An Introduction to Data Analysis and MATLAB

Peter Lamb

Week 1

Data Analysis

Data analysis, an unavoidable part of all experimentation and research, is often the postgrad student's greatest consumer of time and a major source of frustration. Consideration of analysis methods and the available tools is often left to the last minute when important decisions (or mistakes!) have already been made.

The aim of this section of the paper is to introduce and build skills and confidence in a number of useful data analysis tools. Such skills are highly sought after in a wide variety of postgraduate and employment opportunities.

Raw Data

Raw data are the original, unfiltered, unprocessed data that you record during the data collection. These data (note: *data* is plural and *datum* or *data point* is the singular form) are important and you should treat them so. Keep these original data files in a safe location and do not touch them. Make copies of these originals and then perform any analysis on the copies. This way, if you make any mistakes during your analysis you can always go back, make new copies of the originals and begin again. Raw data files should never be edited, this is not only poor practice but also unethical.

Naming convention

Two students, A and B, both test four participants twice, once before an intervention session and once after. The data file names from the two are shown below:

Student A	
Jonesy1.dat	Jonesy2.dat
Mike-T1.dat	Mike2.dat
Richard_1.dat	Rich2.dat
Bill_1.dat	Bill_2.dat

Student B	
JM_T1.dat	JM_T2.dat
MF_T1.dat	MF_T2.dat
RB_T1.dat	RB_T2.dat
WN_T1.dat	WN_T2.dat

To describe the naming convention for Student B you would say: use two letters, one for the first letter of the first name and one for the first of the surname, an underscore '_' and then the test number i.e. T1 for test 1 and T2 for test 2. Now look back and try to describe the convention for Student A.

Why is this important? File names are often a useful way of recording important information about what went on in the data collection session. You may include information like; participant name/id, experiment type/condition, trial number etc. For example, the name 'PP1_G3_T7.dat' could be used to indicate Participant 1, Group 3 and Trial 7. During the exercise in this module you will learn how to create automated data analysis processes to remove repetitive steps. You will learn how to write instructions for a computer to follow to process the data for you (you may be familiar with this already when using macros in Excel). In the case of Student B, you could instruct the computer to look at the first two characters to get the name of the subject and the fifth, or last, character to get the test number. You would have a difficult job instructing anyone how to extract information from the data file names used by Student A's convention!

Take note

When you are collecting data it is useful and often critical to record all available information about the testing session. Record information such as sample rate, amplifier and filter settings, data channel names, equipment serial numbers, tester names, session date and time etc. All this information can be used to track down or exclude any errors/anomalies that may occur during data collection.

Analysis software

After the data have been collected there will almost always be a number of processes that need to be completed before the results of the investigation can be presented. There could be multiple participants, trials, conditions and variables that may need to be cleaned, filtered and averaged before any statistical analysis can be used to determine an outcome. The data may also need to be collated from a number of separate locations and various different file formats.

These processes will undoubtedly be far too onerous to be done manually and you will need to find a software package to assist you. A familiar and easy to use program such as Microsoft Excel is often the first choice for most analysis procedures. This can be sufficient if the requirements are basic and the amount of data is minimal. However, it can be hugely time consuming and computationally inadequate for larger data sets and more complex analysis procedures.

There are other software programs available that are written specifically to facilitate powerful, repeatable and rapid data analysis. Unfortunately, as always, all of the benefits are not without some disadvantages. These software programs often require a period of learning before they can become useful. However, the goal of this module is to show that it is worth it!

MATLAB – The Language of Technical Computing

The name MATLAB (note: always written in all caps) stands for Matrix Laboratory and as the name suggests almost all the work done in MATLAB is based upon the manipulation of matrices. Therefore, it is important to get a basic understanding of what a matrix is, why they are so useful and how MATLAB uses them to tackle some applicable data analysis.

Scalars, Vectors, Matrices, Arrays and Structures

In MATLAB a basic **matrix** is a rectangular array of numbers, arranged in rows and columns (rows × columns). The simplest matrix, a single number in a 1×1 (1 row by 1 column) arrangement, is known as a **scalar**. A single row or column of values is known as a **vector**.

A scalar (1x1):

$$3$$

A row vector (1x5):

$$\begin{bmatrix} 3 & 12 & 5.1 & 9 & 7.02 \end{bmatrix}$$

A column vector (5x1):

$$\begin{bmatrix} 3 \\ 12 \\ 5.1 \\ 9 \\ 7.02 \end{bmatrix}$$

A matrix (4x3):

$$\begin{bmatrix} 3 & 12 & 5.1 \\ 12 & 1 & 11.04 \\ 5.1 & 13 & 3.8 \\ 9 & 7.5 & 8 \end{bmatrix}$$

You will be familiar with displaying data in rows and columns when using a spreadsheet, such as Microsoft Excel. Rows and columns are a very effective way of displaying data in readiness for further analysis. Table 1 shows the lap split times and maximum and minimum speeds for a 10-lap running race.

