Computer Science

MATLAB

Essential MATLAB for Scientists

Chap 3: Fundamentals

2. MATLAB Fundamentals

- The objective of this chapter is to introduce some of the fundamentals of MATLAB programming, including:
  - Variables, operators and expressions
  - Workspace
  - Arrays (including vectors and matrices)
  - Operators, expressions and statements

2. MATLAB Fundamentals

1. Variables

Variables are fundamental to programming. The art of programming is *getting the right values in the right variables at the right time*.

- A variable name (like previously *balance*) must comply with the following two rules:
  - It may consist only of the letters **a-z**, the digits **0-9** and the underscore _.
  - It must start with a letter.
- A variable name may be as long as you like. However, MATLAB remembers the first 31 characters only.

2. MATLAB Fundamentals

1. Variables

- Examples of valid variable names:
  - r2d2
  - pay_day
- Example of invalid variable names:
  - pay-day
  - 2a
  - name$
  - _2a

2. MATLAB Fundamentals

1. Variables

A variable is created simply by assigning a value to it at the command line or in a program, e.g.

  - a = 98

- If you attempt to refer to a non-existent variable, you will get the error message:
  - ??? Undefined function or variable…

2. MATLAB Fundamentals

1. Variables

MATLAB is a case sensitive, which means it distinguishes between uppercase and lowercase letters. Therefore,

- balance,
- BALANCE,
- BaLance

are three different variables.

2. MATLAB Fundamentals

2. The workspace

- clear
  - All variables you create during a session remain in the workspace until your clear them.
    - Type the following and check the result
      - clear <**Enter**>
    - All the variables have been deleted from the workspace
- who
  - The command *who* lists the names of all the variables in your workspace
- whos
  - The command *whos* lists of all the variables in your workspace as well as the size of each variable
- ans
  - The command *ans* lists the names of the last expression evaluated but not assigned to a variable

2. MATLAB Fundamentals
2. The workspace

- Clear *variable_name* removes a particular variable from the workspace.
  - *clear rate balance* removes/deletes both variables rate and balance from the workspace.
- Note that when you run a program, any variable created by it remains in the workspace after the program has run. Therefore, existing variables with the same names got overwritten. Clearing variables could be a solution to such an issue.

2. MATLAB Fundamentals
2. The workspace

- If you often use the same physical or mathematical constants in you MATLAB sessions, you can save them in an M-file and run the latter at the start of a session.
- Example: save the following statements in *myconst.m*:

```
g = 9.81;                     % acceleration due to Earth gravity
avo = 6.023e23;               % Avogadro's number
e = 2.718281828459045;        % base of natural log
pi_4 = pi / 4;
log10e = log10( e );
bar_to_kP = 101.325;          % atmospheres to kilo Pascals
```

2. MATLAB Fundamentals
2. The workspace

- If you run *myconst* at the start of a session, all the variables included in that M-file will be part of the workspace and will be available for the rest of the session or until they are *cleared*.

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

Initializing vectors

Try the following exercise on the command line. You do not need reminding about the command prompt >> any more. So it will no longer appear unless the context absolutely demands it.

- Enter the following statement to create a vector (list) with five elements:
  - x = [1 3 0 -1 5] ;
- Enter the command *disp(x)* to see how MATLAB displays a vector.
- Enter the command whos. Under the heading *size* you will see that x is 1-by-5, which means 1 row and 5 columns.

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Exercise

You can put commas instead of spaces between vector elements. You may try the following:

    - a = [5,6,7]

- Do not forget to put either commas or spaces between elements, as you could end up with something quite different than expected otherwise. You may try the following:

    - y = [130-15]

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Exercise

You can put commas instead of spaces between vector elements. You may try the following:

    - a = [5,6,7]

  - Do not forget to put either commas or spaces between elements, as you could end up with something quite different than expected otherwise. You may try the following:

    - y = [130-15]

  - You can use one vector in a list for another one. You may try the following:

    - a = [1 2 3];
    - b = [4 5];
    - c = [a –b]

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Exercise
  - You may try the following now:
    - a = [1 3 7];
    - A = [a 0 -1]

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Exercise
  - You may try the following now:
    - a = [1 3 7];
    - A = [a 0 -1]
  - Now, you may try the following:
    - x = [ ]
  - Note that the size of x is given as 0-by-0 because x is empty. This means that x is defined but has neither a size nor a value. Making x empty is different than making x = 0. When x = 0 then x has a size 1-by-1. Finally, making x empty is different from clearing x using the command *clear x* which removes x from the workspace and hence make it undefined.

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

Remember that elements in a list must be:

- Enclosed in square brackets, not in round brackets.
- Separated either by spaces or by commas.

