

## Тема 4.2

# Программирование на языке MATLAB

## Вопросы для изучения

4.6 Работа с массивами данных

4.7 Создание и редактирование векторов и матриц

4.8 Выделение подматриц

4.9 Основные поэлементные действия над матрицами. Функции для обработки векторов и матриц

## 4.6 Работа с массивами данных

MatLAB - система, специально предназначенная для осуществления сложных вычислений с векторами и матрицами.

Все данные MatLab представляет в виде массивов.

MATLAB не требует от пользователя предварительного задания размерности и размеров массива. Пользователь может вводить ее постепенно, при этом MATLAB будет динамически перестраивать структуру массива.

Массив - упорядоченная, пронумерованная совокупность однородных данных. У массива должно быть имя.

Под вектором в MatLAB понимается одномерный массив чисел, а под матрицей - двумерный массив.

Доступ к элементам массива осуществляется при помощи индекса. В MatLab нумерация элементов массивов начинается с единицы. Это значит, что индексы должны быть больше или равны единице.

По умолчанию предполагается, что любая заданная переменная является вектором или матрицей. Например, отдельное заданное число система воспринимает как матрицу размером  $(1 \times 1)$ , а вектор-строку из  $N$  элементов - как матрицу размером  $(1 \times N)$ .

Массивы в MATLAB не образуют нового типа данных. Числовые массивы состоят из элементов типа **double**.

Помимо памяти, необходимой для хранения собственно значений числовых элементов (по 8 байт на каждый в случае вещественных чисел и по 16 байт в случае комплексных чисел), при создании массивов MATLAB автоматически выделяет еще и память для управляющей информации. В этой области памяти хранится размерность массива, количество элементов по каждой размерности, тип элементов (вещественные или комплексные) и так далее.

## 4.7 Создание и редактирование векторов и матриц

Для создания одномерного массива используют:

- операцию конкатенации,
- операцию индексации,
- вызов специальных функций.
- специальную операцию, обозначаемую двоеточием.

Операция **конкатенации** обозначается с помощью квадратных скобок [ ].

При использовании операции конкатенации объединяемые в одномерный массив элементы располагаются между открывающей и закрывающей квадратными скобками и отделяются друг от друга либо пробелом, либо запятой.

Например, следующее выражение, использующее операцию конкатенации

```
>> z = [1 2 3]
```

формирует переменную с именем z, являющуюся одномерным массивом, состоящим из трех элементов (вещественных чисел).

Выражение

```
>> z = [1,2,3]
```

по своему результату абсолютно идентично предыдущему.

В MATLAB все одномерные массивы трактуются либо как вектор-строки, либо как вектор-столбцы. При вводе вектор-строк в операциях конкатенации в качестве разделителей использовали либо пробелы, либо запятые.

Следующее выражение, использующее операцию конкатенации, задает вектор-столбец

```
>>b=[1;2;3]
```

состоящий из трех строк, так как точка с запятой в операции конкатенации означает переход на новую строку.

Операция **индексации** для ввода массива основана на возможности доступа к отдельному элементу одномерного массива: после имени массива необходимо указать в круглых скобках индекс (номер) элемента. В итоге третий элемент массива  $z$  обозначается как  $z(3)$ , первый элемент — как  $z(1)$ , второй элемент — как  $z(2)$ .

Например, следующее выражение, использующее операцию индексации

```
>> z(1) = 1;  
>> z(2) = 2;  
>> z(3) = 3;
```

формирует одномерный массив, состоящий из трех элементов (вещественных чисел).

Описанное пошаговое создание массива из трех элементов возможно потому, что MATLAB с каждым новым присваиванием автоматически перестраивает свою служебную информацию о массиве, а также область памяти, отводимой под его элементы.

Описанный способ создания одномерного массива не является эффективным и проигрывает в быстродействии операции конкатенации.

Проигрыш в быстродействии мало заметен когда пользователь вводит всю информацию с клавиатуры, однако становится критичным в программном режиме, когда MATLAB подряд исполняет многочисленные инструкции с массивами

Для экономии ресурсов ЭВМ присваивание значений элементам массива, начиная с последних по номеру элементов и заканчивая первым:

```
>> z(3) = 3;
```

```
>> z(2) = 2;
```

```
>> z(1) = 1;
```

Здесь при выполнении первого же присваивания система MATLAB выделяет память под три вещественных числа, присваивает указанное значение третьему элементу, второму и затем первому элементу.



