

ГРАФИЧЕСКИЕ ВОЗМОЖНОСТИ MATLAB

Учебно-методическое пособие
по направлению 200100.62

Бишкек 2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
КЫРГЫЗСКОЙ РЕСПУБЛИКИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
КЫРГЫЗСКО-РОССИЙСКИЙ СЛАВЯНСКИЙ УНИВЕРСИТЕТ

ЕСТЕСТВЕННО-ТЕХНИЧЕСКИЙ ФАКУЛЬТЕТ

Кафедра «Приборостроение»

ГРАФИЧЕСКИЕ ВОЗМОЖНОСТИ MATLAB

Учебно-методическое пособие
по направлению 200100.62

УДК 681.5.08(072)
Г 78

Рецензенты:

Я. И. Рудав – д-р физ.-мат. наук, проф.,
О. В. Неженко – канд. техн. наук, доц.

Составитель

Г. С. Воронова

Рекомендовано к изданию
кафедрой «Приборостроение»

Г 78 ГРАФИЧЕСКИЕ ВОЗМОЖНОСТИ MATLAB: учебно-методическое пособие по направлению 200100.62 / сост. Г. С. Воронова. Бишкек: КРСУ, 2016. 43 с.: ил.

Система MATLAB обладает мощными графическими возможностями. С ее помощью можно без труда визуализировать как двумерные, так и трехмерные данные.

Наиболее простыми и удобными в использовании являются возможности *высокоуровневой графики*. Высокоуровневые графические функции позволяют без особых усилий построить качественные графики, диаграммы, гистограммы, линии и поверхности. При этом всю работу, связанную с выбором цвета и типа линии, масштаба осей и т. д., система выполняет самостоятельно. Для оформления и редактирования графиков предусмотрены специальные команды пакета MATLAB, инструменты графических окон, а также редактор графиков.

Рекомендуется для студентов 2 курса естественно-технического факультета.

СОДЕРЖАНИЕ

ЦЕЛЬ РАБОТЫ.....	4
ВВЕДЕНИЕ	4
1. Программный продукт MATLAB.....	5
1.1. Основные сведения о MATLAB	5
1.2. Графические команды и функции	5
1.3. График в линейном масштабе.....	8
1.4. График в логарифмическом масштабе.....	12
1.5. График в полулогарифмическом масштабе.....	13
1.6. График в полярных координатах.....	13
1.7. Построение линий и точек в трехмерном пространстве.....	15
2. Формирование двумерных массивов	17
2.1. Формирование двумерных массивов X и Y	17
2.2. Трехмерная сетчатая поверхность.....	18
2.3. Затененная сетчатая поверхность	20
2.4. Затененная поверхность с подсветкой	23
2.5. Масштабирование осей и вывод на экран	24
2.6. Нанесение сетки.....	26
3. Новая графическая система в MATLAB.....	26
3.1. Улучшенный вид – пример 3D графики	26
3.2. Улучшенный вид – пример 2D графики	27
3.3. Графические объекты	28
3.4. Поворачиваемые подписи осей.....	32
3.5. Построение графиков с датами.....	33
3.6. Поддержка многоязычного текста и символов	35
3.7. Множественные цветовые палитры	37
3.8. Вкладки в пользовательских интерфейсах	38
3.9. Другие новые возможности	39
ЛИТЕРАТУРА	42

ЦЕЛЬ РАБОТЫ

Данное учебно-методическое пособие посвящено изучению высокоуровневой графики в среде MATLAB (Matrix laboratory) – высокопроизводительном языке для технических расчетов, который включает в себя вычисления, визуализацию и программирование в удобной среде.

ВВЕДЕНИЕ

Высокоуровневая графика (high level graphics) позволяет пользователю получать результаты в графическом виде, прикладывая минимум усилий с использованием функций из командной строки.

Начиная с версии 4.0, в состав системы MATLAB входит мощная графическая подсистема, которая поддерживает как средства визуализации двумерной и трехмерной графики на экран терминала, так и средства презентационной графики. Следует выделить несколько уровней работы с графическими объектами. В первую очередь, это команды и функции, ориентированные на конечного пользователя и предназначенные для построения графиков в прямоугольных и полярных координатах, гистограмм и столбцовых диаграмм, трехмерных поверхностей и линий уровня, анимации. Графические команды высокого уровня автоматически контролируют масштаб, выбор цветов, не требуя манипуляций со свойствами графических объектов. Соответствующий низкоуровневый интерфейс обеспечивается дескрипторной графикой, когда каждому графическому объекту ставится в соответствие графическая поддержка (дескриптор), на который можно ссылаться при обращении к этому объекту. Используя дескрипторную графику, можно создавать меню, кнопки вызова, текстовые панели и другие объекты графического интерфейса [1].

MATLAB. Исходя из указанной цели, можно выделить частные задачи, поставленные в учебно-методическом пособии:

1. ознакомиться с литературой по данной теме;
2. изучить программу MATLAB;
3. получить представление о высокоуровневой графике;
4. поработать в MATLAB над созданием различных графиков.

Объектом исследования является высокоуровневая графика.

Предметом исследования – программный продукт MATLAB.

1. ПРОГРАММНЫЙ ПРОДУКТ MATLAB

1.1. Основные сведения о MATLAB

Название MATLAB является сокращением от Matrix Laboratory, и первоначально пакет MATLAB разрабатывался как средство доступа к библиотекам программ LINPACK и EISPACK, предназначенных для матричных вычислений. Пакет MATLAB создан компанией MathWorks около тридцати лет назад. Работа сотен ученых и программистов направлена на постоянное расширение его возможностей и совершенствование заложенных алгоритмов. В настоящее время MATLAB является мощным и универсальным средством решения задач, возникающих в различных областях человеческой деятельности. Спектр проблем, исследование которых может быть осуществлено при помощи MATLAB, охватывает: матричный анализ, обработку сигналов и изображений, задачи...

1.2. Графические команды и функции

Начиная с версии 4.0 в состав системы MATLAB входит мощная графическая подсистема, которая поддерживает как средства визуализации двумерной и трехмерной графики на экран терминала, так и средства презентационной графики. Следует выделить несколько уровней работы с графическими объектами. В первую очередь, это команды и функции, ориентированные на конечного пользователя и предназначенные для построения графиков в прямоугольных и полярных координатах, гистограмм и столбцовых диаграмм, трехмерных поверхностей и линий уровня, анимации. Графические команды высокого уровня автоматически контролируют масштаб, выбор цветов, не требуя манипуляций со свойствами графических объектов. Соответствующий низкоуровневый интерфейс обеспечивается дескрипторной графикой, когда каждому графическому объекту ставится в соответствие графическая поддержка (дескриптор), на который можно ссылаться при обращении к этому объекту. Используя дескрипторную графику, можно создавать меню, кнопки вызова, текстовые панели и другие объекты графического интерфейса.

Из-за ограниченного объема данного справочного пособия в него включены только графические команды и функции с минимальными элементами дескрипторной графики. Для более глубокого изучения следует обратиться к документации по системе MATLAB, и, в первую очередь, к книге “Using MATLAB Graphics” (Natick, 1996).

Элементарные графические функции системы MATLAB позволяют построить на экране и вывести на печатающее устройство следующие типы графиков: линейный, логарифмический, полулогарифмический, полярный.

Для каждого графика можно задать заголовок, нанести обозначение осей и масштабную сетку.

Двумерные графики

- PLOT – график в линейном масштабе
- LOGLOG – график в логарифмическом масштабе
- SEMILOGX, SEMILOGY – график в полулогарифмическом масштабе
- POLAR – график в полярных координатах

Трехмерные графики

В системе MATLAB предусмотрено несколько команд и функций для построения трехмерных графиков. Значения элементов числового массива рассматриваются как z-координаты точек над плоскостью, определяемой координатами x и y. Возможно несколько способов соединения этих точек. Первый из них – это соединение точек в сечении (функция plot3), второй – построение сетчатых поверхностей (функции mesh и surf). Поверхность, построенная с помощью функции mesh – это сетчатая поверхность, ячейки которой имеют цвет фона, а их границы могут иметь цвет, который определяется свойством EdgeColor графического объекта surface. Поверхность, построенная с помощью функции surf – это сетчатая поверхность, у которой может быть задан цвет не только границы, но и ячейки; последнее управляется свойством FaceColor графического объекта surface. Уровень изложения данного учебно-методического пособия не требует от студентов знания объектно-ориентированного программирования. Ее объем не позволяет в полной мере описать графическую подсистему, которая построена на таком подходе.