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- The colon operator
  - A vector cans also be generated (initialized) with the colon operator. Enter the following statements:
    - x = 1:10 % elements are the integers 1, 2, …, 10
    - x = 1:0.5:4 % elements are 1, 1.5, 2, 2.5, 3, 3.5, 4
    - x = 10:-1:1 % elements are 10, 9, …, 1
    - x = 1:2:6 % elements are 1, 3, 5
    - x = 0:-2:-5 % elements are 0, -2, -4

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- The colon operator
  - A vector cans also be generated (initialized) with the colon operator. Enter the following statements:
    - x = 1:10 % elements are the integers 1, 2, …, 10
    - x = 1:0.5:4 % elements are 1, 1.5, 2, 2.5, 3, 3.5, 4
    - x = 10:-1:1 % elements are 10, 9, …, 1
    - x = 1:2:6 % elements are 1, 3, 5
    - x = 0:-2:-5 % elements are 0, -2, -4
  - Linspace
    - The function *linspace* can be used to initialize a vector of equally spaced values. The following command creates a vector of 10 equally spaced points from 0 to pi/2 inclusive.
      - linspace(0, pi/2, 10)

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Transposing vectors
  - Vectors seen so far are *row vectors*. Each has one row and several columns. To generate *column vectors*, you simply need to transpose the initial *row vectors*. This is done with a single quote, or apostrophe (’). Enter the following commands:
    - x = 1:5
    - x’

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Transposing vectors
  - Vectors seen so far are *row vectors*. Each has one row and several columns. To generate *column vectors*, you simply need to transpose the initial *row vectors*. This is done with a single quote, or apostrophe (’). Enter the following commands:
    - x = 1:5
    - x’
  - Note that x itself remains a row vector.
  - A column vector can be created using a 1-line command. Try the following:
    - y = [1 4 8 0 -1]’

2. MATLAB Fundamentals
3. Arrays: vectors and matrices

- Subscripts
  - We can refer to particular elements of a vector by means of subscripts. Try the following:
    - r = rand(1,7) % row vector of seven random numbers

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Subscripts
  - We can refer to particular elements of a vector by means of subscripts. Try the following:
    - r = rand(1,7)   % row vector of seven random numbers
    - r(3)

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Subscripts
  - We can refer to particular elements of a vector by means of subscripts. Try the following:
    - r = rand(1,7) % row vector of seven random numbers
    - r(3) % display of the third element of r
    - r(2:4)

2. MATLAB Fundamentals
3. Arrays: vectors and matrices

- Subscripts
  - We can refer to particular elements of a vector by means of subscripts. Try the following:
    - r = rand(1,7) % row vector of seven random numbers
    - r(3) % display of the third element of r
    - r(2:4) % display of the 2nd, 3rd and 4th elements of r
    - r(1:2:7)
    - r([1 7 2 6])

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Subscripts
  - We can refer to particular elements of a vector by means of subscripts. Try the following:
    - r = rand(1,7) % row vector of seven random numbers
    - r(3) % display of the third element of r
    - r(2:4) % display of the 2nd, 3rd and 4th elements of r
    - r(1:2:7)
    - r([1 7 2 6])
  - You can use an empty vector to remove elements from a vector. The following command removes elements 1, 7 and 2 from vector r which makes r a row vector of 4 numbers.
    - r([1 7 2]) = [ ]

2. MATLAB Fundamentals
3. Arrays: vectors and matrices

- A subscript is indicated inside round brackets (parentheses).
- A subscript may be a scalar or a vector.
- In MATLAB, subscripts always start at 1.
- Fractional subscripts are always rounded down. For example, x(1.9) refers to element x(1).

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Matrices
  - You create a matrix just as you do a vector, except that a semicolon is used to indicate the end of a row. Try the following:
    - a = [1 2 3; 4 5 6]   % 2 rows and 3 columns

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Matrices
  - You create a matrix just as you do a vector, except that a semicolon is used to indicate the end of a row. Try the following:
    - a = [1 2 3; 4 5 6]   % 2 rows and 3 columns
  - A matrix may be transposed. Try the following:
    - a'   % 3 rows and 2 columns

2. MATLAB Fundamentals

3. Arrays: vectors and matrices

- Matrices
  - You create a matrix just as you do a vector, except that a semicolon is used to indicate the end of a row. Try the following:
    - a = [1 2 3; 4 5 6]   % 2 rows and 3 columns
  - A matrix may be transposed. Try the following:
    - a'   % 3 rows and 2 columns
  - A matrix can be constructed from column vectors of the same length. Try the following:
    - x = [0:30:180]
    - table = [x' sin(x*pi/180)']

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Numbers
  - Numbers can be represented in MATLAB in the usual decimal form (*fixed point*), with an optional decimal point, e.g.
    - 1.2345
    - -123
    - 0.0001

