

ОСНОВЫ MATLAB

Функции, ячейки, строки, структуры

Юдинцев В. В.

Кафедра теоретической механики

13 сентября 2024 г.



САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

- 1 Функции MATLAB
- 2 Ячейки
- 3 Структуры
- 4 Строки

Функции MATLAB

Математические функции

`sin, cos, tan, cot, acos, asin, ...` – тригонометрические
`log, log10, log2, exp, sqrt, nthroot(x,n), ...`
`sign(a)` – знак числа `a`: -1, 0, +1.

Функции округления

- К ближайшему к нулю целому:
`fix(1.8)=1` но `fix(-1.8)=-1`
- К меньшему целому:
`floor(-1.9)=-2` но `floor(1.9)=1`
- К большему целому:
`ceil(1.3)=2`
- К ближайшему целому:
`round(1.4)=1`, `round(-1.5)=-2`

Функции комплексных чисел

- `abs(z)` – модуль.
- `angle(z)` – аргумент.
- `conj(z)` – комплексно-сопряженное число.
- `imag(z)` – мнимая часть.
- `real(z)` – вещественная часть.
- `isreal(z)` – 1, если z – вещественное число, 0 – мнимое.

Специальные матрицы

- `eye(n)`, `eye(n,m)` – единичные матрицы $n \times n$.

```
1 >> eye(3,4)
2 ans =
3     1     0     0     0
4     0     1     0     0
5     0     0     1     0
```

- `rand(n)`, `rand(n,m)` – матрица псевдослучайных чисел
- `ones(n,m)` – матрица $n \times m$, заполненная единицами
- `zeros(n,m)` – матрица $n \times m$, заполненная нулями

Специальные матрицы

`magic(n)` – квадратная матрица $n \times n$, с одинаковыми суммами элементов по строкам и столбцам и диагоналям.

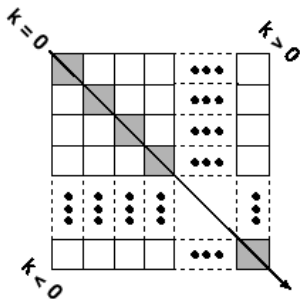
```
1 >> magic(3)
2 ans =
3     8     1     6
4     3     5     7
5     4     9     2
```


Функции над матрицами

- `diag(A)`, `diag(A,k)` – строка, содержащая диагональные элементы матрицы A , если A - матрица ($k=0$ для главной диагонали).
- `diag(V)` – диагональная матрица с элементами вектора V на диагонали, если V - вектор.
- `tril(A)`, `tril(A,k)` – нижняя треугольная матрица, построенная по матрице A .
- `triu(A)`, `triu(A,k)` – верхняя треугольная матрица, построенная по матрице A .

Функции над матрицами

Параметр k в функциях `diag(A,k)`, `tril(A,k)`, `triu(A,k)`



Функции над векторами

- `cross(a,b)` – векторное произведение;
- `dot(a,b)` – скалярное произведение;
- `sum(a)` – сумма элементов вектора;
- `sort(a)` – сортировка вектора; если `a` – матрица, то производится сортировка элементов в столбцах.
Другой вариант вызова `[res,i]=sort(a)`. Переменная `i` содержит индексы элементов исходного вектора `a`, расположенные по порядку сортировки.

Сумма элементов вектора или матрицы

`sum(a)`

Сумма элементов вектора:

```
1 >> a=[1 2 3 4];  
2 >> sum(a)  
3 ans=  
4     10  
5
```

Сумма элементов матрицы:

```
1 >> a=[1 2;  
2       3 4];  
3 >> sum(a)  
4 ans=  
5     4 6  
6
```

Сумма элементов матрицы

Сумма элементов матрицы по заданному измерению:

```
1 >> a=[1 2;  
2       3 4];  
3 >> sum(a,2)  
4 ans =  
5     3  
6     7  
7
```

