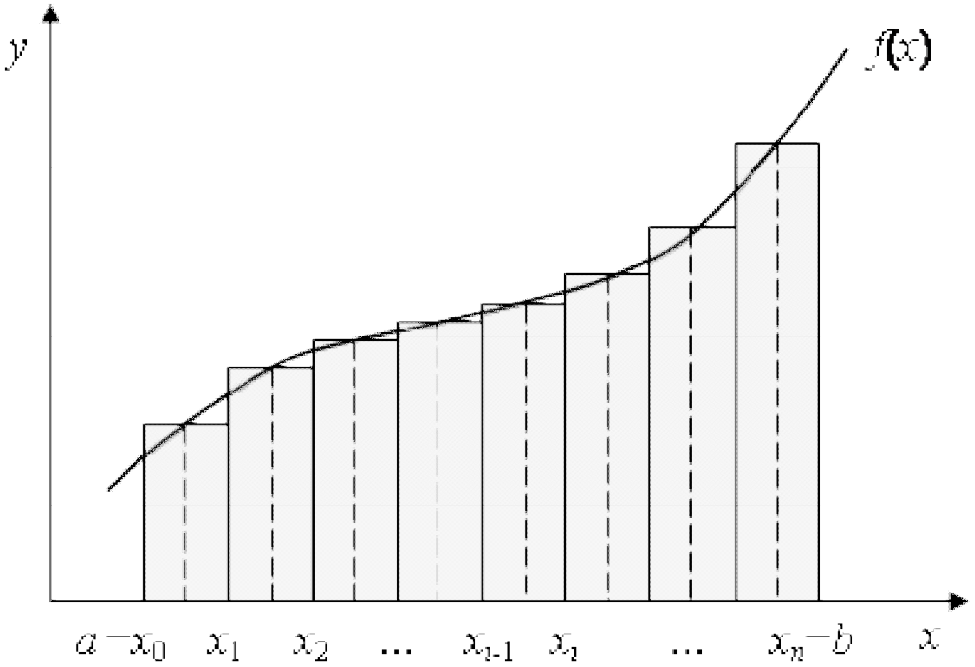
НЕКОТОРЫЕ ЗАДАЧИ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ

Численное интегрирование

Формула центральных прямоугольников

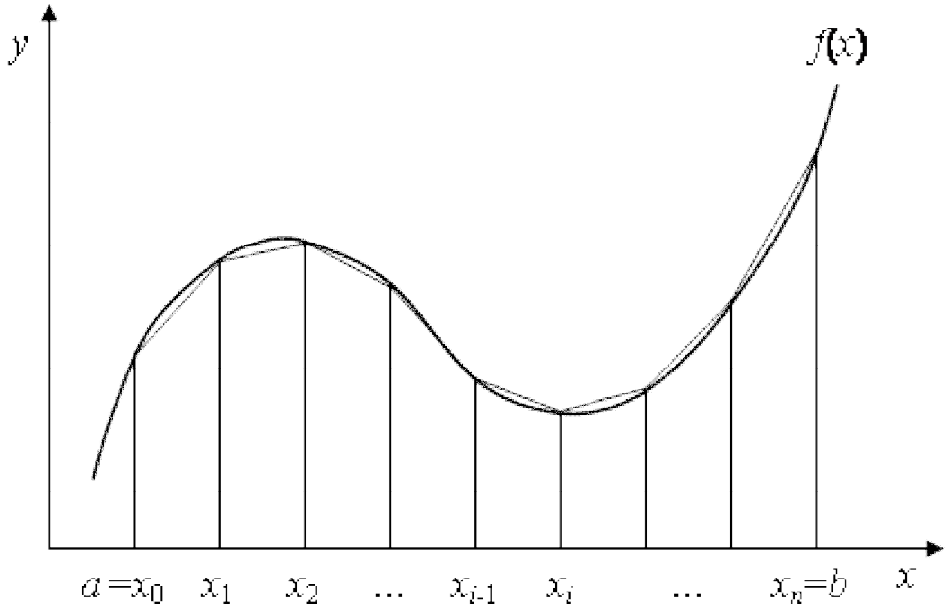


$$h = (b-a)/\ n$$

$$\int_a^b f(x)\,dx \approx h\sum_{i=1}^{n} f\left(x_{i-1} + \frac{h}{2}\right)$$