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Numbers
  - Numbers can be represented in MATLAB in the usual decimal form (*fixed point*), with an optional decimal point, e.g.
    - 1.2345
    - -123
    - 0.0001
  - A number can also be represented in scientific notation, e.g. 1.2345×10^9 that is represented in MATLAB as 1.2345e+9. This is also called *floating point* notation. That number has two parts: the *mantissa*, which is the part on the left side of the symbol *e*, and the *exponent* which is the part on the right side of the symbol *e*.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The following is not scientific notation: 1.2345*10^9. It is actually and expression involving two arithmetic operations (* and ^) and involves more time consumption.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The following is not scientific notation: 1.2345*10^9. It is actually and expression involving two arithmetic operations (* and ^) and involves more time consumption.
- Do not hesitate to use scientific notations for very small and very large numbers.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The following is not scientific notation: 1.2345*10^9. It is actually and expression involving two arithmetic operations (* and ^) and involves more time consumption.
- Do not hesitate to use scientific notations for very small and very large numbers.
- The relative accuracy of numbers is given by the function *eps* which stands for epsilon and is the distance between 1.0 and the next largest floating point number. Type *eps* in the command line to see its value on your computer.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The following is not scientific notation: 1.2345*10^9. It is actually and expression involving two arithmetic operations (* and ^) and involves more time consumption.
- Do not hesitate to use scientific notations for very small and very large numbers.
- The relative accuracy of numbers is given by the function *eps* which stands for epsilon and is the distance between 1.0 and the next largest floating point number. Type *eps* in the command line to see its value on your computer.
- The range of numbers is roughly $\pm 10^{-308}$ to $\pm 10^{308}$. Precise values for your computer are returned by the MATLAB functions *realmin* and *realmax*.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Exercise 1

  Enter the following numbers at the command prompt in scientific notation

  - 1.234 x 10^5
  - -8.765 x 10 – 4
  - 10^{-15}
  - -10^{12}

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Arithmetic operators

Operation	Algebraic form	MATLAB
Addition	a + b	a + b
Subtraction	a – b	a – b
Multiplication	a x b	a * b
Right division	a / b	a / b
Left division	b / a	a \ b
Power	a^b	a ^ b

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Precedence of operators

Precedence	Operator
1	Parentheses (round brackets)
2	Power, left to right
3	Multiplication and division, left to right
4	Addition and subtraction, left to right

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Exercise 2: evaluate the following MATLAB expressions yourself before checking the answers in MATLAB.
  - 1 + 2 * 3
  - 4 / 2 * 2
  - 1 + 2 / 4
  - 1 + 2 \ 4
  - 2 * 2 ^ 3
  - 2 * 3 \ 3
  - 2 ^ (1 + 2)/3
  - $1/2 \times 10^{-1}$

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Exercise 2: evaluate the following MATLAB expressions yourself before checking the answers in MATLAB.
  - $1 + 2 * 3$
  - $4 / 2 * 2$
  - $1 + 2 / 4$
  - $1 + 2 \backslash 4$
  - $2 * 2 \wedge 3$
  - $2 * 3 \backslash 3$
  - $2 \wedge (1 + 2)/3$
  - $1/2 \times 10^{-1}$
- Exercise 3: evaluate the following MATLAB expressions yourself before checking the answers in MATLAB.
  - $\frac{1}{2 * 3}$
  - 2^{2*3}
  - $1.5 \times 10^{-4} + 2.5 \times 10^{-2}$

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The colon operator
  - The colon operator (which is not a division) has a lower precedence than + as the following shows:
    - 1 + 1 : 5

    The addition is carried out first, and then a vector with elements 2, 3, 4, 5 is initialized.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The colon operator
  - The colon operator (which is not a division) has a lower precedence than + as the following shows:
    - 1 + 1 : 5

    The addition is carried out first, and then a vector with elements 2, 3, 4, 5 is initialized.
    - 1 + [1:5]

    results in adding 1 to each element of the vector 1:5. In this context, the addition is called an *array operation*, because it operates on each element of the vector (array).

2. MATLAB Fundamentals

4. Operators, expressions and statements

- The transpose operator
  - The transpose operator has the highest precedence. Try the following:
    - 1:5’

    results in a row vector with the last element 5 transposed into itself since it is a scalar.
    - [1:5]’

    results in transposing the whole row vector into a column vector.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Arithmetic operations on arrays
  - Enter the following statements at the command line:
    - a = [2 4 8];
    - b = [3 2 2];
    - a .* b
    - a ./ b

Operator	Description
. *	Multiplication
. /	Division
. \	Left division
. ^	Power

  - Above are arithmetic operators that operate element by element on arrays.

2. MATLAB Fundamentals

4. Operators, expressions and statements

- Now try the following:
  - a . ^ b

  The *i*th element of the first vector is raised to the power of the *i*th element of the second vector.
- The period (dot) is necessary for the array operations of multiplication, division and exponentiation.
- When array operations are applied to two vectors, both vectors must be the same size.

To summarize

1. Variables, operators and expressions
   - Rules for variables
   - Precedence of operators
   - Arithmetic operators
2. Workspace
   - Save constants

To summarize

3. Arrays (including vectors and matrices)

- Different ways to create vectors
- Transposing vectors
- Subscript
- Matrices

4. Operators

- Number notation