Сортировка

- `sort(a)` – сортировка вектора или матрицы.
- Если `a` – матрица, то производится сортировка элементов в столбцах (по первой размерности)

```
1 >> a = magic(3)
2 a =
3     8     1     6
4     3     5     7
5     4     9     2
6 >> sort(a)
7 ans =
8     3     1     2
9     4     5     6
10    8     9     7
```

Сортировка

Вторым аргументом функции `sort` может быть номер измерения, по которому должна выполняться сортировка:

```
1 >> a = magic(3)
2 a =
3
4     8     1     6
5     3     5     7
6     4     9     2
7
8 >> sort(a,2)
9 ans =
10
11     1     6     8
12     3     5     7
13     2     4     9
```

Сортировка

`[res,i]=sort(a)`. `i` содержит индексы элементов исходного вектора `a`, расположенные по порядку сортировки.

```
1 >> a = magic(3)
2 a =
3     8     1     6
4     3     5     7
5     4     9     2
6 >> [res , i] = sort(a)
7 res =
8     3     1     2
9     4     5     6
10    8     9     7
11 i =
12     2     1     3
13     3     2     1
14     1     3     2
```


Битовые функции

`bitand(a,b)` – поразрядное И

```
1 >> bitand(5,1)
2 ans =
3     1
```

`bitor(a,b)` – поразрядное ИЛИ

```
1 >> bitor(4,2)
2 ans =
3     6
```

`bitset(a,bit,v)` – установка бита в позиции `bit` числа `a`; `v = 1` или `0`

```
1 >> bitset(5,3,0)
2 ans =
3     1
```

Преобразование числа к другому основанию

`dec2bin(a)` – строка с двоичным представлением положительного числа a

```
1 >> dec2bin(5)
2 ans =
3     101
4
```

`dec2hex(a)` – строка с шестнадцатеричным представлением положительного числа a

```
1 >> dec2hex(255)
2 ans =
3     FF
4
```

Преобразование числа к другому основанию

`bin2dec(a)` – преобразование строки с двоичным представлением числа к десятичной системе a

```
1 >> bin2dec('111')
2 ans =
3     7
4
```

`hex2dec(a)` – преобразование строки с шестнадцатеричным представлением числа к десятичной системе a

```
1 >> hex2dec('F1')
2 ans =
3    241
4
```

Работа с множествами

`intersect(A, B)` – пересечение векторов A и B как множеств:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> intersect(a,b)  
4 ans =  
5     1     7
```

`ismember(A, S)` – содержит ли A элементы из S:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> ismember(a,b)  
4 ans =  
5     1     0     0     1     0     1     0     1
```

Работа с множествами

`setdiff(A, B)` – элементы, входящие в A, но отсутствующие в B (A-B):

```
1 >> a = [1 2 3 7 8 7 3 1];
2 >> b = [7 1 5 6];
3 >> setdiff(a,b)
4 ans =
5     2     3     8
```

`setxor(A, B)` – элементы, не входящие в результат пересечения множеств A и B:

```
1 >> a = [1 2 3 7 8 7 3 1];
2 >> b = [7 1 5 6];
3 >> setdiff(a,b)
4 ans =
5     2     3     5     6     8
```

Работа с множествами

`union(A, B)` – объединение множеств:

```
1 >> a = [1 2 3 7 8 7 3 1];
2 >> b = [7 1 5 6];
3 >> union(a,b)
4 ans =
5     1     2     3     5     6     7     8
```

`unique(A)` – элементы вектора A без повторений:

```
1 >> a = [1 2 3 7 8 7 3 1];
2 >> unique(a)
3 ans =
4     1     2     3     7     8
```

Ячейки

Массивы ячеек (cells) могут хранить данные различных типов: числа, строки, структуры, массивы ячеек.

```
1 >> data{1} = 'Текст'
2 data =
3 'Текст'
4 >> data{2} = 1
5 data =
6 'Текст' [1]
7 >> data{5} = [1 2 3]
8 data =
9 'Текст' [1] [] [] [1x3 double]
```


Доступ к элементам массива ячеек

```
1 >> data{1}
2 ans =
3     Текст
4
5 >> data{2}
6 ans =
7     1
8
9 >> data{5}
10 ans =
11     1     2     3
```

