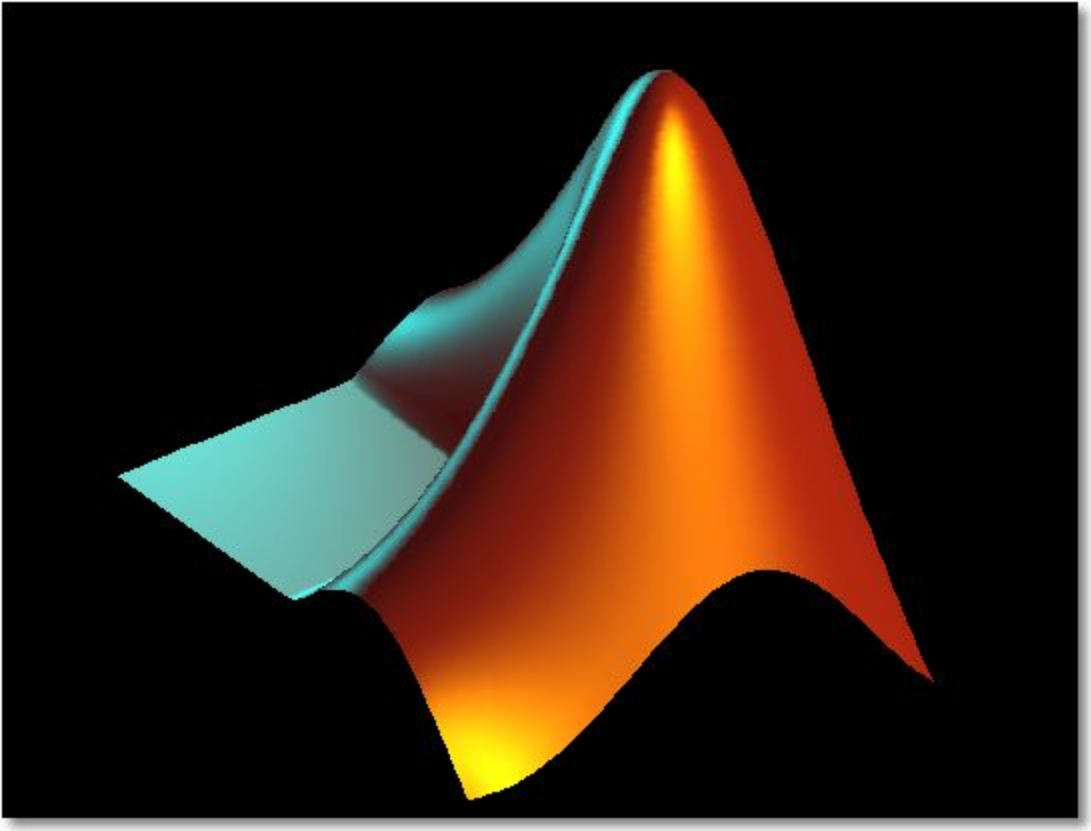


Графические возможности Matlab

Графика в Matlab

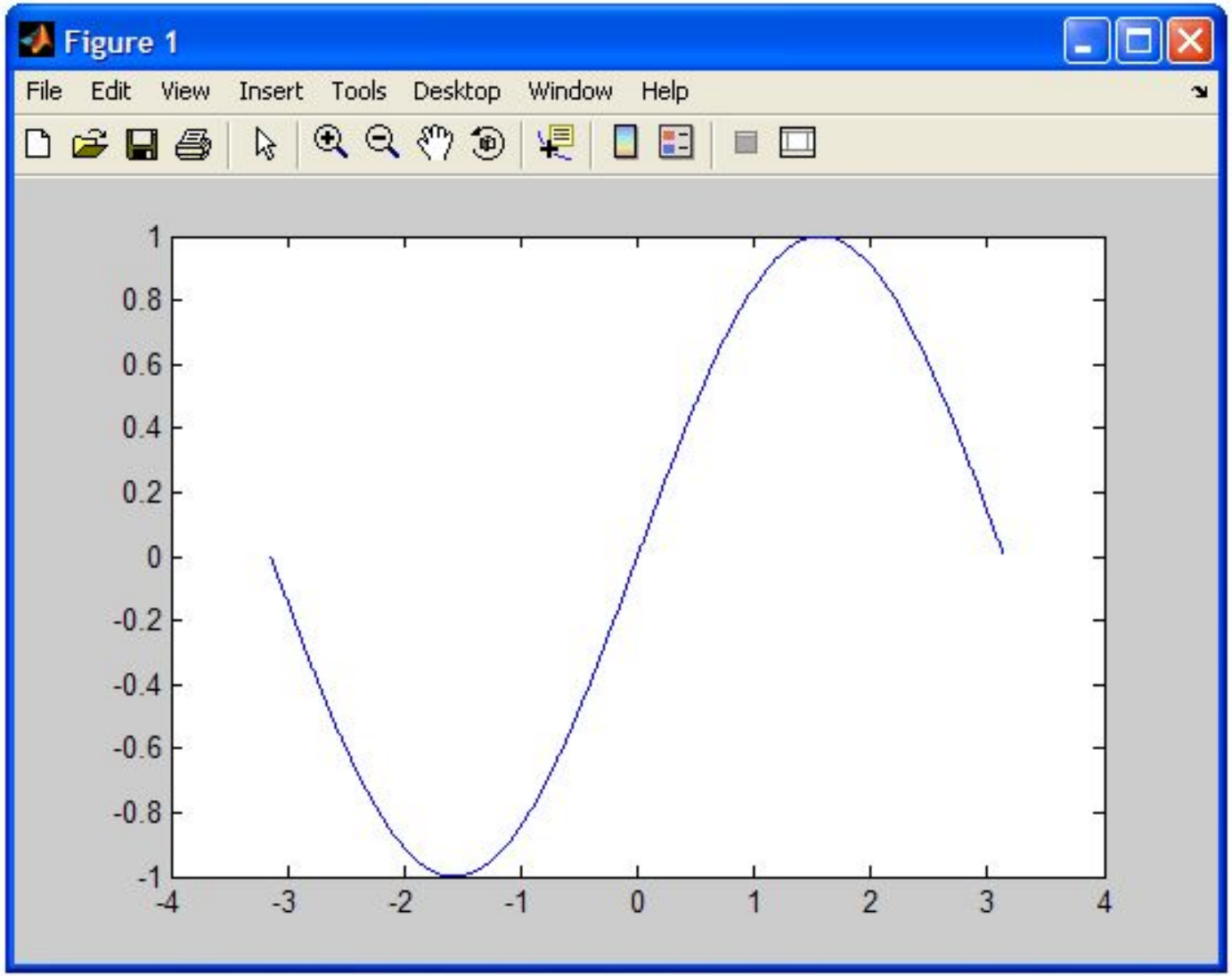
- Высокоуровневая
  - не требует от пользователя детальных знаний о работе графической подсистемы
- Управляемая (*handled*)

доступ к графическим объектам возможен как через инспектор объектов, так и при помощи встроенных функций (дескрипторная графика)

Двумерные (2D-) графики

- Простейший способ построения 2D-графика:
  1. задать область построения (диапазон);
  2. вычислить значение функции на области построения
  3. построить график при помощи одной из встроенных функций Matlab

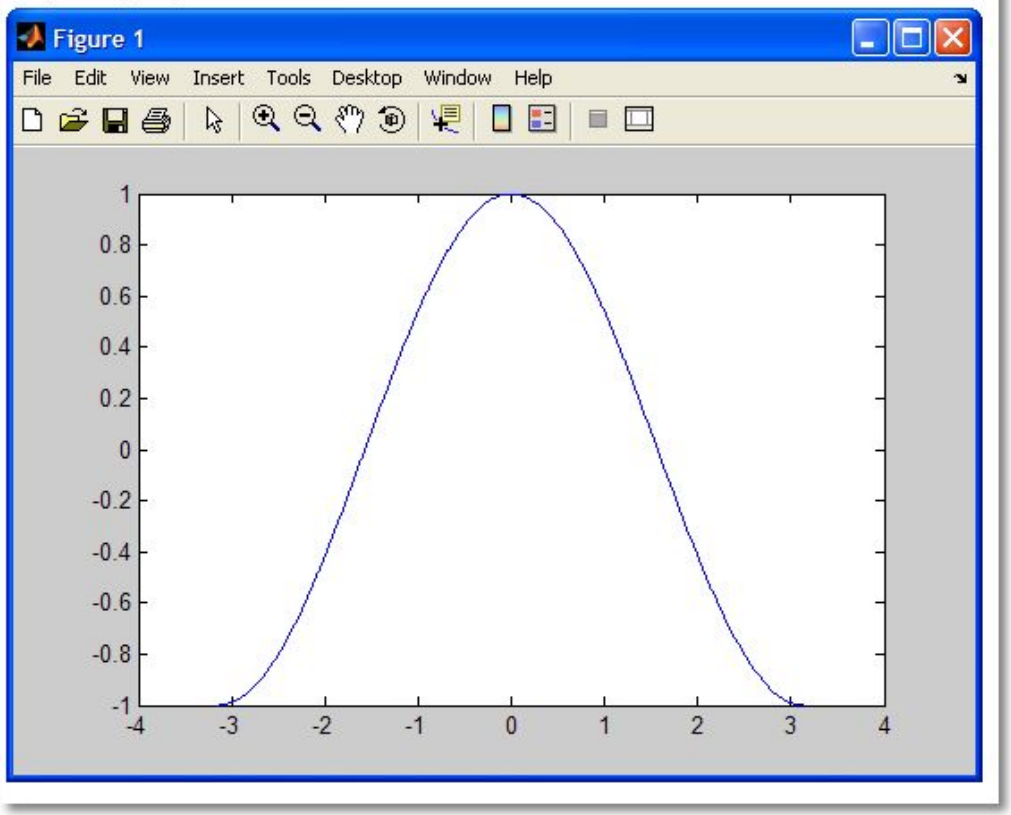
```
>> x = -pi: .01: pi;
>> y = sin(x);
>> plot(x, y)
```



Построение второго графика

- Если сразу же построить другой график, то старый график будет удалён из графического окна

```
>> x = -pi: .01: pi;
>> y = sin(x);
>> plot(x, y)
>> z = cos(x);
>> plot(x, z)
```

