MATLAB/GNU OCTAVE

Dongzhe Chen
University of Arizona
dongzhechen@email.arizona.edu

Peilin Feng
University of Arizona
peilinfeng@email.arizona.edu

April 30, 2020

ABSTRACT

Matlab/Octave is one of the high level programming languages which can help to perform lots of numerical experiments. The design of octave not only has lots of similarities like java python or c, but also some differences which are bettering solving mathematics problems that makes this language special and fun.

1 Introduction

GNU Octave was initial released at 1988, the developer were John W. Eaton and many other programmers. The language contains convenient methods to handle mathematics problems and so on.

2 History

2.1 why was the language designed

Octave helps in solving linear and nonlinear problems numerically, and for performing other numerical experiments using a language that is mostly compatible with MATLAB.

2.2 who designed it

Developer: John W. Eaton and many others

2.3 what is its current status

It is a good language for now and still being used by many programmers.

3 Control Structures

There are many control structures inside Matlab. Just like many other basic language Java, C, python. Matlab has same control structures like for loop, if and else, while loop and so on. There are some similarities and differences.

The similarities between them is the structure looks the same, for loop, if else, and while loop are all having the same structure like Java or C. The unique part of Matlab is there is an extra element 'end' after each control structure. GNU Octave Beginner's Guide [8] has showed an example of if and else structure.

3.1 if and else

if condition 1

do something (body)

elseif condition 2

    do something else (body)

else

    do something else if if condition not met (body)

endif

The example above shows exactly how an if and else control structure looks in Matlab. The only unique part is the endif, by using the 'endif' toward the end of the structure declares that the if and else statement ends here. This unique 'end' not only helps to tell the structure ends here, but also can be used in many different structure. There are also 'endwhile', 'endfor' and so on. It will help to tell programmer when does the specific structure ends. There are some more examples shows while loop and for loop.

3.2 for loop

for i = 1 : 20

    fprintf(i);

endfor

the above for loop example shows a result of 1 to 20. The unique part of this is there is an 'endfor' statement to declare that the for loop ends here.

3.3 while loop

i = 1;

while i < 11

    fprintf(i);

    i++;

endwhile

The above while loop example will give a result of 1 to 10. The unique shows 'endwhile' to delcare this while loop ends here.

4 Data Types

4.1 Single

Single is just like the integer type in java or c. Compare to double, it requires less storage space but has a smaller range. The range is between $3 * 10^{38}$.

An example code :

N = 1;

fprintf('Here shows single type, N = %d', N);

end

The output will be "Here shows single type, N = 1" Just like any other structure, there is and 'end' to show the function ends here.

4.2 Double

Double is just like the double type in java or c. Compare to single, it requires more storage space but has a larger range. The range is between $3 * 10^{300}$.

An example code:

D = $3 * 10^{300}$;

fprintf('Here shows double type, D = %f', D);

end

Just like single, the double is also to declare numeric numbers but with a much bigger size.

4.3 Logical

Just like any other computer science language, matlab's logical uses 1 and 0 to represent true and false. And example :

T = 5 > 3 ;

F = 5 < 3 ;

end

It will give an output of true. Sine 5 > 3 does right and 5 < 3 does wrong.

4.4 Char

'Hello' is an prefect example of Char in matlab. Just like string in other language.

4.5 Cell array

cell array can be represented as 2D array. An example could be :

M = [[zeros(2,5)] ,[ones(2,1)]; 1:5,1;0:2:8,1;[8,7,2,5,9,1]];

end;

output will be :

0 0 0 0 0 1

0 0 0 0 0 1

1 2 3 4 5 1

0 2 4 6 8 1

8 7 2 5 9 1

We can divide this example into multiple cases.

[zeros(2,5)] will return ans =

0 0 0 0 0

0 0 0 0 0

it identifies that the 2 represents 2 rows, and 5 represents 5 columns. The previous zeros shows that the number is 0.

Furthermore, just like [zeros(2,5)], [ones(2,1)] will do the same thing.

[1:5,1; 0:2:8,1] will give us =

1 2 3 4 5 1

0 2 4 6 8 1

It shows that 1:5 means from 1 to 5, and ',' means it doesn't end yet. ';' shows one line end here. 0:2:8 means from 0 to 8 but with and increasing 2 at a time.

4.6 Structure

each structure having named fields capable of storing an array of a different dimension and data type. An example will be :

S(1).name = 'Mike';

S(1).class = 'CSC 372';

S(1).finalGrade = 'A';

S(2).name = 'Bob';

S(2).class = 'CSC 372';

S(2).finalGrade = 'B';

end

The above shows how a structure can contain different data data type.

Continue with the same example, after we typing S(1) information into octave, it will show a scalar structure containing the fields : name = Mike, class = CSC 372, finalGrade = A. But once we entered S(2) info into octave too, it will show a struct array with 1x2 with only name class and finalGrade. But once you type S(1), it will give you all the S(1) information, same thing for S(2).