Lap	Lap Time (s)	Maximum speed (ms^{-1})	Minimum speed (ms^{-1})
1	67.4	7.2	2.4
2	68.2	6.3	5.5
3	70.1	6.1	5.3
4	70.4	5.9	5.1
5	69.3	6.3	5.7
6	71.2	5.8	5.0
7	73.8	5.7	4.9
8	75.2	5.5	4.5
9	73.4	5.6	5.2
10	71.1	8.7	4.8

Table 1: Lap times and speeds for a 10-lap running race

The data could be stored in a matrix as shown in Figure 1.

$$\begin{bmatrix} 1 & 67.4 & 7.2 & 2.4 \\ 2 & 68.2 & 6.3 & 5.5 \\ 3 & 70.1 & 6.1 & 5.3 \\ 4 & 70.4 & 5.9 & 5.1 \\ 5 & 69.3 & 6.3 & 5.7 \\ 6 & 71.2 & 5.8 & 5.0 \\ 7 & 73.8 & 5.7 & 4.9 \\ 8 & 75.2 & 5.5 & 4.5 \\ 9 & 73.4 & 5.6 & 5.2 \\ 10 & 71.1 & 8.7 & 4.8 \end{bmatrix}$$

Figure 1: The data from the Table 1 stored in a 10x4 matrix

This matrix, with rows and columns, is a two-dimensional (2D) matrix. It is possible to have multi-dimensional matrices and when we have more than three dimensions it can become very hard to visualise these arrangements. For a 3D

matrix, in a $3 \times 3 \times 3$ arrangement, the elements could be visualised as a simple rubix cube as shown in Figure 2. The three dimensions could be described as rows, columns and *blocks* of these rows and columns. The highlighted element would be in row 2, column 3 of block 2 (taken from the nearest top left corner).

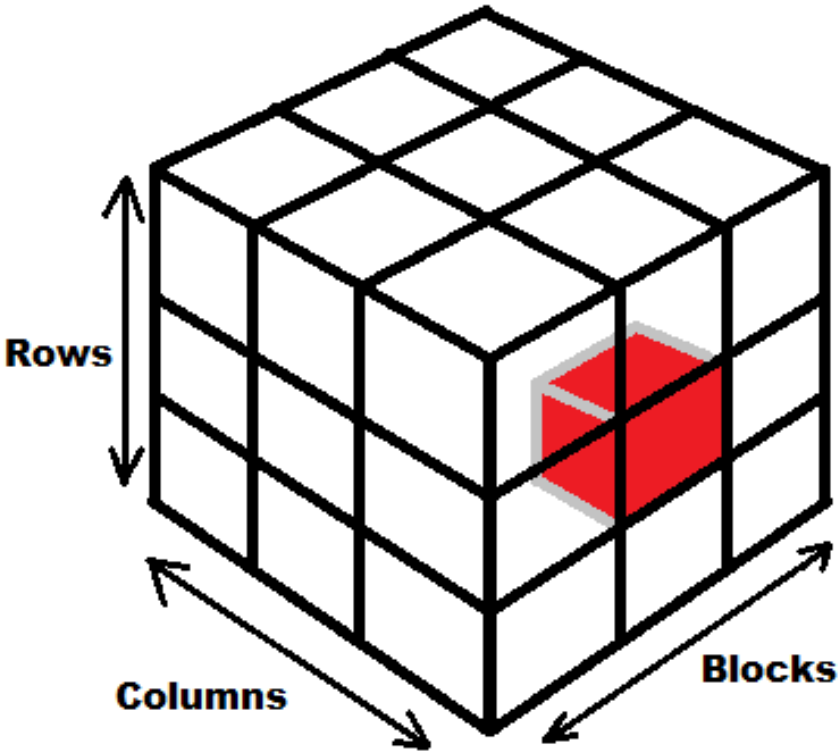

Figure 2: Visualising a 3-dimensional matrix

Figure 3 shows a possible visualisation of a 5D matrix. We could imagine a collection of cubes, arranged themselves in rows and columns - a $2 \times 3 \times 3 \times 3 \times 3$ matrix. To describe the highlighted element we could say; within the cube in row 2, column 2, take the element in row 1, column 3 of block 1.