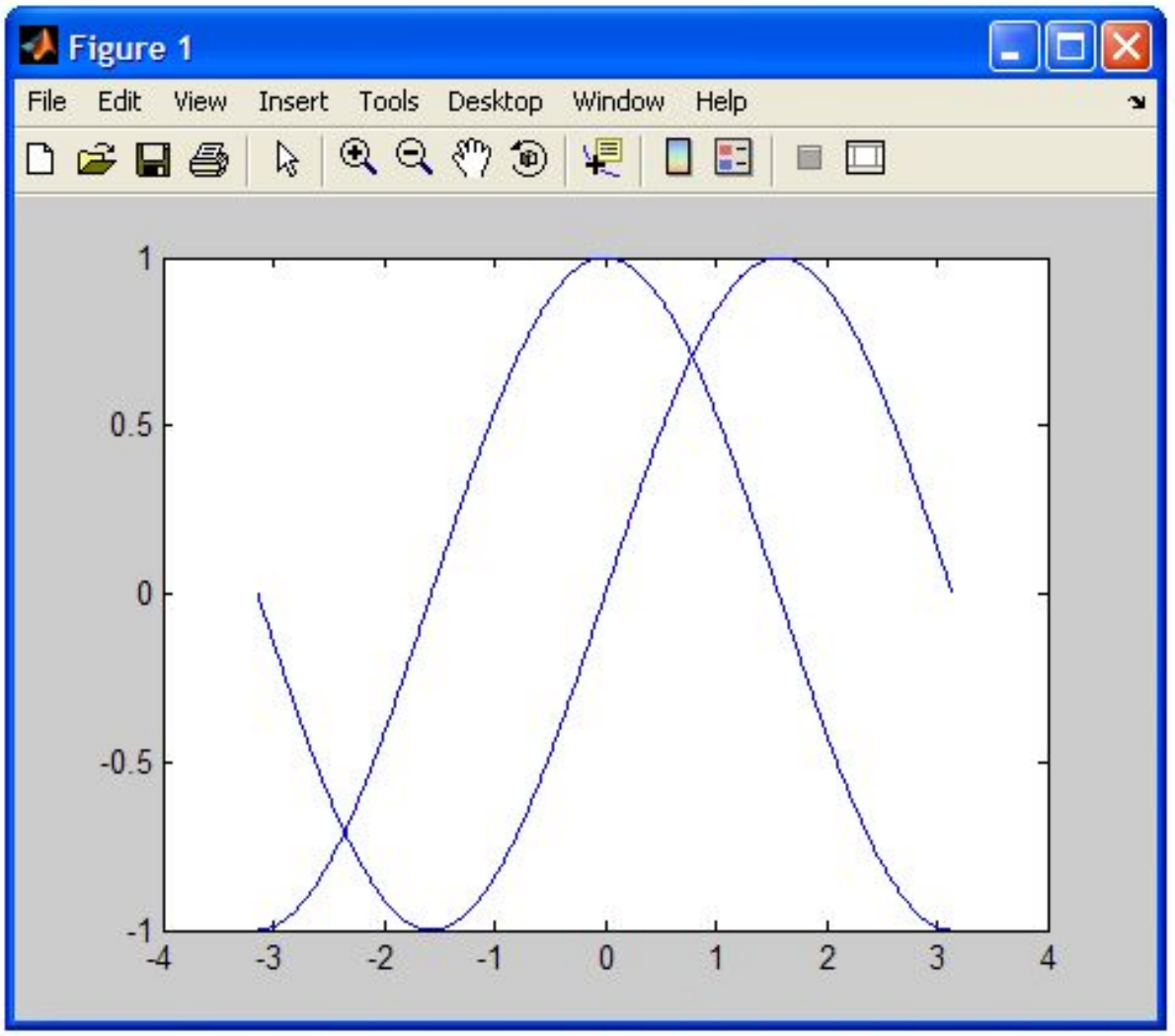


Построение двух графиков в одной системе координат

- Два графика в одной СК можно построить следующими способами:
  1. «закрепить» графическое окно при помощи команды *hold on*
  2. применить одну команду *plot*

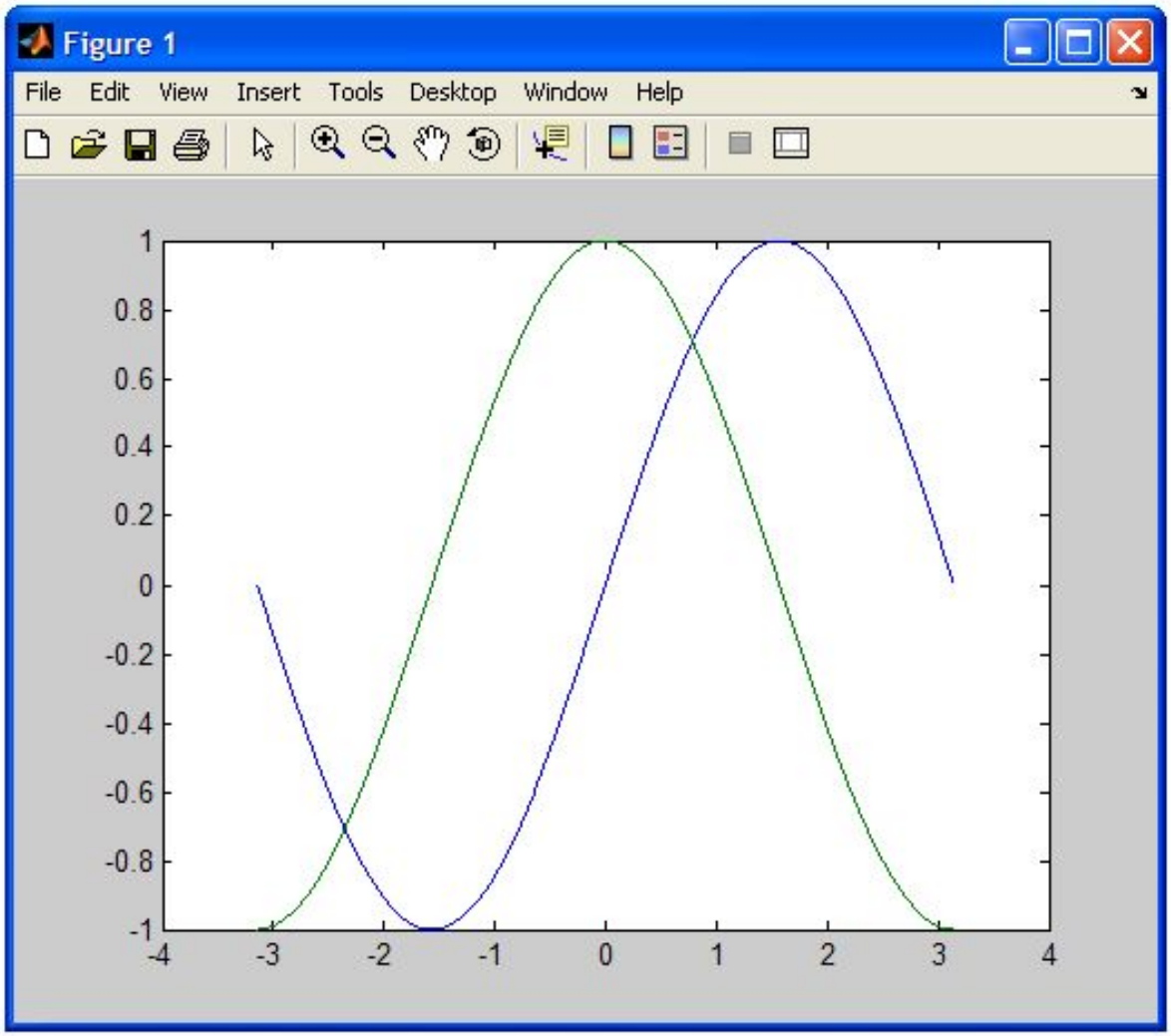
Закрепление графического окна

```
>> x = -pi: .01: pi;
>> y = sin(x);
>> plot(x, y)
>> z = cos(x);
>> hold on
>> plot(x, z)
>> z = cos(x);
```



Параметры команды *plot*

```
>> x = -pi: .01: pi;
>> y = sin(x);
>> z = cos(x);
>> plot(x, y, x, z)
>>
```



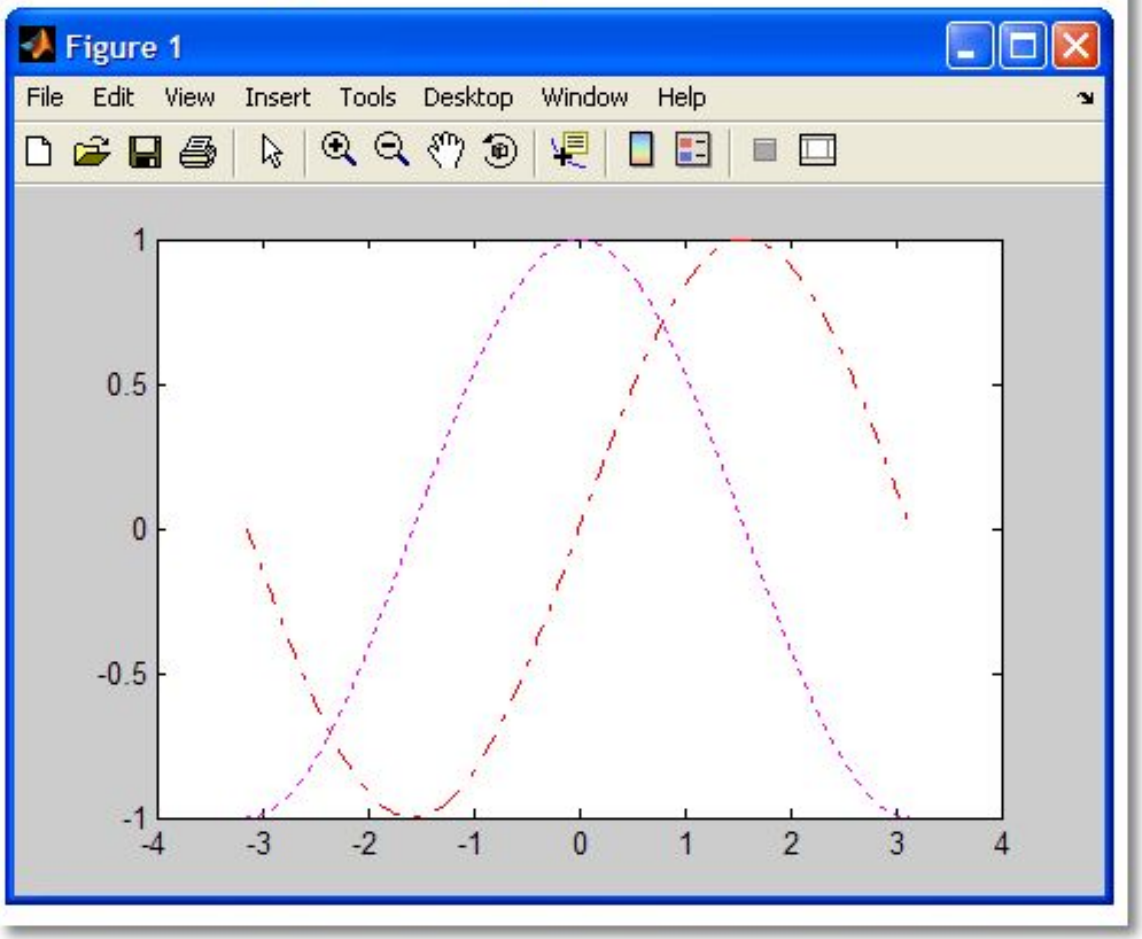
Дополнительные параметры команды *plot*

- В команде *plot* можно задать для каждого графика
  - цвет линии тип маркера тип линии

b	blue	.	point	-	solid
g	green	o	circle	:	dotted
r	red	x	x-mark	-.	dashdot
c	cyan	+	plus	--	dashed
m	magenta	*	star	(none)	no line
y	yellow	s	square		
k	black	d	diamond		
		v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		
		h	hexagram		

