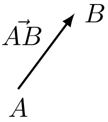
Векторная алгебра. Трехмерная графика

Храмов Д. А.

17.02.2019

Векторы, матрицы и преобразования координат

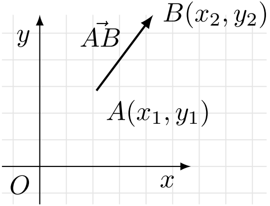
Вектор



Вектор — математический объект, который характеризуется длиной и направлением.

Вектор $\vec{AB}$ — упорядоченная пара точек A (начало вектора) и B (конец вектора).

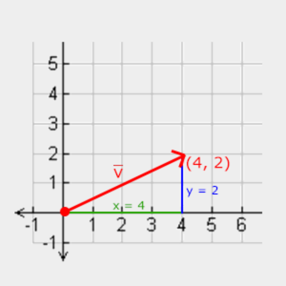
Система координат



Дает возможность ввести понятие координат вектора:
$\vec{AB} = (x_2 - x_1, y_2 - y_1)$.

Внимание! *В программировании, для краткости, "вектором" называют одномерный массив чисел. Такой массив действительно может представлять координаты вектора. Но сам вектор — это геометрический объект, который от выбора системы координат не зависит.*

Длина вектора (норма)



$$||v|| = \sqrt{x^2 + y^2}$$

```
p = [4,2];
norm(p)

ans =
      4.4721
```

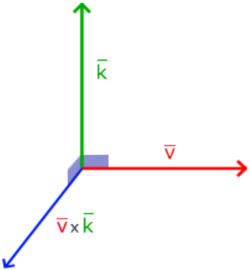
Скалярное произведение векторов (dot product)

$$\vec{v} \cdot \vec{k} = ||\vec{v}|| \cdot ||\vec{k}|| \cdot \cos\theta$$

$$\begin{bmatrix} 0.6 & -0.8 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = (0.6 \cdot 0) + (-0.8 \cdot 1) + (0 \cdot 0) = -0.8.$$

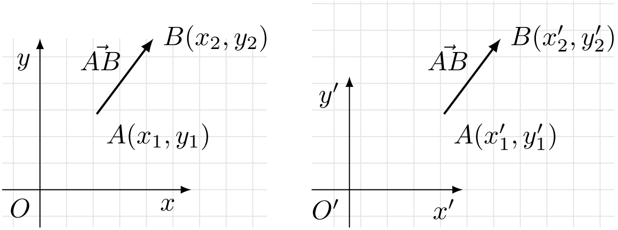
```
v = [0.6,-0.8,0]; k = [0,1,0];
dot(v,k)

ans =

   -0.8000
```

Векторное произведение векторов (cross product)




```
cross(v,k)
```

Координаты вектора в разных системах координат

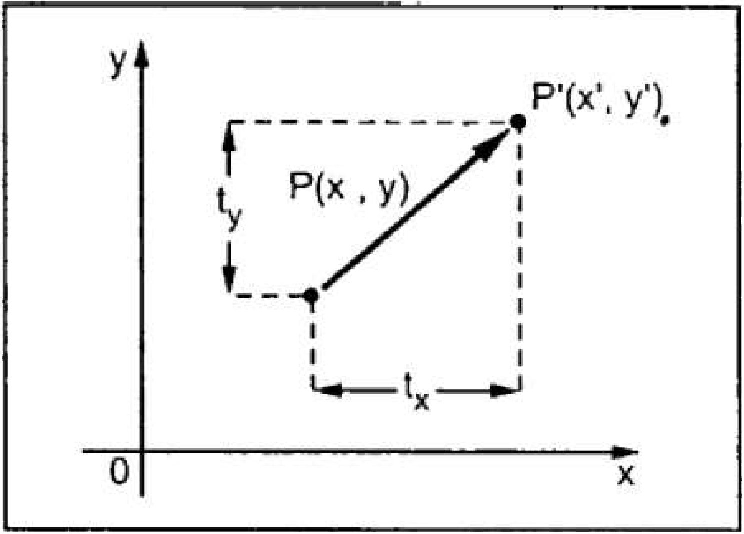


В различных системах координат один и тот же вектор имеет разные координаты.

