Dyson School of Design Engineering

Imperial College London

DE2.3 Electronics 2

Lab Experiment 1: PyBench and Matlab

(webpage: http://www.ee.ic.ac.uk/pcheung/teaching/DE2_EE/)

Objectives

By the end of this experiment, you should have achieved the following:

- How to effective use an electronic logbook for this module;
- How to solder components onto a printed circuit board;
- Completed the construction of the experiment board which will be used throughout the practical sessions of this module;
- Set up the Matlab environment on your personal PC to support the practical work required for this module;
- Learn some basic signal processing and graph plotting features in Matlab;
- Appreciate the time domain and frequency domain view of signals.

Before you start

If you have not done soldering before, I strongly recommend you to watch the following YouTube video: https://www.youtube.com/watch?v=QKbJxytERvg

You are also required to keep an electronic logbook for all the laboratory sessions on Wednesday morning. You will find a useful instruction on how to keep an electronic logbook at the end of this document. While your logbook will not be formally marked, you are expected answer some questions during your oral examination by referring to the electronic logbook.

The PyBench Board

In order to support this module, a dedicated printed circuit board (PCB) was designed. It is given the name "**PyBench**" because the board uses a variation of **Py**thon, the MicroPython, on the Pyboard, in order to implement an Electronics Work**Bench**. A Python programme running on the Pyboard will be used to turn the PyBench board into a signal generator and data acquisition system. PyBench is designed to work with a library class **PyBench.m** running on the PC. Communication between the host PC and the PyBench board is via USB link working as a **virtual communication port** (VCP).

The PyBench board has many sockets, which will host different electronic modules. Some of these modules are the ones that you have already used last year in Electronics 1. Others are new to you.

Building the PyBench Board

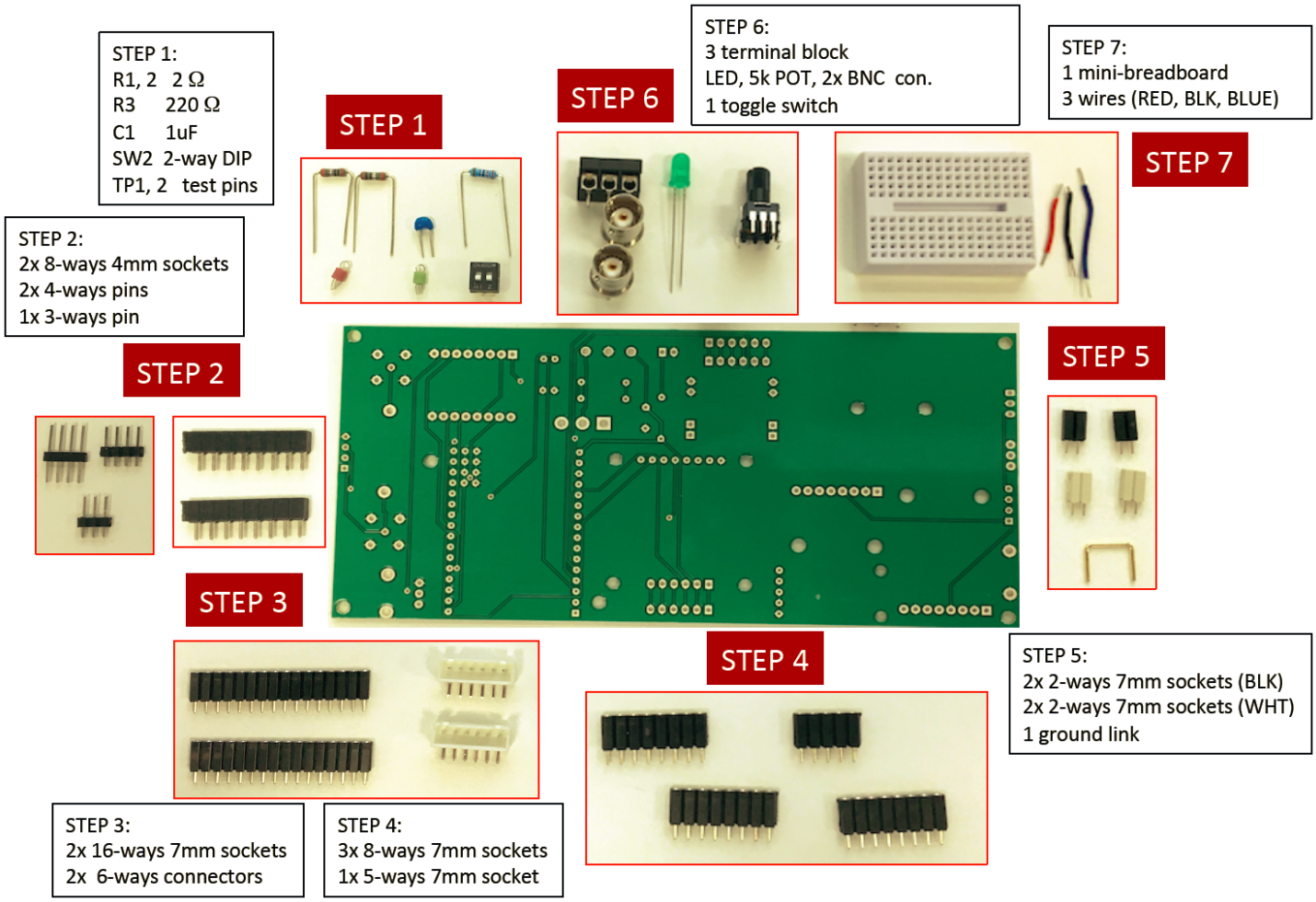
Before you start doing anything, make sure that the PyBench board is the right way up. In making the PyBench board, I made a mistake that resulted in the silkscreen layer being turned off. As a result, all the beautiful labels (in white), which should have been on the component side of the board, were not printed.