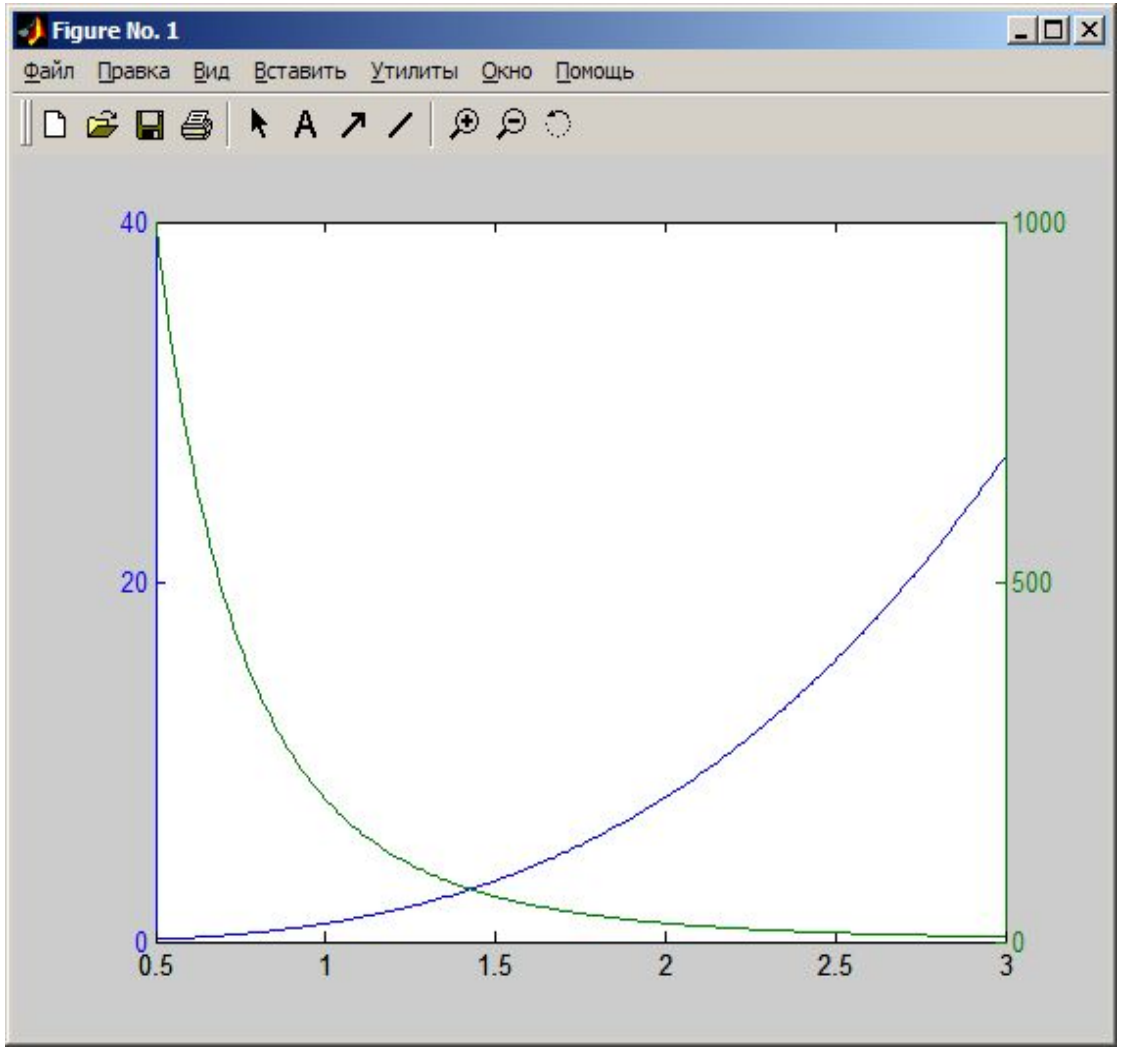
Пример команды `plot`

```
>> x = -pi: .01: pi;
>> y = sin(x);
>> z = cos(x);
>> plot(x, y, 'r-.', x, z, 'm:')
```



Графики в окне с двумя вертикальными осями

```
>> x=0.5:0.01:3;
>> f=x.^3;
>> F=1000*(x+0.5).^-4;
>> plotyy(x,f,x,F)
```

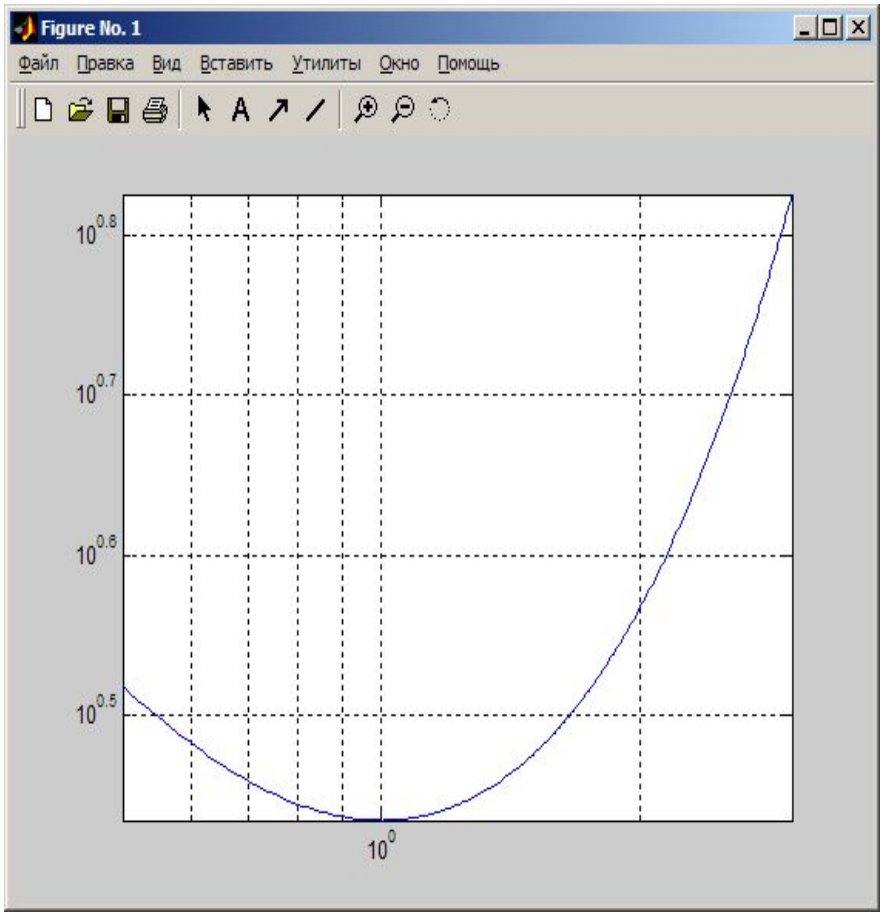


Графики в логарифмическом и полулогарифмическом масштабах

loglog(…)
>>x=logspace(-1,3);
>>loglog(x,exp(x)./x)
>>grid on

Аналогично,
>>semilogx(…)
>>semilogy(…)
Графики в полулогарифмическом масштабе.

Аргументы функций (…) аналогичны аргументам функции plot.



Столбцовые диаграммы.

```
>>subplot (2,1,1)
>>bar(rand(12,3),'stacked')
>>colormap(cool)
>>subplot (2,1,2)
>>barh(rand(5,3),'stacked')
>>colormap(cool)
```

‘stacked’ рисование n столбцов в позиции m друг на друге

barh(…) – столбцовые диаграммы с горизонтальным расположением столбцов

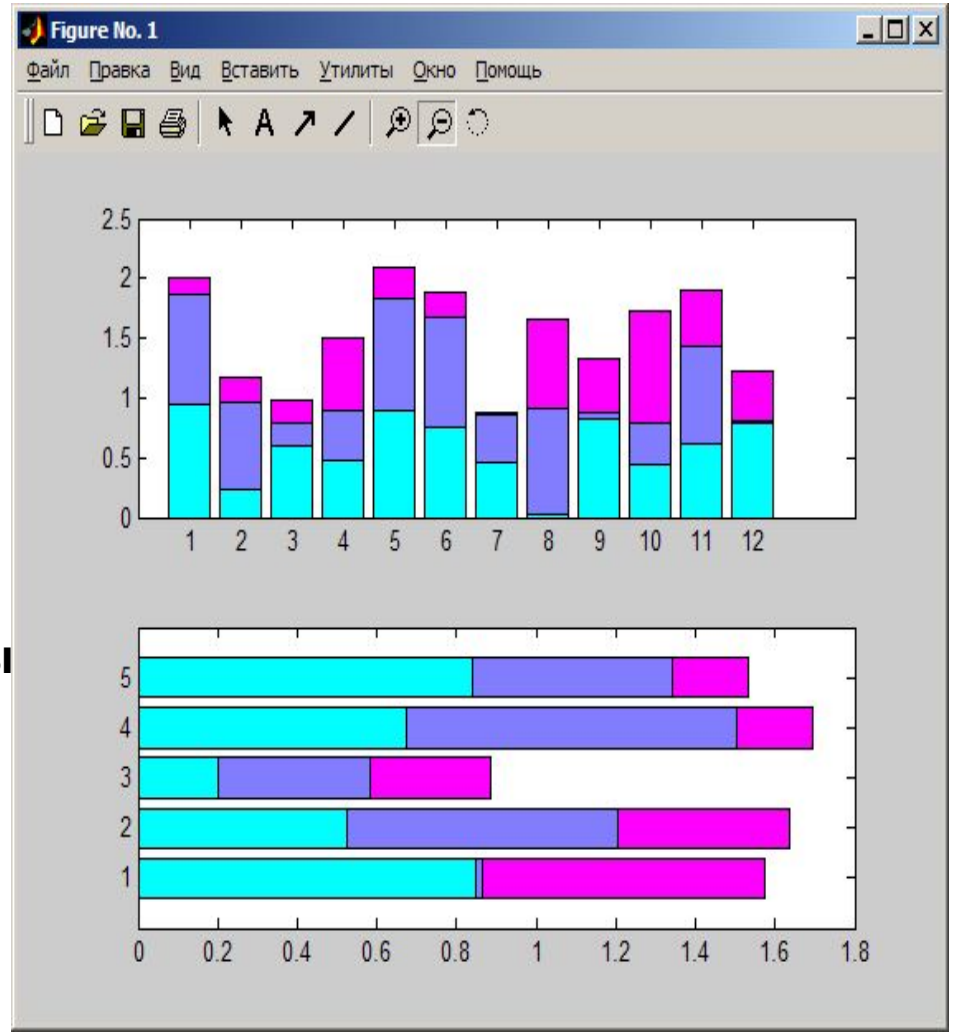
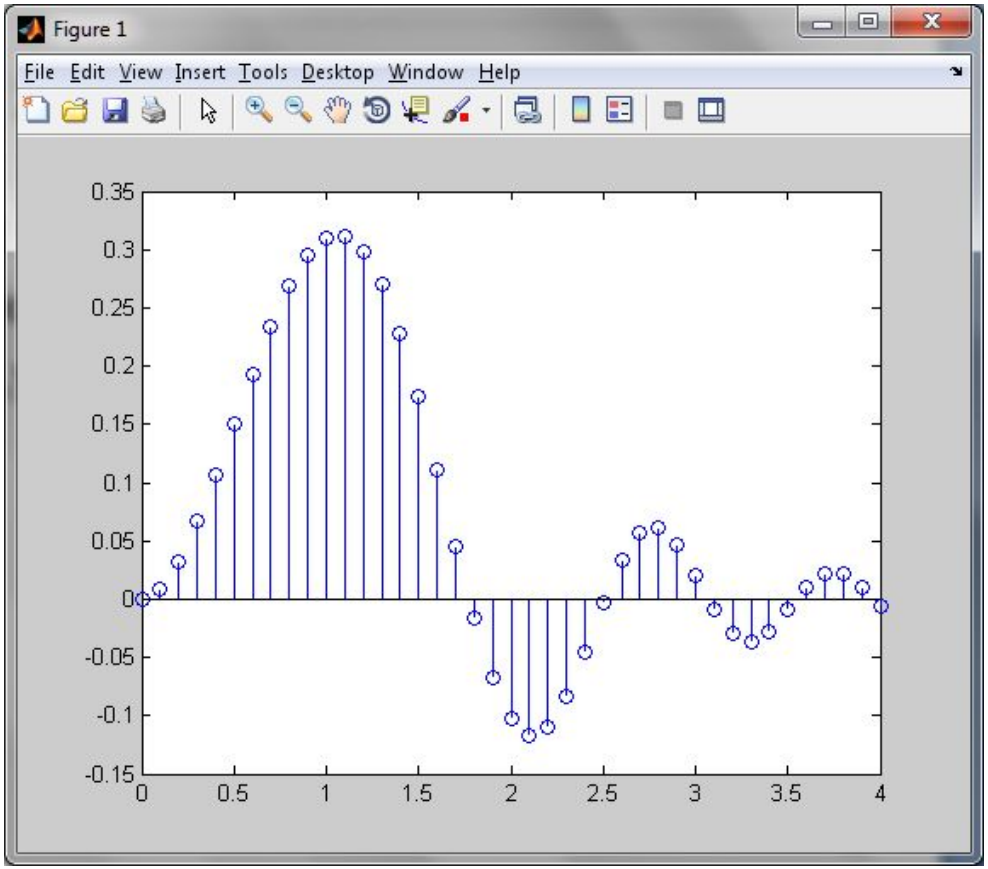
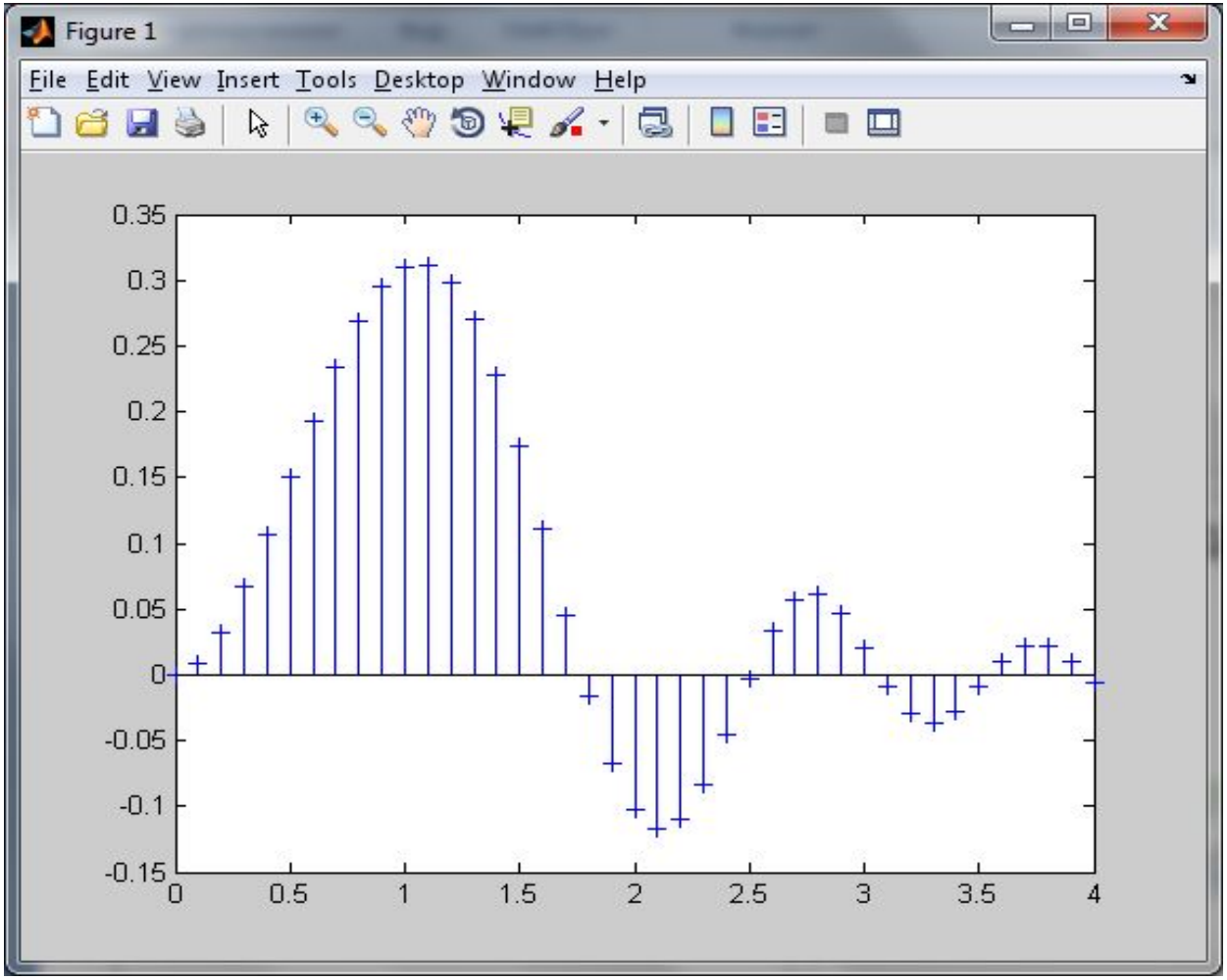


