ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ
ЛЕКЦИЯ №8
«Численное решение обыкновенных дифференциальных уравнений. Модифицированные методы»

Отделение ядерно-топливного цикла

Лектор:
Зав. каф. - руководитель ОЯТЦ ИЯТШ
Горюнов А.Г.

2020

План лекции

8.1 Модификации метода Эйлера, явная и неявная схема.

8.2 Семейство методов Рунге-Кутты.

8.3 Метод Рунге-Кутты 4-го порядка.

8.4 Пошаговый контроль точности.

8.5 Решения дифференциальных уравнений в Matlab.

Информация по курсу:

https://portal.tpu.ru/SHARED/a/ALEX1479/study/Matmod/Tab

8.1 Модификации метода Эйлера, явная и неявная схема

8.1.1 Модификации метода Эйлера

Метод Эйлера состоит в приближенной замене производной в уравнении $y' = f(x, y)$ ее разностным аналогом. Тогда дифференциальное уравнение можно записать в виде:

$$\frac{y_{i+1} - y_i}{x_{i+1} - x_i} = f\left(x_i, y_i\right) \tag{8.1}$$

Расчетная формула метода Эйлера имеет следующий вид:

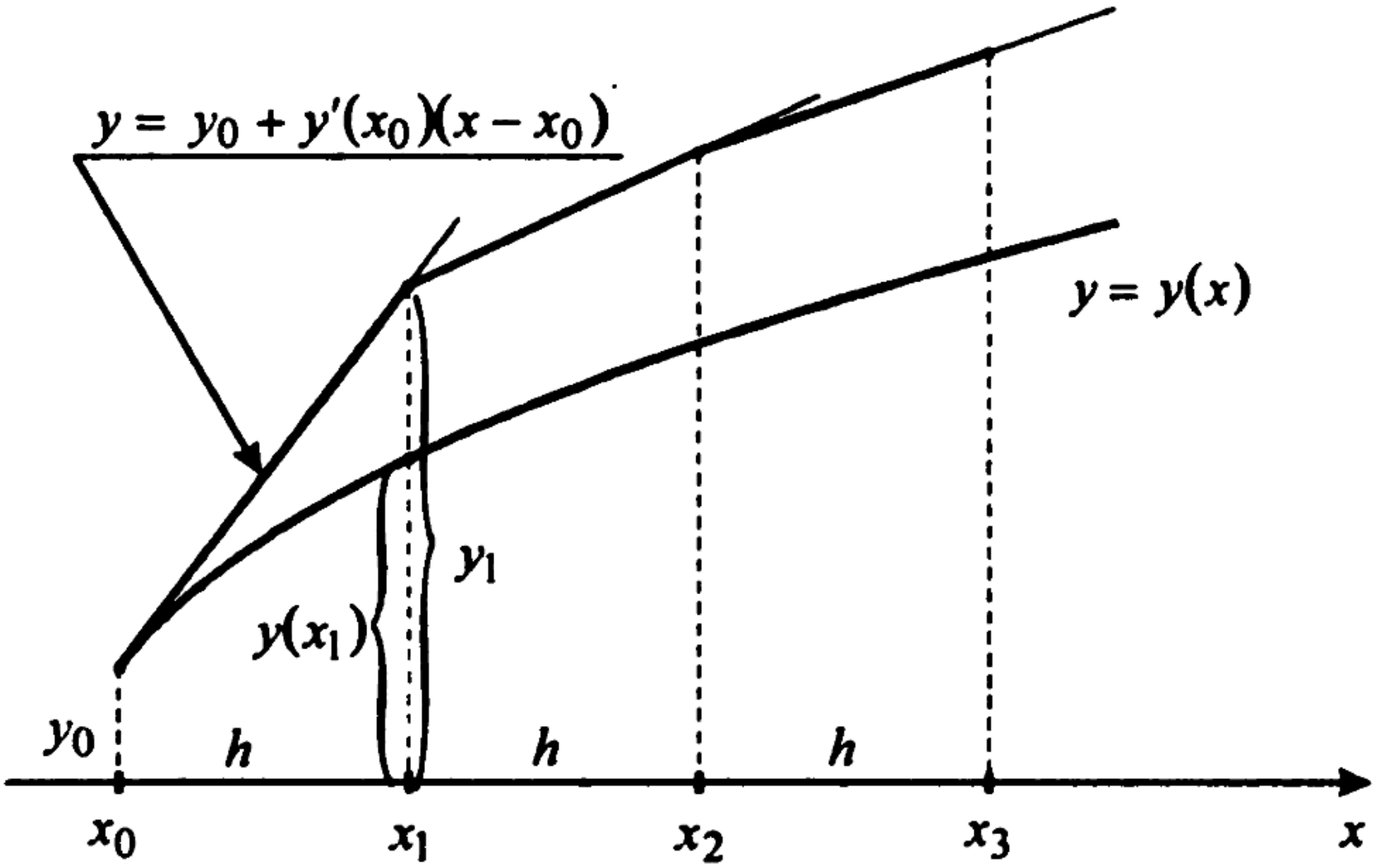
$$\begin{aligned} y_{i+1} &= y_i + h \cdot f\left(x_i, y_i\right) \\ x_{i+1} &= x_i + h, \qquad \left(i = 0,1,2,\ldots,n-1\right) \end{aligned} \tag{8.2}$$

Результатом численного решения задачи Коши будет таблица значений:

x	x_0	x_1	x_2	...	x_n
y	y_0	y_1	y_2	...	y_n

8.1.1 Модификации метода Эйлера

Геометрическая интерпретация метода Эйлера:

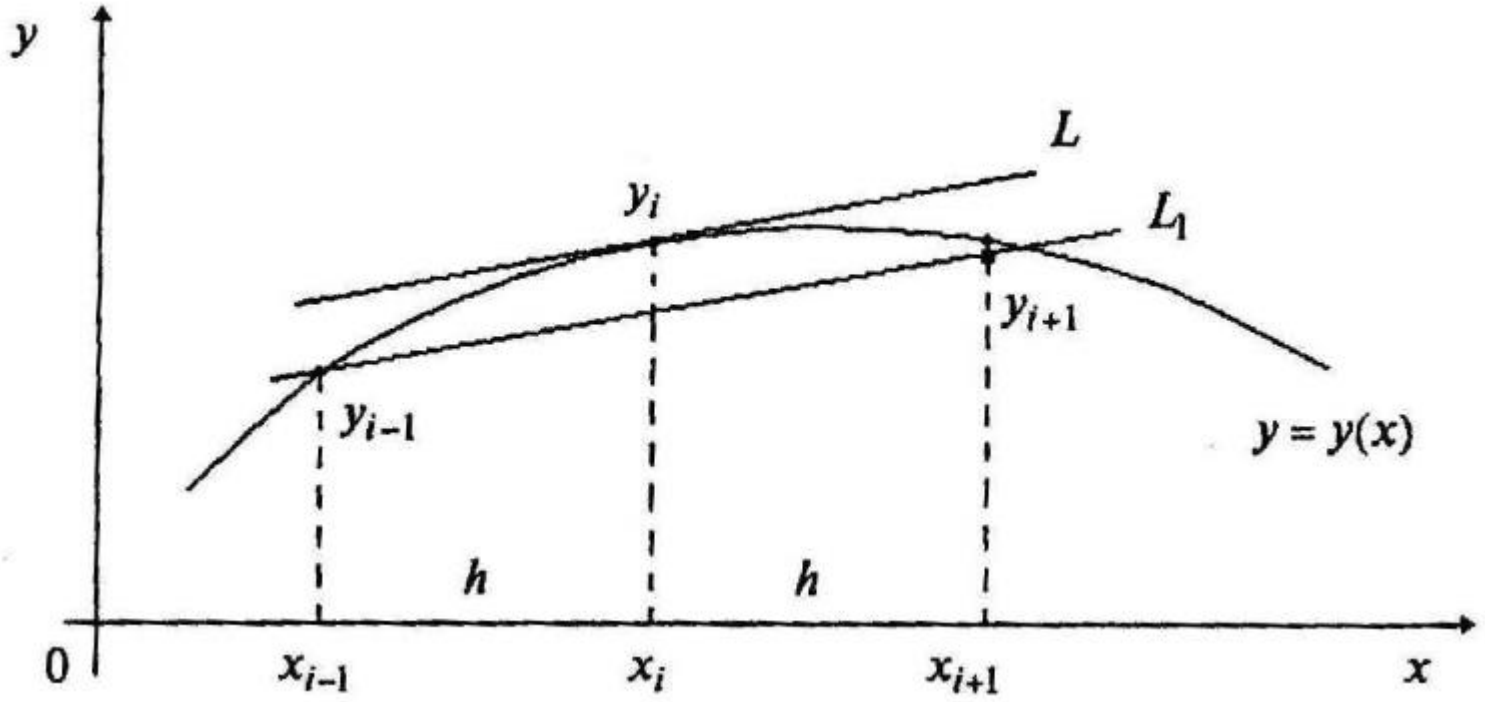


8.1.1 Модификации метода Эйлера

Одной из модификаций является явный метод Эйлера-Коши или уточненный метод Эйлера, применяемый с целью повышения точности расчета:

$$y_{i+1} = y_{i-1} + 2 \cdot h \cdot f\left(x_i, y_i\right), \quad \left(i = 0,1,2,\ldots,n-1\right) \tag{8.3}$$

Геометрическая интерпретация метода Эйлера-Коши:



8.1.1 Модификации метода Эйлера

Рассмотрим еще одну модификацию метода Эйлера. Идея модификации: отрезки должны быть параллельны касательным, которые проведены к графику функции не на левых краях, а «посерединке» интервалов разбиения. Это существенно улучшит качество приближения.

В этом случае формула усложняется:

$$y_{i+1} = y_i + h \cdot f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2} f\left(x_i, y_i\right) \right) \qquad (8.4)$$

Модифицированные методы Эйлера имеют второй порядок точности.

8.1.2 Неявный метод Эйлера

При построении неявного метода Эйлера значение функции *f* вычисляется на *i*+1 шаге, т.е. для решении задачи Коши используется следующее выражение:

$$y_{i+1} = y_i + h \cdot f\left(x_{i+1}, y_{i+1}\right) \qquad (8.5)$$

$$x_{i+1} = x_i + h, \qquad \left(i = 0,1,2,\ldots,n-1\right)$$

Таким образом, для нахождения приближенного значения искомой функции на *i*+1 шаге необходимо решить **нелинейное уравнение** относительно y_{i+1}:

$$y_{i+1} - h \cdot f\left(x_{i+1}, y_{i+1}\right) - y_i = 0 \qquad (8.6)$$

Для решения уравнения (8.6) необходимо использовать численные методы, например, метод Ньютона.

8.1.3 Пример жесткого ОДУ

Рассмотрим пример из литературы, относящийся к динамике. Предположим, что есть частица, с координатами ($x(t)$,$y(t)$), и требуется чтобы ее y-координата всегда оставалась равной нулю. Один из способов добиться этого — добавить слагаемое $-k \cdot y(t)$, к производной $y'(t)$, где k — большая положительная постоянная. Если k достаточно велико, то частица никогда не уйдет далеко от $y(t) = 0$, так как слагаемое $k \cdot y(t)$ всегда приведет $y(t)$ обратно к нулю. Для того, чтобы перемещать частицу как угодно вдоль оси x необходимо дифференциальное уравнение следующего вида:

$$\dot{X}(t) = \begin{cases} \dfrac{dx(t)}{dt} = -x(t) \\ \dfrac{dy(t)}{dt} = -k \cdot y(t) \end{cases} = \begin{pmatrix} -x(t) \\ -k \cdot y(t) \end{pmatrix} \qquad (8.7)$$

Предполагаем, что $y_0 \neq 0$. В этом случае частица будет сильно притягиваться к прямой $y = 0$, и менее сильно — к $x = 0$. Если решать ОДУ на достаточно продолжительном интервале времени, то частица рано или поздно попадет в точку (0,0) и останется в ней.

8.1.3 Пример жесткого ОДУ

Применим метод Эйлера:

$$X_{i+1} = X_i + h \cdot X_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + h \begin{pmatrix} -x_i \\ -k \cdot y_i \end{pmatrix} \tag{8.8}$$

или

$$X_{i+1} = \begin{pmatrix} x_i - h \cdot x_i \\ y_i - k \cdot y_i \end{pmatrix} = \begin{pmatrix} (1-h)\, x_i \\ (1-h \cdot k)\, y_i \end{pmatrix} \tag{8.9}$$

По *y*-компоненте этого уравнения, видно, что при $|1-hk|>1$, вычисленное y_{i+1} будет по модулю больше, чем $|y_i|$. Другими словами, при $|1-hk|>1$ метод Эйлера будет неустойчив: каждый шаг приводит к увеличению y_{i+1} по сравнению с предыдущим значением и приближенное решение будут все дальше отклоняться от нуля. Таким образом, для обеспечения устойчивости метода Эйлера необходимо, чтобы $1-hk>-1$ или $hk<2$. Самый большой шаг, который можно сделать не нарушив устойчивости, должен быть меньше $2/k$.

Теперь, если *k* – большое число, то придется делать очень маленькие шаги. Это означает, что частица будет двигаться к точке (0,0) очень медленно (при моделировании).

8.1.4 Решение жесткого ОДУ

Пусть дано дифференциальное уравнение в векторной форме:

$$\frac{d}{dt}X(t) = f\left(X(t)\right) \tag{8.9}$$

X(t) – вектор столбец.

Формула Эйлера для этого уравнения:

$$X\left(t_{i+1}\right) = X\left(t_i\right) + h \cdot f\left(X\left(t_i\right)\right)$$

или

$$X_{i+1} = X_i + h \cdot f\left(X_i\right) \tag{8.10}$$

Воспользуемся неявной формулой Эйлера:

$$X_{i+1} = X_i + h \cdot f\left(X_{i+1}\right) \tag{8.11}$$

Для того, чтобы решить нелинейное уравнение (8.11) заменим $f(X_{i+1})$ линейной аппроксимацией, основанной на разложении f в ряд Тейлора. Введем обозначение $\Delta X_{i+1} = X_{i+1} - X_0$. Подставив его в уравнение (8.11), получим

8.1.4 Решение жесткого ОДУ

$$X_i + \Delta X_{i+1} = X_i + h \cdot f\left(X_i + \Delta X_{i+1}\right)$$

или

$$\Delta X_{i+1} = h \cdot f\left(X_i + \Delta X_{i+1}\right) \qquad (8.12)$$

Теперь заменим $f(X_i+\Delta X_{i+1})$ следующим приближением (используя разложение в ряд Тейлора)

$$f\left(X_i\right) + f'\left(X_i\right)\Delta X_{i+1}$$

Поскольку $f(X_i)$ является вектором, то производная $f'(X_i)$ является матрицей. Используя это приближение, мы можем записать ΔX_{i+1} как

$$\Delta X_{i+1} = h \cdot \left(f\left(X_i\right) + f'\left(X_i\right)\Delta X_{i+1}\right) \qquad (8.13)$$

или

$$\Delta X_{i+1} - h \cdot f'\left(X_i\right)\Delta X_{i+1} = f\left(X_i\right)$$

8.1.4 Решение жесткого ОДУ

Разделим обе части (8.13) на *h* получим уравнение в матричной форме:

$$\left(\frac{1}{h}I - f'\left(X_i\right)\right)\Delta X_{i+1} = f\left(X_i\right), \tag{8.14}$$

где *I* — единичная матрица (1 на диагонали, остальные 0).