Трактовки преобразования координат:

1. Координаты вектора остаются неизменными, а изменяется система координат.
2. Изменяются координаты вектора, а система координат остается неизменной.

Перенос (трансляция, translation)



Связь новых координат со старыми

$$\begin{aligned} x' &= x + t_x, \\ y' &= y + t_y. \end{aligned}$$

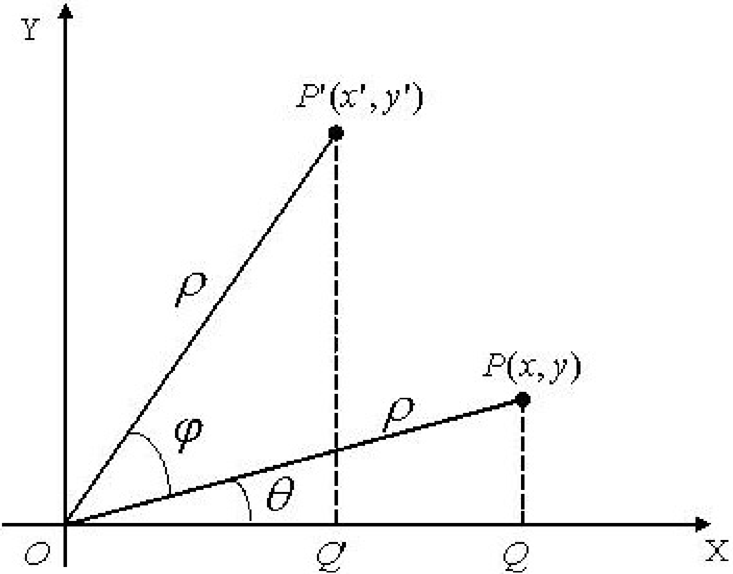
Введем матричные обозначения

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix}, \quad P = \begin{bmatrix} x \\ y \end{bmatrix}, \quad T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}.$$

В матричной записи получим

$$P' = P + T.$$

Поворот (rotation)



Формулы для пересчета старых координат точки P в новые P'

Пусть Q и Q' — проекции точек $P(x, y)$ и $P'(x', y')$ соответственно на ось X.

Тогда из прямоугольного треугольника $OP'Q'$ имеем:

$$\begin{aligned} \cos(\theta+\varphi) &= \frac{x'}{\rho}, \\ \sin(\theta+\varphi) &= \frac{y'}{\rho}. \end{aligned} \quad \Rightarrow \quad \begin{aligned} x' &= \rho\cos(\theta+\varphi), \\ y' &= \rho\sin(\theta+\varphi). \end{aligned}$$

Раскрывая суммы $\cos()$ и $\sin()$, получим

$$\begin{aligned} x' &= \rho\cos\theta\cos\varphi - \rho\sin\theta\sin\varphi, \\ y' &= \rho\sin\theta\cos\varphi + \rho\cos\theta\sin\varphi. \end{aligned} \quad \Rightarrow \quad \begin{aligned} x' &= x\cos\varphi - y\sin\varphi, \\ y' &= x\sin\varphi + y\cos\varphi, \end{aligned}$$

поскольку $x = \rho\cos\theta$, $y = \rho\sin\theta$.