График дискретных отсчетов функции

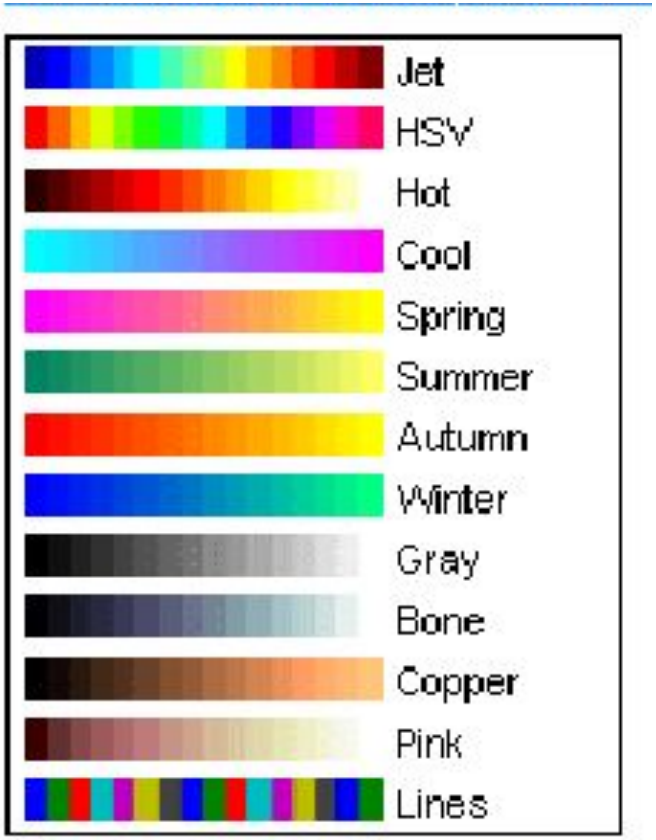
```
>> x=0:0.1:4;
>> y= sin(x.^2).*exp(-x);
>> stem(x,y)
```



Свойства графического объекта



Установка палитры



colormap(map)

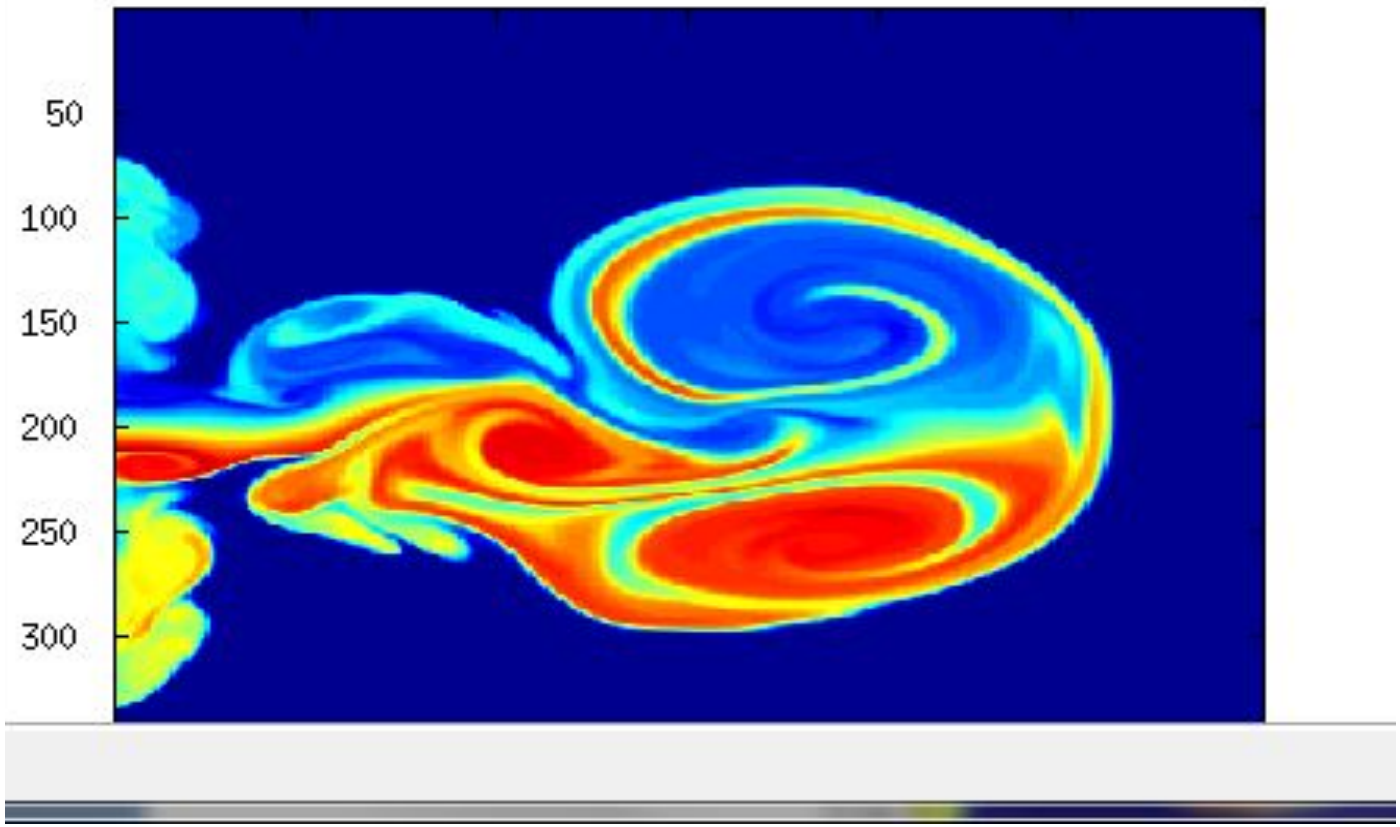
colormap('default')

cmap = colormap

colormap(ax,...)

