DEPARTMENT OF CIVIL, ENVIRONMENTAL & GEOMATIC ENGINEERING
INSTITUTE OF STRUCTURAL ENGINEERING
CHAIR OF STRUCTURAL MECHANICS
STRUCTURAL IDENTIFICATION & HEALTH MONITORING

AN INTRODUCTION TO MATLAB

Abstract

This document aims at introducing MATLAB's environment. The most important features of the software are described and some guidelines about its usage throughout the course are given.

Contents

1 Introduction

MATLAB is a high–level technical computing language and an interactive environment for developing numerical computation and algorithms, as well as for analyzing and presenting data. Using MATLAB, we can solve technical computing problems faster than with traditional programming languages, such as C, C++ and Fortran.

MATLAB can be used in a wide range of applications, including signal and image processing, communications, automatic control, testing and measurement, financial modeling and analysis, and computational biology. Also, there are additional Toolboxes (collections of specialized MATLAB commands) that extend the MATLAB environment to solve particular classes of problems in many different application domains.

MATLAB provides a number of features for documenting and sharing our work. We can integrate our MATLAB code with other languages and applications, and distribute our algorithms and our applications. Such characteristics include, among others:

- High level language for programming technical calculations.
- Development environment for managing code, files and data.
- Interactive tools for iterative design and problem solving.
- Mathematical tools for statistical analysis, Fourier analysis, optimization and numerical analysis.
- Commands for two–dimensional and three–dimensional graphs.

2 MATLAB's Main Parts

MATLAB consists of the following main parts:

2.1 Desktop Tools and Development Environment

This part of MATLAB is the set of tools that help us use it productively. It includes: **MATLAB Desktop**, the **Command Window**, the **Editor**, the **Code Analyzer**, and browsers for viewing help (**MATLAB Documentation**), the **Workspace** and folders.

2.2 Library of Mathematical Functions

This Library is a large collection of computational algorithms ranging from elementary functions like sum, sine and cosine, to more sophisticated functions such as the matrix eigenvalues Fast Fourier Transform.

2.3 Programming Language

The MATLAB language is a high–level matrix language to control flow, functions, data structures, input / output and object–oriented programming. It allows to create programs that do not intend to use again, yet also to create complex application programs intended for reuse.

2.4 Graphics

MATLAB has extensive functions for displaying vectors and matrices as graphs. It includes high–level functions for two–dimensional and three–dimensional data visualization, image processing and animation. It also includes low–level functions that allow you to completely customize the appearance of graphics.

3 Operations and Variables

As mentioned, MATLAB functions both as a computing environment and as a programming language. The basic tool, under which import / export data and call commands, is the **Command Window**. Generally, we should remember the following key points:

- The basic data elements of MATLAB are arrays.
- Variables in capital letters are different from variables with small letters. For example, the variable B is different from the variable b.
- In general, MATLAB does not recognize commands in capital letters.

Elementary operations and variables are performed / defined as follows:

>> 4 * 25 - (2 - 1/2) + 5	basic operations
>> x = sqrt(2)/2	a variable named x, assigned with the value $\sqrt{2}/2$
>> y = x * 180/pi	a variable named y, assigned with the value $180/pi$
>> z = a + b*i	a variable named z, assigned with the value a+b*i

4 Matrices

4.1 Matrix Creation, Operations

The following lines show the creation of matrices and operations among them.

>> A = [1 3 4 5 12 ; 43 4 6 1 7]	a $[2 \times 5]$ matrix
>> B = [8 4 3 6 7 ; 76 89 5 1 2]	a $[2 \times 5]$ matrix
>> C = A + B	matrix addition
>> C = A - B	matrix subtraction
>> C = A ./ B	element–wise matrix division
>> C = A .* B	element–wise matrix multiplication
>> C = A' * B	matrix multiplication
>> Z = [1 2 3 4 5]; or Z = 1:5;	definition of a row vector
>> Z = [1 2 3 4 5]';	definition of a column vector
>> D = Z * Z';	matrix from vector

4.2 Indexing

To extract a particular element of a matrix use the syntax A(i,j), where i refers to the row and j to the column of the matrix that is stored in the variable A. For example:

>> E = [1 3 4;2 -1 3;4 6 7;5 1 -9];	a $[4 \times 3]$ matrix
>> e1 = E(2,1)	definition of e1 variable, equal to 2
>> e2 = E(:,1)	definition of e2 variable, equal to the first column of E
>> e3 = E(1,:)	definition of e3 variable, equal to the first row of E
>> e4 = E(2:3,1:2)	definition of e4 variable, equal to a submatrix of E
>> e5 = E(1:2:4,1:2:end)	definition of e4 variable, equal to a submatrix of E

4.3 Basic Commands

We mentioned that MATLAB contains a huge library of ready–made tools, which perform different functions. The most important of these tools are called MATLAB commands. Let's look at some:

>> mean(Z)	mean value of Z
>> std(Z)	standard deviation of Z
>> max(Z)	max value of Z
>> min(Z)	mim value of Z
>> sum(Z)	sum of the elements of Z

The same commands are also applied to matrices (e.g. mean(E), max(E), min(E), etc.), but with different results.

4.4 Specialized Matrix Functions

Apart from creating matrices with the aforementioned ways, MATLAB enables us to construct some elementary matrices using corresponding commands:

>> zeros(m,n)	a $[m \times n]$ matrix filled with zeros
>> ones(m)	a $[m \times m]$ matrix filled with ones
>> eye(m)	a $[m \times m]$ unit matrix
>> rand(m)	a $[m \times m]$ pseudorandom matrix

5 Polynomials

A set of MATLAB commands deal with polynomials. For a polynomial of the form $a_0 \cdot x^n + a_1 \cdot x^{n-1} + \cdots + a_n$ the following apply:

>> p = [a0 a2 . . . an];	vector of polynomial coefficients
>> p = [1 -12 10 25 16];	defines $x^4 - 12 \cdot x^3 + 10 \cdot x^2 + 25 \cdot x + 16$
>> p = [1 0 0 25 16];	defines $x^4 + 25 \cdot x + 16$
>> r = roots(p);	roots of the polynomial p
>> pp = poly(p);	create a polynomial from its roots

6 Two–Dimensional Plots

One of the most powerful features of MATLAB is its ability to make graphs with great ease. The following lines make the graph of the function $y(t) = sin(t)$

>> t = linspace(0,2*pi,30);	vector of 30 elements, equally spaced from zero to $2 \cdot pi$
>> y = sin(t);	calculates $sin(t)$
>> plot(t,y)	creates the diagram $y = f(t)$
>> grid	places grid on the diagram
>> xlabel('Variable t')	creates a title for the x–axis
>> ylabel('Variable y')	creates a title for the y–axis
>> title('y=sin(t)')	creates a title for the diagram

7 About MATLAB Desktop

7.1 The Workspace

MATLAB Workspace is a graphical interface that allows us to see and change variables / values. The command that opens the Workspace is workspace. For each variable or object, the Workspace provides the following information:

- The name.
- The class (in icon).
- The value.
- Some statistical quantities, where applicable.

Apart from workspace, there are several other commands that manage variables in the MATLAB Workspace. Some of them are listed below:

>> clear	removes all variables from the Workspace
>> whos	lists all the variables in the current Workspace
>> delete	deletes file(s) from current directory

7.2 The Current Folder

After MATLAB start a toolbar is displayed, which indicates the **Current Folder** (or **Current Directory**). We can change this folder with any folder that contains files related to MATLAB. In any case, there are some commands for the management of folders and files. Such are:

>> save filename A	stores variable A in a file named filename
>> save filename	stores all current variables in a file named filename
>> cd	changes current directory
>> load	loads a file in workspace
>> dir or ls	lists the files of the current directory
>> which filename	prints the address of a file called filename

7.3 The Documentation

One of the most useful tools in the MATLAB Desktop is the Documentation. The way in which it is designed and accessed makes almost unnecessary to purchase books for learning the software. The Documentation Browser is shown by typing doc in the Command Window. Other very useful commands are:

>> help	displays help text in Command Window
>> help functionName	displays syntax and description for the function functionName
>> doc functionName	displays the reference page for functionName in the Help browser
>> demo	accesses product demos via Help browser

8 MATLAB's Editor

MATLAB provides the ability to store entire sequences of commands in a special file (with extension *.m), so that we don't have to type them any time. For this reason (and not only, as we will see the following), the

Editor is another important tool of MATLAB. The Editor opens by typing edit in the Command Window.