A common and serious mistake that most student made is to solder components or sockets onto the PCB at an angle. If the component or socket has more than two pins, once solder wrongly, it is a nightmare to fix. To avoid this common mistake:

1. Hold the component in place using some Blu-Tack. Make sure the Blu-Tack is not touching any hold part of the component.
2. Solder ONLY one pin to the PCB.
3. Check that the component is fixed at the straight angle (i.e. the pin or socket is orthogonal to the PCB). If not heat up the joint and adjust accordingly.
4. Solder a second pin at the opposite corner or side.
5. Check again that the angle is correct.
6. Solder the rest.

Step-by-Step guide to building the PyBench board

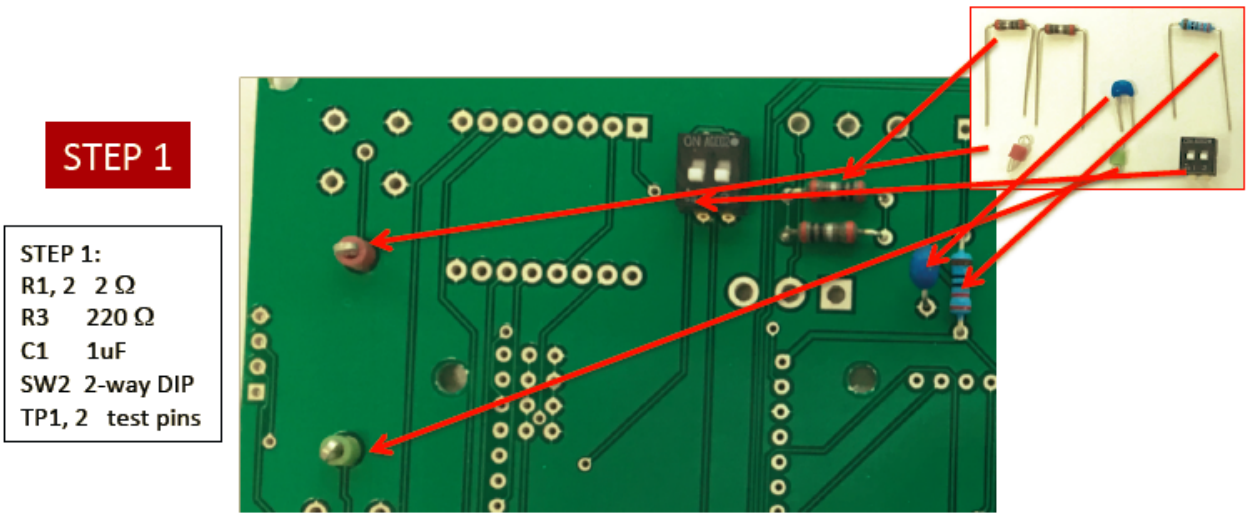
Layout the components for PyBench board as shown below on the laboratory bench.