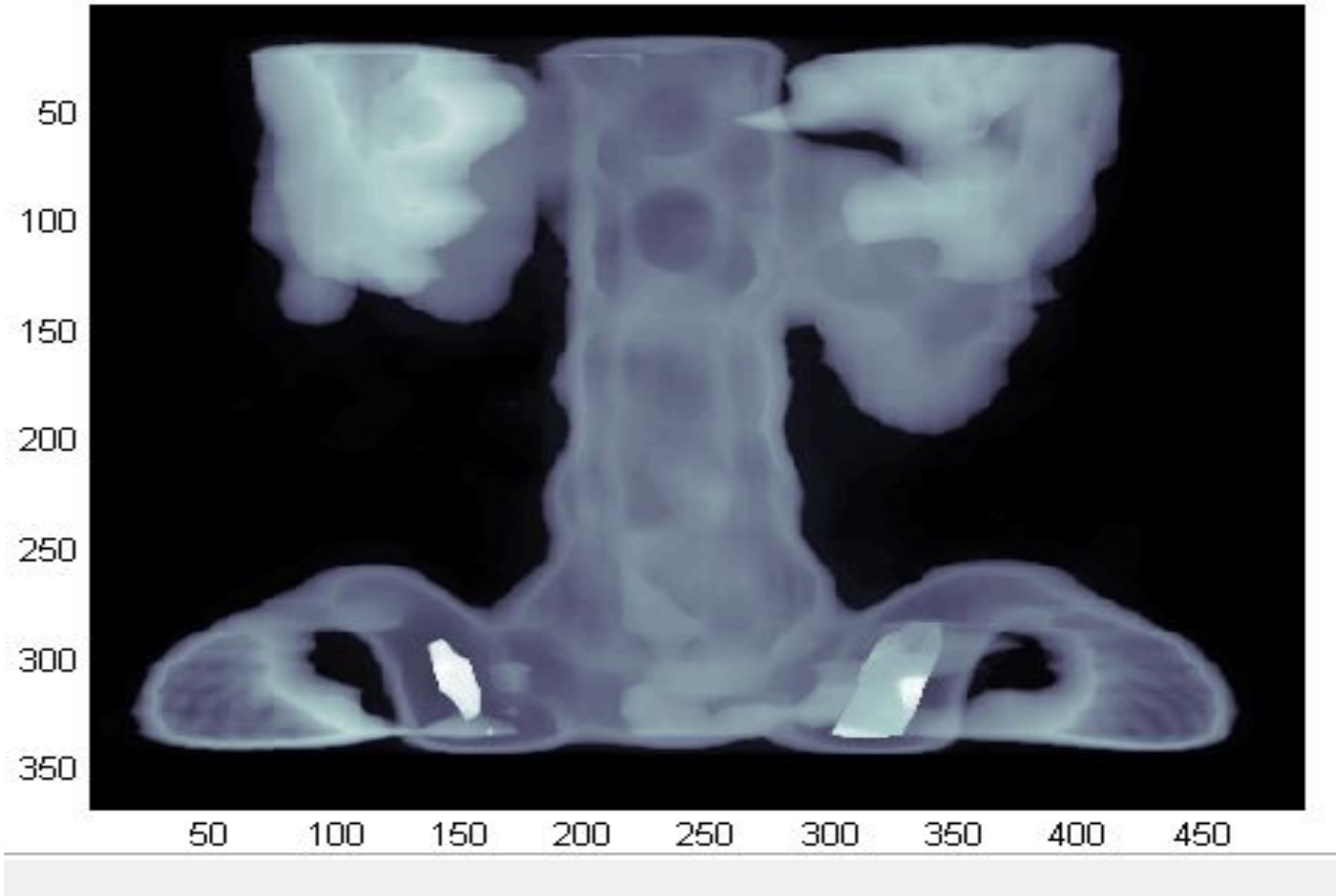
Пример

```
load flujet
image(X)
colormap(jet)
```



Пример

```
load spine
image(X)
colormap bone
```



Гистограммы

N=hist(y,M) – возвращает вектор чисел попаданий для M (скаляр) интервалов.

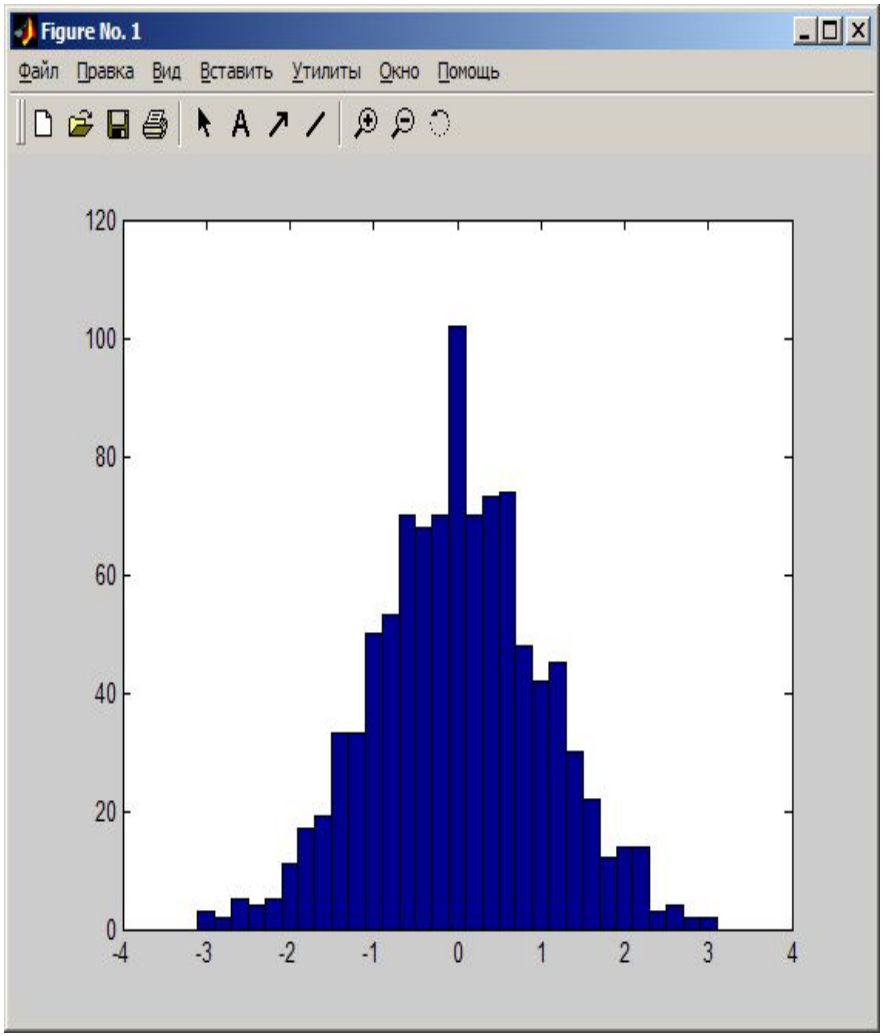
N=hist(y,x) – возвращает числа попаданий элементов вектора y в интервалы, центры которых заданы элементами вектора x.

Пример

```
x=-3:0.2:3;
y=randn(1000,1)
hist(y,x)
h=hist(y,x)
```

В командном окне выведется список попаданий в заданные диапазоны:

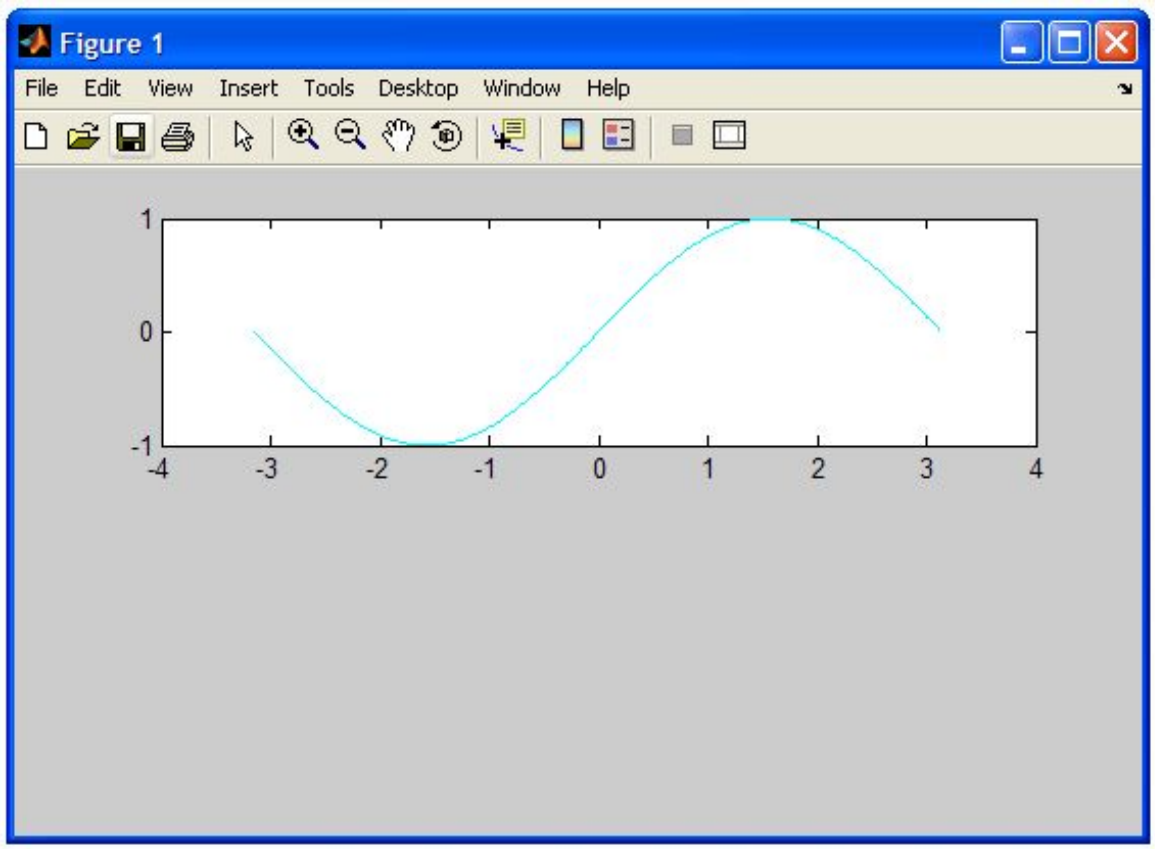
```
h =
3 2  5  4  5 11 17 19 33 33  50 53 70 68 70 102
70 73 74 48 42 45 30 22 1214 14 3 4 2 2
```



Построение нескольких графиков в одном окне в разных подокнах

- Поверхность графического окна можно разделить на зоны, в каждой из которых выводить свой график
- Для этого служит команда *subplot*
- В качестве параметров ей передаётся трёхзначное целое вида *mnk*
- m и n определяют количество графических «подокон» по горизонтали и вертикали
- k задаёт номер графического «подокна»
  - порядок нумерации – по строкам

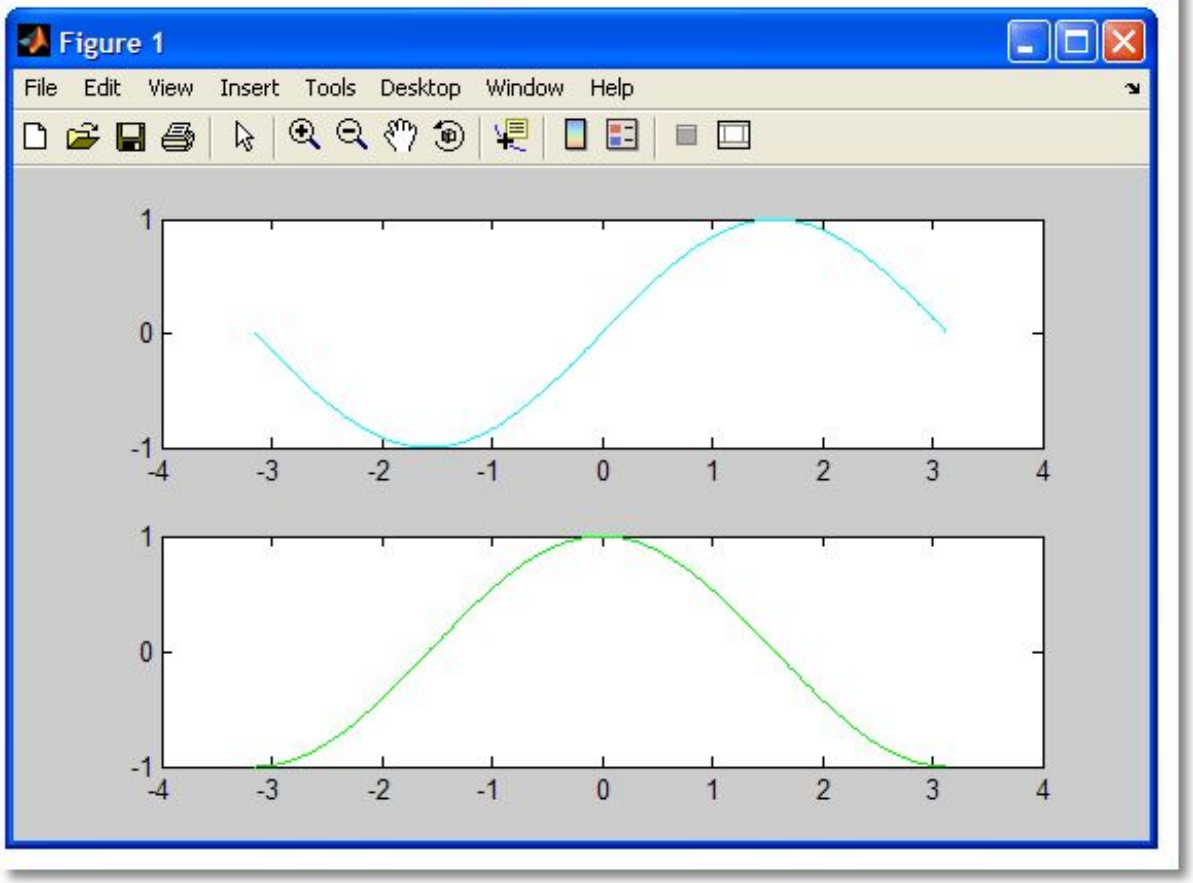
Первый *subplot*




```
>> x = -pi: .01: pi;
>> y = sin(x);
>> z = cos(x);
>> subplot(211); plot(x, y, 'c')
```