Создание массива с **вызовом специальных функций** увеличить быстродействие работы MATLAB примерно в 100 раз.

Во-первых, можно предварительно выделить всю необходимую память под конечный размер массива. Это достигается вызовом функций:

**zeros(m,n)** – создает матрицу размером  $M \times N$  с нулевыми элементами;

**ones(m,n)** – создает матрицу размером  $M \times N$  с единичными элементами;

**eye(m,n)** – создает единичную матрицу размером  $M \times N$ , т.е. с единицами по главной диагонали и остальными нулевыми элементами;

**rand(m,n)** – создает матрицу размером  $M \times N$  из случайных чисел равномерно распределенных в диапазоне от 0 до 1;

Во-вторых постепенно прописать элементы нужными значениями не требует перестройки структуры памяти, отведенной под массив и, следовательно, позволяет сэкономить время. К примеру, для массива  $z$  можно перед присваиваниями сделать следующий вызов функции **ones**:

```
>> z=ones(1,3);
```

сразу создается массив из трех элементов, равных единице. После этого можно осуществить присваивания нужных значений элементам массива

```
>> z(1) =1;
```

```
>> z(2) =2;
```

```
>> z(3) =3;
```

Еще один способ ввода массива, основан на применении **специальной операции, обозначаемой двоеточием** - операцией формирования диапазона числовых значений.

Например, создание одномерного массива чисел в диапазоне от 1 до 3 с приращением 1:

```
>>z = 1:1:3;
```

Сначала включается в формируемый массив левая граница диапазона. Затем к этому числовому значению прибавляется приращение, которое указывается после первого двоеточия. Если сумма не превосходит верхней границы диапазона, то она включается в качестве элемента в формируемый массив. Это все повторяется до тех пор, пока очередное числовое значение не превысит верхнюю границу.

При необходимости изменить элемент сформированного одномерного массива можно применить операцию индексации и операцию присваивания:

$$z(3) = 78;$$

Если, например, второй элемент массива  $z$  должен стать равным среднему арифметическому первого и третьего элементов, нужно выполнить следующую команду:

$$z(2) = (z(1) + z(3)) / 2$$

Запись несуществующего элемента массива означает добавление нового элемента к уже существующему массиву:

$$\gg z(4)=7;$$

Тоже самое действие — «удлинение» массива  $z$ , можно выполнить и с помощью операции конкатенации:

$$\gg z=[z 7];$$

Можно подвергнуть конкатенации и несколько массивов. Например, следующий код

$$\gg y=[z z 9 z]$$

Операцию индексации можно применять как справа от знака операции присваивания, так и слева от него т.е. осуществляется доступ к элементу массива «по чтению» или «по записи».

### Пример

При попытке чтения несуществующего элемента (например, десятого элемента массива  $z$ ) в командном окне появится сообщение об ошибке.

### Пример

Для создания двумерного массива (матрицы) в MATLAB используют:

- операцию конкатенации,
- операцию индексации,
- вызов специальных функций.

Значения элементов матрицы вводятся в квадратных скобках по строкам. При этом элементы строки матрицы разделяются пробелом или запятой, а строки отделяются одна от другой точкой с запятой.

Матрицу  $X$  размером  $3 \times 2$  (первым указывается число строк, вторым — число столбцов), получающуюся в результате операции конкатенации

```
>>X=[1 2 ;3 4 ;5 6]
```

```
1 2
```

```
3 4
```

```
5 6
```

Матрицу X можно сформировать:  
вертикальной конкатенацией векторов-строк:

```
>> X=[[1 2]; [3 4]; [5 6]]
```

или горизонтальной конкатенацией векторов-столбцов:

```
>> X=[[1;3;5],[2;4;6]]
```

Вертикальную и горизонтальную конкатенации можно также осуществить с помощью функции `cat`. Для вертикальной конкатенации ее первый параметр равен единице

```
>>X=cat(1, [1 2] , [3 4] , [5 6])
```

а для горизонтальной конкатенации он равен двум:

```
>>X=cat(2,[1;3;5],[2;4;6])
```

Как и рассмотренные ранее одномерные массивы (векторы), двумерные массивы можно создать с помощью операции индексации, прописывая по отдельности его элементы необходимыми числовыми значениями.