- PLOT3 – построение линий и точек в трехмерном пространстве
- MESHGRID – формирование двумерных массивов X и Y
- MESH, MESHZ, MESHZ – трехмерная сетчатая поверхность
- SURF, SURFC – затененная сетчатая поверхность
- SURFL – затененная поверхность с подсветкой
- AXIS – масштабирование осей и вывод на экран
- GRID – нанесение сетки

- HOLD – управление режимом сохранения текущего графического окна
- SUBPLOT – разбиение графического окна
- ZOOM – управление масштабом графика
- COLORMAP – палитра цветов
- CAXIS – установление соответствия между палитрой цветов и масштабированием осей
- SHADING – затенение поверхностей
- CONTOURC – формирование массива описания линий уровня
- CONTOUR – изображение линий уровня для трехмерной поверхности
- CONTOUR3 – изображение трехмерных линий уровня

Надписи и пояснения к графикам

- TITLE – заголовки для двух- и трехмерных графиков
- XLABEL, YLABEL, ZLABEL – обозначение осей
- CLABEL – маркировка линий уровня
- TEXT – добавление к текущему графику текста
- GTEXT – размещает заданный текст на графике с использованием мыши
- LEGEND – пояснение к графику
- COLORBAR – шкала палитры

Специальная графика

Раздел специальной графики включает графические команды и функции для построения столбцовых диаграмм, гистограмм, средств отображения векторов и комплексных элементов, вывода дискретных последовательностей данных, а также движущихся траекторий как для двумерной, так и для трехмерной графики. Этот раздел получил свое дальнейшее развитие в версии системы MATLAB 5.0, где специальные графические средства улучшены и существенно расширены.

- BAR – столбцовые диаграммы
- ERRORBAR – график с указанием интервала погрешности
- HIST – построение гистограммы
- STEM – дискретные графики
- STAIRS – ступенчатый график
- ROSE – гистограмма в полярных координатах
- COMPASS, FEATHER – графики векторов
- QUIVER – поле градиентов функции
- COMET – движение точки по траектории

- FILL – закраска многоугольника
- COMET3 – движение точки по пространственной траектории
- SLICE – сечения функции от трех переменных
- WATERFALL – трехмерная поверхность
- FILL3 – закраска многоугольника в трехмерном пространстве
- VIEWMTX – вычисление матрицы управления углом просмотра
- VIEW – управление положением точки просмотра

1.3. График в линейном масштабе

PLOT

Синтаксис:

```
plot(y)
plot(x, y)
plot(x, y, s)
plot(x1, y1, s1, x2, y2, s2, ...)
```

Описание:

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.

Команда plot(x, y) соответствует построению обычной функции, когда одномерный массив x соответствует значениям аргумента, а одномерный массив y – значениям функции. Когда один из массивов X или Y либо оба двумерные, реализуются следующие построения:

если массив Y двумерный, а массив x одномерный, то строятся графики для столбцов массива Y в зависимости от элементов вектора x;

если двумерным является массив X, а массив y одномерный, то строятся графики столбцов массива X в зависимости от элементов вектора y;

если оба массива X и Y двумерные, то строятся зависимости столбцов массива Y от столбцов массива X.

Команда plot(x, y, s) позволяет выделить график функции, указав способ отображения линии, способ отображения точек, цвет линий и точек с помощью строковой переменной s, которая может включать до трех символов из следующей таблицы:

Тип линии	Тип точки	Цвет
Непрерывная	Точка	Желтый
Штриховая	Плюс	Фиолетовый
Двойной пунктир	Звездочка	Голубой
Штрих-пунктирная	Кружок	Красный
	Крестик	Зеленый
		Синий
		Белый
		Черный

Если цвет линии не указан, он выбирается по умолчанию из шести первых цветов, с желтого до синего, повторяясь циклически.

Команда `plot(x1, y1, s1, x2, y2, s2, ...)` позволяет объединить на одном графике несколько функций $y_1(x_1)$, $y_2(x_2)$, ..., определив для каждой из них свой способ отображения.

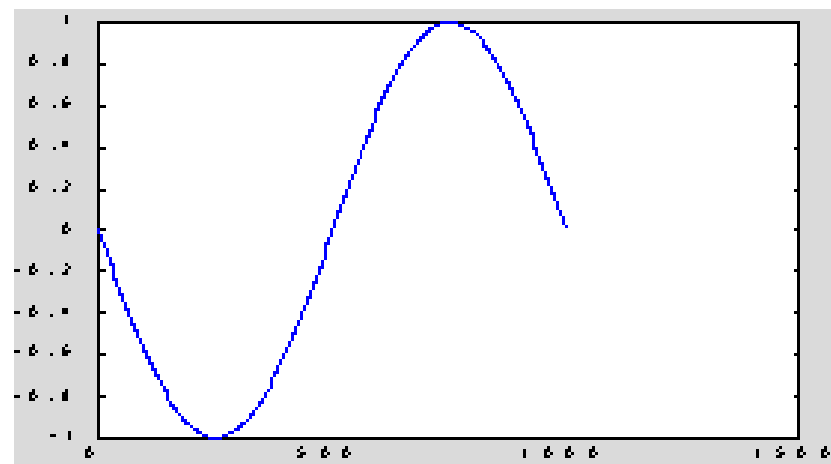
Обращение к командам `plot` вида `plot(x, y, s1, x, y, s2)` позволяет для графика $y(x)$ определить дополнительные свойства, для указания которых применения одной строковой переменной `s1` недостаточно, например при задании разных цветов для линии и для точек на ней.

Примеры:

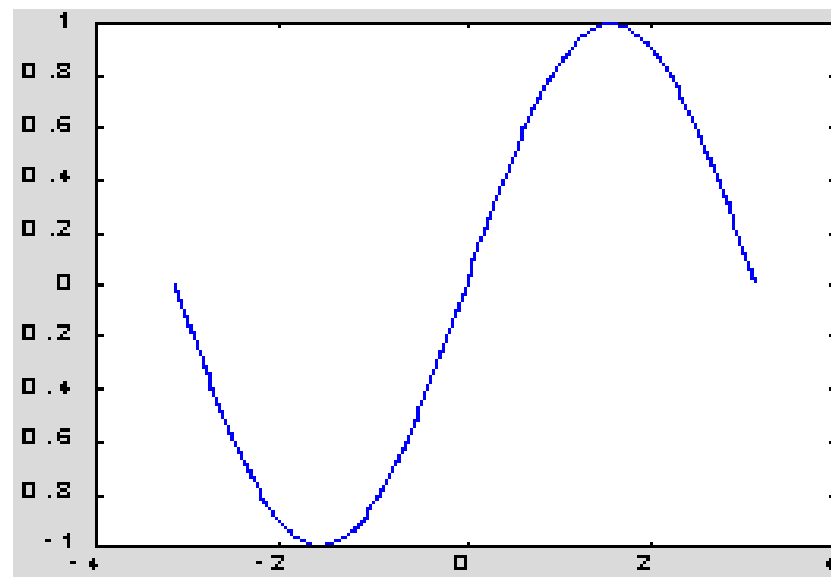
Построим график функции $y = \sin(x)$ на отрезке $[-\pi, \pi]$ с шагом $\pi/500$:

```
x = -pi:pi/500:pi;
y=sin(x);
plot(y)% рис. а
plot(x, y) % рис. б
```

График на рис. а отображает значения одномерного массива y , состоящего из 1001 элемента, как функцию от номера элемента; график на рис. б отображает значения того же массива как функцию элементов массива x .



а)



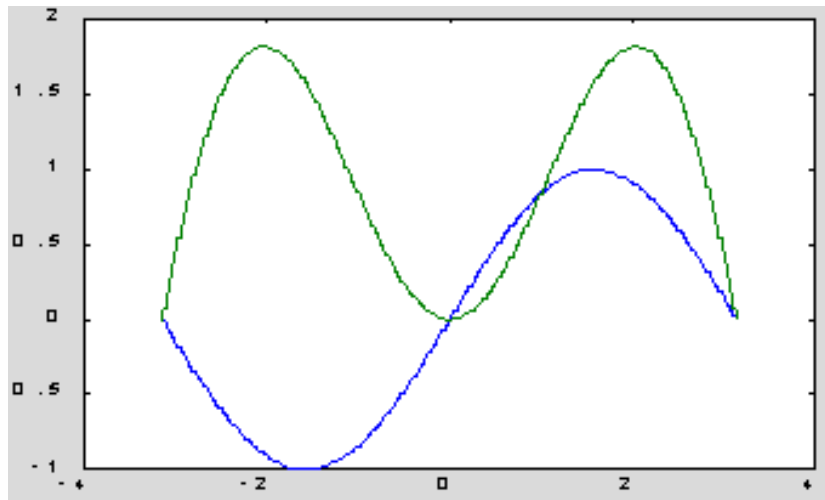
б)

Рассмотрим различные способы применения функции `plot(x, y)` на примере графиков двух функций $y_1 = \sin(x)$ и $y_2 = x \sin(x)$:

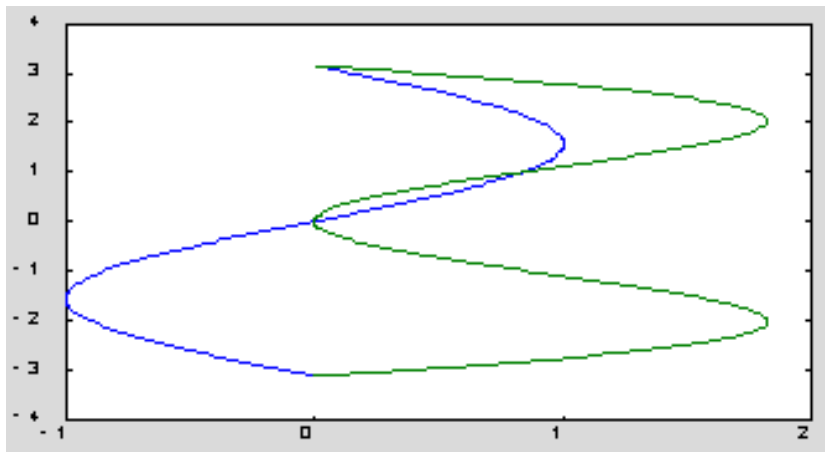
```

x1 = -pi:pi/500:pi;
y1 = sin(x1);
y2 = x1.*sin(x1);
plot(x1', [y1'y2'])% рис. в
plot( [y1' y2'], x1') % рис. г