Follow the steps described below. The principle is simple: **you should build the board from the shortest components to the tallest components**.

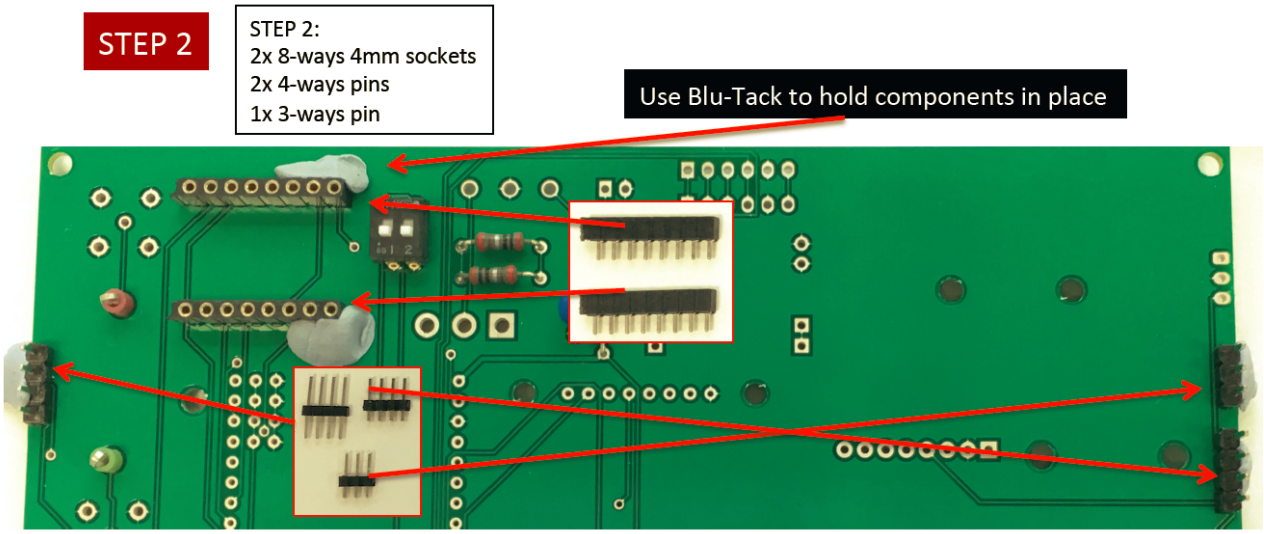
Step 1: resistors, capacitor, test points and DIP switch

You can bend the resistors and capacitor leads slightly to hold the component in place before soldering.



Step 2: Sockets and pins

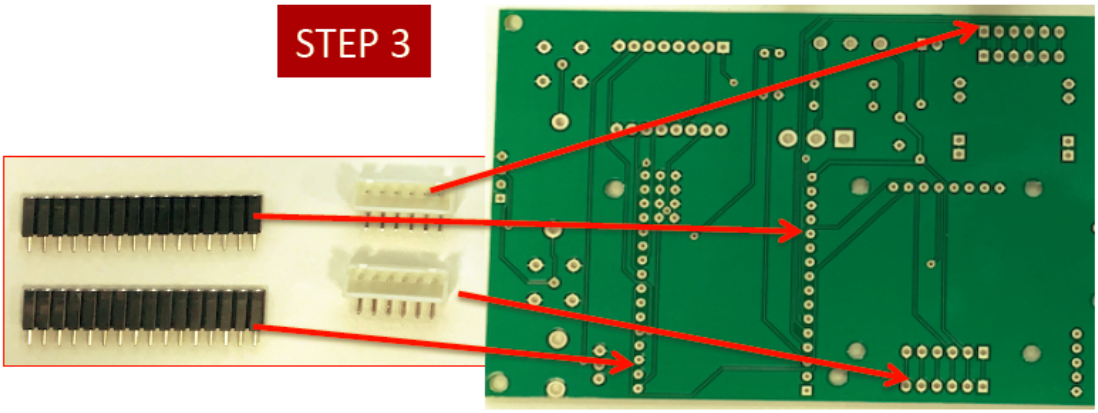
The 8-way low profile sockets may not arrive on time. They were on order for a while now, but were out of stock. If this is the case, I will either provide alternative sockets or we will solder these at the next lab session.