Численное интегрирование

Формула трапеции



$$h = (b-a)/\ n$$

$$\int_a^b f(x)\,dx \approx h\left(\frac{f(a)+f(b)}{2}+\sum_{i=1}^{n} f(x_i)\right)$$

trapz(x,y)

Численное интегрирование

Формула трапеции

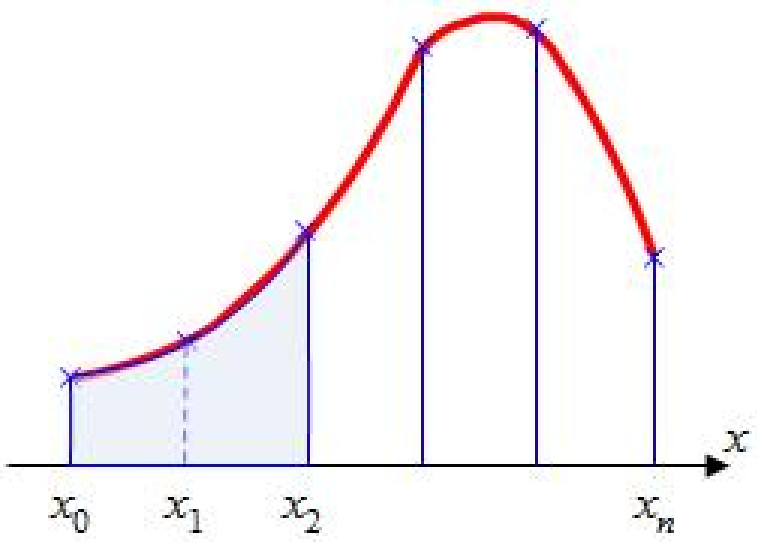
$$\int_0^{\pi} \sin^2(x)\,dx = \left(x/2 - \sin(2x)/2\right)\Big|_0^{\pi} = \pi/2$$

```
% численное интегрирование методом трапеции
x=0:0.01:pi;
y=sin(x).^2;
z=trapz(x,y);
disp (sprintf('Результат = %f (%f)',z,pi/2))
```

Результат = 1.570796 (1.570796)

Численное интегрирование

Метод Симпсона



На каждом отрезке подынтегральную функцию заменим параболой проходящей через точки три точки.

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left(f(a) + f(b) + 4\sum_{i=1}^{n} f(x_{2i-1}) \right) + 2\sum_{i=1}^{n-1} f(x_{2i})$$

Численное интегрирование

Синтаксис:

[I, cnt] = quad('<имя функции>', a, b)
[I, cnt] = quad('<имя функции>', a, b, tol)
[I, cnt] = quad('<имя функции>', a, b, tol, trace)
[I, cnt] = quad8('<имя функции>', a, b)
[I, cnt] = quad8('<имя функции>', a, b, tol)
[I, cnt] = quad8('<имя функции>', a, b, tol, trace)

Переменная cnt содержит информацию о том, сколько раз в процессе интегрирования вычислялась подынтегральная функция.

tol - заданная относительная погрешность.

Когда аргумент trace не равен нулю, строят график, показывающий ход вычисления интеграла.

Численное интегрирование

```
% численное интегрирование методом трапеции
x=0:0.01:pi;
y=sin(x).^2;
[z,cnt]=quad('sin(x).^2', 0, pi, 0.000001)
disp (sprintf('Результат = %f (%f)',z,pi/2))
```

```
z =1.5708
cnt = 41
Результат = 1.570796 (1.570796)
```

Численное решение дифференциальных уравнений

Задача Коши́ состоит в нахождении решения дифференциального уравнения, удовлетворяющего начальным условиям (начальным данным).