```



в)

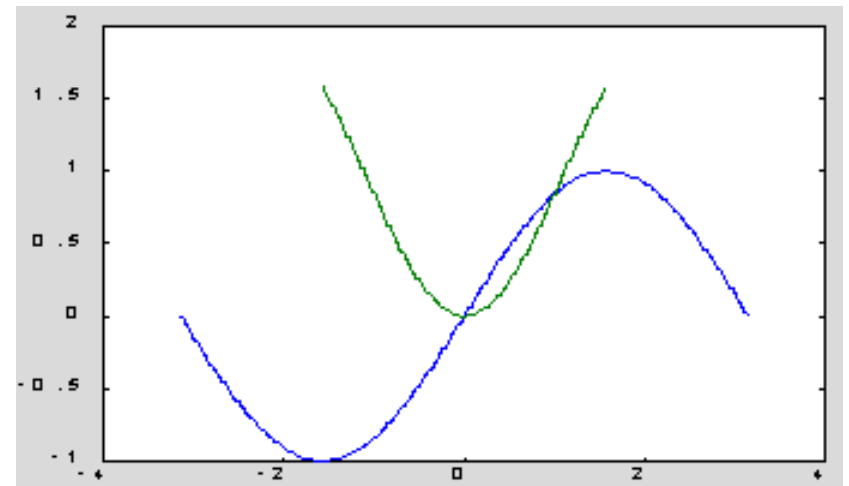


г)

```

x2 = x1/2;
y2 = x2.*sin(x2);
plot([x1' x2'], [y1' y2']) % рис. д

```



д)

Сопутствующие функции и команды: LOGLOG, SEMILOGX, SEMILOGY, POLAR.

1.4. График в логарифмическом масштабе

L GLOG

Синтаксис:

```

loglog(x,y)
loglog(x, y,s)
loglog(x1, y1, s1, x2, y2, s2, ...)

```

Описание:

Команды loglog(...) равносильны функциям plot, за исключением того, что они используют по обеим осям логарифмический масштаб вместо линейного.

Примеры:

Построим график $y = \exp(x)$ в логарифмическом масштабе:

```
x = logspace(-1,2);
```

```
loglog(x, exp(x))
```

```
grid
```

Сопутствующие функции и команды: PLOT, SEMILOGX, SEMILOGY.

1.5. График в полулогарифмическом масштабе

SEMILOGX,
SEMILOGY

Синтаксис:

```
semilogx(x, y)
```

```
semilogy(x, y)
```

```
semilogx(x, y, s)
```

```
semilogy(x, y, s)
```

```
semilogx(x1, y1, s1, x2, y2, s2, ...)
```

```
semilogy(x1, y1, s1, x2, y2, s2, ...)
```

Описание:

Команды semilogx(...) используют логарифмический масштаб по оси x и линейный масштаб по оси y.

Команды semilogy(...) используют логарифмический масштаб по оси y и линейный масштаб по оси x.

Примеры:

Построим график $y = \exp(x)$ в полулогарифмическом масштабе по оси y:

```
x = 0:0.1:100;
```

```
semilogy(x, exp(x))
```

```
grid
```

1.6. График в полярных координатах

POLAR

Синтаксис:

```
polar(phi, rho)
```

```
polar(phi, rho, s)
```

Описание:

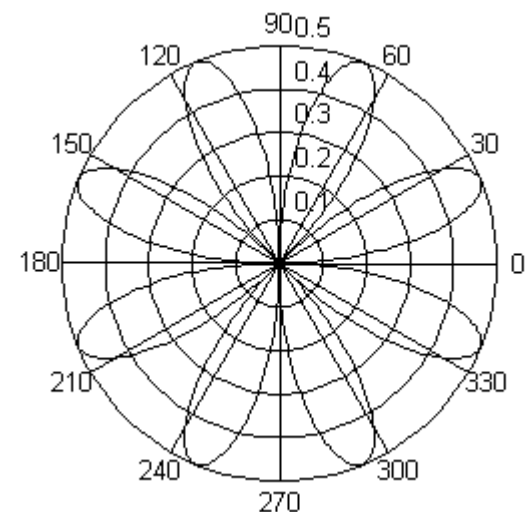
Команды polar(...) реализуют построение графиков в полярных координатах, задаваемых углом phi и радиусом rho.

Примеры:

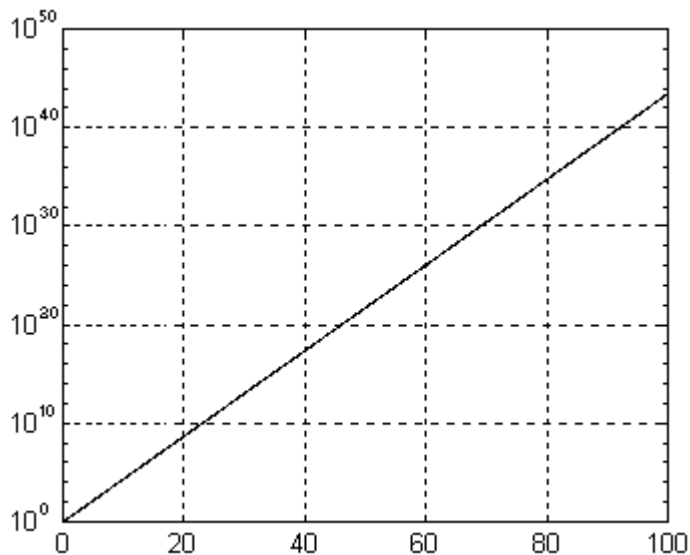
Построим график функции $\rho = \sin(2 * \phi) * \cos(2 * \phi)$ в полярных координатах

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

```
polar(phi, sin(2 * phi) * cos(2 * phi))
```



Сопутствующие функции и команды: PLOT, LOGLOG.



Сопутствующие функции и команды: PLOT, LOGLOG.

1.7. Построение линий и точек в трехмерном пространстве

PLOT3

Синтаксис:

```
plot3(x, y, z)
plot3(X, Y, Z)
plot3(x, y, z, s)
plot3(x1, y1, z1, s1, x2, y2, z2, s2, ...)
```

Описание:

Команды plot3(...) являются трехмерными аналогами функции plot(...).

Команда plot3(x, y, z), где x, y, z – одномерные массивы одинакового размера, строит точки с координатами x(i), y(i), z(i) и соединяет их прямыми линиями.

Команда plot3(X, Y, Z), где X, Y, Z – двумерные массивы одинакового размера, строит точки с координатами x(i, :), y(i, :), z(i, :) для каждого столбца и соединяет их прямыми линиями.

Команда plot3(x, y, z, s) позволяет выделить график функции z(x, y), указав способ отображения линии, способ отображения точек, цвет линий и точек с помощью строковой переменной s, которая может включать до трех символов из следующей таблицы.

Тип линии	Тип точки	Цвет
Непрерывная —	Точка .	Желтый y
Штриховая --	Плюс +	Фиолетовый m
Двойной пунктир :	Звездочка *	Голубой c
Штрих-пунктирная -. .	Кружок o	Красный r
	Крестик x	Зеленый g
		Синий b
		Белый w
		Черный k

Если цвет линии не указан, он выбирается по умолчанию из шести первых цветов, с желтого до синего, повторяясь циклически.

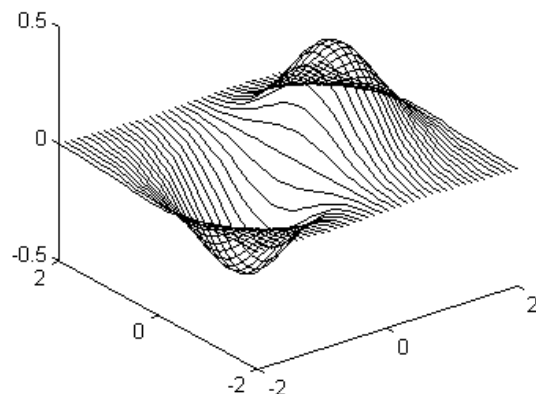
Команда plot3(x1, y1, z1, s1, x2, y2, z2, s2, ...) позволяет объединить на одном графике несколько функций z1(x1, y1), z2(x2, y2), ..., определив для каждой из них свой способ отображения.

Обращение к команде plot3 вида plot3(x, y, z, s1, x, y, z, s2) позволяет для графика z(x, y) определить дополнительные свойства, для указания которых применения одной строковой переменной s1 недостаточно, например, при задании разных цветов для линии и для точек на ней.

