**Задание**:

Написать программу-функцию polimorf(). У функции предусмотреть варианты с разными входными и выходными параметрами.

Рассмотреть следующие варианты:

1. **Ведение базы данных**. Входные параметры: массив структур. Организовать форматный вывод данных в файл. Выходных данных: нет.
2. **Решение системы линейных уравнений**. Входные параметры: матрица и вектор столбец. Выходной параметр: решение СЛАУ. Сделать форматный вывод на экран.
3. **Ведение базы данных – статистический анализ**. Найти студента с минимальным средним баллом, максимальным средним баллом, средний балл по всем студентам. Входные параметры: массив структур. Выходные параметры: ФИО студента с минимальным баллом и минимальный балл, ФИО студента с максимальным баллом и максимальный балл, средний балл по группе студентов.

function polimorf

% Данные для проверки работы функции polimorf1()

% Вариант: два входных параметра и один выходной

% Матрица системы A и вектор свободных членов b

A=[2 3 1;3 1 -1;2 -1 3];

b=[11;2;9];

x=polimorf1(A,b);

fprintf('Решение СЛАУ:\n');

fprintf('%7.2f ',x);

%Проверка

c=A\*x;

fprintf('\n\nПроверка:\n');

fprintf('%7.2f ',c);

%Вариант: один входной аргумент, выходных нет

z=struct('Department','ПМ-ПУ','Group','103','Name','Панькина Наталья Андреевна','Marks',...

struct('Subject',{'Математический анализ','Алгебра','Геометрия'},'Rating',{4 5 5}));

z(2)=struct('Department','ПМ-ПУ','Group','103','Name','Нечипорук Антон Алексеевич','Marks',...

struct('Subject',{'Математический анализ','Алгебра','Геометрия'},'Rating',{4 5 4}));

z(3)=struct('Department','ПМ-ПУ','Group','103','Name','Иванов Иван Иванович','Marks',...

struct('Subject',{'Математический анализ','Алгебра','Геометрия'},'Rating',{3 4 3}));

polimorf1(z);

%Вариант: один входной аргумент и один выходной

% cel\_m{1}.FIO='ФИО с мин. баллом';

% cel\_m{1}.ball=3;

% cel\_m{2}.FIO='ФИО с мах. баллом';

% cel\_m{2}.ball=5;

% cel\_m{3}=3;

cel\_m=polimorf1(z);

fprintf('Имя студента с min баллом:%s.Балл: %d\n',cel\_m{1}.FIO,cel\_m{1}.ball);

fprintf('Имя студента с max баллом:%s.Балл: %d\n',cel\_m{2}.FIO,cel\_m{2}.ball);

fprintf('Средний балл по группе:%f\n',cel\_m{3});

end

function varargout=polimorf1(varargin);

if nargin==1 & nargout==0

z=varargin{1};

fprintf('\n\n Вывод базы данных:\n');

for j=1:length(z)

fprintf('\n%5s %5s %20s \n',z(j).Department,z(j).Group,z(j).Name);

for i=1:3

fprintf('%25s %5d\n',z(j).Marks(i).Subject,z(j).Marks(i).Rating);

end

end

end

if nargin==1 & nargout==1

z=varargin{1};

n1=[];

for i=1:length(z)

n1=[n1 sum([z(i).Marks.Rating])];

end

i=find(n1==min(n1));

j=min(n1);

i1=find(n1==max(n1));

j1=max(n1);

i2=mean(n1);

cel\_m1{1}.FIO=z(i).Name;

cel\_m1{1}.ball=j;

cel\_m1{2}.FIO=z(i1).Name;

cel\_m1{2}.ball=j1;

cel\_m1{3}=i2;

varargout{1}=cel\_m1;

end

if nargin==2 & nargout==1

varargout{1}=varargin{1}^(-1)\*varargin{2};

end

end

>> polimorf

Решение СЛАУ:

