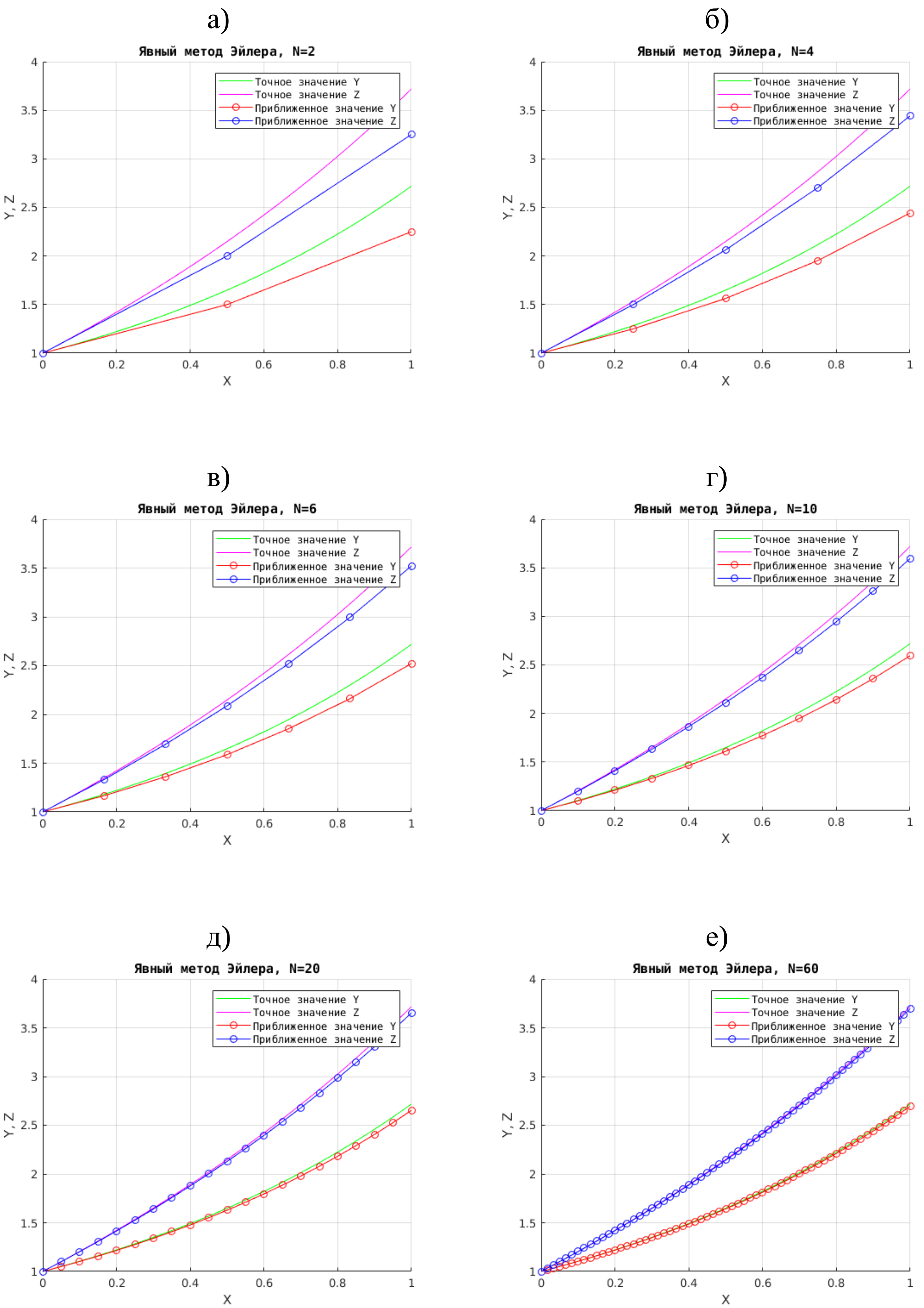

Рисунок 1 – Сравнение точности приближения при различных значениях числа частичных отрезков разбиения в явном методе Эйлера