Примеры:

Построим график функции $z = x * \exp(-x^2 - y^2)$ в трехмерном пространстве.

```
[X, Y] = meshgrid([-2 : 0.1 : 2]);
Z = X .* exp(- X.^ 2 - Y.^ 2);
plot3(X, Y, Z)
```



Сопутствующие функции и команды: PLOT, AXIS, VIEW, MESH, SURF.

2. ФОРМИРОВАНИЕ ДВУМЕРНЫХ МАССИВОВ

2.1. Формирование двумерных массивов X и Y

MESHGRID

Синтаксис:

`[X, Y] = meshgrid(x, y)`

`[X, Y] = meshgrid(x)`

Описание:

Функция `[X, Y] = meshgrid(x, y)` задает сетку на плоскости x – y в виде двумерных массивов X , Y , которые определяются одномерными массивами x и y . Строки массива X являются копиями вектора x , а столбцы – копиями вектора y . Формирование таких массивов упрощает вычисление функций двух переменных, позволяя применять операции над массивами.

Функция `[X, Y] = meshgrid(x)` представляет собой упрощенную форму записи для функции `[X, Y] = meshgrid(x, x)`.

Примеры:

Определим двумерные массивы и вычислим функцию, $z = x e^{(-x^2 - y^2)}$, заданную на квадрате $-2 \leq x \leq 2$, $-2 \leq y \leq 2$.

`[X, Y] = meshgrid(-2 : 0.2 : 2);`

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

Сопутствующие функции и команды: MESH, SURF.

2.2. Трехмерная сетчатая поверхность

MESH, MESH C, MESH Z

Синтаксис:

`mesh(X, Y, Z, C)` `meshc(X, Y, Z, C)` `meshz(X, Y, Z, C)`

`mesh(x, y, Z, C)` `meshc(x, y, Z, C)` `meshz(x, y, Z, C)`

`mesh(Z, C)` `meshc(Z, C)` `meshz(Z, C)`

`mesh(X, Y, Z)` `meshc(X, Y, Z)` `meshz(X, Y, Z)`

`mesh(x, y, Z)` `meshc(x, y, Z)` `meshz(x, y, Z)`

`mesh(Z)` `meshc(Z)` `meshz(Z)`

Описание:

Команда `mesh(X, Y, Z, C)` выводит на экран сетчатую поверхность для значений массива Z , определенных на множестве значений массивов X и Y . Цвета узлов поверхности задаются массивом C . Цвета ребер определяются свойством `EdgeColor` объекта `surface`. Можно задать одинаковый цвет для всех ребер, определив его в виде вектора `[r g b]` интенсивности трех цветов – красного, зеленого, синего. Если определить спецификацию `none`, то ребра не будут прорисовываться. Если определить спецификацию `flat`, то цвет ребер ячейки определяется цветом того узла, который был первым при обходе этой ячейки. Поскольку одни и те же ребра обходятся несколько раз, то цвета будут замещаться. Если определить спецификацию `interp`, то будет реализована линейная интерполяция цвета между вершинами ребра.

Применение функции `shading` после обращения к функции `mesh` изменяет спецификации свойств `EdgeColor` и `FaceColor` согласно следующей таблице.

Свойство	Применяемая функция		
	mesh	shading flat	shading interp
EdgeColor	flat	flat	interp
FaceColor	Цвет фона	Цвет фона	Цвет фона

Команда `mesh(x, y, Z, C)` выполняет ту же функцию, но вместо двумерных массивов `X`, `Y` использует их одномерные проекции, так что, если `length(x) = n`, а `length(y) = m`, то `[m, n] = size(Z)`. В этом случае узлы сетчатой поверхности определяются тройками $\{x(j), y(i), Z(i, j)\}$, где вектор `x` определяет столбцы массива `Z`, а `y` – строки.

Команда `mesh(Z, C)` использует сетку, которая определяется одномерными массивами `x = 1 : n` и `y = 1 : m`.

Команды `mesh(X, Y, Z)`, `mesh(x, y, Z)`, `mesh(Z)` используют в качестве массива цвета `C = Z`, то есть цвет в этом случае пропорционален высоте поверхности.

Группа команд `meshc(...)` в дополнение к трехмерным поверхностям строит проекцию линий постоянного уровня.

Группа команд `meshz(...)` в дополнение к трехмерным поверхностям строит плоскость отсчета на нулевом уровне, закрывая поверхность, лежащую ниже этого уровня.

Функция `h = mesh(...)` возвращает дескриптор `h` для графического объекта `surface`.

Примеры:

Построим трехмерную поверхность функции $z = x * \exp(-x^2 - y^2)$ с проекциями линий постоянного уровня.

```
[ X, Y ] = meshgrid([ -2 : 0.1 : 2 ]);
Z = X . * exp(- X .^ 2 - Y .^ 2);
meshc(X, Y, Z)
```

Теперь построим эту же функцию с пьедесталом отсчета

```
meshz(X, Y, Z)
```

Сопутствующие функции и команды: SURF, WATERFALL

2.3. Затененная сетчатая поверхность

URF,
SURFC

Синтаксис:

`surf(X, Y, Z, C)` `surfc(X, Y, Z, C)`

`surf(x, y, Z, C)` `surfc(x, y, Z, C)`

`surf(Z, C)` `surfc(Z, C)`

`surf(X, Y, Z)` `surfc(X, Y, Z)`

`surf(x, y, Z)` `surfc(x, y, Z)`

`surf(Z)` `surfc(Z)`

Описание:

Команда `surf(X, Y, Z, C)` выводит на экран сетчатую поверхность для значений массива `Z`, определенных на множестве значений массивов `X` и `Y`. Цвет ячейки определяется массивом `C`. Цвет ребер – черный, определяется свойством `EdgeColor`, специфицированным как `[0 0 0]`. Можно задать одинаковый цвет для всех ребер, определив его в виде вектора `[r g b]` интенсивности трех цветов – красного, зеленого, синего. Если определить спецификацию `none`, то ребра не будут прорисовываться.

Применение функции `shading` после обращения к функции `surf` изменяет спецификации свойств `EdgeColor` и `FaceColor` графического объекта `surface` согласно следующей таблице.

Свойство	Применяемая функция		
	surf	shading flat	shading interp
EdgeColor	[0 0 0]	none	none
FaceColor	flat	flat	interp

Команда `surf(x, y, Z, C)` выполняет ту же функцию, но вместо двумерных массивов `X`, `Y` использует их одномерные проекции, так что, если `length(x) = n`, а `length(y) = m`, то `[m, n] = size(Z)`. В этом случае узлы сетчатой поверхности определяются тройками $\{x(j), y(i), Z(i, j)\}$, где вектор `x` определяет столбцы массива `Z`, а `y` – строки.

Команда `surf(Z, C)` использует сетку, которая определяется одномерными массивами `x = 1 : n` и `y = 1 : m`.

Команды `surf(X, Y, Z)`, `surf(x, y, Z)`, `surf(Z)` используют в качестве массива цвета $C = Z$, то есть цвет в этом случае пропорционален высоте поверхности.

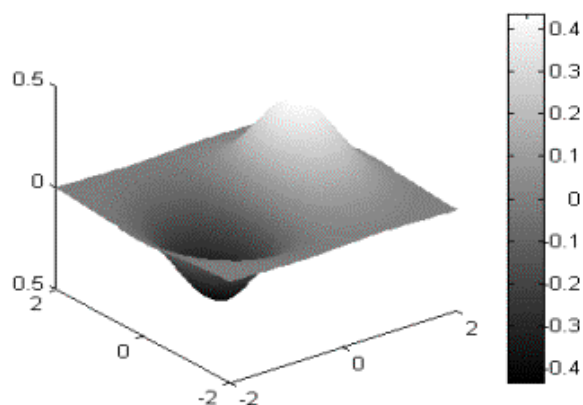
Группа команд `surfc(...)` в дополнение к трехмерным затененным поверхностям строит проекцию линий постоянного уровня.

Функция `h = surf(...)` возвращает дескриптор `h` для графического объекта `surface`.

Примеры:

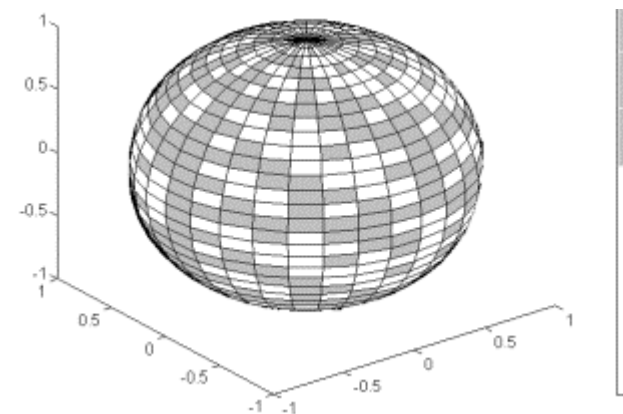
Построим трехмерную затененную поверхность функции $z = x \cdot \exp(-x^2 - y^2)$ со шкалой затененности.

```
[X, Y] = meshgrid([-2 : 0.1 : 2]);
Z = X .* exp(-X.^2 - Y.^2);
surf(X, Y, Z)
colormap(gray)
shading interp
colorbar
```



Рассмотрим пример сферы, которая раскрашена в соответствии с матрицей Адамара (Hadamard), часто используемой в теории кодирования сигналов и составленной только из двух чисел 1 и -1.