1.00 2.00 3.00

Проверка:

11.00 2.00 9.00

Вывод базы данных:

ПМ-ПУ 103 Панькина Наталья Андреевна

Математический анализ 4

Алгебра 5

Геометрия 5

ПМ-ПУ 103 Нечипорук Антон Алексеевич

Математический анализ 4

Алгебра 5

Геометрия 4

ПМ-ПУ 103 Иванов Иван Иванович

Математический анализ 3

Алгебра 4

Геометрия 3

Имя студента с min баллом:Иванов Иван Иванович.Балл: 10

Имя студента с max баллом:Панькина Наталья Андреевна.Балл: 14

Средний балл по группе:12.333333

# §9. Стандартная и специализированная графика MATLAB

**Функции одной переменной (построение графика на плоскости):**

Для отображения функции одной переменной на плоскости используются графики в декартовой (прямоугольной) системе координат. При выводе на экран задаются координаты узловых точек функции. Эти точки соединяются друг с другом отрезками прямых линий. Совокупность узловых точек функции задается двумя векторами: вектором абсцисс и вектором соответствующих ординат.

Вывод графика на экран в декартовой системе координат осуществляется командой **plot.**

Функция **plot** перед выводом графика очищает текущее графическое окно (если оно есть). После выполнения в командной строке (или в программе пользователя) команды **hold on** графики, формируемые функциями **plot,** при новом обращении к **plot** сохраняются в графическом окне. Отмена режима "наложения" графиков осуществляется командой **hold off**. Масштабно-координатная сетка наносится командой **grid on**, а убирается командой **grid off**. Подписи к осям делают функции **xlabel и ylabel**, а заголовок – **title**.

Существует несколько вариантов использования этой команды:

1. при вызове функции с двумя аргументами **plot(x,y),** где оба вектора одинаковой размерности, будет построен график зависимости **y(x).**

**Пример:**

**x=0:pi/100:2\*pi;**

**y=sin(x);**

**plot(x,y)**

1. можно строить графики нескольких функций одним вызовом функции **plot**.

**Пример:**

**x=0:pi/100:2\*pi;**

**y1=sin(x); y2=cos(x);**

**plot(x,y1,x,y2)**

1. для наложения можно использовать функцию ***hold*** с аргументами ***on*** (разрешить наложение) или off (запретить наложение).

**Пример:** % действие как в предыдущем примере

**x=0:pi/100:2\*pi;**

**y1=sin(x); y2=cos(x);**

**plot(x,y1)**

**hold on**

**plot(x,y2)**

**Создание двух графиков в двух окнах:**

**x1 = 0:0.01:2\*pi;   
y1 = sin(x1);   
    
x2 = 0:0.01:pi;   
y2 = cos(x2);   
    
figure(1);** % создание окна с номером 1  **plot(x1, y1);** % рисование первого графика **figure(2);** % создание графического окна с номером 2 **plot(x2, y2);**% рисование 2-го графика во 2-м окне

В некоторых случаях большего удобства представления информации можно достичь, отображая два графика **в одном графическом окне**. Это достигается путем использования функции **subplot()**, имеющей следующий синтаксис:

**subplot(<число строк>, <число столбцов>, <номер графика>)**

Рассмотрим пример отображения двух графиков друг под другом вышеприведенных функций синуса и косинуса.

**x1 = 0:0.01:2\*pi;   
y1 = sin(x1);   
    
x2 = 0:0.01:pi;   
y2 = cos(x2);   
    
figure(1);   
subplot(2,1,1);** % делим окно на 2 строки и один столбец **plot(x1,y1);** % отображение первого графика **subplot(2,1,2);**  % строим 2-ю координатную ось  **plot(x2,y2);** % отображаем 2-й график в новых осях

Аналогичным образом можно выводить два и более графиков в столбец, в виде таблицы и т.п. Кроме того, можно указывать точные координаты расположения графика в графическом окне. Для этого используется параметр **position** в функции **subplot()**:

**subplot(**'**position**'**, [left bottom width height]);**

**left** – смещение от левой стороны окна;

**bottom** – смещение от нижней стороны окна;

**width**, **height** – ширина и высота графика в окне.

Все эти переменные изменяются в пределах от 0 до 1.