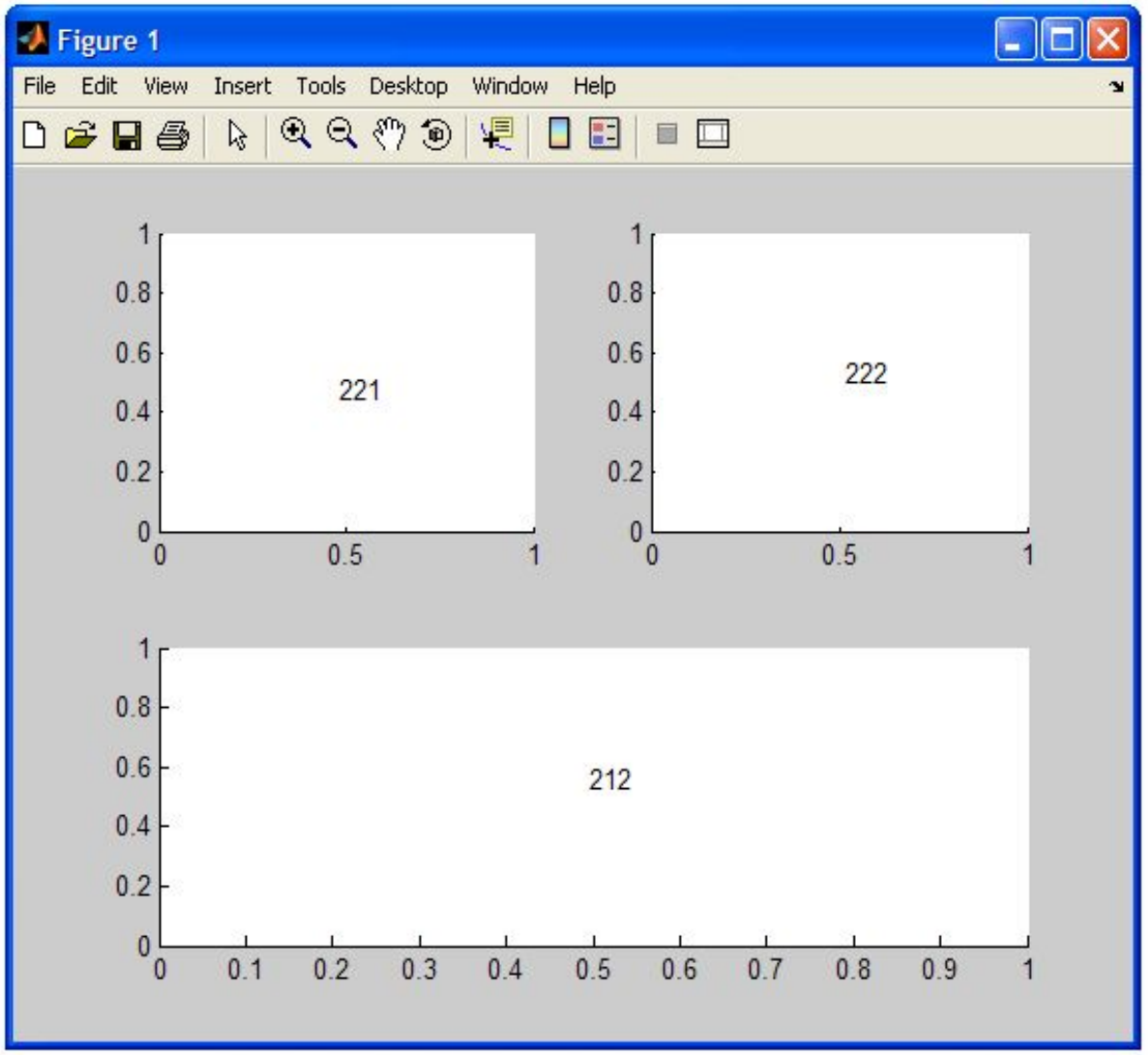
Второй *subplot*

```
>> x = -pi: .01: pi;
>> y = sin(x);
>> z = cos(x);
>> subplot(211); plot(x, y, 'c')
>> subplot(212); plot(x, z, 'g')
```



Более хитрый пример *subplot*

```
>> subplot(221);
>> subplot(222);
>> subplot(212);
>>
>>
```



subplot

Варианты команды

Subplot (m,n,p) или subplot(mnp)

m число рядов

n число колонок

p номер окна: номер отсчитывается вдоль рядов с переходом на новый ряд по исчерпанию

subplot или clf reset

удаляет все подокна и возвращает графическое окно в обычное состояние.

Позиционирование текста с помощью «мыши»

gtext(‘string’)

на графике появляется перемещаемый мышью маркер в виде крестика. Установив маркер в нужное место, достаточно щелкнуть кнопкой мыши для вывода текста.

Построение графиков в разных графических окнах

- Создать новое графическое окно можно командой *figure*
- Команда *figure* создаёт графическое окно и возвращает указатель на него:

```
h = figure
```

- Активизировать ранее созданное окно можно командой *figure(h)*

figure : пример использования 1

```
>> x = -pi: .01: pi;
>> y = sin(x);
>> z = cos(x);
>> plot(x, y)
>> figure
>> plot(x, z, 'r')
```