5 Subprograms

5.1 Class

Just like other programing languages, matlab has class as well. The unique part of matlab is that it starts with classdef title in the beginning. For each class, it mainly has two parts. An example [1] will be :

```
classdef someclass
    properties
    endproperties
    methods
    endmethods
endclassdef
```

properties declare the property of this class, and methods declare its methods. Inside methods, it can include different functions. It brings another unique element inside matlab. Octave provides each function with an automatic variable called nargin. Whenever a function is called, the nargin will tell the number of arguments that has been passed into the function. So nargin could be used as a checking condition to see whether if there is valid arugment passed into the function.

Once the class is created, can create a main file that declares the class. For example, if one class named BasicClass has been created, then in main, can declare x = BasicClass(a). Then it will follow the class that you create. The example will be :

```
classdef BasicClass
    properties
        Value
    endproperties
    methods
        function obj = BasicClass(val)
            if nargin == 1
                obj.Value = val;
            endif
        endfunction
        function a = get.Value(obj)
            a = obj.Value;
        endfunction
    endmethods
```

endclassdef

Inside main, it can declare x = BasicClass(1:5)

the output will be 1 2 3 4 5

6 Extra Interesting

6.1 Bar

Bar is a very interesting element in octave, it can easily create an graph. From Octavfe Forge [5] it gives an example of how to use bar.

y = rand(11, 1);

h = bar (y);

set (h, "ydata", sort (rand (11,1)));

title ("bar() graph");

the title declares the title name for the bar graph, from above example, the title is named bar() graph. And the set method will set the bar based on the info you filled out. There are a list of methods to use bar. For example:

:bar(y), : bar(x,y), :bar(...,w), :h=bar(...,prop, val, ...) Produce a bar graph from two vectors of X-Y data, if only one argument is given, y, it is taken as a vector of Y values and the X coordinates are the range 1:numel (y) [5].

6.2 2D plot

Two-Dimensional Plots is another very unique element that matlab/octave has. In octave, it is very easy to generate a 2D plot by using plot(). The plot function allows you to create simple x-y plots with linear axes [4]. One simple example will be :

x = linspace(-2,2,5);

y = $x.^2$;

G = plot(x,y);

as a result, it will show a 2D graph. linspace function returns a row vector when both base and limit are scalars.If one, or both, inputs are vectors, then linspace transforms them to column vectors and returns a matrix where each row is an independent sequence between base(rown), limit(rown) [6]. From above example, linspace(-2,2,5) will return -2, -1, 0, 1, 2.

6.3 3D plot

Just like two-dimensional plot, three-dimensional is also very useful and unique in octave. Similarly to 2D plot, we need to get x value and y value, the difference is that when in 2D plot, we only need to evenly spaced x values, but in 3D plot we need to evenly spaced y values as well. In order to find z, since it is 3D plot. We need spaced on a grid where z is taken of a point (x,y). An example will be:

x = linspace(-2,2,5);

y = linspace(-2,2,5);

[xx,yy] = meshgrid(x,y);

mesh(xx,yy,4-(xx^2+yy^2));

From the example above, [xx, yy] = meshgrid(x,y) will sign values for xx and yy.

xx will be:

-2 -1 0 1 2

-2 -1 0 1 2

-2 -1 0 1 2

-2 -1 0 1 2

-2 -1 0 1 2

yy will be :

-2 -2 -2 -2 -2

-1 -1 -1 -1 -1

0 0 0 0 0

1 1 1 1 1

2 2 2 2 2

7 Summary

Matlab/Octave has been a great tool to deal with numerical experiments among programmers. This language including many same structures like java or python. Just like if and else statement, while loop and so on. The unique different is that it has an 'end' statement at the end of each structure. Matlab also has unique structure that many other programming language don't have. Just like array parts and graph parts. In octave, it is very easy to declare an array and change its elements. Whats more, it is also very convenient to generate graph based on given data in octave. To sum up, matlab/octave has been a convenient and useful language to help many programmers to write their programs.

References

[1] class def events.

[2] Gnu octave: Simple examples.

[3] Matlab - gnu octave tutorial - tutorialspoint.

[4] Two-dimensional plots.

[5] The Octave-Forge Community. Bar, Jan 2017.

[6] The Octave-Forge Community. linspace, Jan 2017.

[7] John Wesley Eaton, David Bateman, and Søren Hauberg. *Gnu octave*. Network thoery London, 1997.

[8] Jesper Schmidt Hansen. *GNU Octave: Beginner's Guide: Become a Proficient Octave User by Learning this High-level Scientific Numerical Tool from the Ground Up*. Packt Publishing Ltd, 2011.

[8] [7] [3] [2] [5] [6] [1] [4]