**Пример: %** фрагмент программы отображения графика функции синуса в центре графического окна

**x1 = 0:0.01:2\*pi;   
y1 = sin(x1);   
    
subplot(**'**position**'**, [0.33 0.33 0.33 0.33]);   
plot(x1,y1);**

В данном примере функция **subplot()** смещает график на треть от левой и нижней границ окна и рисует график с шириной и высотой в треть графического окна. В результате получается эффект рисования функции синуса по центру основного окна.

**Оформление графиков**

Пакет **MATLAB** позволяет отображать графики с разным цветом и типом линий, показывать или скрывать сетку на графике, выполнять подпись осей и графика в целом, создавать легенду и многое другое.

Рассмотрим наиболее важные функции, позволяющие делать такие оформления на примере двумерных графиков.

Функция **plot()** позволяет менять цвет и тип отображаемой линии. Для этого используются дополнительные параметры, которые записываются следующим образом:

**plot(x, y, <**'**цвет линии, тип линии, маркер точек**'**>);**

Цвет линии графика:

|  |  |
| --- | --- |
| **Маркер** | **Цвет линии** |
| **c** | **голубой** |
| **m** | **фиолетовый** |
| **y** | **желтый** |
| **r** | **красный** |
| **g** | **зеленый** |
| **b** | **синий** |
| **w** | **белый** |
| **k** | **черный** |

Тип линии графика:

|  |  |
| --- | --- |
| **Маркер** | **Цвет линии** |
| **-** | **непрерывная** |
| **--** | **штриховая** |
| **:** | **пунктирная** |
| **-.** | **штрих-пунктирная** |

Тип точек графика:

|  |  |
| --- | --- |
| **Маркер** | **Цвет линии** |
| **.** | **точка** |
| **+** | **плюс** |
| **\*** | **звездочка** |
| **o** | **кружок** |
| **x** | **крестик** |

**Замечание: plot(y)**

Команда **plot(y)** строит график элементов одномерного массива y в зависимости от номера элемента. Если элементы массива y комплексные, то строится график **plot(real(y), imag(y))**. Если Y - двумерный действительный массив, то строятся графики для столбцов; в случае комплексных элементов их мнимые части игнорируются.

Примеры записи функции **plot()** с разным набором маркеров:

x = 0:0.1:2\*pi;   
y = sin(x);   
    
subplot(2,2,1); plot(x,y,'r-');   
subplot(2,2,2); plot(x,y,'r-',x,y,'ko');   
subplot(2,2,3); plot(y,'b--');   
subplot(2,2,4); plot(y,'b--+');

Иногда требуется выделить отдельный фрагмент графика и только его показать целиком. Для этого используется функция **axis()**, которая имеет следующий синтаксис:

**axis( [ xmin, xmax, ymin, ymax ] )**

Воспользуемся данной функцией для отображения графика функции синуса в пределах от 0 до 2pi:

**x = 0:0.1:2\*pi;   
y = sin(x);   
    
subplot(1,2,1);   
plot(x,y);   
axis([0 2\*pi -1 1]);   
    
subplot(1,2,2);   
plot(x,y);   
axis([0 pi 0 1]);**

Рассмотрим возможности создания подписей графиков, осей и отображения сетки на графике.