Формулы преобразования координат в матричной записи

$$x' = x\cos\varphi - y\sin\varphi,$$
$$y' = x\sin\varphi + y\cos\varphi,$$

Вспомнив, что

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix}, \quad P = \begin{bmatrix} x \\ y \end{bmatrix}.$$

запишем формулы преобразования координат в матричном виде

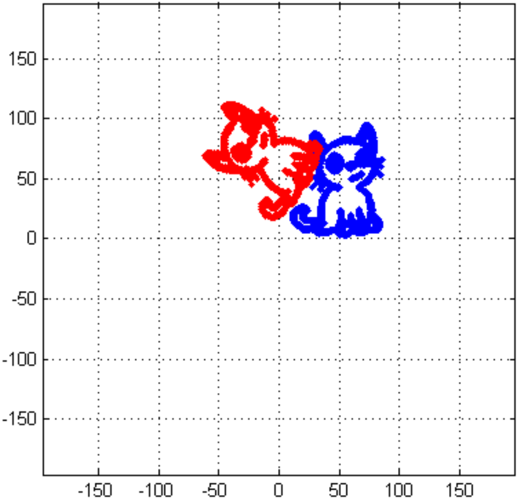
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

$$P' = RP,$$

где R – матрица поворота:

$$R = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}.$$

Пример. Преобразование координат изображения



Код примера

```
A = imread('kitten.bmp');
[N,M] = size(A);
[i,j] = find(~A);
x = j; y = M-i;
plot(x,y,'.')
v = 2*max([N,M]);
grid on
axis([-v v -v v])
axis square
hold on

R = @(phi) [cos(phi),sin(phi); -sin(phi),cos(phi)];

a =-pi/3;
B = R(a)*[x';y'];
plot(B(1,:),B(2,:),'r.')
```

Задача 1.

Для любого из файлов *.bmp в папке `pics` последовательно выполнить следующие преобразования изображения:

- поворот на угол 65°;
- масштабирование с коэффициентами $s_x = 1$, $s_y = 1.5$;
- перенос начала координат на $t_x = 20$, $t_y = 30$.

Матрица преобразования координат

Для произвольной матрицы А размерности 2 х 2 можно записать аналогично

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

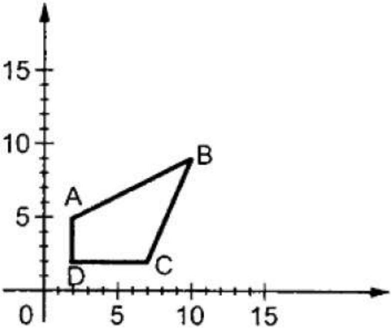
или

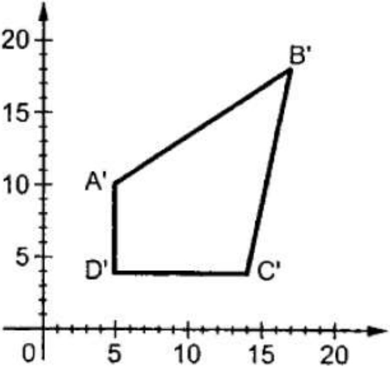
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix},$$

где A – матрица преобразования координат

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}.$$

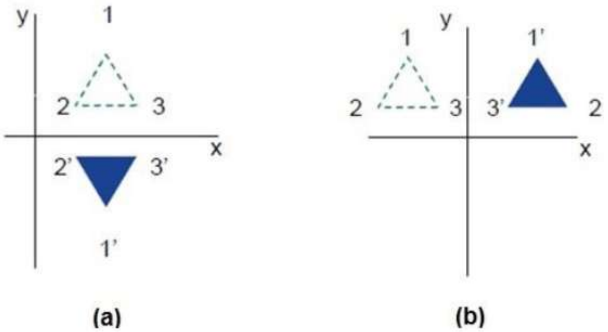
Масштабирование (scaling)





$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Отражение (reflection)



$$\text{a)} \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{b)} \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Последовательные преобразования

Пусть координаты точки $P(x, y)$ преобразуются сначала в координаты $P'(x', y')$ при помощи матрицы B, а затем в координаты $P''(x'', y'')$ при помощи матрицы A