Например, рассмотренный ранее массив X можно создать следующим образом

```
>> X(1,1)=1; X(1,2)=2;  
>> X (2 ,1) =3 ; X (2, 2) =4 ;  
>> X (3,1) =5 ; X (3,2) =6 ;
```

где для доступа («чтению») к отдельным элементам используются круглые скобки (операция индексации), внутри которых через запятую перечисляются индексы. Здесь первым указывается номер строки, вторым — номер столбца.

Как и в случае одномерных массивов, это решение является неэффективным, так как по мере присваивания MATLAB приходится перестраивать структуру массива. Проблема преодолевается, если присваивание

```
>> X (3,2) =6;
```

поместить первым.

Кроме того, можно сразу создать двумерный массив нужного размера функциями `ones` или `zeros`, у этих функций первый параметр задает число строк, а второй число столбцов. :

```
>> ones (3,2)
```

или

```
>> zeros (3,2)
```

а затем осуществить присваивания отдельным элементам нужных значений (порядок присваивания не имеет значения).

Редактирование двумерных массивов и обращение к их элементам производится с использованием операции индексации.

Пример



## 4.8 Выделение подматриц

Выделение блоков матриц осуществляется индексацией при помощи двоеточия например

```
>>X1 = X(2:3,2:3)
```

Для выделения из матрицы столбца или строки (то есть массива, у которого один из размеров равен единице) следует в качестве одного из индексов использовать номер столбца или строки матрицы, а другой индекс заменить двоеточием без указания пределов. Например, запишите вторую строку матрицы X в вектор x

```
>>x = X(2, :)
```

При выделении блока до конца матрицы можно не указывать ее размеры, а использовать элемент end:

```
>>x = X(2, 2:end)
```

В MatLab парные квадратные скобки [ ] обозначают пустой массив, который, в частности, позволяет удалять строки и столбцы матрицы.

Для удаления строки следует присвоить ей пустой массив, например, удаление первой строки квадратной матрицы M:

```
>> M = [2 0 3; 1 1 4; 6 1 3];
```

```
>> M(1,:)=[];
```

Аналогичным образом удаляются и столбцы. Для удаления нескольких идущих подряд столбцов (или строк) им нужно присвоить пустой массив например, удаление второго и третьего столбца в массиве M

```
>> M(:, 2:3) = []
```

## 4.9 Основные поэлементные действия над матрицами. Функции для обработки векторов и матриц

Базовые операции над векторами и матрицами – сложение, вычитание, умножение матрицы на число, умножение матрицы на матрицу – выполняются с применением обычных знаков арифметических операций, по правилам, предусмотренным в математике, дополнительно используют матричные арифметические операции.

Функция	Обозначение (синтаксис)
Сложение	+ (M1+M2)
Вычитание	- (M1-M2)
Матричное умножение	* (M1*M2)
Поэлементное умножение массивов	.* (M1.*M2)
Возведение матрицы в степень	^ (M1^ x)
Поэлементное возведение массива в степень	.^ (M1.^ x)
Деление матриц слева направо	/ (M1 / M2)
Поэлементное деление массивов слева направо	./ (M1 ./ M2)
Деление матриц справа налево	\ (M1 \ M2)
Поэлементное деление массивов справа налево	.\ (M1 .\ M2)

Если A и B — массивы одинаковых размеров, то допустимы следующие выражения

$\gg C = A+B;$

$\gg O = A-B;$

где элементы массивов C и D равны сумме или разности соответствующих элементов массивов A и B. Таким образом, эти операции выполняются поэлементно и порождают массивы тех же размеров, что и исходные операнды.

В случае, когда один из операндов является скаляром:

$\gg A+5$

скаляр предварительно расширяется до массива размером с матричный операнд. Например, из скаляра 5 сначала генерируется матрица  $[5 \ 5 \ 5; 5 \ 5 \ 5]$ , которая и складывается далее поэлементно с матрицей A.

Для поэлементного перемножения и поэлементного деления массивов одинаковых размеров применяются операции, обозначаемые комбинациями двух символов: «.\*» и «./».

Кроме операции «./», называемой операцией правого поэлементного деления, есть еще операция левого поэлементного деления «.\». При этом выражение  $A./B$  приводит к матрице с элементами  $A(k,m)/B(k,m)$ , а выражение  $A.\ B$  приводит к матрице с элементами  $B(k,m)/A(k,m)$ .

Важно помнить, что при сложении или вычитании матрицы должны иметь одинаковые размеры, а при умножении матриц число столбцов первой матрицы должно совпадать с числом строк второй матрицы. Невыполнение этих условий вызывает появление в командном окне сообщения об ошибке.