ОДУ первого порядка, разрешённое относительно производной

$$\frac{dy}{dx} = f(x, y)$$

$$y_0 = y(x_0)$$

Решением задачи Коши является функция, определённая на интервале *[a,b]*, включающем , являющаяся решением дифференциального уравнения и удовлетворяющая начальному условию.

Численное решение дифференциальных уравнений

Задача Коши́ состоит в нахождении решения дифференциального уравнения, удовлетворяющего начальным условиям (начальным данным).

ОДУ первого порядка, разрешённое относительно производной

$$\frac{dy}{dx} = f(x, y)$$

$$y_0 = y(x_0)$$

Решением задачи Коши является функция, определённая на интервале *[a,b]*, включающем , являющаяся решением дифференциального уравнения и удовлетворяющая начальному условию.

Численное решение дифференциальных уравнений

Система n–го порядка ОДУ первого порядка, разрешённая относительно производных

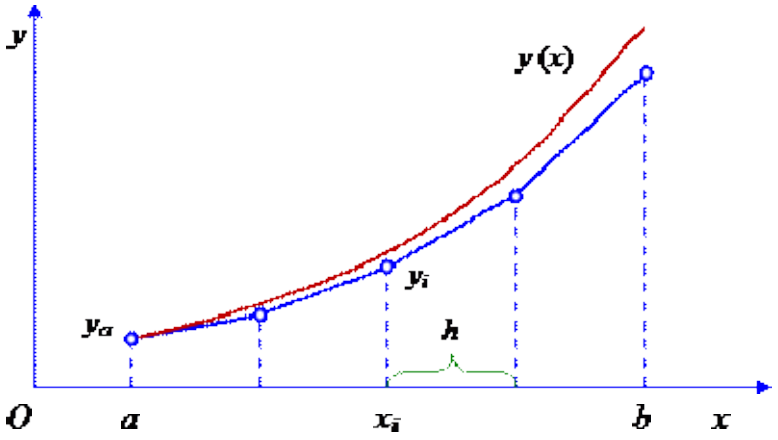
$$\begin{cases} y_1' = f_1\left(x, y_1, \ldots, y_n\right) \\ \ldots \\ y_n' = f_n\left(x, y_1, \ldots, y_n\right) \\ y_{01} = y_1\left(x_0\right) \\ \ldots \\ y_{0n} = y_n\left(x_0\right) \end{cases}$$

Численное решение дифференциальных уравнений

ОДУ n-го порядка, разрешённое относительно старшей производной

$$\begin{cases} y^{(n)} = f\left(x, y, y^{(1)}, \ldots, y^{(n-1)}\right) \\ y_{01} = y\left(x_0\right) \\ \ldots \\ y_{0n} = y^{(n-1)}\left(x_0\right) \end{cases}$$

Метод Эйлера



$$\frac{dy}{dx} \approx \frac{y(x+\Delta x)-y(x)}{\Delta x}$$

$$y(x+\Delta x) = y(x) + \Delta x \cdot f(x,y)$$

$$x_1 = x_0 + \Delta x \qquad y(x_1) = y(x_0) + \Delta x \cdot f(x_0, y_0)$$

$$x_n = x_{n-1} + \Delta x \qquad y(x_n) = y(x_{n-1}) + \Delta x \cdot f(x_{n-1}, y_{n-1})$$

Погрешность численного решения ~Δx.

Метод первого порядка точности.

Уменьшаем шаг до тех пор, пока решение не перестанет зависеть от шага.