Разрешая это соотношение относительно ΔX_{i+1}, получим:

$$\Delta X_{i+1} = \left(\frac{1}{h}I - f'\left(X_i\right)\right)^{-1} f\left(X_i\right). \tag{8.15}$$

Применим неявный метод для решения уравнения (8.7). В этом случае $f(X(t))$ равно

$$f\left(X\left(t\right)\right) = \begin{pmatrix} -x\left(t\right) \\ -k \cdot y\left(t\right) \end{pmatrix}$$

Дифференцирование по *X* дает

$$f'\left(X\left(t\right)\right) = \frac{\partial}{\partial X} f\left(X\left(t\right)\right) = \begin{pmatrix} -1 & 0 \\ 0 & -k \end{pmatrix}. \tag{8.16}$$

8.1.4 Решение жесткого ОДУ

Тогда матрица $(1/h)I - f'(X_i)$ будет равна

$$\begin{pmatrix} \frac{1}{h}+1 & 0 \\ 0 & \frac{1}{h}+k \end{pmatrix} = \begin{pmatrix} \frac{1+h}{h} & 0 \\ 0 & \frac{1+k\cdot h}{h} \end{pmatrix}. \qquad (8.17)$$

Обращая эту матрицу и умножая на $f(X_i)$, получим

$$\Delta X_{i+1} = \begin{pmatrix} \frac{1+h}{h} & 0 \\ 0 & \frac{1+k\cdot h}{h} \end{pmatrix}^{-1} \begin{pmatrix} -x_i \\ -k\cdot y_i \end{pmatrix} = \begin{pmatrix} \frac{h}{1+h} & 0 \\ 0 & \frac{h}{1+k\cdot h} \end{pmatrix} \begin{pmatrix} -x_i \\ -k\cdot y_i \end{pmatrix} = -\begin{pmatrix} \frac{h}{1+h}x_i \\ \frac{h}{1+k\cdot h}k\cdot y_i \end{pmatrix}.$$

В итоге получаем:

$$X_{i+1} = \begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} - \begin{pmatrix} \frac{h}{1+h}x_i \\ \frac{h}{1+k\cdot h}k\cdot y_i \end{pmatrix} \qquad (8.18)$$

В общем случае для жесткого ОДУ мы не сможем сделать шаг произвольного размера, но мы сможем сделать его **гораздо большим**, чем если бы использовали явный метод.

8.2 Семейство методов Рунге-Кутты

Для построения разностной схемы интегрирования воспользуемся разложением функции:

$$\frac{dy}{dx} = f\left(x, y(x)\right), \quad 0 < x \le T, \quad y(0) = y_0 \tag{8.19}$$

в ряд Тейлора:

$$y\left(x_{i+1}\right) = y\left(x_i\right) + y'\left(x_i\right)h + y''\left(x_i\right)\frac{h^2}{2} + \ldots$$

Заменим вторую производную в этом разложении выражением

$$y''\left(x_i\right) = \left(y'\left(x_i\right)\right)' = f'\left(x_i, y\left(x_i\right)\right) \approx \frac{f\left(\tilde{x}, \tilde{y}\right) - f\left(x_i, y\left(x_i\right)\right)}{\Delta x}$$

где

$$\tilde{x} = x_i + \Delta x, \quad \tilde{y} = y\left(x_i + \Delta x\right)$$

Причем Δx подбирается из условия достижения наибольшей точности записанного выражения. Для дальнейших выкладок произведем замену величины «y с тильдой» разложением в ряд Тейлора:

$$\tilde{y} = y\left(x_i + \Delta x\right) = y\left(x_i\right) + y'\left(x_i\right)\Delta x + \ldots$$

8.2 Семейство методов Рунге-Кутты

Для исходного уравнения (8.19) построим вычислительную схему:

$$y_{i+1} = y_i + f\left(x_i, y_i\right)\cdot h + \frac{h^2}{2\Delta x}\left(f\left(x_i + \Delta x, y_i + y_i'\Delta x\right) - f\left(x_i, y_i\right)\right)$$

которую преобразуем к виду:

$$y_{i+1} = y_i + h\left[\left(1 - \frac{h}{2\Delta x}\right)\cdot f\left(x_i, y_i\right) + \frac{h}{2\Delta x} f\left(x_i + \Delta x, y_i + y_i'\Delta x\right)\right] =$$

$$= y_i + h\left[\left(1 - \frac{h}{2\Delta x}\right)\cdot f\left(x_i, y_i\right) + \frac{h}{2\Delta x} f\left(x_i + \frac{\Delta x}{h}h, y_i + f\left(x_i, y_i\right)\frac{\Delta x}{h}h\right)\right]$$

Введем следующие обозначения:

$$\alpha = \frac{h}{2\Delta x},\quad \beta = 1 - \frac{h}{2\Delta x},\quad \gamma = \frac{\Delta x}{h},\quad \delta = f\left(x_i, y_i\right)\frac{\Delta x}{h}$$

Эти обозначения позволяют записать предыдущее выражение в форме:

$$y_{i+1} = y_i + h\left[\beta\cdot f\left(x_i, y_i\right) + \alpha\cdot f\left(x_i + \gamma\cdot h, y_i + \delta\cdot h\right)\right] \qquad (8.20)$$

8.2 Семейство методов Рунге-Кутты

Очевидно, что все введенные коэффициенты зависят от величины Δ*x* и могут быть определены через коэффициент α, который в этом случае играет роль параметра:

$$\beta = 1-\alpha,\quad \gamma = \frac{1}{2\alpha},\quad \delta = f\left(x_i, y_i\right)\frac{2}{\alpha}$$

Окончательно схема Рунге-Кутты принимает вид:

$$y_{i+1} = y_i + h\left[\left(1-\alpha\right)\cdot f\left(x_i, y_i\right) + \alpha\cdot f\left(x_i + \frac{h}{2\alpha}, y_i + f\left(x_i, y_i\right)\frac{h}{2\alpha}\right)\right] \qquad (8.21)$$

Та же схема в форме разностного аналога уравнения (8.19):

$$\frac{y_{i+1}-y_i}{h} = \left(1-\alpha\right)\cdot f\left(x_i, y_i\right) + \alpha\cdot f\left(x_i + \frac{h}{2\alpha}, y_i + f\left(x_i, y_i\right)\frac{h}{2\alpha}\right) \qquad (8.22)$$

При α = 0 получаем как частный случай уже известную схему Эйлера:

$$y_{i+1} = y_i + h\cdot f\left(x_i, y_i\right)$$

При *α* = 1:

$$y_{i+1} = y_i + h\cdot f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}\cdot f\left(x_i, y_i\right)\right) \qquad (8.23)$$

8.3 Метод Рунге-Кутты 4-го порядка

Пусть на отрезке [*a, b*] требуется найти приближенное численное решение дифференциального уравнения:

$$y' = f\left(x, y\right) \tag{8.24}$$

при начальных условиях $y(x_0) = y_0$.