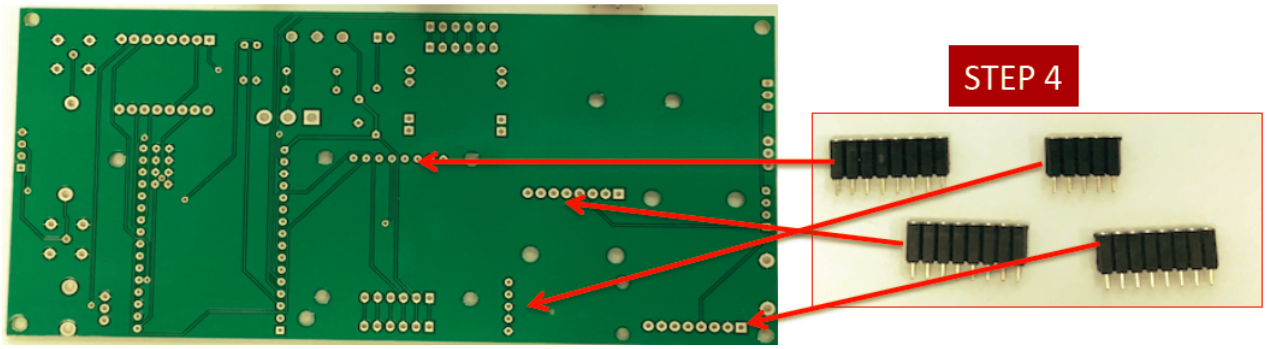
Step 3: Pyboard and motor sockets

Pyboard requires two 16 pin single-in-line sockets. It is absolutely vital that you solder these two sockets precisely at right angles to the PCB. One trick to use is the actually use the Pyboard as a template. First plug the sockets onto the Pyboard pins. Then align the pins to the PCB. Solder only ONE PIN on each socket to the PCB. Check for mounting angle and adjust as necessary before completing the job.

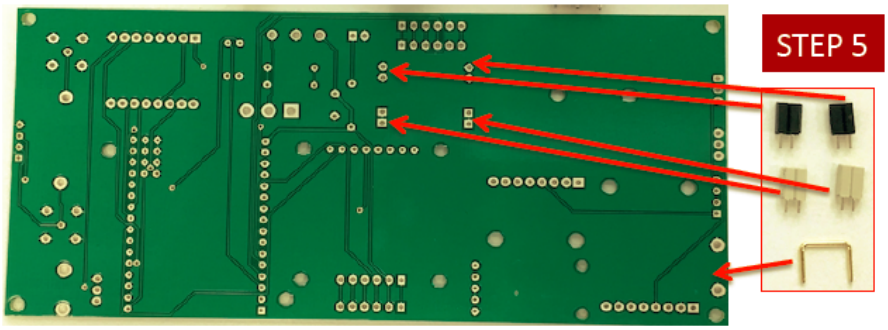
When soldering the white motor connectors, make sure that the notches on the connectors are facing INSIDE of the PCB.



Step 4: OLED, IMU, UART Friend and Microphone amplifier sockets



Step 5: Ground link and DC-DC converter sockets



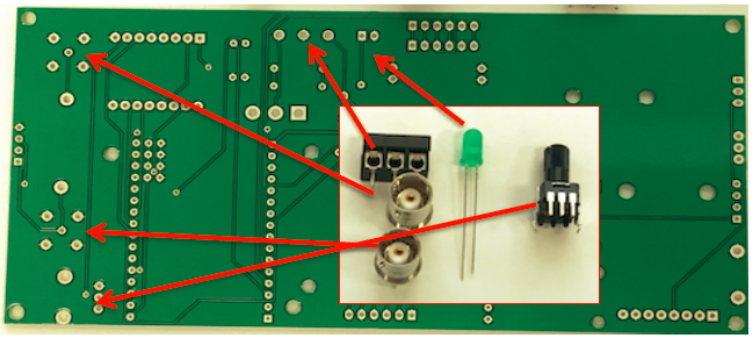
STEP 5:
2x 2-ways 7mm sockets (BLK)
2x 2-ways 7mm sockets (WHT)
1 ground link