## 4.9 Функции для формирования и обработки векторов и матриц

В MATLAB имеются ряд встроенных функций для создания векторов и матриц. С полным списком функций и примерами их использования можно познакомиться, выполнив из командной строки команду **help elmat** и на предыдущих слайдах.

Количество элементов в одномерном массиве возвращает функция **length**:

```
>> length( z )
```

Для того, чтобы узнать размеры двумерного массива и «геометрию» векторов (вектор-столбцы или вектор-строки), нужно использовать функцию **size**:

```
>> size(X)
```

Для нахождения числа измерений массива используется функция `ndims`:

```
>> ndims(a)
```

аналогом является использование

```
>> length(size(M))
```

Количество измерений в массиве-всегда больше или равно 2 (всегда есть строка и столбец).

Если после формирования двухмерного массива  $X$  потребуется, не изменяя элементов массива, изменить его размеры, можно воспользоваться функцией

```
reshape( X, M, N )
```

где  $M \times N$  — новые размеры массива  $x$  ( $M$  — число строк,  $N$  — число столбцов). Если количество элементов в массиве  $x$  не равно произведению  $M$  на  $N$ , то MATLAB выдаст сообщение об ошибке.

```
>> reshape(X,2,3)
```

Для нахождения векторного произведения предназначена специальная функция `cross`:

```
>> u=[1 2 3];
```

```
>> v=[3 2 1];
```

```
>> cross(u,v)
```

скалярное произведение векторов вычисляется с помощью функции общего назначения `sum`, вычисляющей сумму всех элементов векторов (для матриц эта функция вычисляет суммы для всех столбцов).

```
>> sum(u)
```



Для вычисления скалярного произведения также можно использовать функцию `dot`:

```
>> dot(u,v)
```

Длина вектора вычисляется с помощью функции `norm`:

```
>> norm(u)
```

Угол между векторами вычисляется на основе определения скалярного произведения, в соответствии с которым оно равно произведению длин векторов на косинус угла между ними. Отсюда находим выражение для вычисления угла между ранее заданными векторами  $u$  и  $v$ :

```
>> phi=acos(dot(u,v)/(norm(u) *norm(v) ));
```

Ранее рассмотренные операции отношения и логические операции выполняются в случае массивов поэлементно, поэтому размеры обеих операндов должны быть одинаковы.

Функция **prod** вычисляет произведение элементов столбцов матрицы. К примеру, для матрицы

```
>> A=[1 1 1;2 2 2;3 3 3]
```

она возвращает следующий результат:

```
>> prod(A)
6     6     6
```

Функции **max** и **min** ищут соответственно максимальный и минимальный элементы массивов. Для векторов они возвращают единственное числовое значение, а для матриц они порождают набор, соответственно, максимальных или минимальных элементов, вычисленных для каждого столбца, например:

```
>>max (A)
3     3     3
```

```
>>[m,k]=max(a)
```

- k содержит номер максимального элемента в векторе a

```
>>[m,k]=min(a)
```

- k содержит номер минимального элемента в векторе a

Функция `sort` сортирует в возрастающем порядке элементы одномерных массивов, а для матриц она производит такую сортировку для

```
>>sort(a) – по возрастанию
```

```
>> sort (-a) – по убыванию
```

```
>>[a1,ind]=sort(a)
```

- `ind` является вектором из целых чисел от 1 до `length(a)`, который соответствует проделанным перестановкам.

Функция `mean` вычисляет вычисление среднего арифметического элементов . К примеру, для матрицы `A`

```
>> m=mean(a)
```

Поменять местами строки матрицы с ее столбцами можно операцией транспонирования, которая обозначается знаком (апостроф). Например,

```
>> A=[1 1 1;2 2 2;3 3 3]
```

```
1 1 1  
2 2 2  
3 3 3
```

```
>> B=A'
```

```
1 2 3  
1 2 3  
1 2 3
```

Для квадратных матриц на своих местах остаются элементы главной диагонали квадратной матрицы, а остальные «отражаются симметрично» относительно этой диагонали.

Вектор-строки операцией транспонирования преобразуются в вектор-столбцы, и наоборот.

Вычисление обратной матрицы можно делать путем вызова функции

`inv(имя матрицы)`

или возводя матрицу в степень -1.

Вычисление определителя производится путем вызова функции

`det(имя матрицы)`