Разбиваем отрезок [*a*, *b*] на *n* равных частей точками

$$x_i = x_0 + i \cdot h \tag{8.25}$$

где $i = 0,1,2,\ldots,n$; $h = (b - a)/h$ – шаг интегрирования (решения) дифференциального уравнения.

На каждом шаге интегрирования заданного ОДУ (8.24) искомая функция $y(x)$ аппроксимируется рядом Тейлора (8.26),

$$y_{i+1} = y_i + y_i' h + \frac{y_i''}{2!} h^2 + \frac{y_i'''}{3!} h^3 + \frac{y_i^{(4)}}{4!} h^4 + \ldots \tag{8.26}$$

усеченным до члена ряда, содержащего h^4 .

8.3 Метод Рунге-Кутты 4-го порядка

Каждое последующее приближенное значение y_{i+1} искомого решения y определяется через текущее значение y_i с помощью рекуррентных формул по вычислительной схеме:

$$y_{i+1} = y_i + \Delta y_i \tag{8.27}$$

где

$$\Delta y_i = \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{8.28}$$

$$k_1 = h \cdot f\left(x_i, y_i\right) \tag{8.29}$$

$$k_2 = h \cdot f\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right) \tag{8.30}$$

$$k_3 = h \cdot f\left(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right) \tag{8.31}$$

$$k_4 = h \cdot f\left(x_i + \frac{h}{2}, y_i + k_3\right) \tag{8.32}$$

8.4 Пошаговый контроль точности

Автоматическое изменение шага в ходе решения систем дифференциальных уравнений необходимо, если решение требуется получить с заданной точностью. При высокой точности (погрешность $\varepsilon = E = 10^{-3}$) и решении в виде кривых с сильно различающейся крутизной автоматическое изменение шага обеспечивает уменьшение общего числа шагов в несколько раз, резко уменьшает вероятность возникновения числовой неустойчивости, дает более равномерное расположение точек графика кривых (решений) при их выводе на печать.

Метод Рунге — Кутта с автоматическим изменением шага заключается в том, что после вычисления $y_{j(i+1)}$ с шагом h все вычисления проводятся повторно с шагом h/2. Полученный результат $y^*_{j(i+1)}$ сравнивается с $y_{j(i+1)}$. Если с $|y_{j(i+1)} - y^*_{j(i+1)}| < e$, вычисления продолжают с шагом h, в противном случае шаг уменьшают. Если это неравенство слишком сильное, шаг, напротив, увеличивают. При той же погрешности $R \sim (h^s)$ лучшие результаты дает описанный ниже метод.

Метод Рунге — Кутта — *Мерсона с автоматическим изменением шага* обеспечивает приближенную оценку погрешности на каждом шаге интегрирования. Погрешность интегрирования имеет порядок $п^5$. Этот метод реализуется следующим алгоритмом:

8.4 Пошаговый контроль точности

1. Задается число уравнений ***N***, погрешность $\varepsilon = E$.начальный шаг интегрирования ***h = H*** и начальное значение $x=x_0$, $y_1(x_0)=y_0$, $y_2(x_0)=y_2$, …, $y_N(x_0) = y_N$

2. С помощью 5 циклов с управляющей переменной ***j****= 1, 2, ..., **N*** вычисляются коэффициенты:

$$K_{0j} = h \cdot f_j\left(x_i, y_{ji}\right)$$

$$K_{1j} = h \cdot f_j\left(x_i + \frac{1}{3}h, y_{ji} + \frac{1}{3}K_{0j}\right)$$

$$K_{2j} = h \cdot f_j\left(x_i + \frac{1}{3}h, y_{ji} + \frac{1}{6}K_{0j} + \frac{1}{6}K_{1j}\right) \qquad (8.33)$$

$$K_{3j} = h \cdot f_j\left(x_i + \frac{1}{2}h, y_{ji} + \frac{1}{8}K_{0j} + \frac{3}{8}K_{2j}\right)$$

$$K_{4j} = h \cdot f_j\left(x_i + h, y_{ji} + \frac{1}{2}K_{2j} + \frac{3}{2}K_{2j} + 2K_{3j}\right)$$

3. Находится (в последнем цикле) значение

$$y_{j(i+1)} = y_{ji} + \left(K_{0j} + 4K_{3j} + K_{4j}\right)/6 \qquad (8.34)$$

и погрешность

$$R_{j(i+1)} = \left(-2K_{0j} + 9K_{2j} - 8K_{3j} + K_{4j}\right)/30$$

8.4 Пошаговый контроль точности

4. Проверяется выполнение условий

$$\left|R_{j(i+1)}\right| \le E, \quad \left|R_{j(i+1)}\right| \ge E/30$$

Если первое условие не выполняется, делится шаг h на 2 и повторяются вычисления с п. 2, восстановив начальные значения y_{ji}. Если это условие выполняется и выполняется второе условие, значения $x_{i+1} = x_i + h$ результат фиксируется. Если второе условие не выполняется, шаг h увеличивается вдвое и вычисления опять повторяются с п. 2.

8.5 Решения дифференциальных уравнений в Matlab.

8.5.1 Аналитическое решение дифференциальных уравнений в Matlab

Найдем аналитическое решение задачи Коши для дифференциального уравнения:

$$\frac{dy}{dx} = x^3 - 2y \qquad (8.35)$$

$$y(0) = 0$$

```
1 -    clc;
2 -    dsolve('Dy = x^3 - 2*y','x') % Определяем аналитическое
3                                   % решение дифференциального уравнения
4      %Определяем частное решение диф.ур. при y(0) = 0 (решение задачи Коши)
5 -    dsolve('Dy = x^3 - 2*y','y(0) = 0','x')
```