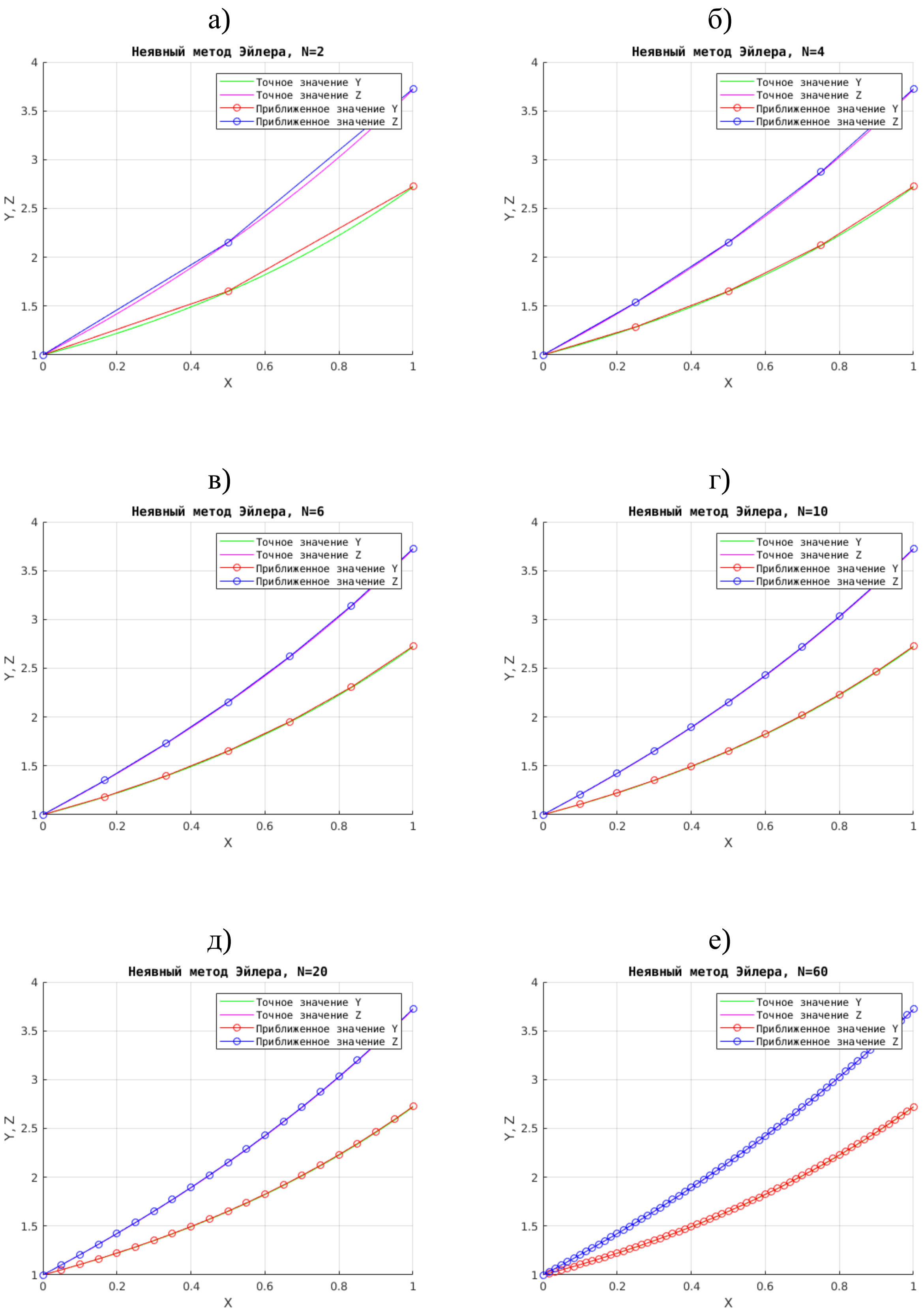

Рисунок 2 – Сравнение точности приближения при различных значениях числа частичных отрезков разбиения в неявном методе Эйлера

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 неявным методом Эйлера (рис.4) сходится к точному решению быстрее, чем приближение явным методом (рис.3):