In the following example, we open the Editor and we re–write the same commands that we used to plot the function $y(t) = sin(t)$

```
% program mytest
t = linspace(0,2*pi,30);
y = sin(t);
plot(t,y),grid
xlabel('Variable t')
ylabel('Variable y')
title('My first plot')
```

We store the file with name myTest.m, we type in the Command Window myTest and the same diagram appears, as before.

9 Notes

1. The best way to start using MATLAB is by navigating across the Help Browser. Type doc matlab on the Command Window and the Help Browser will open at the specified location.
2. When you are at the Help Browser, observe that the left side contains the navigation tree and the right side contains the documentation.
3. Navigate across the contents of MATLAB (you don't have to look at other Toolboxes) and get a first insight about its capabilities. You should start with the **Getting Started** Section.
4. One of the very important aspects of MATLAB is the manipulation of matrices. You should become familiar with matrix commands and operations as quickly as possible. To this, it is important to understand exactly how indexing works. MATLAB contains analytical documentation for indexing, located in **MATLAB** → **Language Fundamentals** → **Matrices and Arrays** → **Array Indexing**.

10 Exercises

1. Create a script with name resetAll.m which will perform the following operations, on the order they appear:
   (a) It will clear all the variables of the Workspace.
   (b) It will closes all opened Figure windows.
   (c) It will clear the Command Window.
2. (Hankel matrices)
   (a) Create a variable v, equal to a pseudorandom vector of size $[5 \times 1]$.
   (b) Create a variable u, equal to a pseudorandom vector of size $[5 \times 1]$. The first element of u must be equal to the last element of v.
   (c) Create a variable H, equal to a Hankel matrix of size $[5 \times 5]$. The first column of H must be equal to v and the last column of H must be equal to u. (Note: you must find the command that creates Hankel matrices)

(d) Create a variable HH, equal to the transpose of H. What are the differences between the two matrices? (Note: you must find the command that calculates the transpose)

(e) Observe the matrix H and derive a relation among its elements.

3. (Statistics)

(a) Create a variable p, equal to a pseudorandom vector of size $[100 \times 1]$.

(b) Create a variable q, equal to a pseudorandom vector of size $[100 \times 1]$.

(c) Multiply p and q appropriately, to create a matrix of size $[100 \times 100]$. Store this matrix to the variable A.

(d) Find the absolute minimum and the absolute maximum of A.

(e) Find the line of A with the largest mean value. Find the line of A with the smallest mean value.

(f) Find the line of A with the largest variance. Find the line of A with the smallest variance.

(g) Find the column of A with the largest mean value. Find the column of A with the smallest mean value.

(h) Find the column of A with the largest variance. Find the column of A with the smallest variance.

(i) Calculate the sum of all elements of A using the minimum number of commands.

4. (Indexing) For the matrix A that you created in the previous step:

(a) At the Command Window, display the element A(10,84).

(b) At the Command Window, display a submatrix of A that starts from the element A(10,84) and has size $[12 \times 6]$.

(c) Create a variable A1 that contains the even columns of A (the 2nd, the 4th, the 6th, etc).

(d) Create a variable A2 that contains the odd rows of A (the 1st, the 3rd, the 5th, etc).

(e) Create a variable B, equal to a matrix of size $[10 \times 10]$, the elements of which are equal to those of a submatrix of A that starts from the element A(73,37).

5. (Editor) Create a script with name artificialEarthquake.m and write the following code:

```
% program artificialEarthquake
t = (0:0.01:(60-0.01))';
N = length(t);
A = 1; B = 20; C = 1000;
x = A*randn(N,1);
n = (A/B)*randn(N,1);
a = [zeros(C,1);hann(1500);zeros(N-1500-C,1)];
y = a.*x + n;
plot(t,y),grid
xlabel('Time (s)')
ylabel('Acceleration (m/s^2)')
title('Artificial Earthquake Acceleration')
```

(a) Explain the action of every line.

(b) Modify A, B and C and observe the results.

(c) Select values for A, B, C and calculate the peak ground acceleration.