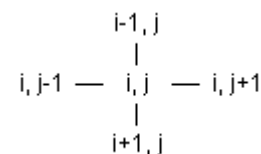
```
k = 5; n = 2 ^ k - 1;
[X, Y, Z] = sphere(n);
C = hadamard(2 ^ k);
surf(X, Y, Z, C);
colormap([1 1 1; 2/3 2/3 2/3])
colorbar
```



Алгоритм:

В общем виде, для задания поверхности можно использовать два независимых параметра i и j , которые изменяются непрерывно, например в прямоугольнике $1 \leq i \leq m$, $1 \leq j \leq n$; тогда поверхность будет определяться тремя функциями $x(i, j)$, $y(i, j)$, $z(i, j)$. Когда i и j целые числа, они задают прямоугольную сетку с целочисленными значениями для узлов. Функции $x(i, j)$, $y(i, j)$, $z(i, j)$ становятся двумерными массивами X , Y , Z размера $m \times n$. Четвертая функция – цвет ячеек поверхности $c(i, j)$ задает четвертую матрицу C .

Каждая точка сетчатой поверхности имеет в общем случае четырех соседей, как показано на следующей схеме.



Такая прямоугольная сетка приводит к разбиению поверхности на ячейки, ограниченные четырьмя ребрами. Каждый внутренний узел поверхности имеет четырех соседей, узел на границе – трех, узел в углу поверхности – двух.

Цвет такой сетчатой поверхности может быть задан двумя способами: либо цветом одной из вершин, либо цветом в центре ячейки.

Рассмотрим, как в этом случае действует функция закрашки `shading`. Если задано значение `shading interp`, то цвет ячейки определяется

как билинейная функция местных координат. Если задано значение shading faceted (принято по умолчанию) или shading flat, то цвет ячейки постоянен и определяется цветом верхней левой вершины, как показано на следующей схеме:



Сопутствующие функции и команды: MESH, MESH C, SURFL, WATERFALL.

2.4. Затененная поверхность с подсветкой

SURFL

Синтаксис:

surfl(X, Y, Z, s)	surfl(Z, s)
surfl(X, Y, Z, s, k)	surfl(Z, s, k)
surfl(X, Y, Z)	surfl(Z)

Описание:

Команда surfl(X, Y, Z, s) выводит на экран затененную поверхность с подсветкой для значений массива Z, определенных на множестве значений массивов X и Y. Направление на источник света может быть задано с помощью вектора $s = [S_x, S_y, S_z]$ в декартовых координатах или вектора $s = [az, elev]$ в сферических координатах. По умолчанию азимут $az = -37.5^\circ$, возвышение $elev = 30^\circ$. Подсветка учитывает модели рассеяния, отражения и зеркального эффекта освещения поверхности.

Команда surfl(X, Y, Z, s, k) позволяет управлять параметрами рассеяния, отражения и зеркального эффекта, используя вектор $k = [ka, kd, ks, spread]$, который учитывает эффекты отраженного света ka, диффузного отражения kd, зеркального отражения ks и зеркального распространения spread. По умолчанию вектор k имеет значения [0.55 0.6 0.4 10].

Команда surfl(X, Y, Z) использует значения параметров по умолчанию.

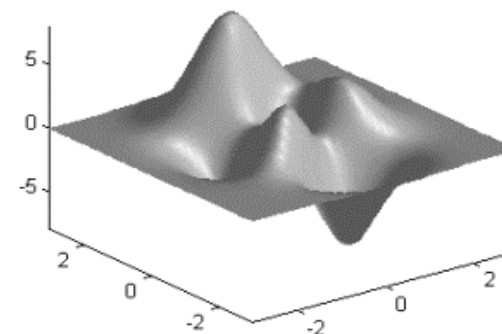
Команды surfl(Z, ...) строят графики, не учитывая истинных значений массивов X и Y.

Из-за того, что алгоритм surfl вычисляет нормали к поверхности, необходимо, чтобы входные матрицы имели размер по крайней мере 3 x 3.

Примеры:

Построим изображение функции peaks, используя подсветку.

```
[X, Y] = meshgrid(-3 : 1/8 : 3);
Z = peaks(X, Y);
surfl(X, Y, Z)
shading interp
colormap(gray)
```



Сопутствующие функции и команды: SHADING.

Задание осей координат

2.5. Масштабирование осей и вывод на экран

AXIS

Синтаксис:

```
axis([xmin xmax ymin ymax])
axis([xmin xmax ymin ymax zmin zmax])
axis('auto')
axis(axis)
v = axis
axis('ij')
axis('xy')
```

```
axis('square')
axis('equal')
axis('off')
axis('on')
[s1, s2, s3] = axis('state')
axis(s1, s2, s3)
```

Описание:

Команда `axis` обеспечивает преемственность предшествующих версий системы MATLAB, ориентированных на символьную обработку, с версиями 4.x, ориентированными на графический интерфейс.

Команда `axis([xmin xmax ymin ymax])` устанавливает масштаб по осям *x*, *y* для активного графического окна.

Команда `axis([xmin xmax ymin ymax zmin zmax])` устанавливает масштаб по осям *x*, *y*, *z* для активного графического окна.

Команда `axis('auto')` возвращает масштаб по осям к штатным значениям (принятым по умолчанию).

Команда `axis(axis)` фиксирует текущие значения масштабов для последующих графиков, как если бы был включен режим `hold`.

Функция `v = axis` возвращает вектор-строку масштабов по осям для активного графика. Если график двумерный, то *v* имеет 4 компонента; если трехмерный – 6 компонентов.

Команда `axis('ij')` перемещает начало отсчета в левый верхний угол, сохраняет положение осей и реализует отсчет по вертикальной оси из верхнего левого угла (матричная система координат).

Команда `axis('xy')` возвращает декартову систему координат; начало отсчета находится в нижнем левом углу; ось *x* горизонтальна и размечается слева направо, ось *y* вертикальна и размечается снизу вверх.

Команда `axis('square')` устанавливает одинаковый диапазон изменения переменных по осям.

Команда `axis('equal')` устанавливает масштаб, который обеспечивает одинаковые расстояния между метками по осям *x* и *y*.

Команда `axis('image')` устанавливает масштаб, который обеспечивает квадратные размеры пикселей.

Команда `axis('normal')` восстанавливает полноразмерный масштаб, отменяя масштабы, установленные командами `axis('square')` и `axis('equal')`.

Команда `axis('off')` снимает с осей их обозначения и маркеры.

Команда `axis('on')` восстанавливает на осях их обозначения и маркеры.

Функция `[s1, s2, s3] = axis('state')` возвращает строку, определяющую вектор состояния объекта `axes`:

```
s1 = 'auto' | 'manual'.
```

```
s2 = 'on' | 'off'.
```

```
s3 = 'xy' | 'ij'.
```

Команда `axis(s1, s2, s3)` устанавливает параметры объекта `axes` в соответствии с вектором состояния `[s1, s2, s3]`. По умолчанию этот вектор принимает значения `['auto', 'on', 'xy']`.

Сопутствующие функции и команды: SUBPLOT.

2.6. Нанесение сетки

GRID

Синтаксис:

```
grid on
grid off
grid
```

Описание:

Команда `grid on` наносит координатную сетку на текущие оси.

Команда `grid off` удаляет координатную сетку.

Команда `grid` выполняет роль переключателя с одной функции на другую.

Команды группы `grid` выполняют установку свойств `'XGrid'`, `'YGrid'`, `'ZGrid'` объекта `axes`.

Сопутствующие функции и команды: TITLE, XLABEL, YLABEL.

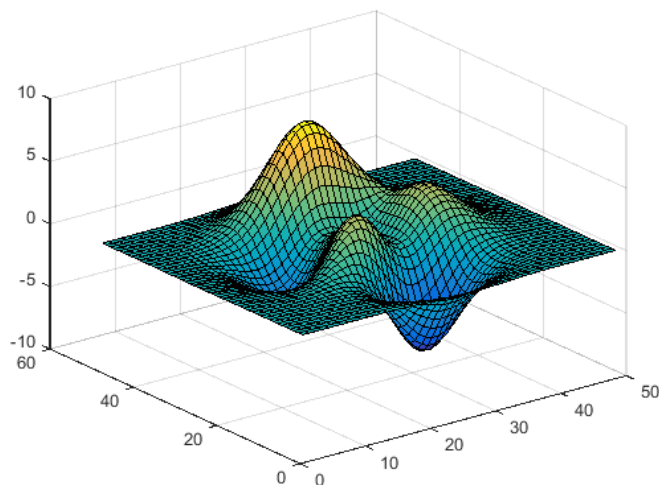
3. НОВАЯ ГРАФИЧЕСКАЯ СИСТЕМА В MATLAB

В релизе MATLAB R2014b появилась новая графическая система.