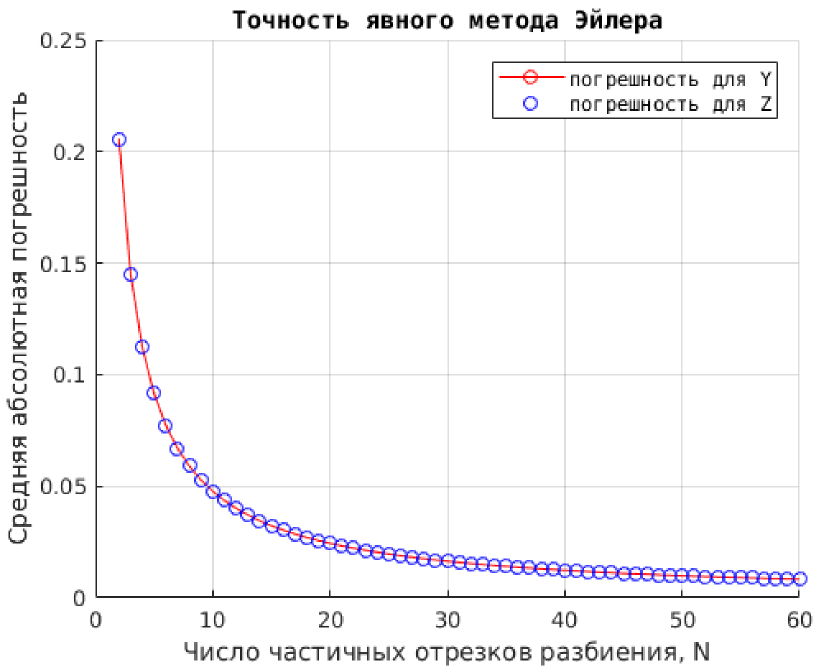

Рисунок 3 – Зависимость средней ошибки приближения
от числа частичных отрезков разбиения, явный метод Эйлера

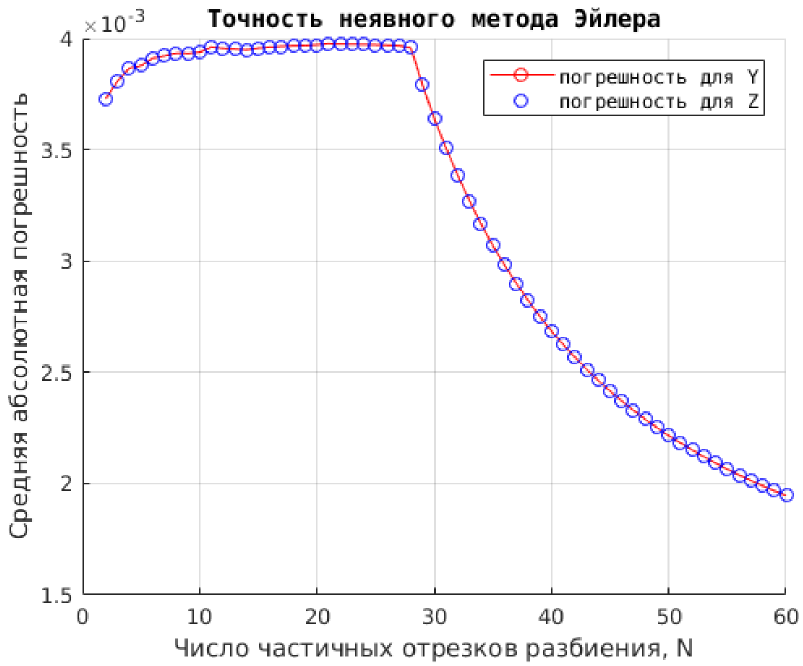

Рисунок 4 – Зависимость средней ошибки приближения
от числа частичных отрезков разбиения, неявный метод Эйлера:

Как видно из рис.4, точность неявного метода Эйлера даже при достаточно малых значениях числа частичных отрезков разбиения не превышает значения $4*10^{-3}$. Некоторое снижение точности (до отметки $N \approx 28$) в случае неявного метода объясняется, очевидно, накоплением ошибки округления, связанной с использованием внутреннего алгоритма метода касательных Ньютона, которая, однако, затем начинает снижаться.

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 неявного 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, 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. При этом приближение неявным методом Эйлера сходится к точному решению быстрее, чем приближение явным методом. Суммарная погрешность обусловлена как использованием приближенных методов расчета, так и ошибками округления.

3. Наилучшим в плане сходимости способом приближения является неявный метод Эйлера; в силу особенностей расчета он дает хорошую сходимость при меньших значениях n, чем явный метод.

4. Неявный метод Эйлера в силу особенностей вычисления требует программирования на ЭВМ дополнительных вложенных циклов с реализацией метода касательных Ньютона, что налагает дополнительные требования используемым ресурсам. Тем не менее, в тех случаях, когда требуется достижение высокой точности приближения при имеющемся малом количестве отрезков разбиения, он оказывается гораздо предпочтительнее.

5. В случае использования неявного 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 **F3.m**

```
f function F3=F3(X, Y)

F3 = [Y(1).^2./(Y(2)-X); Y(1)+1];

end
```

Модуль MatLab **F4.m**

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

% Точное решение

Y = exp(X);

Z = X + exp(X);

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 **EilerImplicitMethod.m**

```
function [X, Y, Z] = EilerImplicitMethod(a, b, N, stPy, stPz)
% Функция для приближения неявным 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
% Используем встроенную функцию неявного метода, она возвращает два массива:
% в первом - значения X, во втором - столбцы решений Y и Z для этих X
[X, Y]=ode15s('F3',X,[stPy,stPz]); % <- подставляем начальные условия
Z = Y(:, 2); % Вытаскиваем второй столбец
Y = Y(:, 1); % и первый столбец
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

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

[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
```