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = B \begin{bmatrix} x \\ y \end{bmatrix}, \quad \begin{bmatrix} x'' \\ y'' \end{bmatrix} = A \begin{bmatrix} x' \\ y' \end{bmatrix}.$$

Тогда матрица C, выполняющая преобразование от $P(x, y)$ к $P''(x'', y'')$,

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = AB \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x'' \\ y'' \end{bmatrix} = C \begin{bmatrix} x \\ y \end{bmatrix},$$

запишется в виде

$$C = AB$$

или

$$C = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix} = AB.$$

Однородные координаты

Большинство операций по преобразованию координат выполняется умножением матриц, тогда как перенос – суммированием.

Для единообразия, в компьютерной графике вводят однородные координаты для плоскости

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},$$

а вместо матриц размерности 2 x 2 (для плоскости) используют матрицы размерности 3 x 3, что позволяет записать все преобразования, используя только умножение матриц. В том числе и перенос

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Некоммутативность умножения матриц

Выполним в разной последовательности масштабирование и перенос начала отсчета

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Получим

$$ST = \begin{bmatrix} S_x & 0 & S_x t_x \\ 0 & S_y & S_y t_y \\ 0 & 0 & 1 \end{bmatrix}, \quad TS = \begin{bmatrix} S_x & 0 & t_x \\ 0 & S_y & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Тем не менее, произведение некоторых видов матриц коммутативно. Например, диагональные матрицы переставлять можно. Вспомним геометрический смысл осуществляемого ими преобразования.

Обратное преобразование

Преобразование, обратное тому, что осуществляется матрицей A выполняется при помощи обратной к A матрицы A^{-1}.

Для прямой и обратной матриц справедливо

$$AA^{-1} = E,$$

где E – единичная матрица.

Трехмерная графика

Построение кривых и точек в пространстве

`plot3(x,y,z)`

`x`, `y` и `z` — векторы координат точек линии (равной длины):

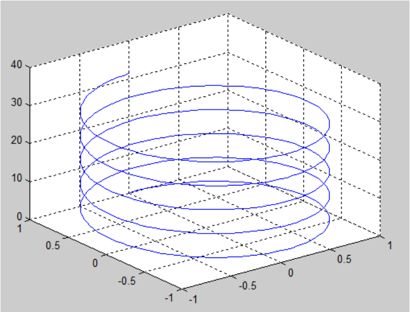
`x = x(t), y = y(t),  z = z(t).`

Функция `plot3` удобна для отображения точек данных. Нужно лишь поставить маркер «точка». Для этой же цели используется `scatter3`.

Пример. Винтовая линия

Винтовая линия – растущая вверх окружность единичного радиуса.

```
t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t)
grid on
```



Задача 2

Построить графики пространственных кривых

1. $$\begin{aligned} x &= \frac{a\cos t}{t}, \\ y &= \frac{a\sin t}{t}, \\ z &= at, \end{aligned}$$
   где $a = 2$, $\pi \le t \le 10\pi$.
2. $$\begin{aligned} x &= 10\cos t - 5\cos 2t, \\ y &= 10\sin t - 5\sin 2t, \\ z &= 10t, \end{aligned}$$
   $0 \le t \le 10\pi$.

Тип маркера – “точка”, цвет и стиль линии выбрать самостоятельно.

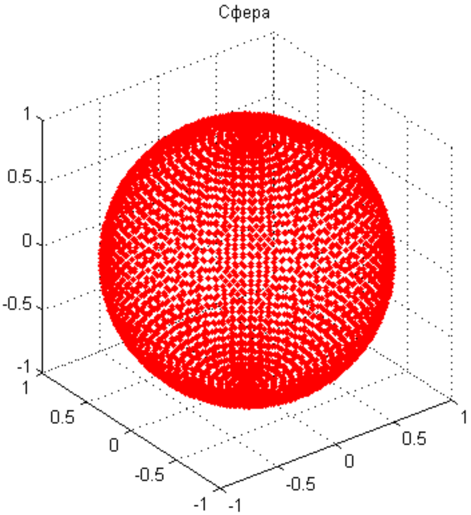
Сохранить текст программы в m-файле, а рисунок — в файле формата `*.fig`.