Command Window

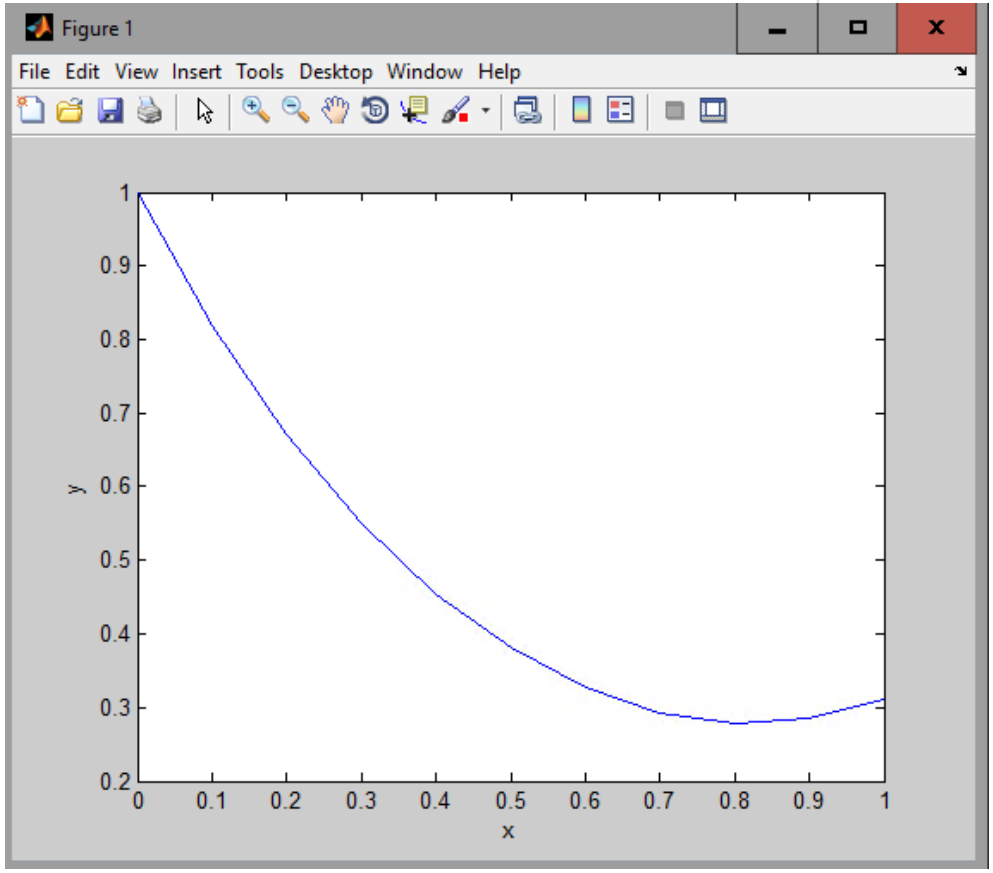
```
ans =

(3*x)/4 - (3*x^2)/4 + x^3/2 + C4*exp(-2*x) - 3/8


ans =

(3*x)/4 + (3*exp(-2*x))/8 - (3*x^2)/4 + x^3/2 - 3/8
```

8.5.2 Аналитическое решение задачи Коши для (8.35) на интервале [0,1] по *x*

```
clc;
dsolve('Dy = x^3 - 2*y','x'); % Определяем аналитическое
                              % решение дифференциального уравнения
%Определяем частное решение диф.ур. при y(0) = 0 (решение задачи Коши)
FDY = dsolve('Dy = x^3 - 2*y','y(0) = 0','x');
FDY
v = symvar(FDY); % получаем список переменых
dya = @(X) double(subs(FDY,v,X)); % создаем функцию - аналитического решения

% Решаем на интервале [0,1] по x
x = 0:0.1:1;
dya_x = dya(x);
plot(x,dya_x);
```

Command Window

```
FDY =

(3*x)/4 + (11*exp(-2*x))/8 - (3*x^2)/4 + x^3/2 - 3/8
```



8.5.3 Решение задачи Коши для (8.35) на интервале [0,1] по *x* методом Эйлера

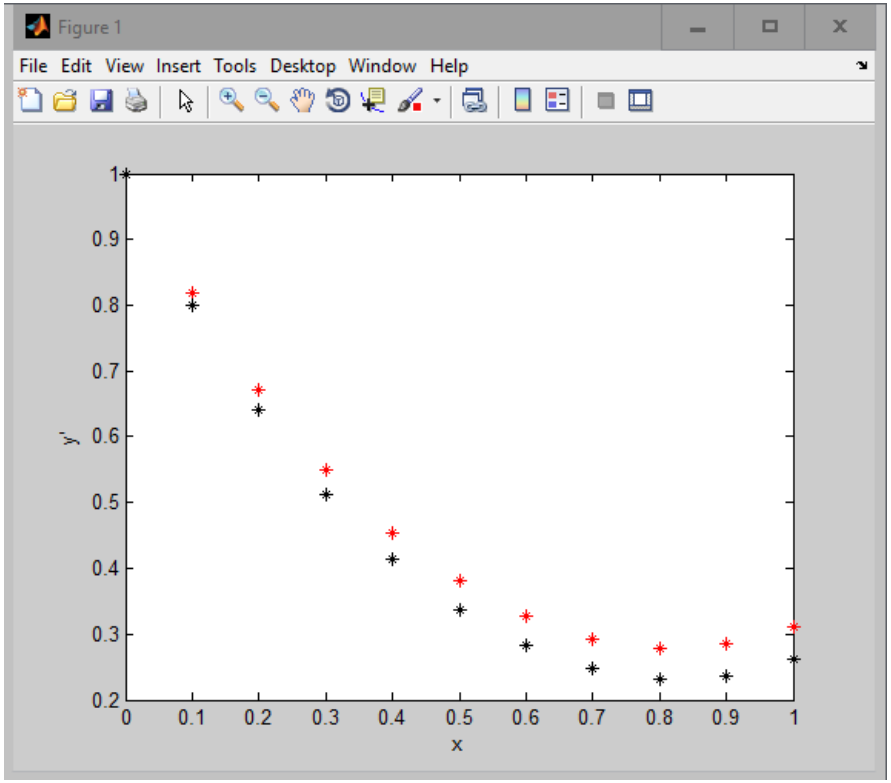
```
clc;
dsolve('Dy = x^3 - 2*y','x'); % Определяем аналитическое
                              % решение дифференциального уравнения
%Определяем частное решение диф.ур. при y(0) = 0 (решение задачи Коши)
FDY = dsolve('Dy = x^3 - 2*y','y(0) = 0','x');
FDY

v = symvar(FDY); % получаем список переменых
dya = @(X) double(subs(FDY,v,X)); % создаем функцию - аналитического решения

FY = sym('x^3 - 2*y'); %%правая часть диф.ур.
v2 = symvar(FY); % получаем список переменых
f = @(X,Y) double(subs(FY,v2,{X,Y})); % создаем функцию - аналитического решения

% Решаем на интервале [0,1] по x
x = 0:0.1:1;
dya_x = dya(x);
%plot(x,dya_x);

%Решаем методом Эйлера
y0 = 0; % начальные условия

dye_x = [];
dye_x(1) = y0;
n = length(x);
h = x(2) - x(1);
for i=1:n-1
    dye_x(i+1) = dye_x(i) + h*f(x(i),dye_x(i));
end
plot(x,dya_x,'r*',x,dye_x,'k*');

```

8.5.3 Решение задачи Коши для (8.35) на интервале [0,1] по *x* методом Эйлера

```
35    %% Определяем СКО для метода Эйлера
36 -  delta = [];
37 -  sko_e = 0;
38 -  for i=1:n
39 -      delta(i) = dya_x(i) - dye_x(i);
40 -      sko_e = sko_e + (delta(i))^2;
41 -  end
42 -  sko_e = sqrt(sko_e/(n));
43 -  sko_e = 100*sko_e/(max(dya_x) - min(dya_x))
44
```