Step 6: Power terminal, BNC sockets, LED and Potentiometer

The terminal block should have the connecting holes facing outwards. The cathode of the LED is the side that has a flat edge on the case and has a shorter leg. This terminal should be on SQUARE pad left of the PCB.

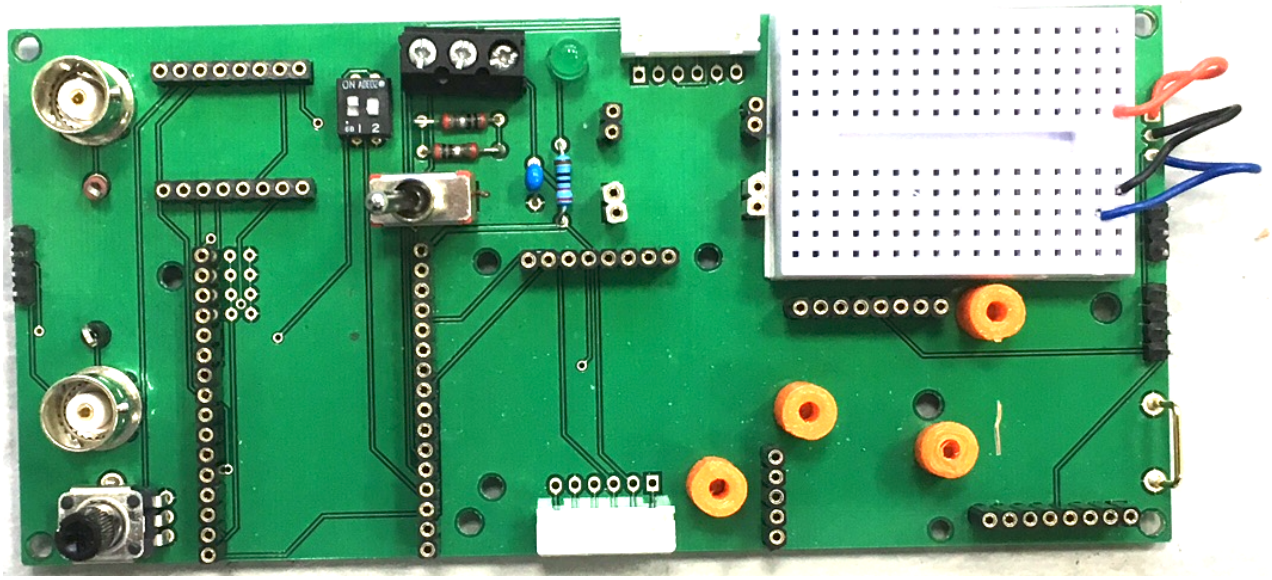
STEP 6

STEP 6:
3 terminal block
LED, 5k POT, 2x BNC con.
1 toggle switch



Step 7: The miniature breadboard and standoffs

Solder the three coloured wires (RED=5V, BLACK=GND, BLUE=3.3V) to the three holes on the PCB as shown. Peel off the backing sheet of the breadboard. Stick this down on the PCB as far left towards the 2-ways sockets as possible (touching would be perfect!). Screw the orange standoffs (four) to the holes indicated. These provide support for the plug-in modules. **Here is a photo of the completed board:**



Step 8: Testing the board

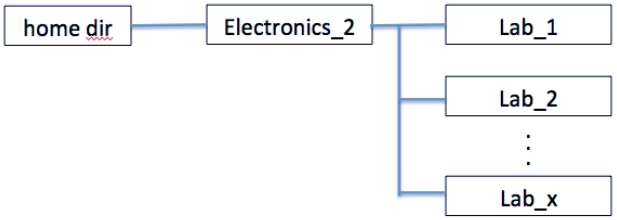
Having completed the board, compare it with the photograph below to check that everything is in place. Take the board to the TEST STATION, where I will run through a diagnostic test programme to verify that your PyBench board is working properly.