  Функции оформления графиков:

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **grid [on, off]** | **Включает/выключает сетку на графике** |
| **title('заголовок графика')** | **Создает надпись заголовка графика** |
| **xlabel('подпись оси Ox')** | **Создает подпись оси Ox** |
| **ylabel('подпись оси Oy')** | **Создает подпись оси Oy** |
| **text(x,y, 'текст')** | **Создает текстовую надпись в координатах (x,y).** |

Рассмотрим работу данных функций в следующем примере:

**x = 0:0.1:2\*pi;   
y = sin(x);   
    
plot(x,y);   
axis([0 2\*pi -1 1]);   
grid on;   
title('The graphic of sin(x) function');   
xlabel('The coordinate of Ox');   
ylabel('The coordinate of Oy');   
text(3.05,0.16,'\leftarrow sin(x)');**

Таким образом, используя описанный набор функций и параметров, можно достичь желаемого способа оформления графиков в системе **MATLAB.**.

**Построение графиков в полярных координатах**

Например, построим график функции **r = sin(3ᵩ)** в полярных координатах:

**phi = 0 : 0.01 : 2\*pi;**

**r = sin( 3\* phi );**

**polar( phi , r )**

**Построение графика в пространстве. Построение трехмерной линии**

Используют функцию **plot3**.

Если **x,y,z**, векторы одинаковой размерности, то команда **plot3(x,y,z)** позволяет строить трехмерную линию, проходящую через точки, координаты которых являются компонентами векторов **x,y,z**.

**Пример:**

**t=0:pi/50:10\*pi;**

**plot3(sin (t),cos (t), t)**

**axis square**

**grid on**

**MATLAB** обладает рядом инструментов для визуализации графиков в трехмерном пространстве. Такие задачи обычно возникают при отображении графиков функций типа ****.

Функция **meshgrid** позволяет преобразовывать область, заданную входными аргументами в массивы, которые могут использоваться для вычисления функций двух переменных и построения графиков.

**Пример:**

**x=-1:0.1:1;               % координаты точек по оси Ox   
y=-2:0.1:2;             % координаты точек по оси Oy**

**[X,Y]=meshgrid(x,y); % матрицы координат точек по осям Ox и Oy**

**Z=exp(-X.^2-Y.^2);**

**plot3(X,Y,Z);**

**grid on**

Можно добиться лучшей визуализации, используя функцию

**mesh(X,Y,Z);   % отображение графика в виде сетки**

**grid on;**

**colorbar**

Благодаря использованию функции **mesh()** получается график, образованный интерполяцией точек массивов X, Y и Z линиями по осям Ox и Oy. Кроме того, цветом указывается уровень точки по оси Oz: от самого малого значения (синего) до самого большого (красного) и производится удаление «невидимых» линий. Это позволяет лучше визуально оценивать структуру трехмерного графика по сравнению с функцией plot3().

Для построения поверхностей используется функция **surf.**

Аргументы функции такие же, как и у функции mesh.

**surf(X,Y,Z);        % отображение непрерывной поверхности**

**colorbar**

К новым методам дополнительного оформления трёхмерных графиков можно отнести возможность вызывать функцию **mesh** с суффиксами ***z*** и ***c*** (**meshz** и **meshc**), а функцию **surf** с суффиксом ***c*** (**surfc**). Использование суффикса ***z*** приводит к построению "графика с пьедесталом". Например,

**[X,Y] = meshgrid( -2 : 0.1 : 2 );**

**Z = X .\* exp( - X.^2 - Y.^2 );**

**meshz( X, Y, Z )**

Функции с суффиксом ***c*** помимо собственно трёхмерного графика строят ещё и так называемые линии уровня.

**Пример:**

**[X,Y,Z] = peaks(30);**

**surfc(X,Y,Z);**

**colormap( hsv );**

**axis([-3 3 -3 3 -10 5]);**

 Функция с именем **peaks** (является некоторой масштабированной комбинацией стандартных гауссовых функций) часто применяется в справочной системе **MATLAB** для наглядной иллюстрации графических функций.