It is possible to have a multi-dimensional dataset that is made up from simple parameters you may use in a standard research project. For example: 10 participants are in a study with 7 conditions; 8 trials are completed for each condition in 6 separate testing sessions; during each trial, 11 signals are recorded consisting of 4,000 data points each. This could produce a 6D matrix in a $10 \times 7 \times 8 \times 6 \times 11 \times 4000$ arrangement but it is not worth imagining how this might look!!

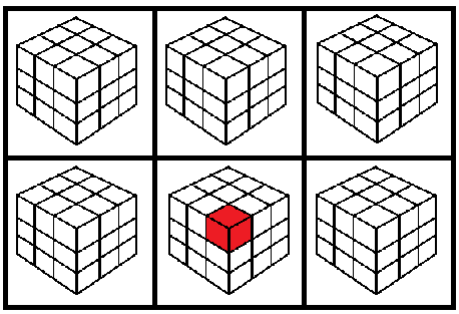

Figure 3: Visualising a 5-dimensional matrix

In MATLAB, an **array** is a collection of variables (these could be text, scalars, vectors, matrices, etc.) as opposed to a matrix which is a collection of numbers. For example, an array could be used to store a number of matrices. Figure 4 shows an array containing three items.

$$\left\{ \begin{bmatrix} 12 & 5 & 7 & 3 \\ 2 & 3 & 20 & 9 \\ 15 & 6 & 8 & 11 \\ 4 & 1 & 21 & 18 \end{bmatrix} \begin{bmatrix} 2 & 13 & 9 & 7 \\ 12 & 11 & 20 & 16 \\ 4 & 9 & 8 & 8 \\ 17 & 6 & 19 & 1 \end{bmatrix} \begin{bmatrix} 2 & 12 & 4 & 1 \\ 17 & 14 & 2 & 21 \\ 6 & 19 & 10 & 9 \\ 22 & 5 & 16 & 18 \end{bmatrix} \right\}$$

Figure 4: An array containing three 4×4 matrices

A **structure** is probably the most complicated item you will come to use in MATLAB. A structure is a collection of related data, often in different forms (scalars, text, matrices etc). For example, a subject of an experiment could be seen as a structure, 'SUBJECT'. Associated with this subject is their name (text), their age, height and weight (1×3 matrix) and their test data of 10 samples of 5 variables (10×5 matrix). This structure is shown in Figure 5 as an example.