Задача 3

Загрузить координаты точек из файла `data.csv`. Отобразить точки при помощи `plot3` или `scatter3`.

Пример. Построение графика единичной сферы с помощью окружностей

```
n = 50; m = 50;
t1 = -pi:pi/n:pi;
t2 = [-pi/2:pi/m:pi/2]';
X = cos(t2)*cos(t1);
Y = cos(t2)*sin(t1);
E = ones(size(t1));
Z = sin(t2)*E;
plot3(X,Y,Z,'r.'), grid, title('Сфера'), axis square
```


Сфера
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Задача 4

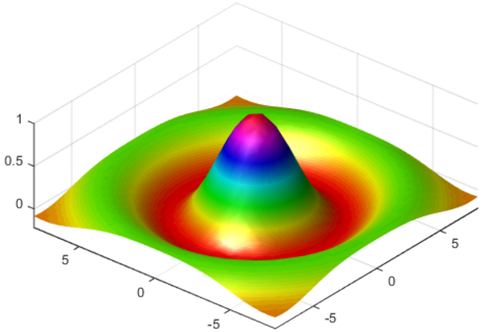
В рассмотренном выше примере построения графика единичной сферы добавить ввод параметров `n` и `m` с клавиатуры и установить стиль линии – “сплошная”. Задав по своему усмотрению `n` и `m`, построить единичную сферу.

Построить единичную сферу с помощью функции `sphere(n)`.

Сохранить код программы в файле.

Поверхность в трехмерном пространстве

Задается функцией двух переменных: $z = f(x, y)$.



Каждой точке с координатами x и y соответствует «высота» z (значение функции).

Как задать область определения

Чтобы построить график функции одной переменной $y = f(x)$, мы изменяли аргумент x с заданным шагом и вычисляли соответствующие значения функции y (`linspace`).

В пространственном случае будем поступать аналогично – строить сетку: изменять значения аргументов x и y с постоянным шагом и вычислять соответствующие значения функции z.

Для простоты будем рассматривать только прямоугольные области определения.

Пример. Построить график функции $z = x^2 + y^2$ в области $0 \le x \le 1$, $0 \le y \le 1$ с шагом 0.5.

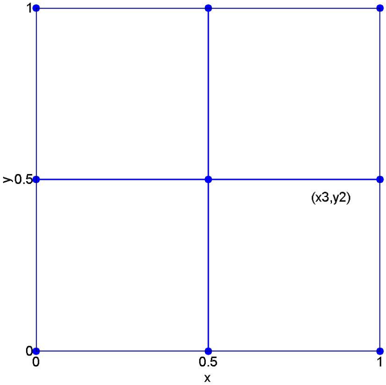
В традиционном языке программирования для вычисления значений функции понадобятся циклы

```
x = 0:0.5:1; y = 0:0.5:1;
for j=1:length(y)
  for i=1:length(x)
    Z(i,j)=x(i)^2+y(j)^2;
  end
end
...рисующая массив z функция...
```

Для MATLAB так тоже можно, но не эффективно: действия над массивом в целом реализуются быстрее, чем циклы по элементам.

Но мы не можем записать `Z = x.^2 + y.^2`, поскольку массивы `x`, `y` — одномерные, а `Z` — двумерный (значения функции в узлах сетки).

Как быть?




```
X =[0 0.5 1;
    0 0.5 1;
    0 0.5 1];

Y =[0   0   0;
    0.5 0.5 0.5;
    1   1   1];
```

Строки матрицы X состоят из копий вектора x. Число этих копий равно числу элементов в векторе y.