Метод Эйлера

Пример 1

$$\frac{dy}{dx} = -2xy \qquad y(0) = 1$$

$$\frac{dy}{y} = -2x\,dx \qquad \ln y = -x^2 + C$$

$$y(x) = e^{-x^2}$$

Метод Эйлера

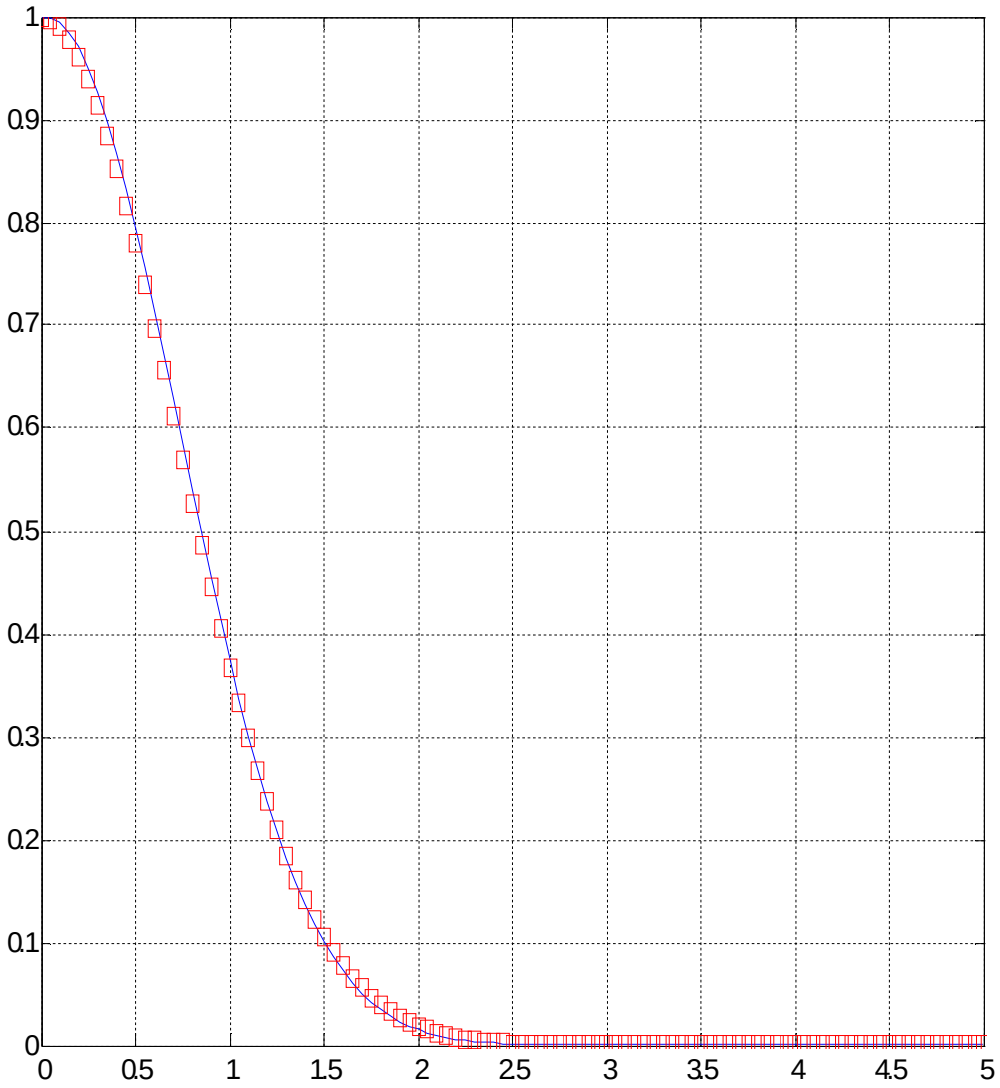
```
% MyEuler.m    m–функция ////////////////////////////////////////
function [X,Y]=MyEuler (y0, x0, x1, N)
dx=(x1-x0)/N;
x(1)=x0;
y(1)=y0;
for i=1:N
    x(i+1)=x(i)+dx;
    y(i+1)=y(i)+dx*fno(x(i),y(i));
end
X=x;
Y=y;

function z=fno (x,y)
z=-2*x*y;
% MyEuler.m    m–функция ////////////////////////////////////////
```

Метод Эйлера

```
% Untitled.m    файл сценария (script-файл)
x0=0;
y0=1;
x1=5;
N=100;
[x,y]=MyEuler (y0,x0,x1,N);
plot (x,y, x, exp(-x.^2),'rs'); grid;
% Untitled.m    файл сценария (script-файл)
```

В m–функции по умолчанию внутренние переменные локальны, а у файла сценария они глобальны.