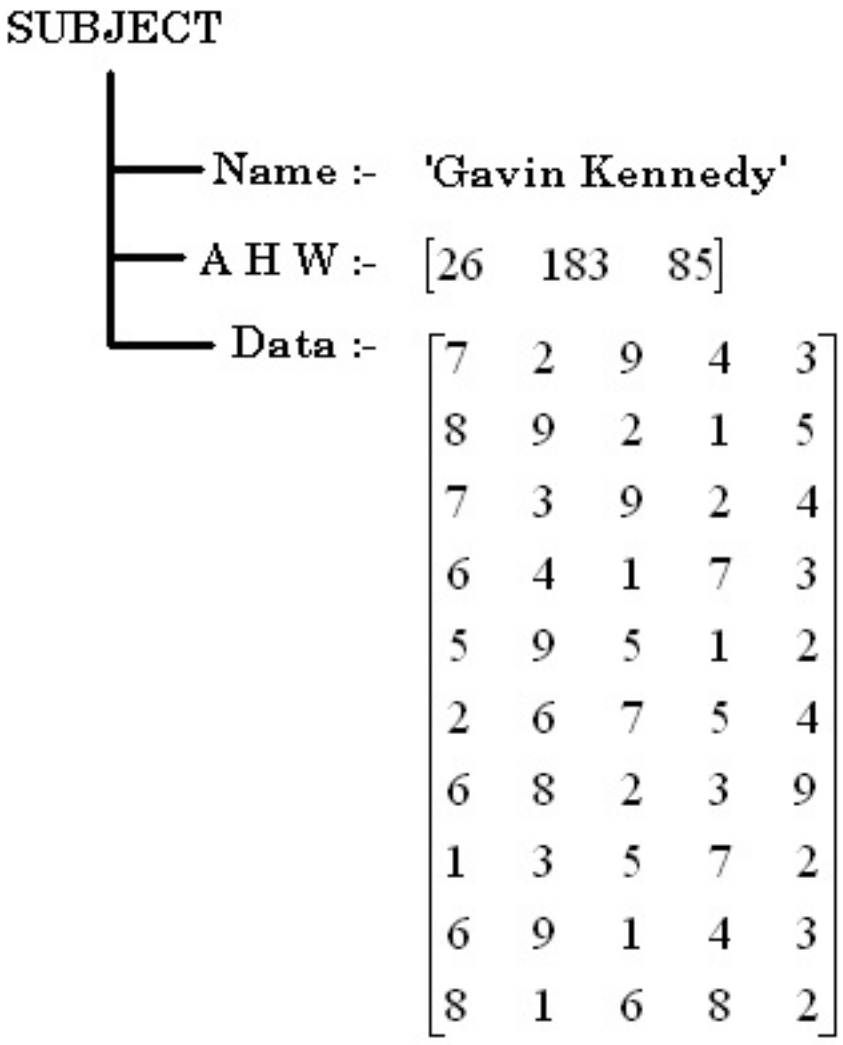

Figure 5: An example of a structure

A **structure** variable can also be arranged to store a collection, or array, of individual structures. Figure 6 shows the array of structures, called SUBJECT, containing two items.

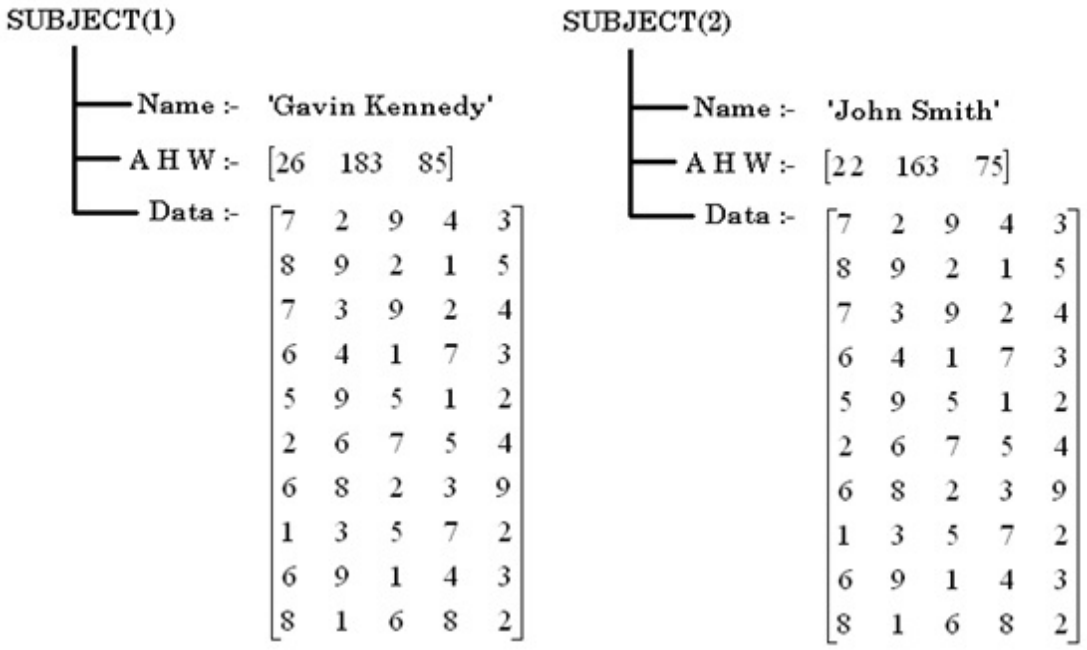

Figure 6: Two structures that make up the structure array called SUBJECT

An introduction to MATLAB

MATLAB has a vast collection of functions and resources, making finding the one that may be useful to you seem very daunting. However, there are a number of different ways of getting to grips with what is available.

The product help is a great way of browsing through the catalogue of resources. Open the **Product Help** by selecting the **Help→Product Help** from the MATLAB menu. From here you can search by typing relevant key words in the search box or simply browse through the functions in the Contents listing. Each toolbox has its own branch which contains **Getting Started** guides, **Functions**, **Examples** and **Demos**. Within the **Function** collection the functions are grouped into a number of different categories.

Another, less formal, option is search popular internet forums. General queries / searches, such as "how to index vector elements in matlab", will likely result in you being directed to the online MATLAB documentation; however, more specific queries will likely bring you to Forums such as stackoverflow, stackexchange or MATLAB's own forum – matlabcentral. matlabcentral also hosts many user-

contributed functions on their File Exchange that can be downloaded and run for your own purposes.

In addition to the built-in MATLAB functions you will begin to create your own custom functions and scripts. It is a good idea to create a directory somewhere on your computer specifically for your custom functions.

Please watch the following MATLAB demonstration videos before continuing. These can be found by searching for "Getting Started with MATLAB" in the 'Search Documentation' box in the top right corner. You should see the following videos listed under **Videos**.

1. Getting Started with MATLAB
2. Working in the Development Environment