Matlab Foundamental

Variables

In Matlab, you don't need to declare any variable before using it, which means that you can use a new variable whenever you want to.

As I have said in the previous parts, Matlab is the abbreviation of "Matrix Laboratory", so it has tight relationship with matrices. Actually, every variable in Matlab is a matrix, even if it main contain only one element. What's more, the size of the matrices is dynamic, which mean that you can change the size of it whenever you like.

While accessing elements in matrices, you might know that the statement is A(i,j), which indicates the ith row and jth column of matrix A. You should always be clear that the first index means the row number and second means column number.

Another thing is that, you can use the same method to get a part of the matrix, not only a single number, for example, if you want to get all elements in 1,3,5th rows and 2,4,6th columns of matrix A, you can write A([1,3,5],[2,4,6]), and it will generate a 3*3 matrix.

However, matrices can also have different types. When you are programming in this course, there are three main types you will see.

Double

Here you can just regard "double" has "numbers". It will occur most frequently in your programming.

Char

Char means "character", it means the the letters in a word, some symbols and also single digits as well(pay attention that the number here is not like double, we treat numbers here(0-9) as a single character). To declare that you are treating something as a character, use '' to contain it. For example, 'b' means the letter b, '1' means the character number 1 while 1 indicate a real number 1.

In fact, char variables are also stored in the form of real numbers(more accurately integers), every character has a number to label it, and we call this number "ASCII Code".

I think you should remember three of the ASCII codes. For digit '0', the number is 49; for letter 'A', it is 65, while 97 is for the lowercase letter 'a'. Digit '1' is right after '0', and 'B' is after 'A', 'b' is after 'a'. And so forth, you can know the exact ASCII code for the ten digits and all lowercase and uppercase letters.

And because these numbers are stored in integers, we can also conduct mathematical operations on them. For example

```
'b'+1
ans =
    99

'b' + 'c'
ans =
    197
```

Because the ASCII code for 'b' is 98 and 99 for 'c', so we will have this kind of calculation results. We can also use char() to let the answer be displayed in the form of a character

```
char('b'+1)
ans =
    'c'
```

Please note that the maximum for all ASCII codes is 127, so anything larger than 127 is not available to be turned into characters. (Matlab use larger nubers to store special symbols and characters in other languages, so you can also get an answer here. But you cannot do the same thing in C and C++)

Now if you put many characters into a column vector, the characters will form a word or sentence. Here we call a group of characters a string. For example

```
['U','M','J','I']
ans =
    'UMJI'
```

So here 'UMJI' is a string, and it is also a column vector whose elements are all char variables.

Logical

It is another different type of variable. A single logical variable can only can only be 0 or 1. It is like binary but it is not completely binary numbers. Here you can regard 0 as **false** and 1 as **true**.

For logical variables we have three main kind of operations: and(&&) or(||) and not(~). Here listed is the usage of the three:

```
1||0
ans =
    logical
    1

1&&0
ans =
    logical
    0

~0
ans =
    logical
    1
```

I believe you have already had basic knowledge about logic so I will not explain more about it.

Statements and Operations

For this part I don't want to explain too much because the things are very intuitive for you and you can search for them easily on the internet. For every part I will just list those functions and operations that you might need and you can explore them yourself.

Input & Output

```
A = input('Please input A:');

display(A);

fprintf('The answer of the question is %d',answer);

fout = fopen('output.txt');
fprintf(fout,'The answer of the question is %d',answer);
fclose(fout);
```

Condition Statement

```
if condition
    operations1
else
    operations2
end

if condition1
    operations1
elseif condition2
    operations2
else
    operations3
end

switch expression
    case value1
        operations1
    case value2
        operations2
    case value3
        operations3
    otherwise
        operations4
end
```

Loop Statements

```
for i = 1:m
    operations
end

while condition
    operations
end
```

Functions in Matlab

Built-in functions

The most important feature for Matlab is that it has a huge built-in funtion library. In most cases you will use functions concerned with mathematical calculation and matrix processing. Use "help" in the command line to get familiar with those built-in functions.