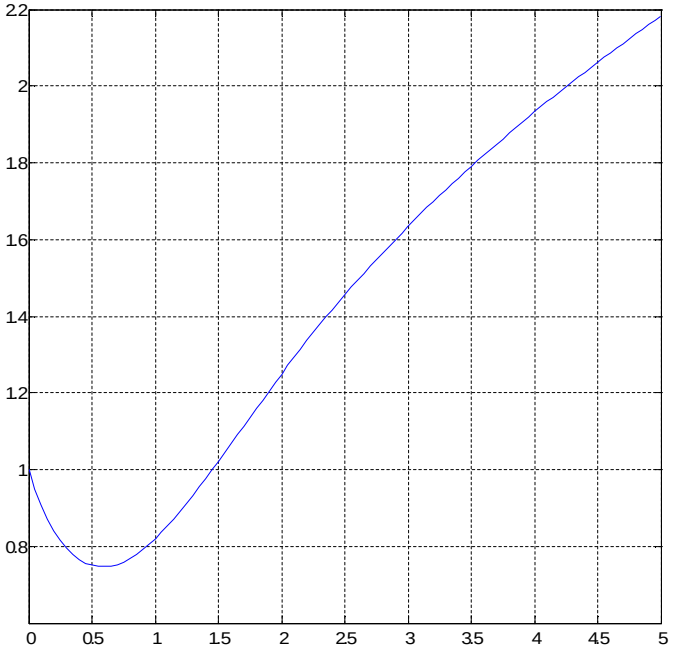
Метод Эйлера



Метод Эйлера

Пример 2 $\frac{dy}{dx} = x - y^2$ $y(0) = 1$

```
function z=fno (x,y)
z=x-y^2;
End

x0=0;
y0=1;
x1=5;
N=100;
[x,y]=MyEuler (y0,x0,x1,N
plot (x,y ); grid;
```



Методы Рунге-Кутты

Классический метод Рунге – Кутты имеет четвёртый порядок точности.

$$y_{n+1} = y_n + \frac{\Delta x}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

$$k_1 = f\left(x_n, y_n\right)$$

$$k_2 = f\left(x_n + \Delta x/2, y_n + k_1 \Delta x/2\right)$$

$$k_3 = f\left(x_n + \Delta x/2, y_n + k_2 \Delta x/2\right)$$

$$k_4 = f\left(x_n + \Delta x, y_n + k_3 \Delta x\right)$$

Методы Рунге-Кутты

Для решения систем ОДУ в *MatLab* ***реализованы различные методы названные решателями-солверами*** *(solver).*

Все решатели *(ode45, ode23, ode133, ode15s, ode23s, ode23t, ode23tb)* ***могут решать системы уравнений явного вида*** *y' = F(t, y).*
Решатели *ode15s, ode23s, ode23t, ode23tb* ***могут решать уравнения неявного вида*** *F(t, y, y') = 0.*

•ode45 – одношаговые явные методы Рунге-Кутта 4-го и 5-го порядка {начальная проба решения}. Во многих случаях он дает хорошие результаты;
•ode23 – одношаговые явные методы Рунге-Кутта 2-го и 3-го порядка. При умеренной жесткости системы ОДУ и низких требованиях к точности этот метод может дать выигрыш в скорости решения;
•ode133 – многошаговый метод Адамса-Башворта-Мултона переменного порядка. Это адаптивный метод, который может обеспечить высокую точность решения;
•ode15s – многошаговый метод переменного порядка (от 1-го до 5-го, по умолчанию 5), использующий формулы численного дифференцирования. Это адаптивный метод, его стоит применять, если решатель ode45 не обеспечивает решения;
•ode23s – одношаговый метод, использующий модифицированную формулу Розенброка 2-го порядка. Может обеспечить высокую скорость вычислений при низкой точности;
•ode23t – метод трапеций с интерполяцией. Этот метод дает хорошие результаты при решении задач, описывающих осцилляторы с почти гармоническим выходным сигналом;
•ode23tb – неявный метод Рунге-Кутта в начале решения и метод, использующий формулы обратного дифференцирования 2-го порядка в последующем. При низкой точности этот метод может оказаться более эффективным, чем ode15s.