Функция cell

Формирование пустой матрицы ячеек

```
1 >> a = cell(3,4)
2 a =
3     []     []     []     []
4     []     []     []     []
5     []     []     []     []
```

Присвоение значения элементу новой матрицы

```
1 >> a{2,3} = 'элемент 2,3'
2 a =
3     []     []     []     []
4     []     []     'элемент 2,3'   []
5     []     []     []     []
```

Функции для работы с ячейками

Преобразование числовой матрицы в матрицу ячеек

```
1 >> a=magic(5)
```

```
2 a =
```

```
3     17     24     1     8     15
4     23     5     7    14    16
5     4     6    13    20    22
6    10    12    19    21     3
7    11    18    25     2     9
```

```
1 >> b = mat2cell(a,[2 3],[2 1 2])
```

```
2 b =
```

```
3     [2x2 double]     [2x1 double]     [2x2 double]
4     [3x2 double]     [3x1 double]     [3x2 double]
```

Функции для работы с ячейками

Преобразование числовой матрицы в матрицу ячеек

```
1 >> a=magic(5)
2 a =
3     17     24     1     8     15
4     23     5     7    14    16
5     4     6    13    20    22
6    10    12    19    21     3
7    11    18    25     2     9
```

```
1 >> b = mat2cell(a,[2 3],[2 1 2])
2 b =
3     [2x2 double]     [2x1 double]     [2x2 double]
4     [3x2 double]     [3x1 double]     [3x2 double]
```

Преобразование матрицы ячеек в числовую матрицу

Функция `mat2cell`:

```
1 >> b = mat2cell(a,[2 3],[2 1 2])
2 b =
3     [2x2 double]     [2x1 double]     [2x2 double]
4     [3x2 double]     [3x1 double]     [3x2 double]
5 >> cell2mat(b)
6 ans =
7     17     24     1     8     15
8     23     5     7    14    16
9     4     6    13    20    22
10    10    12    19    21     3
11    11    18    25     2     9
```

Структуры

Структура

Структура – это элемент данных, который может содержать разнотипные поля:

```
имя = struct('поле1', значение, 'поле2', значение, ...)
```

Структура, описывающая параллелепипед:

```
1 box=struct('height', 100, 'width', 10, 'depth', 5)
2 box=
3     height : 100
4     width  : 10
5     depth  : 5
```

Доступ к полям структуры

- Считывание значения поля

```
1 >> box.height
2 ans =
3     100
```

- Имя поля может быть задано строкой

```
1 >> box.( 'height' )
2 ans =
3     100
```


Список полей структуры

```
1 >> names = fieldnames(box)
2 names =
3     'height'
4     'width'
5     'depth'
6     'mass'
```

Изменение значения поля

```
1 >> box.height = 20
2 box =
3     height: 20
4     width: 10
5     depth: 5
```

Добавление поля

Для добавления поля к структуре необходимо присвоить значение новому полю, как существующему:

```
1 >> box.mass=5
2 box =
3     height : 20
4     width  : 10
5     depth  : 5
6     mass   : 5
```

Создание структуры

Создание новой структуры без использования функции `struct`:

```
1 >> sphere2.radius=2  
2 sphere2 =  
3     radius : 2
```

Строки

Строковые переменные

Строки создаются при помощи одинарных кавычек

```
1 >> a = 'To be or not to be?'
```

Строка это массив символов

```
1 >> whos
2  Name           Size           Bytes   Class   Attributes
3
4  a              1x19           38     char
```

Склейка строк

Для склейки двух строк их можно объединить в матрицу-строку:

```
1 >> S1 = 'String 1 ';
2 >> S2 = 'String 2';
3 >> [S1 S2]
4 ans =
5
6     'String 1 String 2'
```

или использовать функцию `strcat`

```
1 >> strcat(S1, S2)
2 ans =
3
4     'String 1String 2'
```

Во втором случае пробелы в конце строк игнорируются.