Функция **surf()** может использоваться в режиме:

**shading interp;         % интерполяция тени на гранях графика**

которая интерполирует цвет на гранях для получения более гладкого изображения поверхности. Также существует возможность менять цветовую карту отображения графика с помощью функции

**colormap( <карта> );    % установка цветовой карты**

Например, карта с именем hot, используемая по умолчанию может быть заменена на любую другую доступную **(hot, hsv, gray, pink, cool, bone copper)** или созданную самостоятельно.

Следует отметить, что все три функции **plot3(), mesh() и surf()** могут быть использованы и с одним аргументом Z, который интерпретируется как матрица со значениями точек по оси Oz.

Для масштабирования отдельных участков трехмерных графиков, также как и в случае с двумерными графиками, используется функция:

**axis([xmin xmax ymin ymax zmin zmax])**

Для оформления трехмерных графиков можно пользоваться описанными ранее функциями: **text, xlabel, ylabel, zlabel, title, grid [on/off], subplot.**

Наконец, для трёхмерных графиков существует возможность изменять точку их обзора, т.е. положение виртуальной камеры с помощью функции

**view([az el]);**

где **az** – угол азимута; **el** – угол возвышения. Изменение первого угла означает вращение плоскости xOy вокруг оси Oz против часовой стрелки. Угол возвышения есть угол между направлением на камеру и плоскостью xOy.

Когда выполняются функции **mesh** или **surf**, то по умолчанию устанавливаются **значения az = -37.5°, el = 30°.**

Эти значения в любой момент времени можно изменить функцией

**view( [ az , el ] )**

**Пример:** Для рассмотренной выше функции

**figure(1)**

**[X,Y] = meshgrid( -2 : 0.1 : 2 );**

**Z = X .\* exp( - X.^2 - Y.^2 );**

**meshz( X, Y, Z )**

**figure(2)**

**[X,Y] = meshgrid( -2 : 0.1 : 2 );**

**Z = X .\* exp( - X.^2 - Y.^2 );**

**meshz( X, Y, Z )**

**view( [ -15 , 20 ] )**

**Специализированная графика Matlab**

Для отображения векторных или матричных данных используются столбцовые диаграммы. Для построения таких графиков используются функции **bar, bar3, barh, bar3h.**

**y=[5 2 1; 8 7 3; 9 8 6; 5 5 5; 4 3 2];**

**subplot(2,2,1), bar(y)**

**subplot(2,2,2), bar3(y)**

**subplot(2,2,3), barh(y)**

**subplot(2,2,4), bar3h(y)**

**Построение гистограммы**

**Пример:**

**x=-5:0.005:5;**

**y=randn(100000000,1);**

**hist(y,x)**

% Вывод графиков с измененной толщиной линии и размером маркера

**x=linspace(0,1,50); %** равномерное разбиение от 0 до 1, количество точек : 50

**x1=[-0.5:0.05:1.5];**

**y=x.\*(1-x)+0.1;**

**plot(x,y,'xr',x1,(x1+0.5)/5,'b','LineWidth',2,'MarkerSize',8)**

**grid on**

% Вывод графического окна с заданной позиции и заданного размера

% с именем, но без вывода номера

**Poz=[100 100 500 400];**

**Fig=figure('Position',**Poz**,'NumberTitle','**off**','Name',**'Графическое окно'**)**

**x=linspace(0,2\*pi,500);**

**y=exp(-0.5\*x).\*sin(4\*pi\*x);**

**z=exp(-005\*x);**

**plot(x,y,x,z,':r',x,-z,':r','LineWidth',2)**

**grid on**

**% Fig = 1**

**%** Пример с объединением графиков

**x = linspace(0,2\*pi);**

**subplot(2,2,1);**

**plot(x,sin(x)); axis([0 2\*pi -1.5 1.5]); title('sin(x)');**

**subplot(2,2,2);**

**plot(x,sin(2\*x)); axis([0 2\*pi -1.5 1.5]); title('sin(2x)');**

**subplot(2,2,3);**

**plot(x,sin(3\*x)); axis([0 2\*pi -1.5 1.5]); title('sin(3x)');**

**subplot(2,2,4);**

**plot(x,sin(4\*x)); axis([0 2\*pi -1.5 1.5]); title('sin(4x)');**

**pause(7);**

**subplot(2,2,[3 4]);**

**plot(x,sin(2\*x)); axis([0 2\*pi -1.5 1.5]); title('sin(2x)');**

**Построение графика вектор-функциии**

График функции одной переменной, описанной в файл-функции Name.m, на отрезке  строит также и функция **fplot**:

**fplot('Name',[x\_min,x\_max],tol,LineSpec)**

где **tol** – точность, **LineSpec** – спецификация кривых (толщина линии, маркер, цвет – табл. 1). *tol* и *LineSpec –* не обязательные аргументы.

При обращении

**[x,y]=fplot('Name',[x\_min,x\_max],tol,LineSpec)**

в *Workspace* формируются вектор-столбец *x* узловых точек и вектор-столбец *y* значений функции в этих точках.

Функция **fplot** может построить несколько графиков, если выходным аргументом функции Name является вектор:

**function F=Name(x)**

**F=[f1(x),…,fn(x)];**

По умолчанию первый график рисуется синим цветом, второй – зеленым, третий – красным.

***Пример****. Построить графики функций  и  на промежутке .*

***Вариант решения***:

Описание файл-функции, вычисляющей  и :

function F=Fun(x)

F=[sin(x),x.\*x-x-1];

Графики будут построены после выполнения команды

fplot('Fun',[0,pi])

или

fplot(@Fun,[0,pi])

Если заранее не задавать файл-функцию, вычисляющую  и , то график может быть построен после выполнения команды

fplot('[sin(x),x^2-x-1]',[0,pi])

**Построение графика функции, заданной в неявном виде.** График функции, заданной уравнением , в прямоугольнике  (по «умолчанию» ) строится с помощью функции **ezplot**:

**ezplot(@(x,y)f(x,y),[x\_min,x\_max,y\_min,y\_max])**

***Пример.*** *Построить конхоиду Никомеда, задаваемую уравнением* 

*с параметрами  и  в прямоугольнике :*

***Вариант решения***:

a=0.4; b=1;

F=@(x,y)(x-a).^2\*(x.^2+y.^2)-b^2\*x.^2;

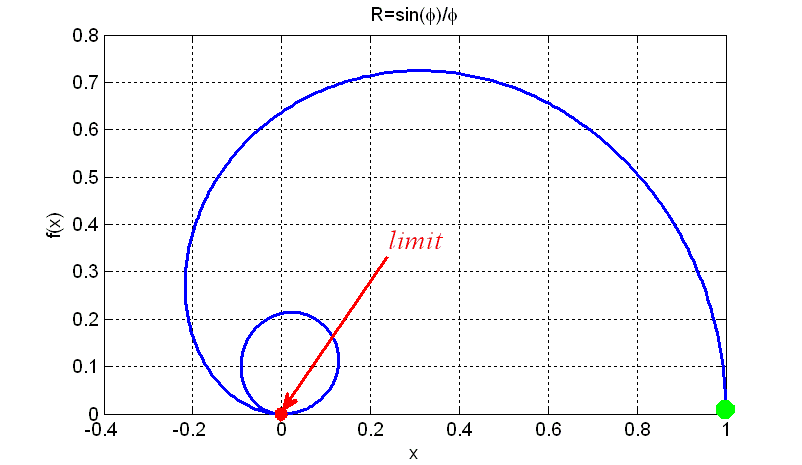
ezplot(F,[-1 2 -4 4]);

grid on

График конхоиды приведен на рис.

|  |
| --- |
| Konhoida |

**Пример:** *Построить график функции *



Вариант решения:

%f=sin(x)/x

clear all

fi=linspace(0.01,2\*pi,500);