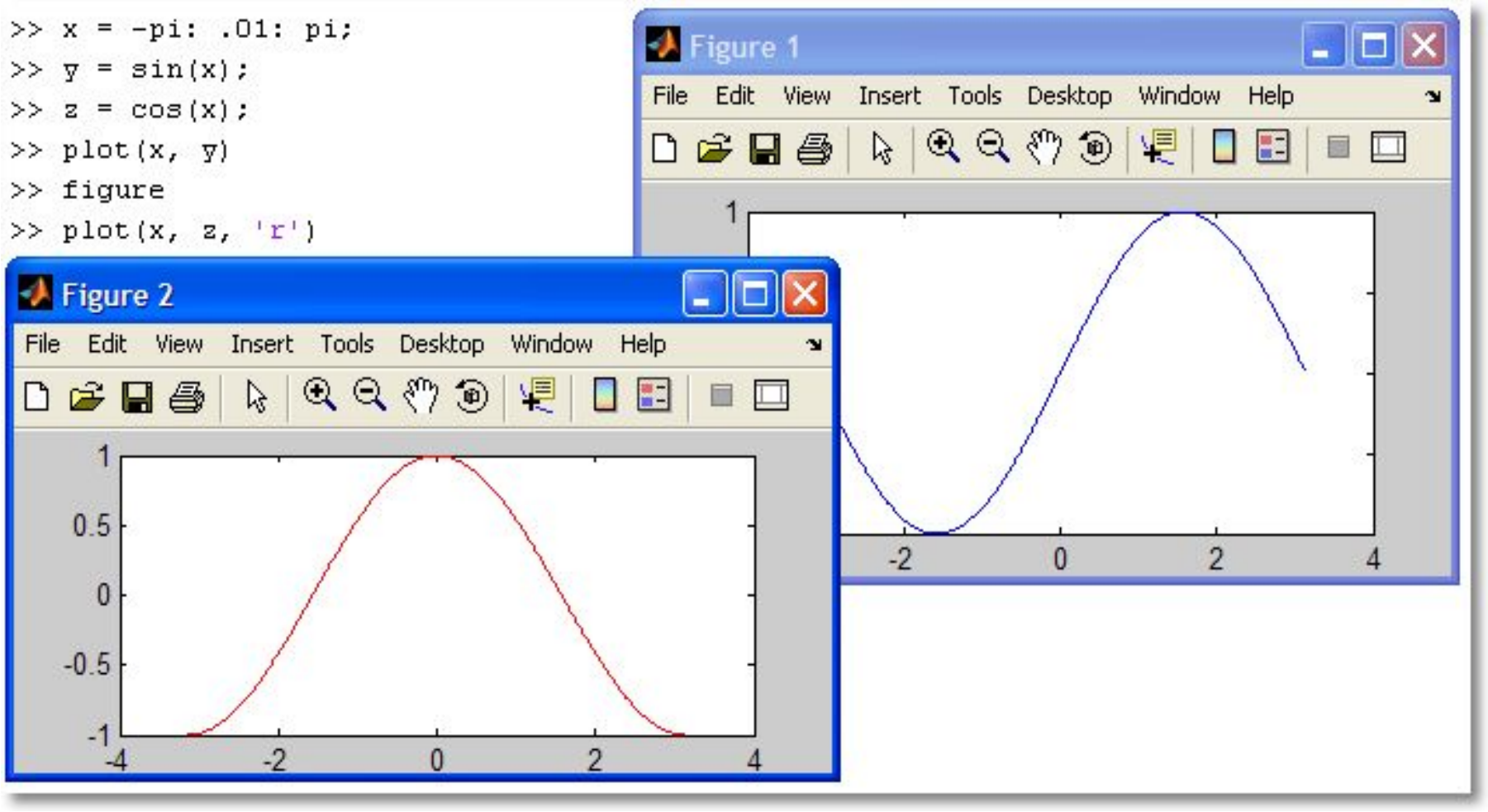


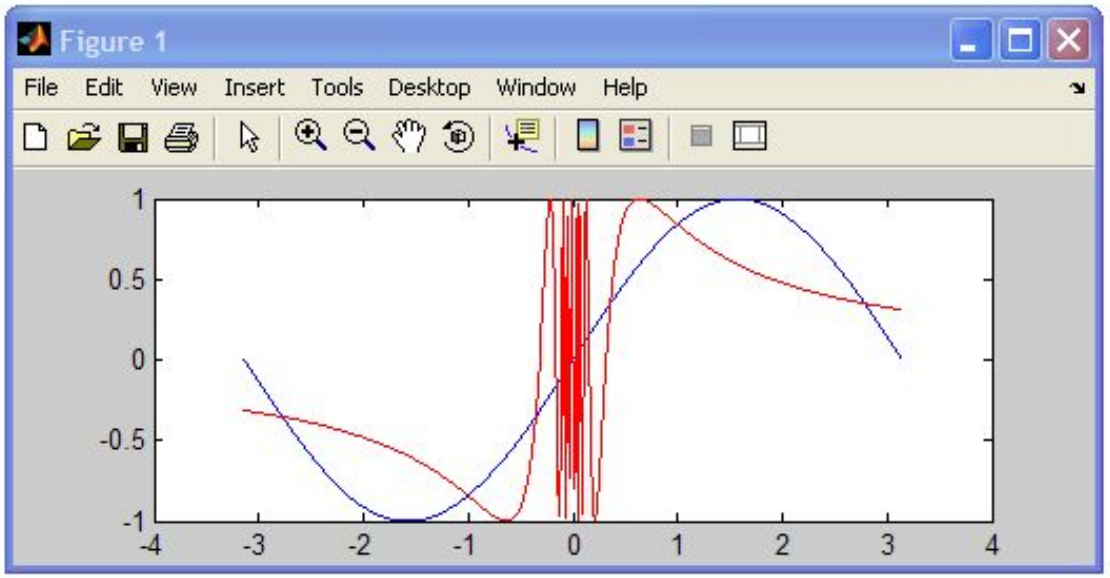
figure : пример использования 2

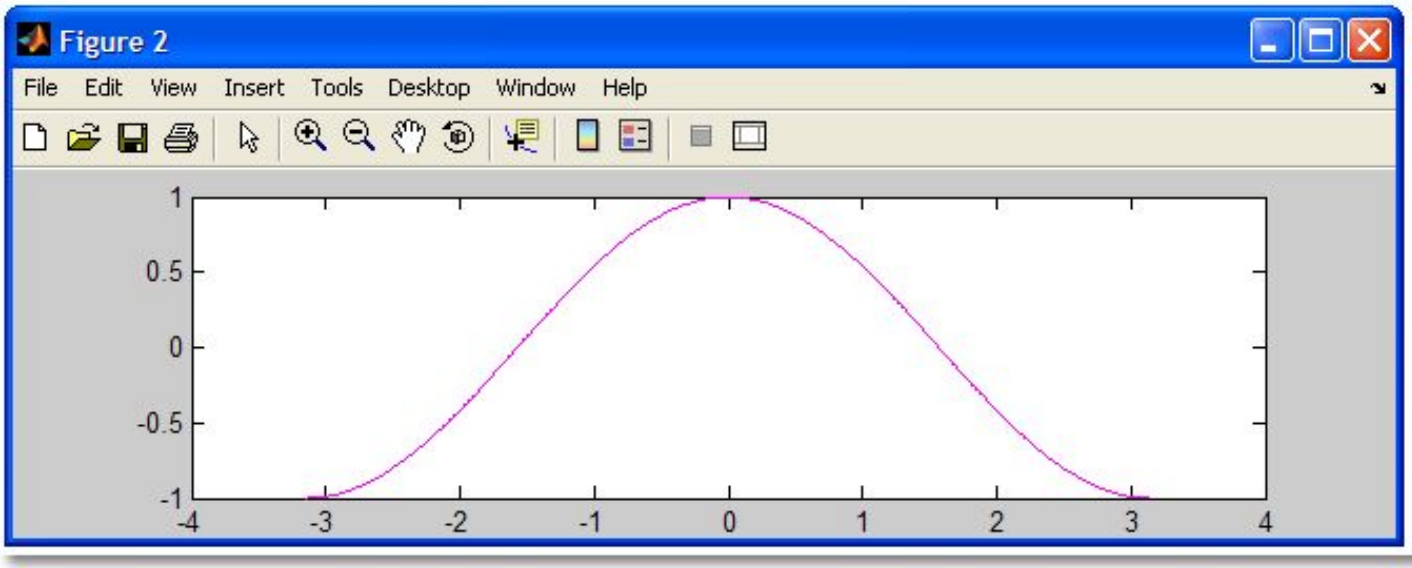
```
>> x = -pi: .01: pi;
>> y = sin(x);
>> z = cos(x);
>> v = sin(1./x);
>> f = figure

f =

     1

>> plot(x, y)
>> figure
>> plot(x, z, 'm')
>> figure(f)
>> hold on
>> plot(x, v, 'r')
```



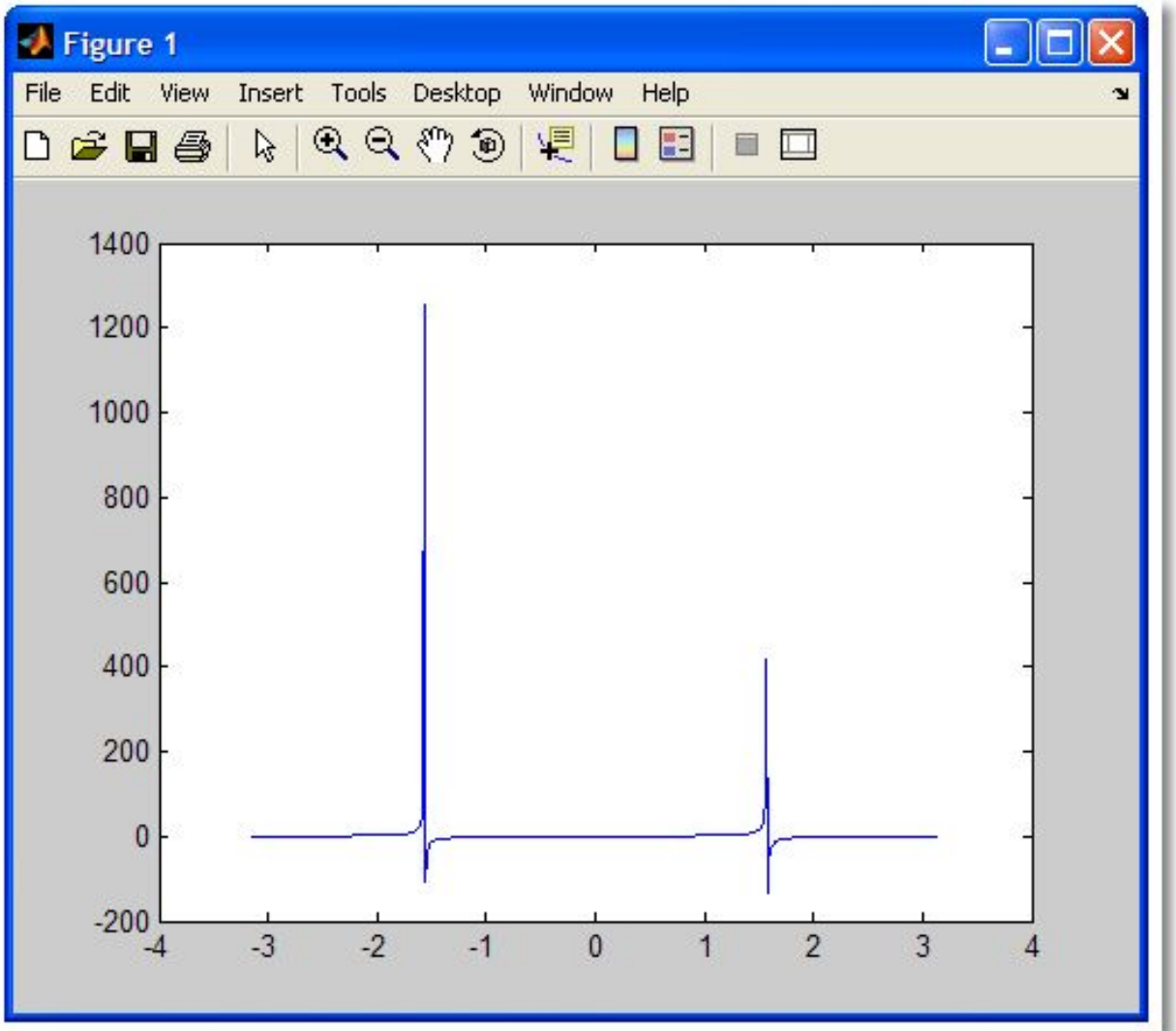


Axis: управление масштабом

- Команда
  `axis([Xmin Xmax Ymin Ymax])`
  задаёт область построения графиков по осям X и Y
- Используется, если результат автомасштабирования неудовлетворителен

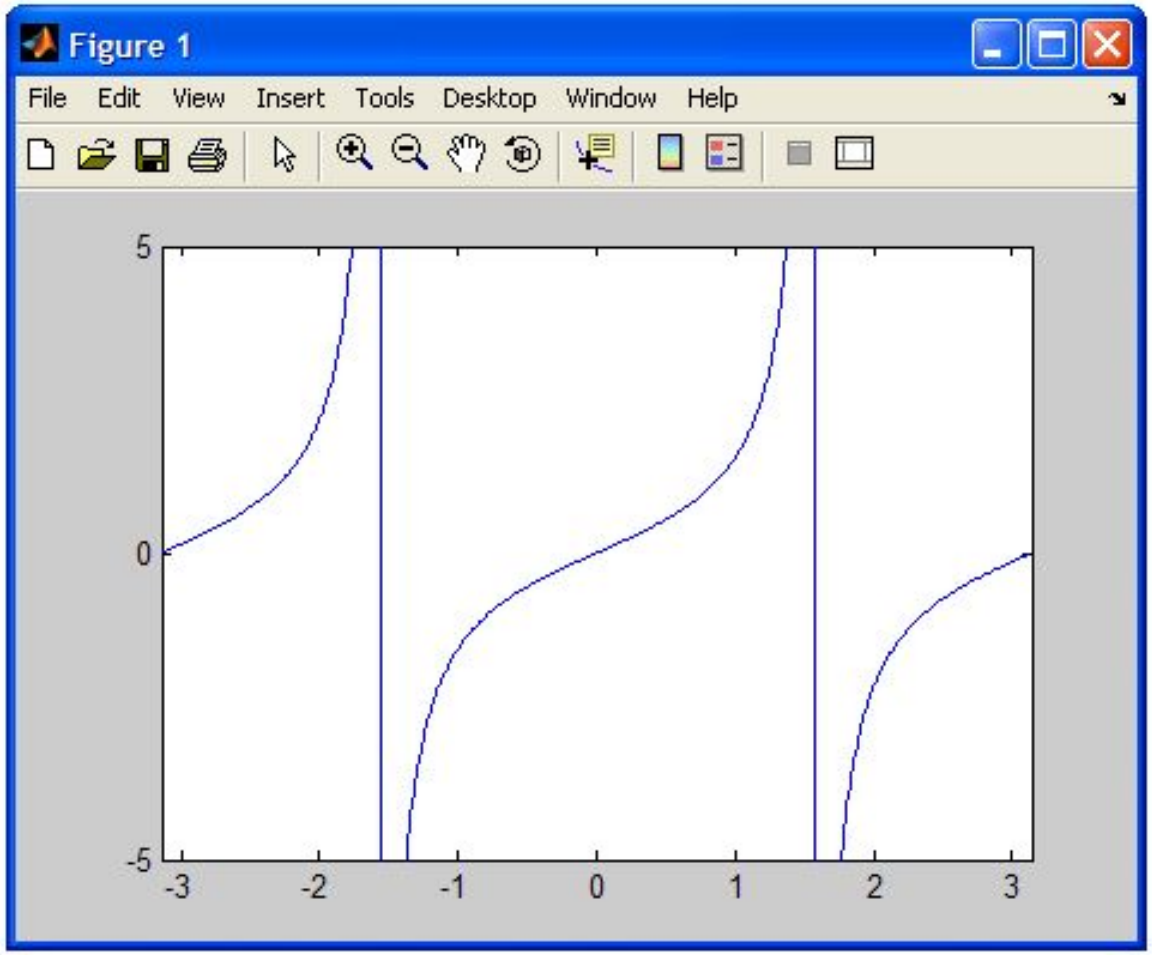
Axis не используется

```
>> x = -pi: .01: pi;
>> y = tan(x);
>> plot(x, y)
>>
```



`Axis` используется

```
>> x = -pi: .01: pi;
>> y = tan(x);
>> plot(x, y), axis([-pi pi -5 5])
```