Here I will also list some useful functions for you:

```
min, max, sum, mean, find %statistic functions
sin(trigonalmetric), mod, round %mathematical functions
eye, zeros, ones, rand %special matices
size, repmat, numel %matrix processing
strcmp, strcat, strfind %string processing
```

Tip: In your matlab exam there will be a question asking you to explore the usage of a matlab function that you probably dont use in assignments. Based on my experience, this is a function concerning with image processing.

Return values and arguments

Here are two basic concepts about function: return value and argument.

The argument means the "condition" of the function, which is the things contained in the (). You should know how to correctly put things into the argument list, and pay attention that the same function may have different usages if you use different types of arguments. Use "help" to see the function's requirements for arguments.

Return value is the "result" of the function. I said that you need to get familiar with functions, the most important part is to understand what its result will be, not only the result in mathematical concepts but also the form it will be presented. Another thing is that a function might have multiple return values(just in Matlab). Take "max" as an example:

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

[B,C] = max(A)
B =
    8 9 7
C =
    1 3 2
```

As you can see, the first return value indicates the maximum number of every column in A while C returns their locations.

Of course, in many situations you only need one of the two return values. So if you just use the first return value(B), then you just write

```
B = max(A)
```

However, if you want to get the second(C) only, you need to write

```
[~,C] = max(A)
```

Here it uses a ~ to creat a null variable in the place of B.

In conclusion, the most important part of studying functions is getting clear about the arguments and return values.

Functions written by yourselves

You may learn that, actually you can write the function yourself. And the process is not so difficult. Just create a script with the same name of your function(it is important). And go on like this:

```
function [y1,y2,...] = func1(x1,x2...)
    expressions

end
```

The y1,y2... are the return values and x1,x2... are the arguments. If you only have one return value, you don't need a [] here.

And you should implement the function in the expressions part.

Important: make sure that you have assigned a value to the return values before the function ends.

Tip: All variables in Matlab are matrices with dynamic size, so are return values of functions. But if you can work out the size of the return values based on arguments and your mathematical knowledge, you can initialize the size of the return values using zeros() function. This is useful when there are loop statements or complex mathematical calculations in your function, which means that you cannot complete the implementation using just one line.

Using Matlab to plot

This is one of the most powerful features of Matlab. Most of the issues are simple and will be taught clearly in the lecture, and here are some important points and tips for you.

1. "hold on" is a command to keep the plot you have drawn staying on the canvas when you want to add more things to the plot. If you don't use it, the things you have drawn before will disappear.
2. "plot" is not the only drawing function. You should use different plotting functions based on the question. For example, you can use functions like "scatter" and "contour". The arguments of them are similar to those of "plot".
3. "plot" is a function, and it does have a return value. I suggest you declare a variable and assign it the return value of the functions. Then the variable can represent the plot you have just drawn, and in Matlab there are many plot-concerned functions which need this.

Structures

This is a new terminology for you and I think structures in Matlab is very useful. Sometimes when you want your single variable to have multiple "properties", you can use a structure.

```
student = struct('Name','Zhang Yichi','Age',20,'Grade','A+')
student =
  struct with fields:
     Name: 'Zhang Yichi'
      Age: 20
    Grade: 'A+'

student.Grade
ans =
    'A+'
```

We use expression like A.prop1 to access the property called prop1 in structure A.

And in fact we can build an array of tructures using the same function

```
students = struct('Name',{'Zhang Yichi';'Fu Qichen'},'Age',[20;21],'Grade',{'A-
';'A+'})
students =
  2×1 struct array with fields:
    Name
    Age
    Grade

students(2).Grade
ans =
    'A+'
```

In fact there is another thing called class in Matlab. It works like structures but they are somewhat more powerfull, because they can have not only self-contained properties, but also self-contained functions(we usually name it methods) as well. But it is more complex to create a class and use it. When you learn C++, you will learn the usage of classes in C++.

P.S.

From this chapter, I will leave some practice questions for you and you can try to solve them. You can find them in the "Practices" directory. I will also post the answers for the questions.