R=sin(fi)./fi;

x=R.\*cos(fi);

y=R.\*sin(fi);

plot(x,y,'LineWidth',2)

grid on

hold on

plot(x(1),y(1),'g','Marker','\*','LineWidth',12)

plot(x(end),y(end),'r','Marker','\*','LineWidth',8)

xlabel('x')

ylabel('f(x)')

title('R=sin(\phi)/\phi')

hold off

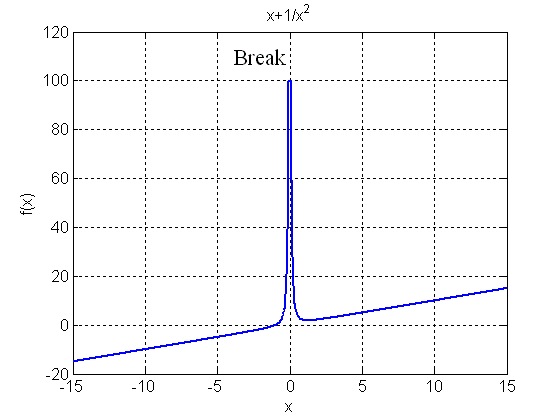
**Пример***:* *построить график встроенной функции* humps(x)*. Указать на графике точку, соответствующую максимальному значению функции. Указать на графикe множество значений функции, лежащих в интервале* [20,40].

**Вариант решения:**

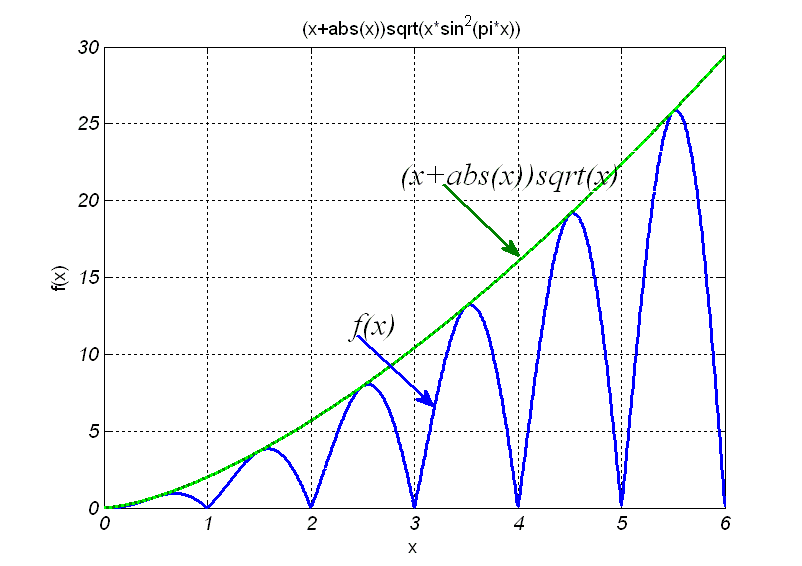
|  |  |
| --- | --- |
| x = 0:.01:2;  y = humps(x);  plot(x,y)  [v,ind] = max(y)  hold on  plot(x(ind),y(ind),'ro')  x(ind)  ind = find(20<=y & y<=40);  plot(x,y,x(ind),y(ind),'o')  grid |  |

**Построить графики функций:**

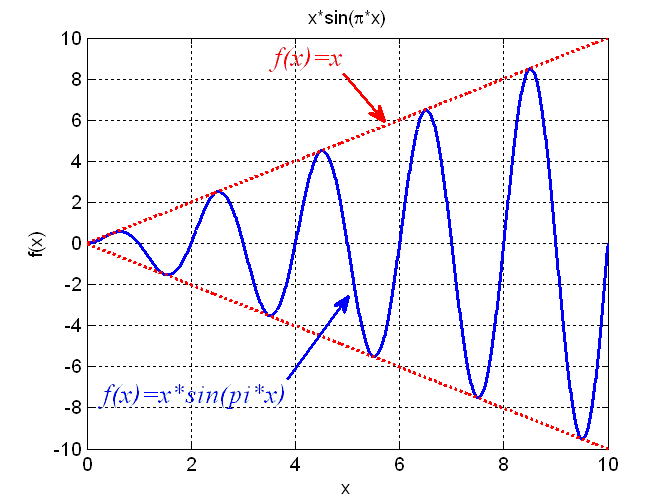
1. 