Аналогично, y-координаты образуют матрицу Y. Здесь уже столбцы матрицы представляют собой y-координаты узлов сетки, а число столбцов равно числу элементов вектора x.

Теперь каждому `Z(i,j)` соответствует `X(i,j)` и `Y(i,j)`:

```
x = 0:0.5:1; y = 0:0.5:1;
for j=1:length(y)
  for i=1:length(x)
     X(i,j)=x(j); Y(i,j)=y(i);
     Z(i,j)=X(i,j)^2+Y(i,j)^2;
  end
end
...рисующая массив z функция...
```

meshgrid — формирование прямоугольной координатной сетки

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

`x`, `y` – векторы, задающие диапазоны изменения по осям Ox, Oy.
`X`, `Y` — матрицы элементов сетки.

```
[X,Y] = meshgrid(0:1,0:1);
Z = X.^2 + Y.^2;  % поэлементное действие!
```

Построение графиков функций двух переменных

График функции 2-х переменных – это поверхность в пространстве.

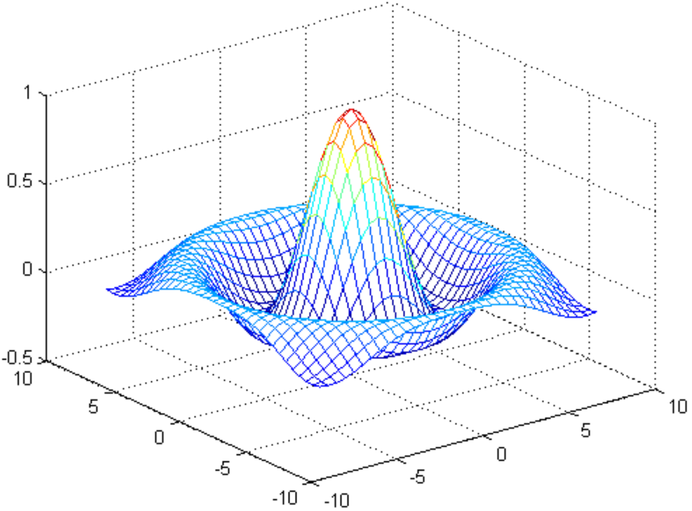
Поверхности строятся одним из следующих способов:

- `mesh(X,Y,Z)` – проволочный каркас поверхности.
- `surf(X,Y,Z)` – грани, соединяющие ребра каркаса.

Каркас окрашен в зависимости от значения функции в данной точки. Например, большие значения функции окрашены в более теплые цвета (максимальное – в ярко красный), меньшие – в более холодные (наименьшее – в синий).

Принцип раскраски граней тот же, что у каркаса.

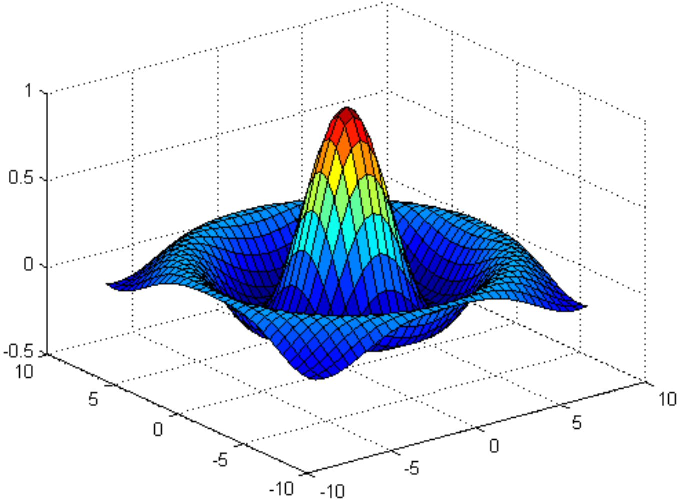
mesh



Код примера

```
[X,Y] = meshgrid(-8:.5:8);
R = sqrt(X.^2 + Y.^2) + eps;
Z = sin(R)./R;
mesh(X,Y,Z)
```