3.1. Улучшенный вид – пример 3D графики

Новая цветовая палитра (`colormap`) по умолчанию, `parula`, упрощает интерпретацию данных путем более точного их представления. Она упорядочена от темных к светлым тонам и визуально воспринимается однородно. Также у фигур теперь новый цвет фона по умолчанию, а координатная сетка теперь серого цвета, так что ваши данные отображаются более выразительно. Графическое сглаживание (`anti-aliasing`) улучшает вид линий и текста на графике.

surf(peaks)



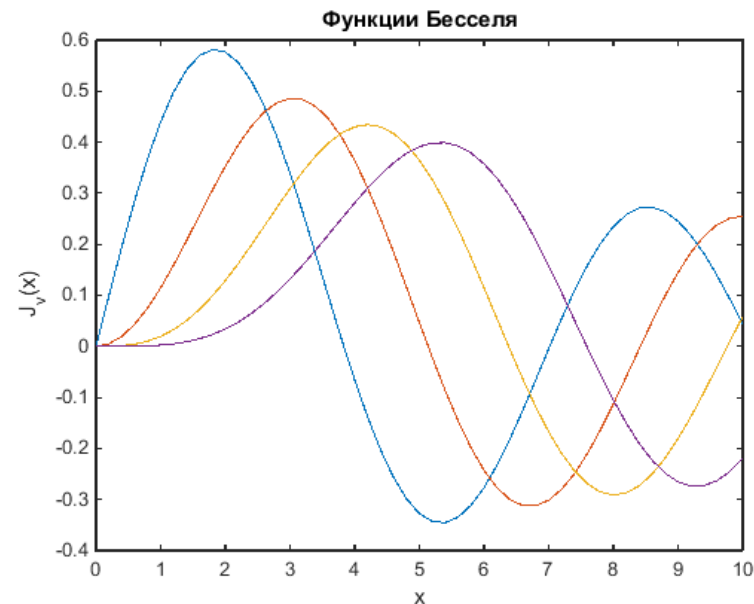
3.2. Улучшенный вид – пример 2D графики

Заголовки теперь используют полужирное начертание и по умолчанию большего размера. Линии используют новые цвета по умолчанию, которые проще отличать друг от друга, а также применяется новый порядок следования цветов. При использовании hold on порядок цветов теперь поддерживается.

load besseldata

```
plot(x,y1)
hold on
plot(x,y2,x,y3)
plot(x,y4)
hold off
```

```
xlabel('x')
ylabel('J_v(x)')
title('Функции Бесселя');
```



3.3. Графические объекты

Новая графическая система использует стандартные объекты MATLAB. Это дает большие возможности программного управления, а также легкой настройки ваших графиков.

Графические объекты теперь отображают свои наиболее распространенные свойства в командном окне.

s = surf(peaks)

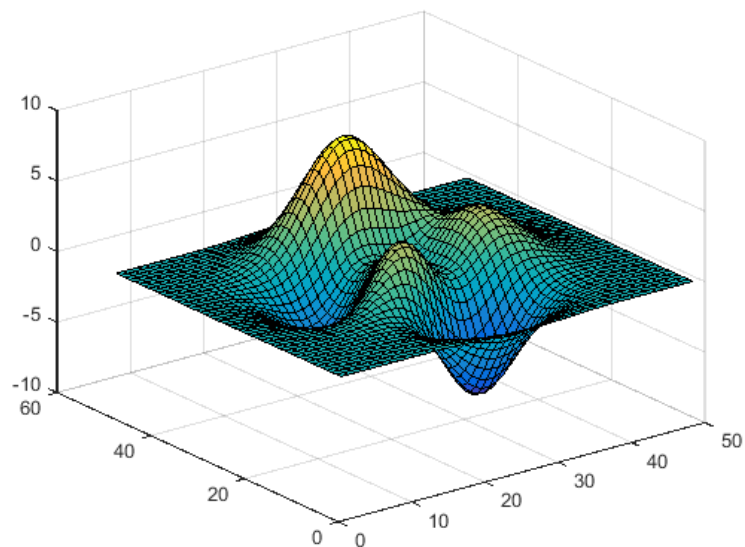
s =

Surface with properties:

```
EdgeColor: [0 0 0]
LineStyle: '-'
FaceColor: 'flat'
FaceLighting: 'flat'
FaceAlpha: 1
XDat a: [1x49 double]
YDat a: [49x1 double]
```

ZDat a: [49x49 double]
 CDat a: [49x49 double]

Use GET to show all properties

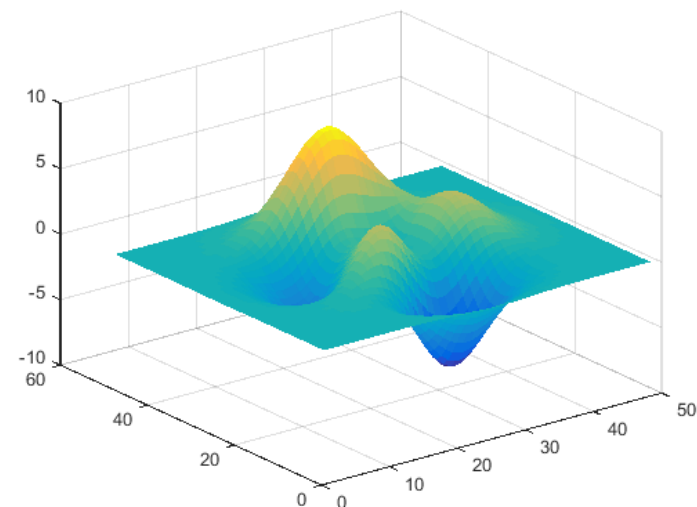


Вы можете обращаться и изменять индивидуальные графические свойства с использованием нотации `object.property` – подобно тому, как вы работаете с полями структуры.

s.LineWidth
s.EdgeColor = 'none';

ans =

0.5000



Использование методов `set` и `get` для свойств объектов все еще работает в новой графической системе. Например, вы можете получить (`get`) все свойства объекта.

get(s)

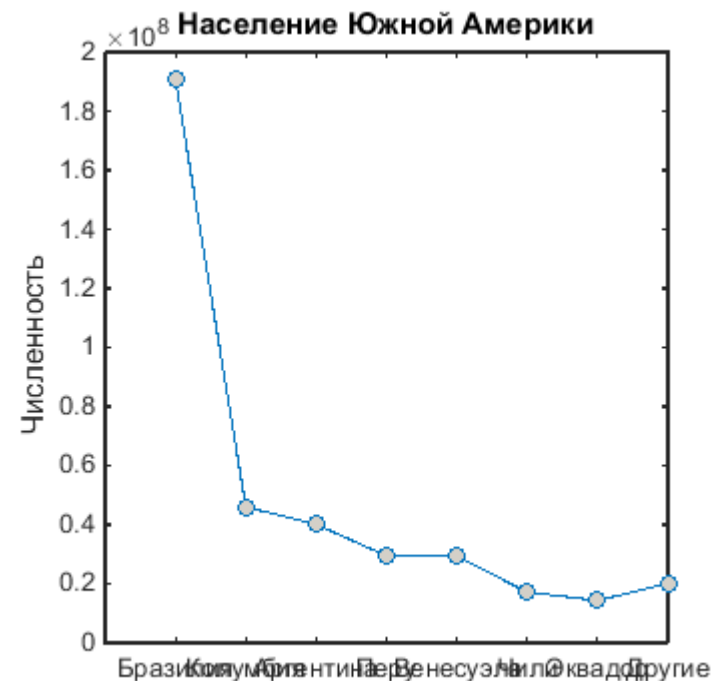
AlignVertexCenters: 'off'
 AlphaDat a: 1
 AlphaDataMapping: 'scaled'
 AmbientStrength: 0.3000
 Annotation: [1x1 matlab.graphics.eventdata.Annotation]
 BackFaceLighting: 'reverselit'
 BeingDeleted: 'off'
 BusyAction: 'queue'
 ButtonDownFcn: "
 CDat a: [49x49 double]
 CDataMapping: 'scaled'
 CDataMode: 'auto'
 CDataSource: "
 Children: []
 Clipping: 'on'
 CreateFcn: "
 DeleteFcn: "
 DiffuseStrength: 0.6000

DisplayName: "
 EdgeAlpha: 1
 EdgeColor: 'none'
 EdgeLighting: 'none'
 FaceAlpha: 1
 FaceColor: 'flat'
 FaceLighting: 'flat'
 FaceNormals: [48x48x3 double]
 FaceNormalsMode: 'auto'
 HandleVisibility: 'on'
 HitTest: 'on'
 Interruptible: 'on'
 LineStyle: '-'
 LineWidth: 0.5000
 Marker: 'none'
 MarkerEdgeColor: 'auto'
 MarkerFaceColor: 'none'
 MarkerSize: 6
 MeshStyle: 'both'
 Parent: [1x1 Axes]
 Selected: 'off'
 SelectionHighlight: 'on'
 SpecularColorReflectance: 1
 SpecularExponent: 10
 SpecularStrength: 0.9000
 Tag: "
 Type: 'surface'
 UIContextMenu: []
 UserData: []
 VertexNormals: [49x49x3 double]
 VertexNormalsMode: 'auto'
 Visible: 'on'
 XData: [1x49 double]
 XDataMode: 'auto'
 XDataSource: "
 YData: [49x1 double]
 YDataMode: 'auto'
 YDataSource: "
 ZData: [49x49 double]
 ZDataSource: "