Command Window

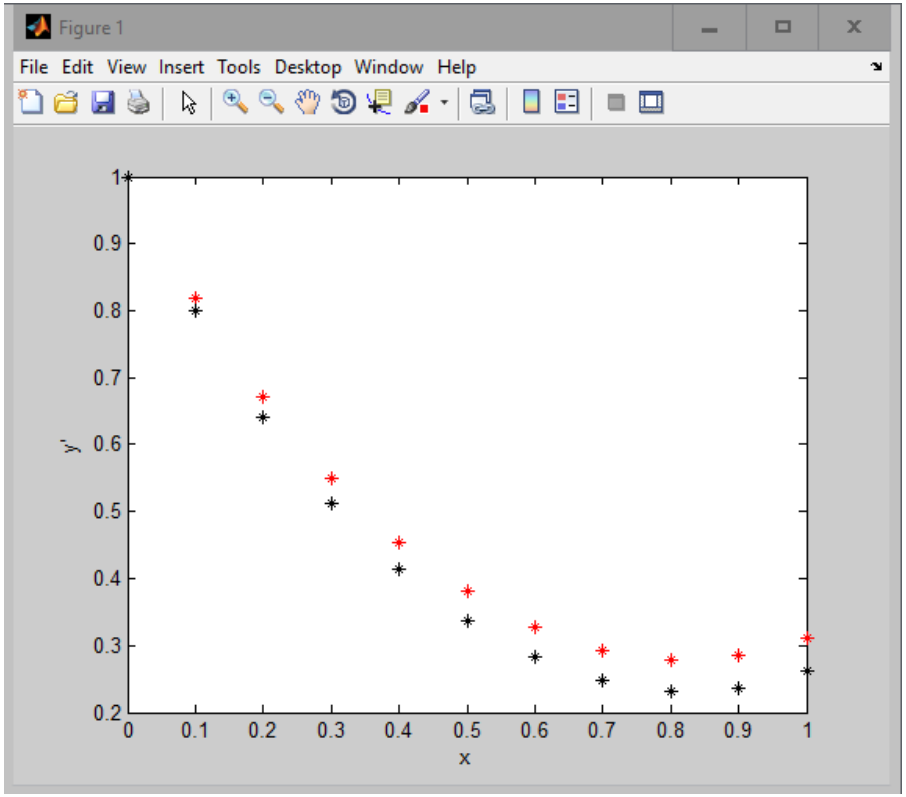
```
FDY =

(3*x)/4 + (11*exp(-2*x))/8 - (3*x^2)/4 + x^3/2 - 3/8


sko_e =

    5.5051
```

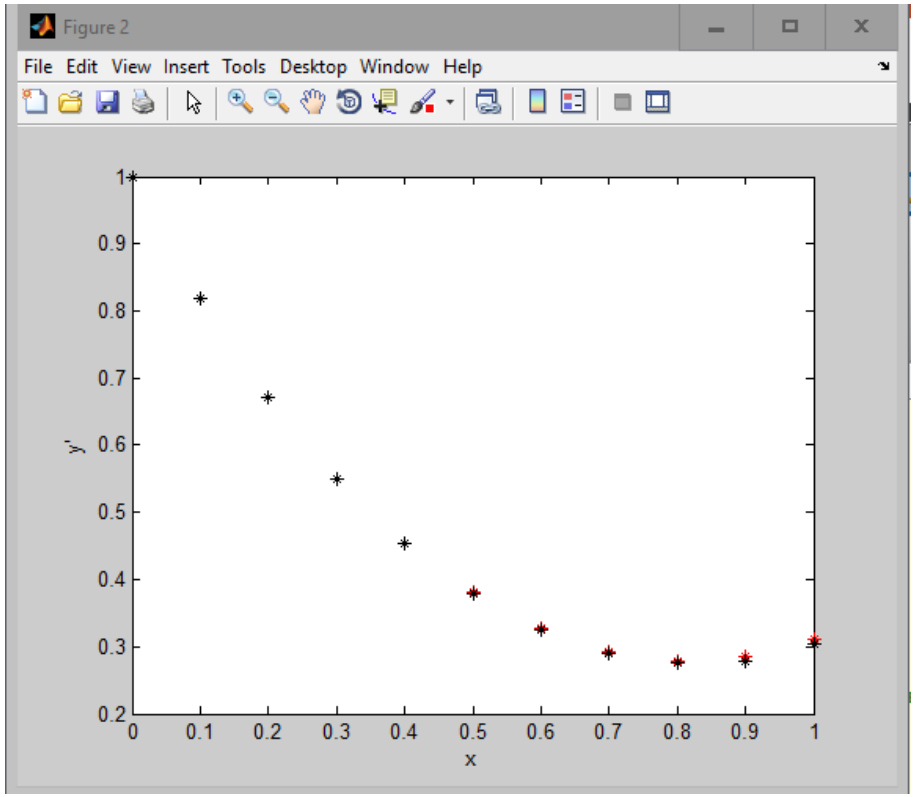


8.5.4 Решение задачи Коши для (8.35) на интервале [0,1] по *x* методом Рунге-Кутта 4-го порядка

```
45      %Решаем методом Рунге-Кутта 4-го порядка
46 -    dyr_x = [];
47 -    dyr_x(1) = y0;
48 -    n = length(x);
49 -    h = x(2) - x(1);
50 -    for i=1:n-1
51 -        k1 = h*f(x(i),dyr_x(i));
52 -        k2 = h*f(x(i)+h/2,dyr_x(i)+k1/2);
53 -        k3 = h*f(x(i)+h/2,dyr_x(i)+k2/2);
54 -        k4 = h*f(x(i)+h/2,dyr_x(i)+k3);
55 -        dyr_x(i+1) = dyr_x(i) + (1/6)*(k1 + 2*k2 + 2*k3 + k4);
56 -    end
57
58 -    figure; plot(x,dya_x,'r*',x,dyr_x,'k*');
59
60      %% Определяем СКО для метода Рунге-Кутта 4-го порядка
61 -    delta = [];
62 -    sko_r = 0;
63 -    for i=1:n
64 -        delta(i) = dya_x(i) - dyr_x(i);
65 -        sko_r = sko_r + (delta(i))^2;
66 -    end
67 -    sko_r = sqrt(sko_r/(n));
68 -    sko_r = 100*sko_r/(max(dya_x) - min(dya_x))
```

8.5.4 Решение задачи Коши для (8.35) на интервале [0,1] по *x* методом Рунге-Кутта 4-го порядка