surf



Алгоритм построения графика функции двух переменных

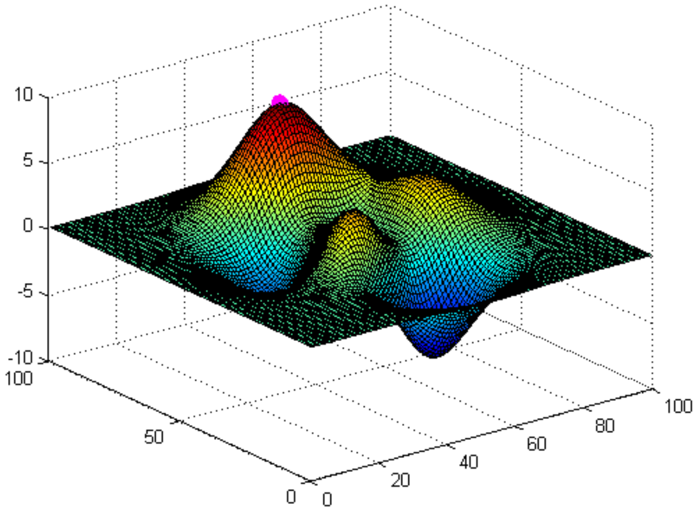
1. Сгенерировать матрицы `X`,`Y` с координатами узлов сетки (`meshgrid`).
2. Вычислить значения функции в узлах сетки: `Z = f(X,Y)`.
3. Использовать одну из графических функций MATLAB: `surf`, `mesh`,...

Пример: найдем максимальное значение функции peaks и выведем его на график

```
Z = peaks(100);
surf(Z), hold on
Zmax = max(max(Z));    % максимум по столбцам и строкам
[i,j] = find(Z==Zmax); % индексы максимального элемента
% Отмечаем максимум большой фиолетовой точкой
plot3(j,i,Z(i,j),'m.','MarkerSize',30)
```

`i` и `j` — не координаты, а индексы максимального элемента массива, координаты же этого элемента: `x=j`, `y=i`.

Если данные распределены неравномерно, то вместо `meshgrid` используются `griddata` или `TriScatteredInterp` (в последних версиях MATLAB).


10
5
0
-5
-10
100
50
0
0
20
40
60
80
100

Задача 5

Построить график поверхности $z = x\exp(-x^2 - y^2)$ в области $-3 \le x, y \le 3$ с помощью следующих графических функций:

- mesh
- surf
- contour

Шаг сетки выбрать по своему усмотрению.

Сделать подписи к осям координат и заголовок графика. В заголовке нужно указать отображаемую на графике функцию.

[x,y] = meshgrid(-5:0.1:5,-5:0.1:5);

Вспомогательные графические функции

Большинство вспомогательных функций, определяющих внешний вид графика в плоском случае, работают и в трехмерной графике.

- `axis` – управление свойствами осей координат сохраняют свои свойства и в трехмерной графике. Только добавляется третья ось координат: `axis([xmin xmax ymin ymax zmin zmax])`
- Для обозначения по оси z используется функция `zlabel`: `zlabel('строка')`

Кроме этого существуют функции, специально предназначенные только для трехмерной графики. Например:

- `hidden on/off` – {включить}/выключить удаление невидимых линий.

Ссылки

1. LearnOpenGL - Transformations – иллюстрации к норме и произведениям векторов.
2. TutorialsPoint: 2D Transformation – картинки с переносом, масштабированием и отражением.
3. Representing Data as a Surface - MATLAB & Simulink – график поверхности, код примера построения графика `sinc`.
4. MATLAB Plot Gallery – просто красивые и полезные примеры.

Список задач

- Задача 1
- Задача 2
- Задача 3
- Задача 4
- Задача 5