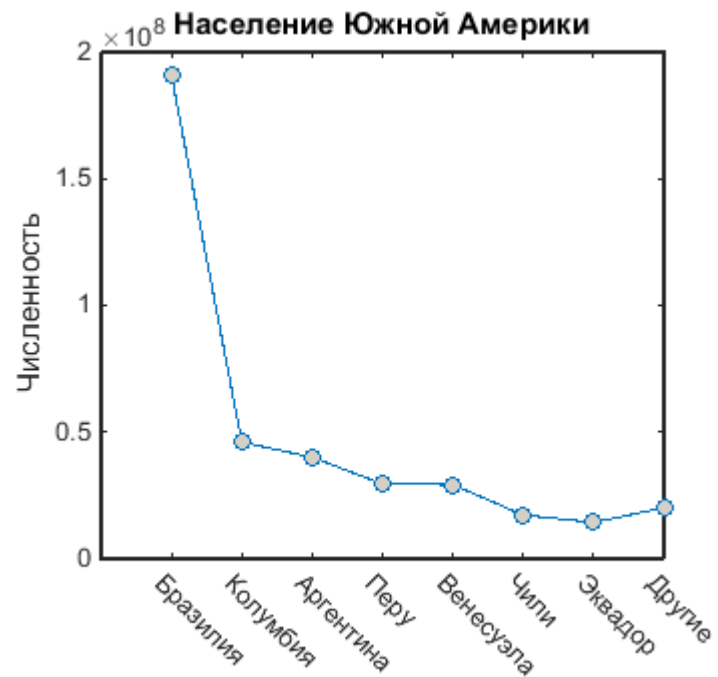
3.4. Поворачиваемые подписи осей

Дополнительно появились новые возможности в графической системе. Подписи к засечкам осей теперь можно поворачивать.

`open('PopulationPlot.fig')`



`ax = gca;`
`ax.XTickLabelRotation = -45;`



3.5. Построение графиков с датами

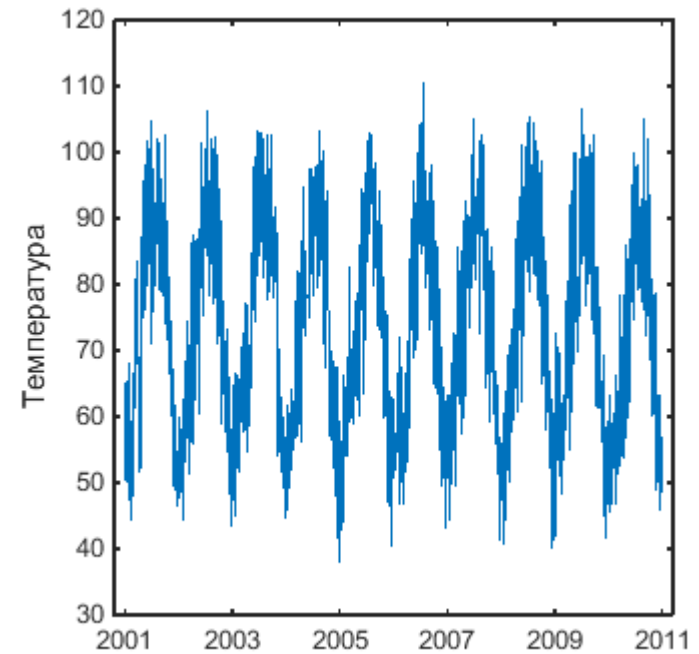
При использовании команды `plot` поддерживается новый тип данных `datetime`.

```
load tempdata
class(time)

plot(time,temperature)
ylabel('Температура')
```

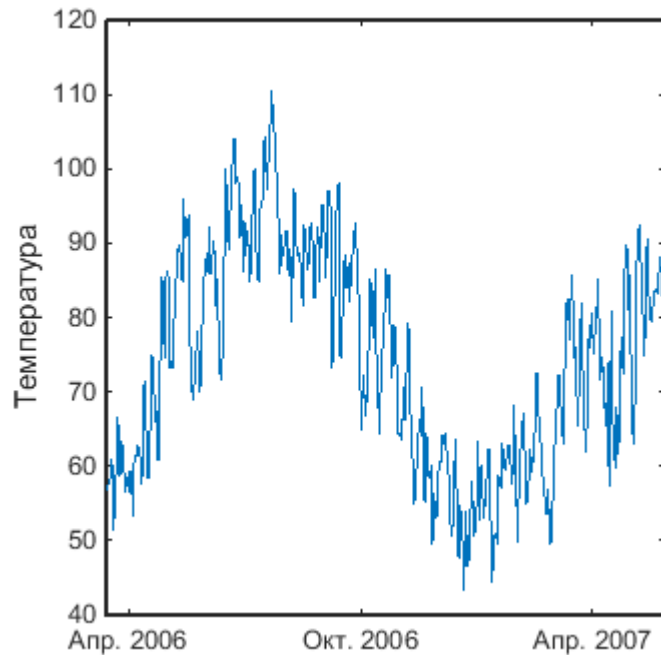
ans =

datetime



Подписи засечек осей форматируются и автоматически обновляются при изменении масштаба или перетаскивании графика.

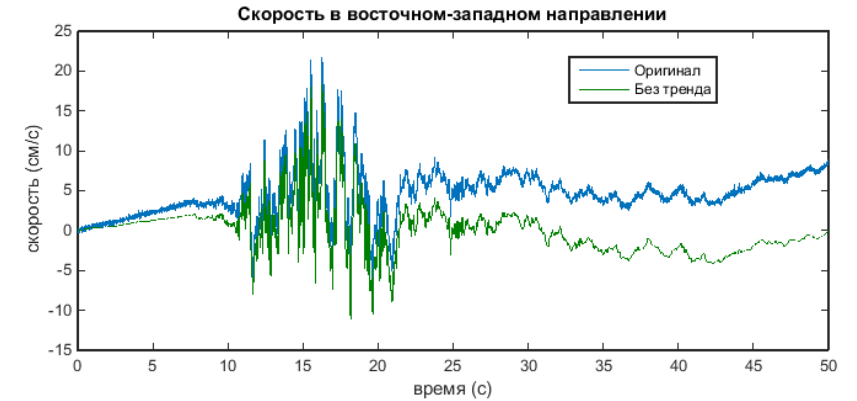
```
ax = gca;
ax.XLim = [7.3275e + 05 7.3319e + 05];
```



3.6. Поддержка многоязычного текста и символов

Теперь можно использовать символы Unicode в заголовках, подписях осей и пользовательских интерфейсах. Целочисленные значения для символов Unicode можно найти по адресу <http://unicode-table.com>.

```
load unicolabels
open('SeismicData.fig')
```



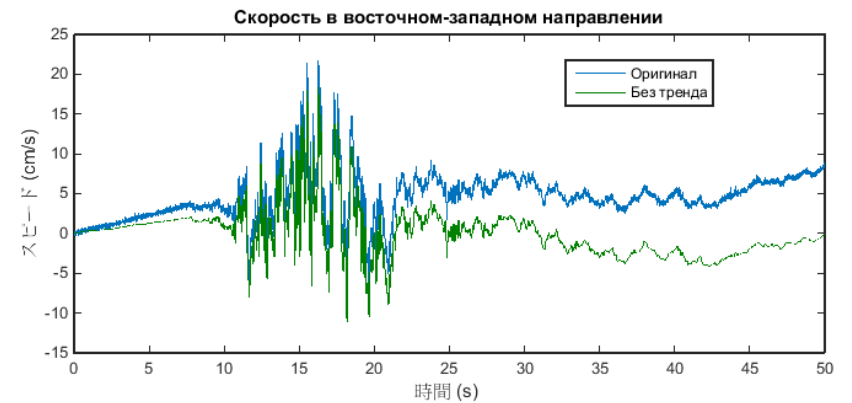
```
disp(xLabel);
```

```
時間 (s)
```

```
disp(double(xLabel(1)));
```

```
26178
```

```
ax = gca;
ax.XLabel.String = xLabel;
ax.YLabel.String = yLabel;
```



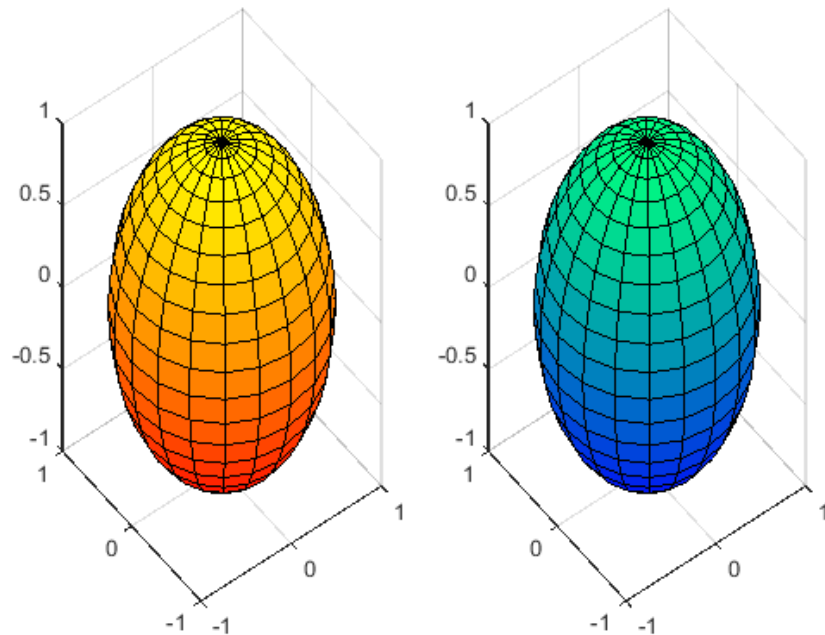
3.7. Множественные цветовые палитры

Теперь можно использовать отдельные цветовые палитры для каждой оси на фигуре.