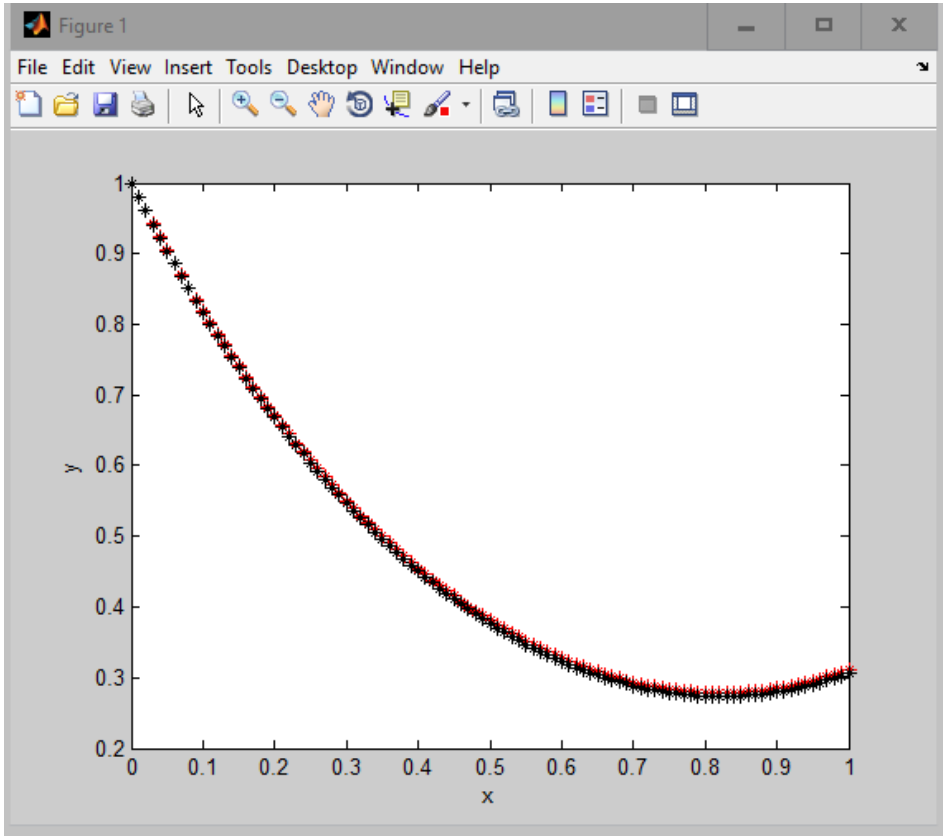


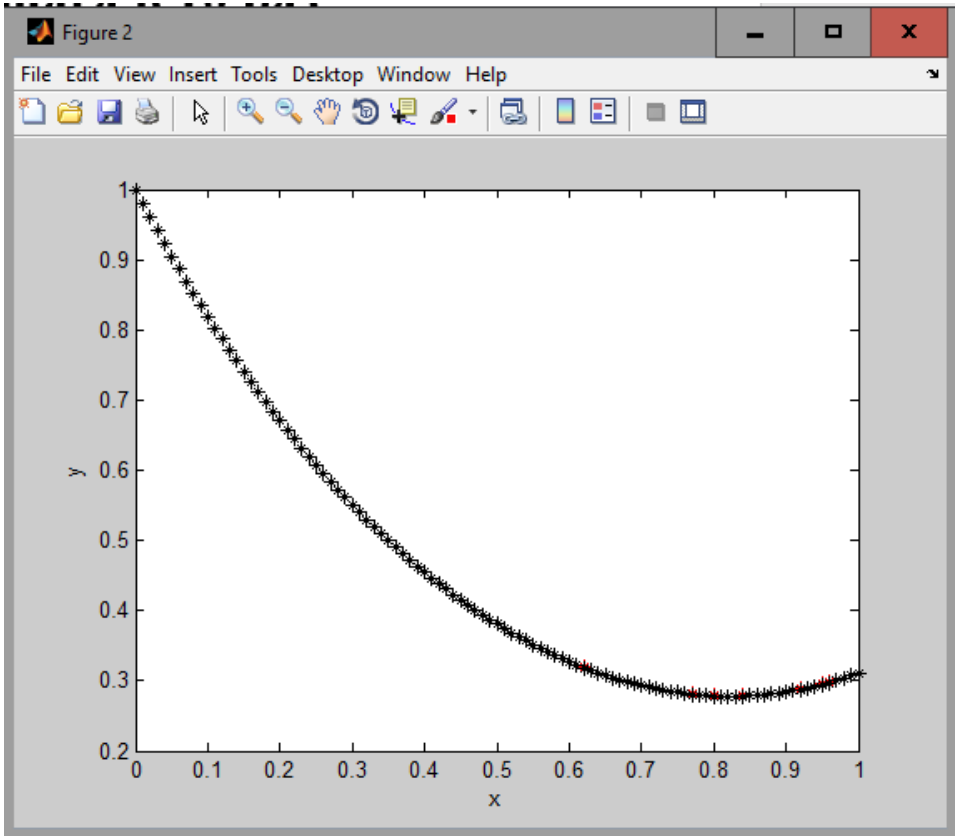


Command Window

```
sko_e =

    5.5051


sko_r =

    0.3878
```

8.5.5 Решение задачи Коши для (8.35) на интервале [0,1] при уменьшении шага в 10 раз





Command Window

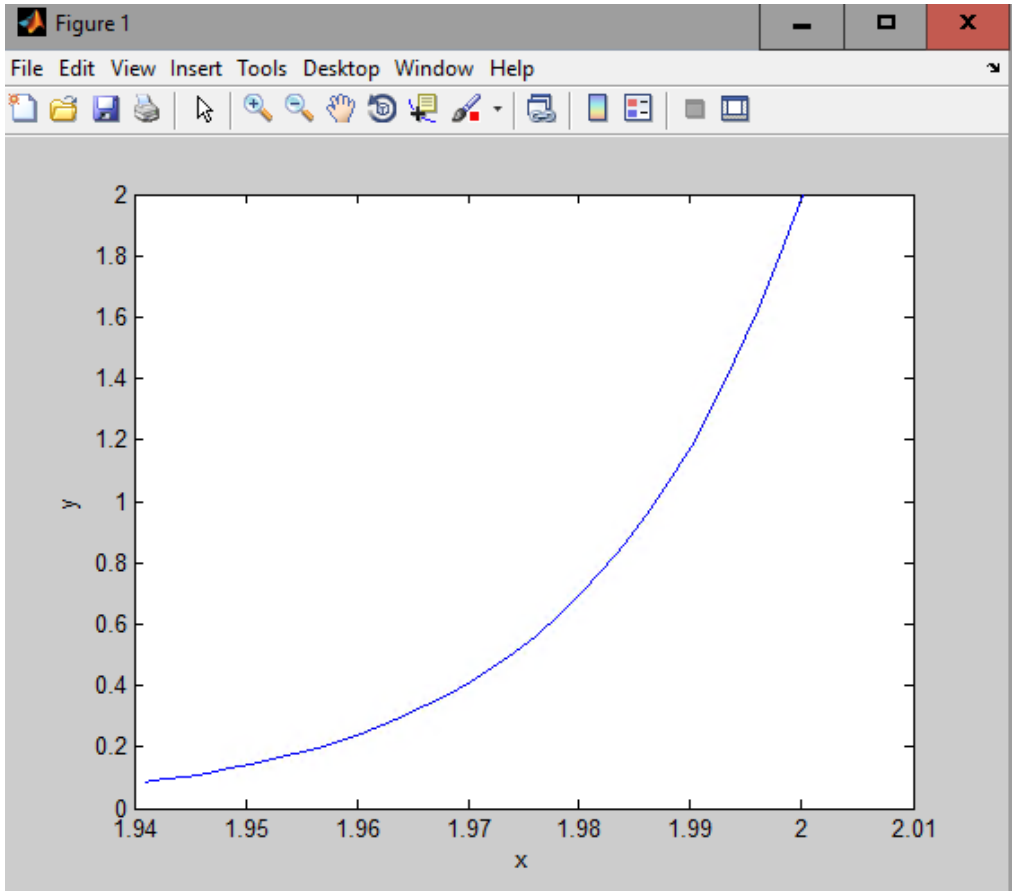
```
sko_e =

    0.5236


sko_r =

    0.0306
```