1. Серпантин Ньютона ,
2. ,
3. ,
4. ,
5. 

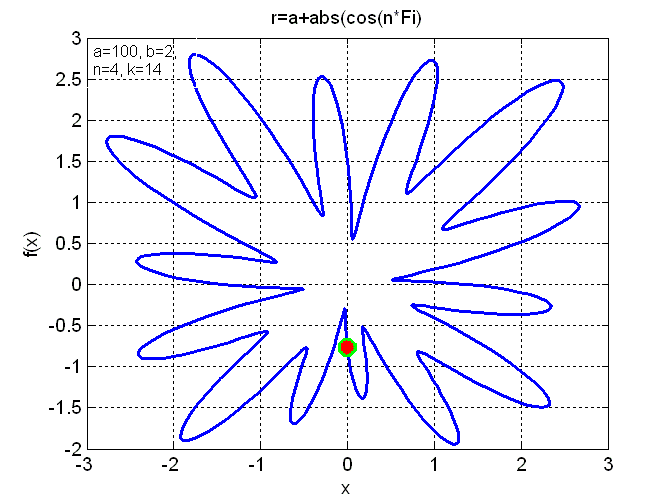


1. 

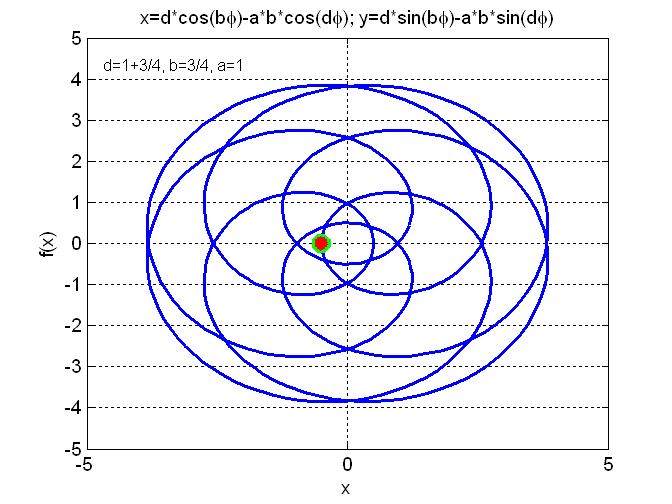


1. , 

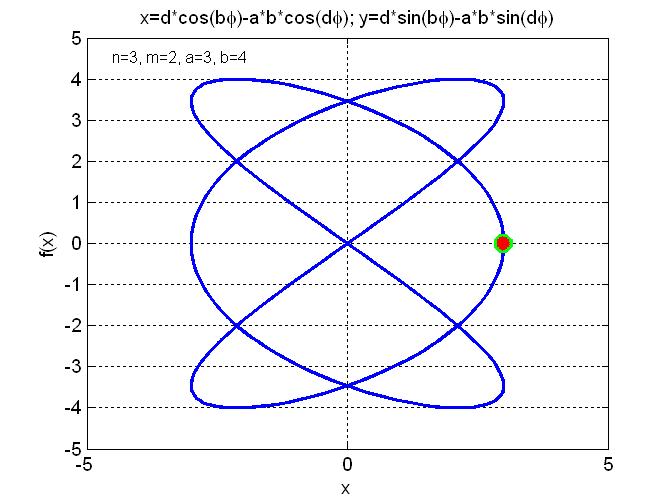




1. 



1. 



|  |  |  |
| --- | --- | --- |
|  | ,  , |  |