figure

```
ax1 = subplot(1,2,1);  
sphere  
colormap(ax1, autumn)
```

```
ax2 = subplot(1,2,2);  
sphere  
colormap(ax2, winter)
```



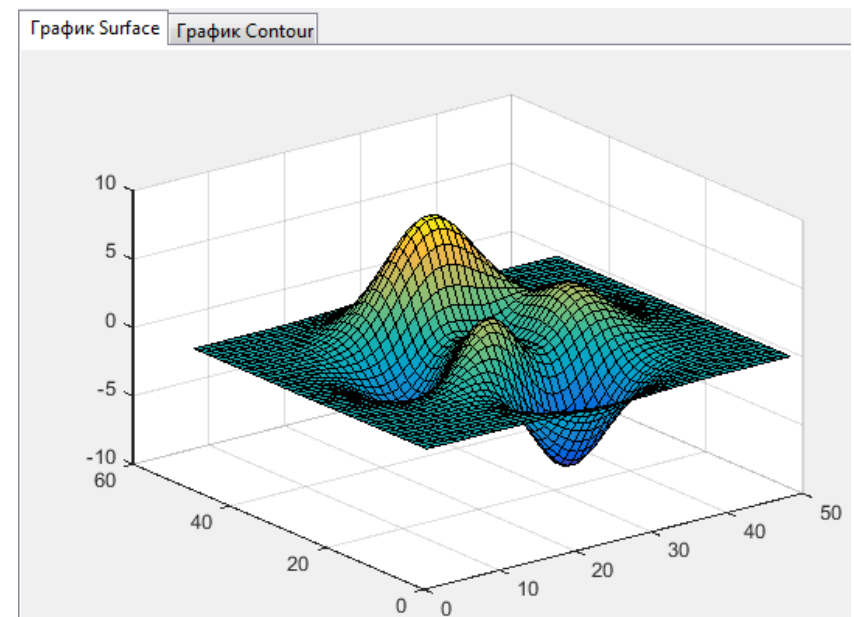
3.8. Вкладки в пользовательских интерфейсах

f = figure;

```
tabgp = uitabgroup(f);  
tab1 = uitab(tabgp,'Title','График Surface');  
tab2 = uitab(tabgp,'Title','График Contour');
```

```
ax1 = axes('Parent',tab1);  
surf(peaks,'Parent',ax1);
```

```
ax2 = axes('Parent',tab2);  
contourf(peaks,'Parent',ax2);
```



tabgp.SelectedTab

ans =

Tab (График Surface) with properties:

Title: 'График Surface'
 BackgroundColor: [0.9400 0.9400 0.9400]
 Position: [0.0036 0.0071 0.9911 0.9286]
 Units: 'normalized'

Use GET to show all properties

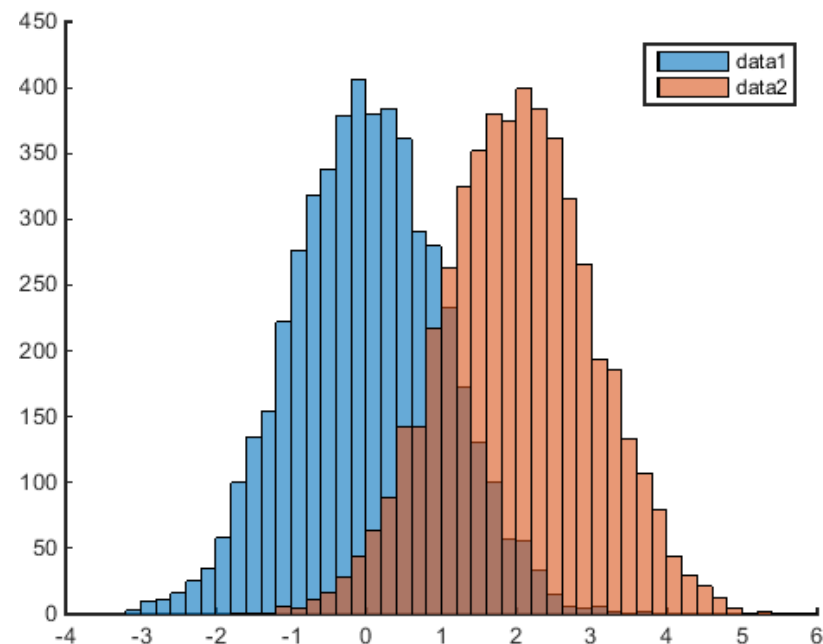
3.9. Другие новые возможности

В новой графической системе появилось много новых возможностей, включая использование символов для подписей засечек осей, добавление изображений на график, построение более сложных гистограмм, анимация линий и создание диаграмм с категориальными данными.

```
data1 = randn(5000,1);
data2 = randn(5000,1)+ 2;
```

```
figure
h1 = histogram(data1);
hold on
h2 = histogram(data2);
hold off
```

legend show

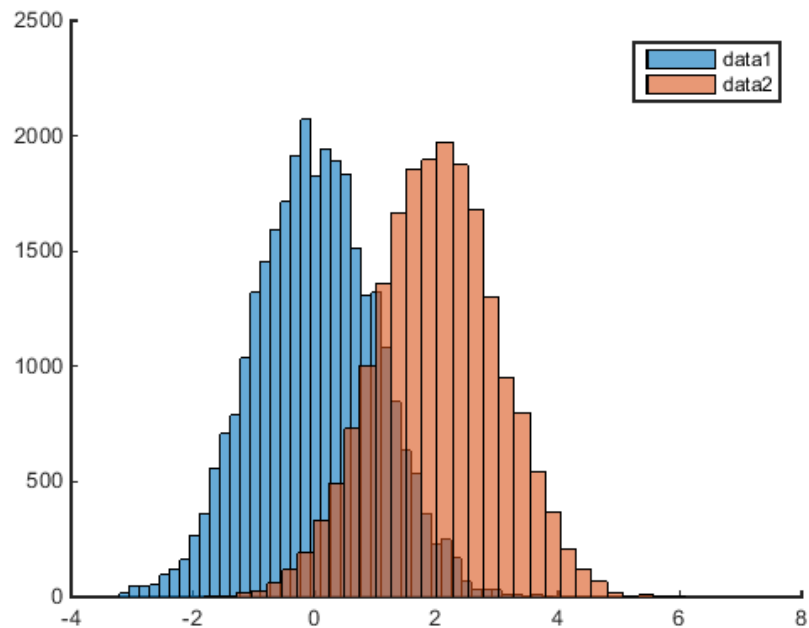


Можно легко управлять количеством прямоугольников. Можно настраивать тип нормализации. В этом примере, мы сделаем высоту каждого прямоугольника равным отношению числа измерений в прямоугольнике к ширине прямоугольника.

```
morebins(h1); %увеличить на 10%
morebins(h1); %увеличить на 10%
```

```
fewerbins(h2); %уменьшить на 10%
fewerbins(h2); %уменьшить на 10%
```

```
h1.Normalization = 'countdensity';
h2.Normalization = 'countdensity';
```



ЛИТЕРАТУРА

1. Кондрашов В., Королев С. Matlab как система программирования научно-технических расчетов. М.: Мир, Институт стратегической стабильности Минатома РФ, 2002.
2. Дорф Р., Бишоп Р. Современные системы управления: Перевод с английского. Лаборатория базовых знаний, 2002.
3. Дьяконов В., Абраменкова И. MATLAB. Обработка сигналов и изображений: специальный справочник. Питер, 2002.
4. Дьяконов В. MATLAB: учебный курс. Питер, 2000.
5. Потемкин В. Система MATLAB: справочное пособие. М.: Диалог-МИФИ, 1997.
6. Поршнев С. В. MATLAB 7. Основы работы и программирования: учебник. М.: Бином. Лаборатория знаний, 2006. 320 с.
7. <http://unicode-table.com>.

Составитель
Галина Станиславовна Воронова

ГРАФИЧЕСКИЕ ВОЗМОЖНОСТИ МАТЛАВ
Учебно-методическое пособие по направлению 200100.62

Редактор *А. И. Шевченко*
Компьютерная верстка – *Ю. Ф. Атаманов*

Подписано в печать 23.05.16.
Формат 60х84¹/₁₆
Офсетная печать. Объем 2,75 п.л.
Тираж 100 экз. Заказ 202

Отпечатано в типографии КРСУ
720048, г. Бишкек, ул. Горького, 2