Step 9: Mounting the PyBench board

Mount the PyBench board on the black plastic baseboard with the 5mm metal standoffs provide. Congratulations! You have now completed building the PyBench board. From now on, your Lab Group own the corresponding PyBench board for the week. For example, if you are group 5 for Lab 1, you and your partner have the use of PyBench-05 for the entire week until you move to another group the following week.

Getting started with Matlab

For the rest of this lab session, you will be making sure that either yours, or your lab partner's, laptop computer will have Matlab running properly. Furthermore, you will also investigate the ideas covered in Lectures 1 and 2 in this laboratory.



Before you start, I strongly recommend you to create a directory structure for all the Matlab codes you will be creating for this module. A possible structure will looks something like this. Trust me – a little effort now will save you lots of time later.

Exercise 1: Sinusoidal signal generation

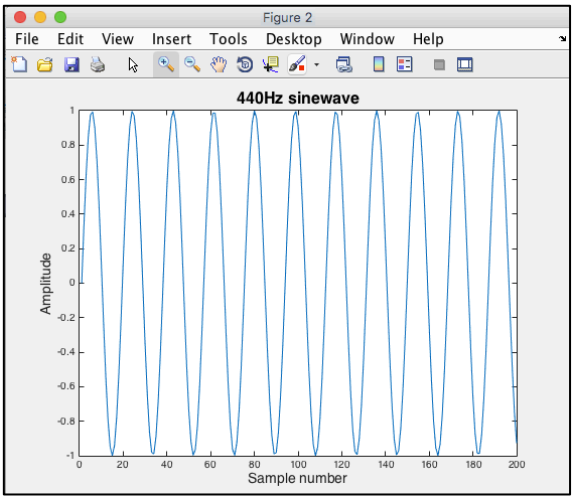
Enter the following function to generate a sinusoidal signal using the filename: **sine_gen.m**. Test the function to produce the plot as shown.

```
function [sig] = sine_gen(amp, f, fs, T)% Function to generate a sinewave
%
%   ....   with a sampling frequency fs for a duration T
%
%   usage:     signal = sine_gen(1.0, 440, 8192, 1)
%
% author: Peter YK Cheung, 17 Jan 2017

    dt = 1/fs;
    t = 0:dt:T;
    sig = amp*sin(2*pi*f*t);
```

Test this function in the interactive mode of Matlab:

```
>> s1 = sine_gen(1.0, 440, 8192, 1);
>> plot(s1(1:200));
>> xlabel('\fontsize{14}Sample number');
>> ylabel('\fontsize{14}Amplitude');
>> title('\fontsize{16}440Hz sinewave');
```



Exercise 2: Spectrum of the signal

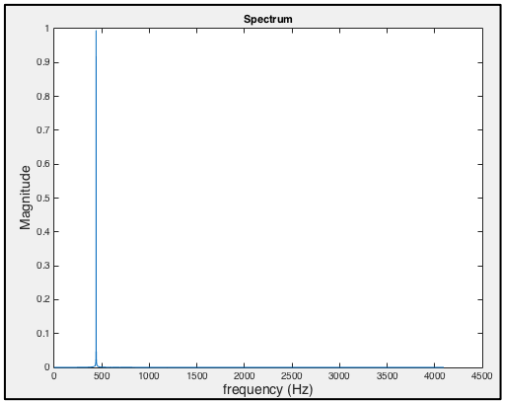
Enter the following function using the filename: **plot_spec.m**. This function uses the Matlab built-in function "fft" to calculate the frequency spectrum of the signal. Don't worry about exactly how it works for now. (I deliberately want you to type the code into Matlab instead of doing cut-and-paste. In this way, there is a higher chance that you will remember some of the syntax of Matlab.)

```
function plot_spec(sig, fs)

% Function to plot frequency spectrum of sig
%   usage:
%            plot_spectrum(sig, 8192)
%
% author: Peter YK Cheung, 17 Jan 2017
    magnitude = abs(fft(sig));
    N = length(sig);
    df = fs/N;
    f = 0:df:fs/2;
    Y = magnitude(1:length(f));
    plot(f, Y/(N)*2)
    xlabel('\fontsize{14}frequency (Hz)')
    ylabel('\fontsize{14}Magnitude');
```