Syntax

```
[t,y] = ode45(odefun,tspan,y0)
[t,y] = ode45(odefun,tspan,y0,options)
[t,y,te,ye,ie] = ode45(odefun,tspan,y0,options)
sol = ode45(___)
```

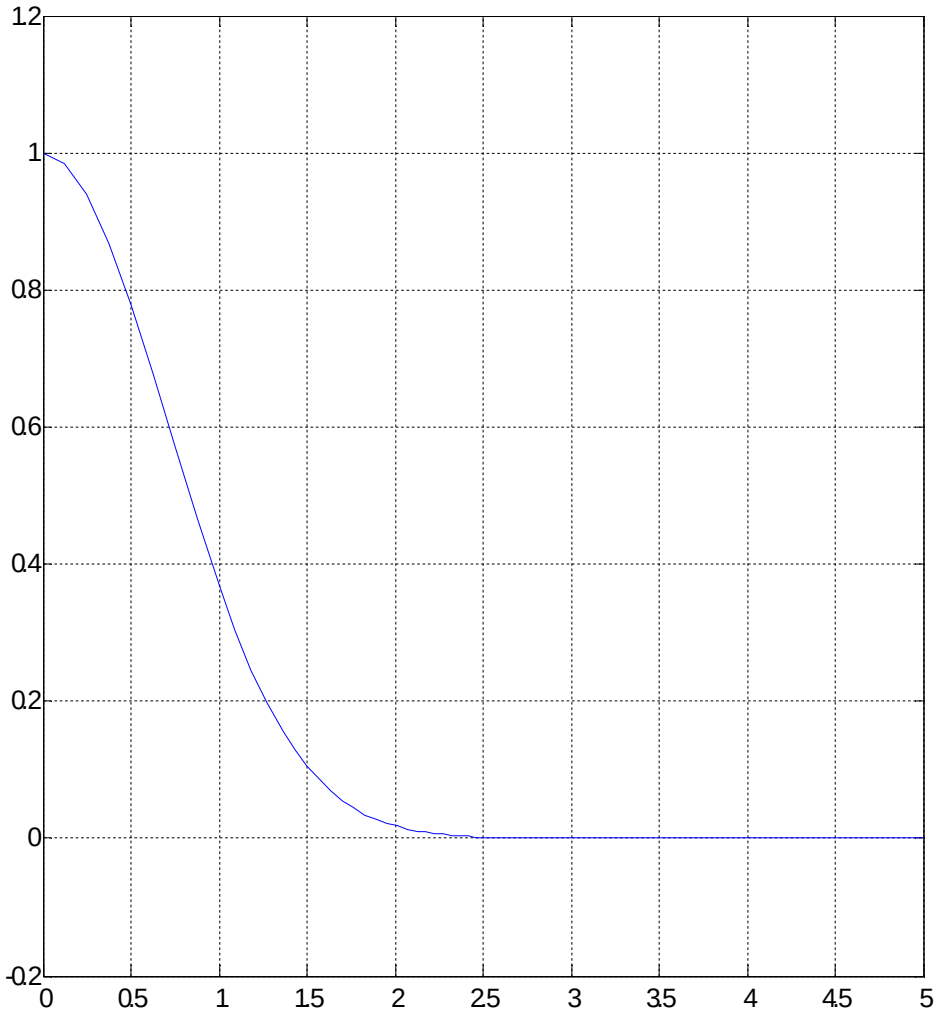
https://www.mathworks.com/help/matlab/ref/ode45.html