8.5.6 Решение ОДУ (8.7) методом Эйлера

FILE EDIT NAVIGATE

Lecture_8_2.m

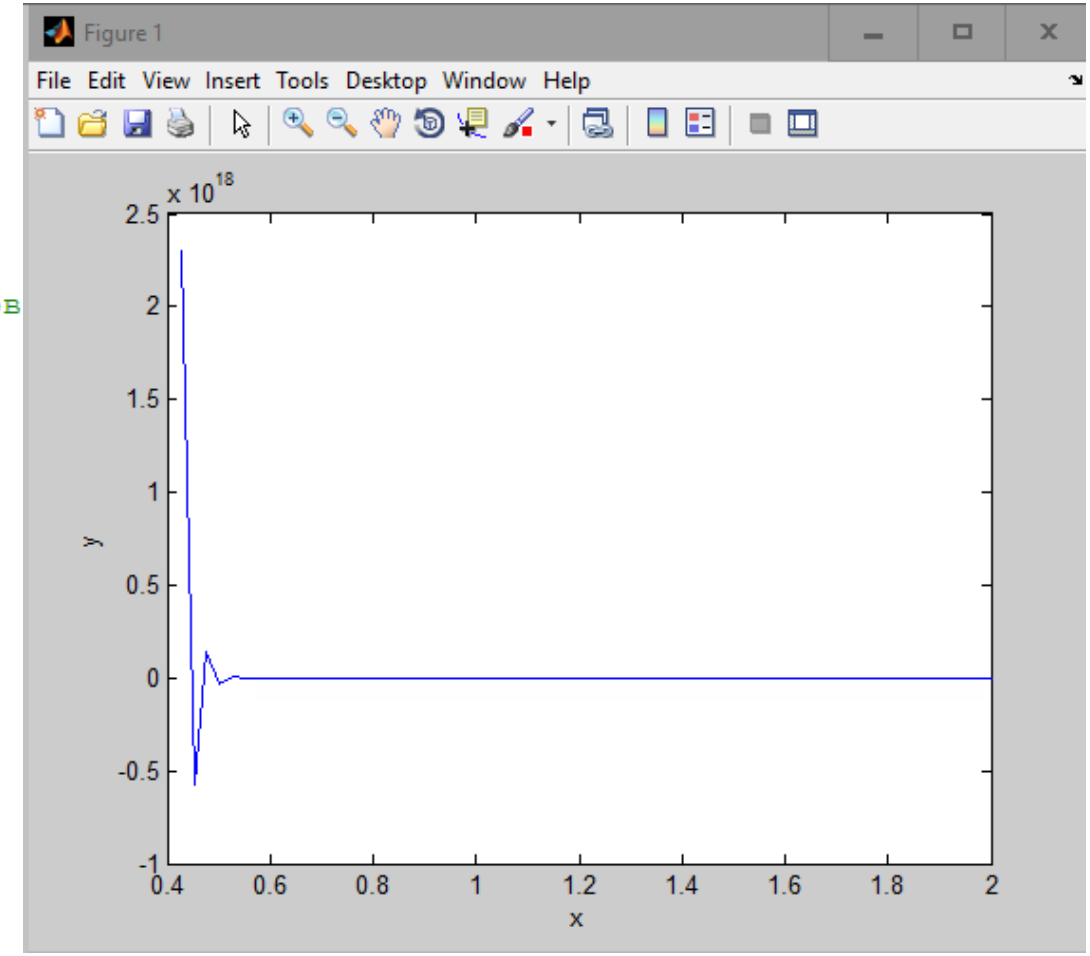
```
1 -    clc;
2 -    x = [];
3 -    y = [];
4 -    x(1) = 2;
5 -    y(1) = 2;
6
7 -    k = 100;
8
9 -    n = 30; % задаем количество шагов
10
11     %Решаем методом Эйлера
12 -   h = 0.001;
13     %h = 0.05;
14 -   for i=1:n
15 -       x(i+1) = (1 - h)*x(i);
16 -       y(i+1) = (1 - h*k)*y(i);
17 -   end
18 -   plot(x,y);
```



8.5.7 Решение ОДУ (8.7) методом Эйлера (увеличиваем шаг с 0,001 до 0,05)

Lecture_8_2.m

```
1   clc;
2   x = [];
3   y = [];
4   x(1) = 2;
5   y(1) = 2;
6
7   k = 100;
8
9   n = 30; % задаем количество шагов
10
11  %Решаем методом Эйлера
12  %h = 0.001;
13  h = 0.05;
14  for i=1:n
15      x(i+1) = (1 - h)*x(i);
16      y(i+1) = (1 - h*k)*y(i);
17  end
18  plot(x,y);
```

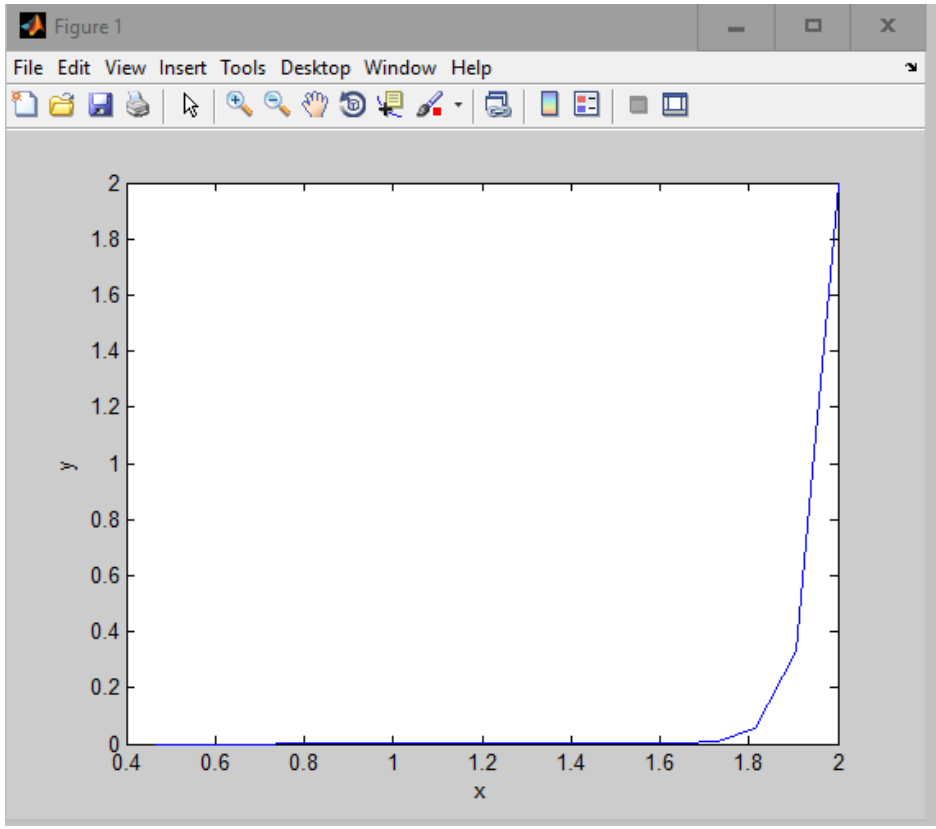


Решение ОДУ «развалилось»!!!

8.5.8 Решение ОДУ (8.7) в неявной форме (шаг решения 0,05)

Lecture_8_3.m

```
1  clc;
2  x = [];
3  y = [];
4  x(1) = 2;
5  y(1) = 2;
6
7  k = 100;
8
9  n = 30; % задаем количество шагов
10
11 %Решаем методом Эйлера
12 %h = 0.001;
13 h = 0.05;
14 for i=1:n
15     x(i+1) = x(i) - (h/(1+h)*x(i));
16     y(i+1) = y(i) - ((h/(1+k*h))*k*y(i));
17 end
18 plot(x,y);
```



8.5.9 Решение ОДУ (8.7) в неявной форме (шаг решения 0,5)

Lecture_8_3.m

```
1 -    clc;
2 -    x = [];
3 -    y = [];
4 -    x(1) = 2;
5 -    y(1) = 2;
6
7 -    k = 100;
8
9 -    n = 30; % задаем количество шагов
10
11     %Решаем методом Эйлера
12     %h = 0.001;
13     %h = 0.05;
14 -   h = 0.5;
15 -   for i=1:n
16 -       x(i+1) = x(i) - (h/(1+h)*x(i));
17 -       y(i+1) = y(i) - ((h/(1+k*h))*k*y(i));
18 -   end
19 -   plot(x,y);
```