Test this function in the interactive mode of Matlab and you should see the following frequency spectrum plot. You can zoom onto the peak frequency at 440Hz using the button.

```
>> s1 = sine_gen(1.0, 440, 8192, 1);
>> plot_spec(s1,8192);
>> title('Spectrum')
```



Exercise 3: Two tones

Now generate two sinewaves, one at 440Hz (amplitude 1.0v) and another at 1000Hz (amplitude 0.5v), using a sampling frequency of 8192Hz and each having a duration of 1.0 second. Add these together and plot the waveform.

Plot the spectrum of the combined signal.

Exercise 4: Two tones + noise

Assuming that your two tone signal is called “sig”. Create a noisy version of this signal using:

```
noisy = sig + randn(size(sig));
```

Now, plot noisy and its spectrum.

What have you learned from this exercise?

Exercise 5: Projection using dot product

Let us now treat the two sinusoid signal s1 (440Hz) and s2 (1000Hz) as two vectors. You can find the projection of s1 on s2 by computing their **dot product** (also known as **inner product**) in Matlab:

```
dot_product = s1*s2';
```

What value do you get? Now create another sinewave at 441Hz, and find the dot product of s1 on s3. What do you get?

Finally, find the dot_product of (s1 + s2) on s1. What do you get?

I will be explaining all the Matlab code and the ideas behind this laboratory session during the next lecture/tutorial.

Guidelines on the use of electronic logbook

Throughout your study here at Imperial College, you will be required to keep a log of your practical work. In many companies, such practice is mandatory. The purposes for keeping the a logbook are:

1. It helps you to plan your work and, even more importantly, reflect on what you have done. While you are learning, there is a tendency just to "hack" the hardware or software until you get the results you are expecting. Keeping a logbook will force you to tidy-up, think about what you have done, ask yourself questions and finally record your conclusions and thoughts.
2. It helps you to remember what you have done and learned in the future. Spending time "now" will save you time in the future. Indeed, any practical assessment you part-take will almost certain require you to look back on your logbook to find the answers.

You must remember that:

1. Logbook is a personal tool for YOU to assist your learning. It is not the vehicle to get marks.
2. Logbook is NOT a report. It does not need to be tidy, use perfect English, or beautifully laid out. However, an unstructured logbook is also not good – it will be difficult for you to find stuff in it in the future.
3. The most effective way to keep a logbook is "contemporaneous", meaning that you write it as you go along, immediately before or after the practical work. You loose many benefits if you wait until after the lab session to "write it up" because it does not help you in reflection and planning.

In summary a good logbook is much more than a record of what you have done. It is also an effective learning tool that helps your understanding and memory.

Electronic Logbook

While conventional logbook keeping involves pen and paper, modern engineering, particularly with electrical and computer engineering, much of what you do involve programs and electronic information such as screenshots of a scope or computer. Therefore increasingly there is a case for keeping electronic, instead of paper, logbook.

There are various application programmes available to help you to keep an eLogbook. Here we suggest three possible tools in order of skill level required:

Microsoft Word – this is very straight forward, but a Word document only has a sequential or linear structure. It is therefore not as easy as other methods to find the information you need. However, everyone knows how to use Word and therefore there is no overhead in learning this.

Microsoft OneNote – This is a very useful tool from Microsoft designed not only for eLogbook, but for general note taking. It is easy to learn and use, and it is free for all students at Imperial. You may find a good tutorial on OneNote here: https://www.youtube.com/watch?v=WH8U_AhyEJw

Github – This requires the highest level of skills, but has the advantage of combining document formatting, repository and revision control. You need to spend more time and effort to learning and to use Github, but the result is most impressive and satisfying. Here are two helpful tutorials on Github:

https://guides.github.com/activities/hello-world/

https://www.youtube.com/watch?v=0fKg7e37bQE