Simple ODEs that have a single solution component can be specified as an anonymous function in the call to the solver. The anonymous function must accept two inputs (t,y) even if one of the inputs is not used.

```
tspan = [0 5];
y0 = 1;
[t,y] = ode45(@(x,y) -2*x*y, tspan, y0);
```

```
func1=@(x,y) -2*x*y; % правая часть ОДУ
X0=0;
XN=5;
Y0=1;
[X,Y]=ode45(func1,[X0,XN],Y0);
```

```
function f=prim1(x,y)
f= -2*x*y;
end

[x,y]=ode113(@prim1,[0 5],1);
```


1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

Методы Рунге-Кутты

Пример 3

$$y'' + \frac{1}{2}y' + 2y = 0 \qquad y(0) = 0 \qquad y'(0) = 1$$

$$z_1 = y \qquad z_2 = y'$$

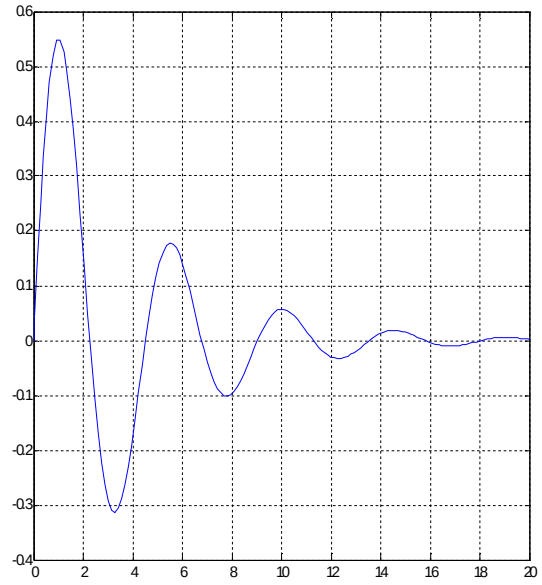
$$\begin{cases} z_1' = z_2 & z_1(0) = 0 \\ z_2' = -\frac{1}{2}z_2 - 2z_1 & z_2(0) = 1 \end{cases}$$

$$Z' = f(Z)$$

Методы Рунге-Кутты

Пример 3

```
func1=@(x,y) [y(2);-0.5*y(2)-2*y(1)];
X0=0; XN=20; Y0=0; YP0=1;
[X,Y]=ode45(func1,[X0,XN],[Y0, YP0]);
plot(X,Y(:,1)); grid;
```



Модель «хищник – жертва»

Модель «хищник – жертва». Имеются два биологических вида. Особи первого вида являются пищей для особей второго вида (хищников). Численности популяций в начальный момент времени известны. Требуется определить численность видов в произвольный момент времени. Математической моделью задачи является система дифференциальных уравнений Лотки – Вольтерра.

Ареол обитания не совпадает

x – популяция травоядных

a – коэффициент рождаемости травоядных

$$\frac{dx}{dt} = ax$$

y – популяция хищников

c – коэффициент убыли хищников

$$\frac{dy}{dt} = -cy$$

Модель «хищник – жертва»

Единый ареол обитания.

Частота встреч хищников и травоядных пропорциональна xy.
Хищник, встретив травоядное, с вероятностью b съедает его.

$$\frac{dx}{dt} = ax - bxy$$

Сытый хищник производит потомство с вероятностью d.