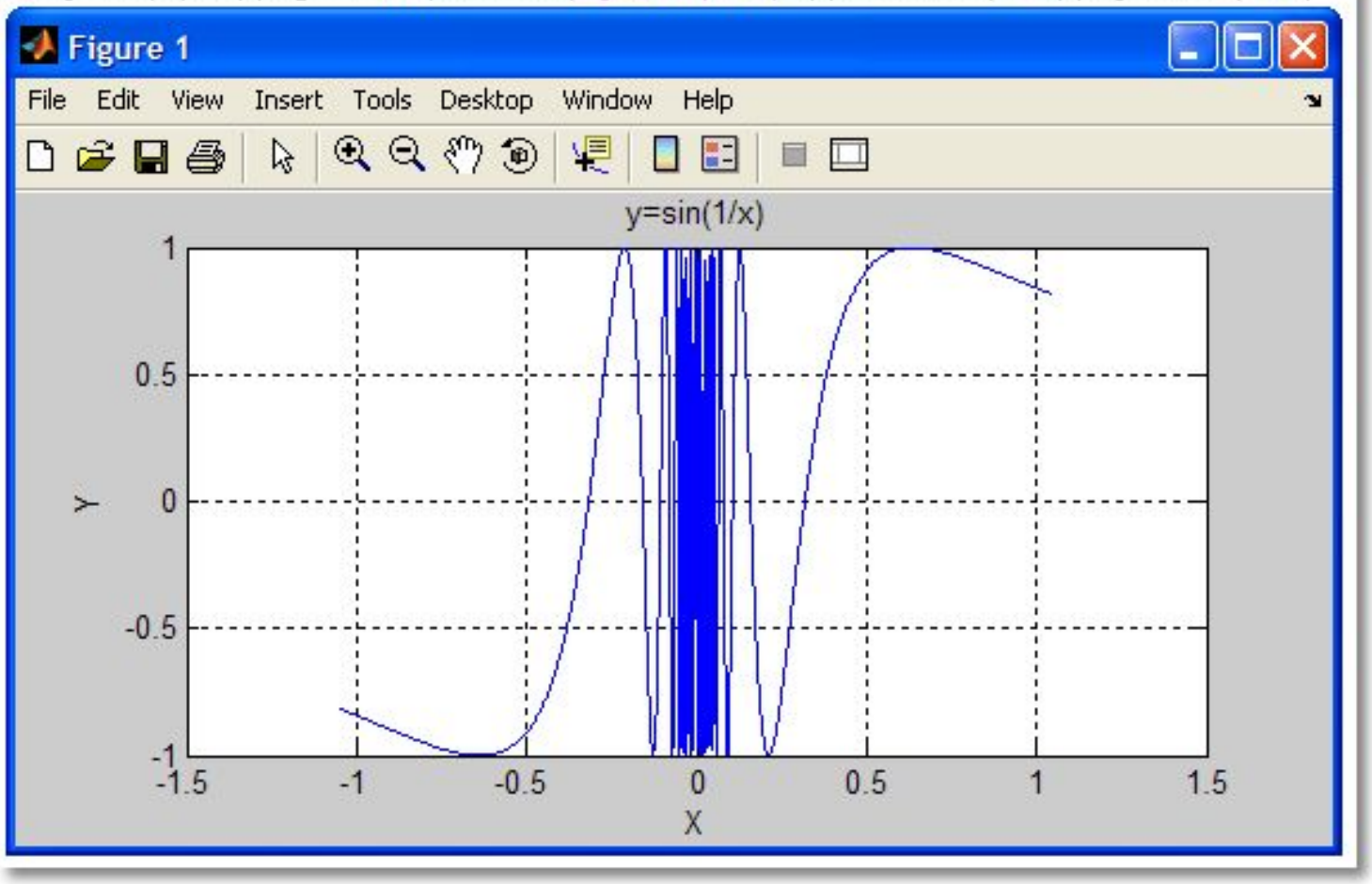


Оформление графиков

- Для графиков можно задать
  - масштабную сетку: `grid on`
  - заголовок: `title('заголовок')`
  - подписи осей: `xlabel('текст')` и `ylabel ('текст')`
- В заголовках и подписях можно использовать нотацию системы TeX

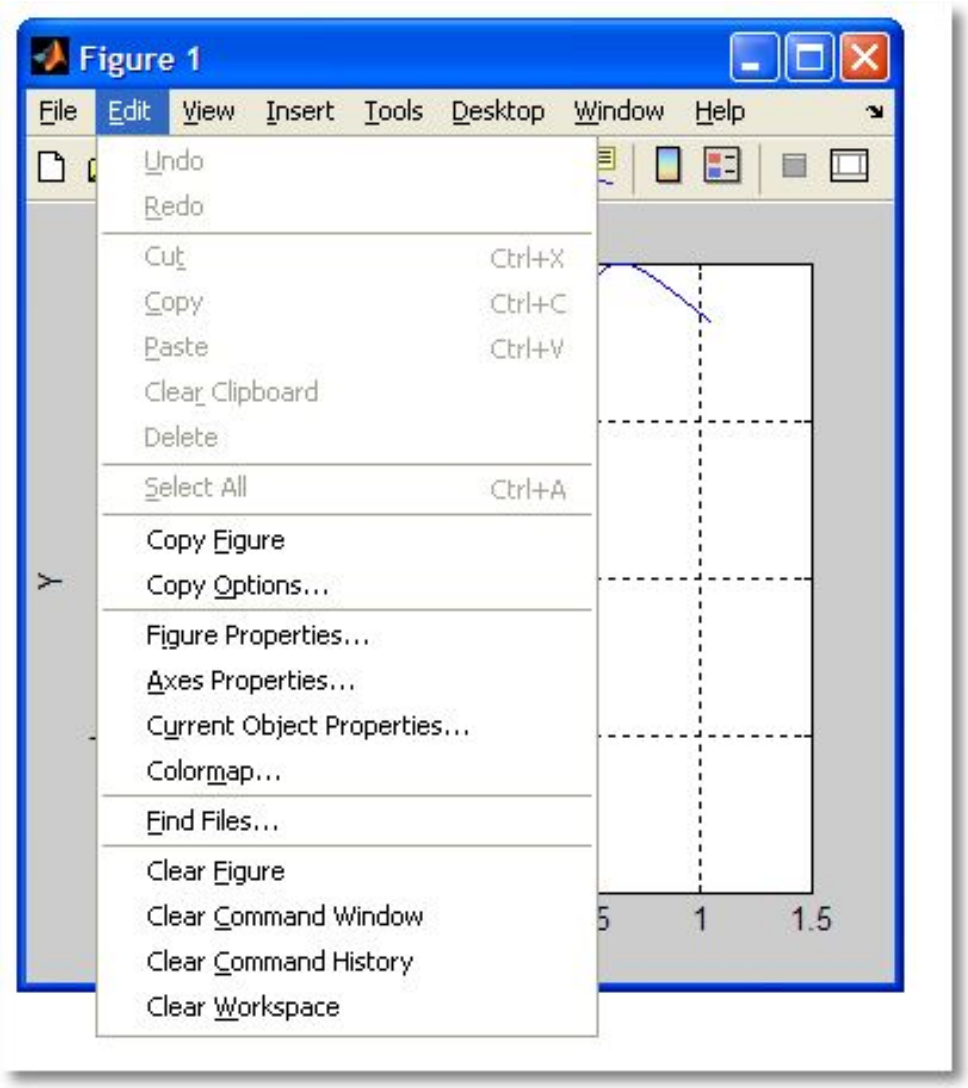
Пример оформления графика

```
>> x = -pi/3: .001: pi/3;
>> v = sin(1./x);
>> plot(x, v), grid on, title('y=sin(1/x)'), xlabel('X'), ylabel('Y')
```



Форматирование графиков

- Доступно из меню `Edit`:

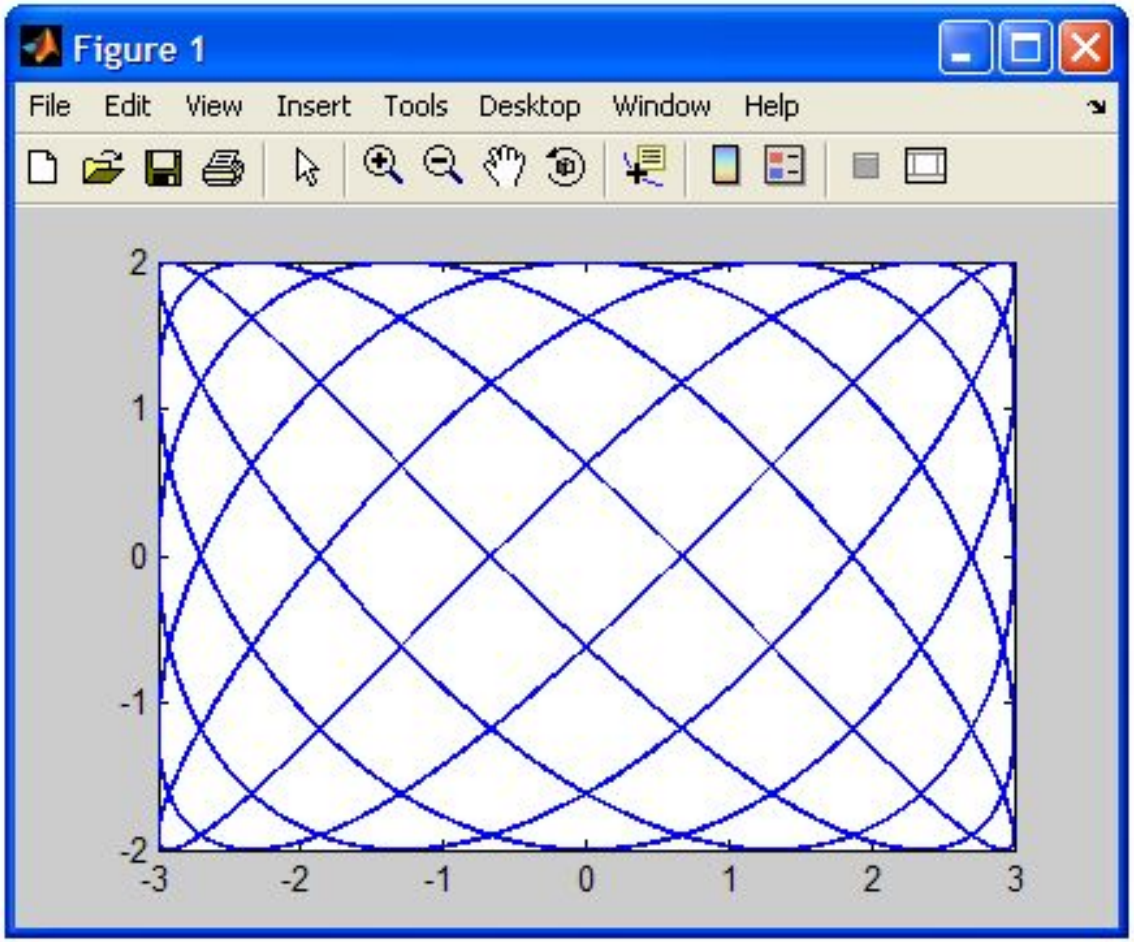


Графики функций, заданных параметрически

- Строятся при помощи оператора *plot*
- Вначале задаётся диапазон построения t
- Затем вычисляются *x(t)* и *y(t)*
- И строится график

Графики функций, заданных параметрически

```
>> t = 0: .01: 100;
>> x = 3*cos(5*t);
>> y = 2*sin(7*t);
>> plot(x, y)
>>
```



Графики функций, заданных параметрически

- Графики параметрических функций часто возникают в физических приложениях
- Независимая переменная t в этом случае имеет смысл времени, x и y – координаты
- Для построения динамического графика можно использовать функцию `comet(x,y)`

Функции в полярной СК

- Строятся аналогично графикам функций в декартовой системе
- Для построения используется команда *polar*

Функции в полярной СК

```
>> phi = 0: .01: 15;
>> r = phi;
>> polar (phi, r, 'g')
>>
```