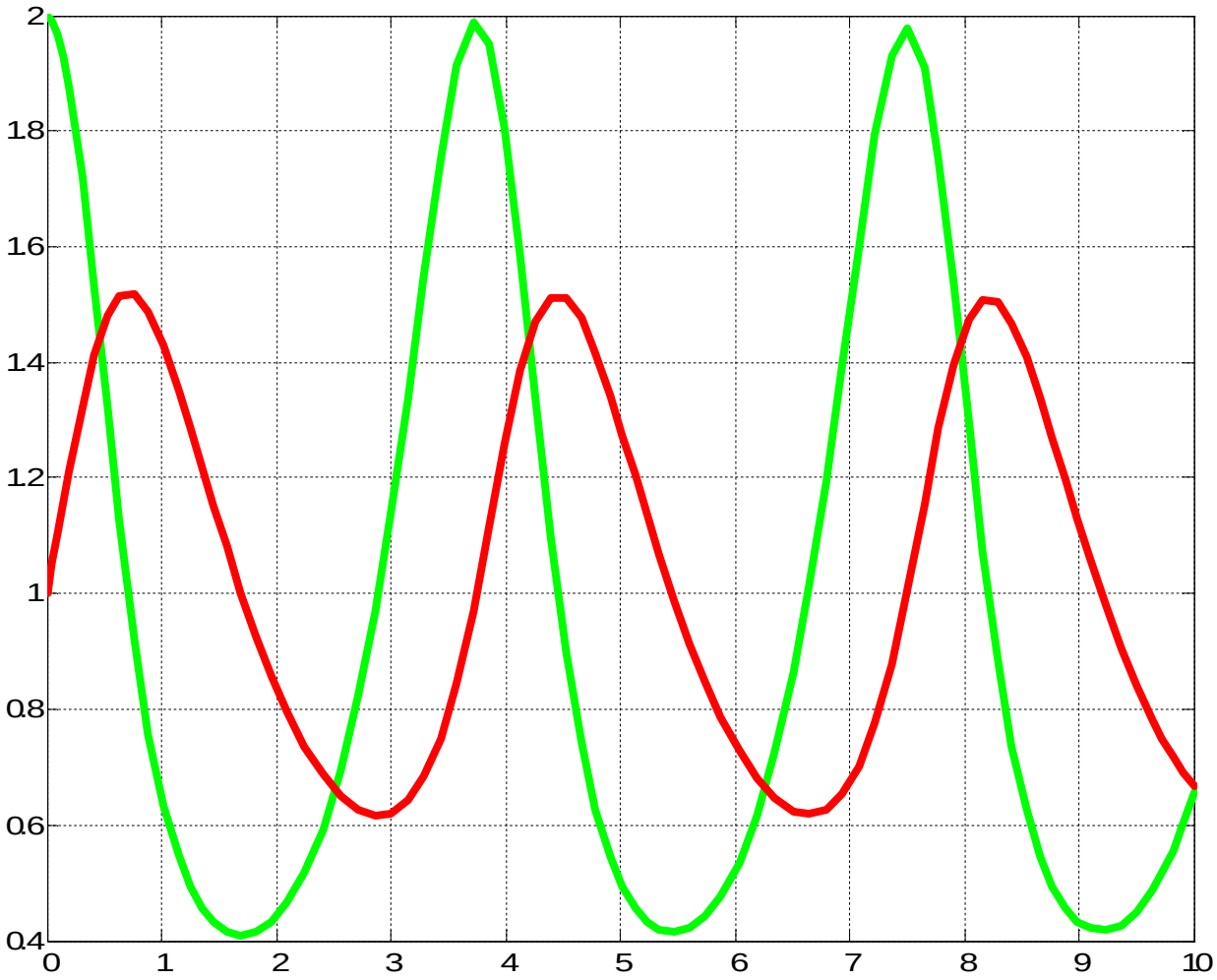
$$\frac{dy}{dt} = -cy + dxy$$

Модель «хищник – жертва»

$$\begin{cases} \frac{dx}{dt} = (a - by)x \\ \frac{dy}{dt} = (dx - c)y \end{cases}$$

$a=b=3$, $c=d=1$, $x(0)=2$, $y(0)=1$.

```
% решение системы уравнений Лотки - Вольтерра
fun=@(x,y) [3*y(1).*(1-y(2)); y(2).*(y(1)-1)];

Y0=[2; 1]; % вектор начальных условий
[T,Y]=ode45(fun,[0 10],Y0); % решаем систему
plot(T,Y(:,1),'g',T,Y(:,2),'r','LineWidth',3 );
grid;
```


2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0
1
2
3
4
5
6
7
8
9
10