Faculty of Science, Engineering and Technology

SWINBURNE UNIVERSITY OF TECHNOLOGY

ENG10004
Digital and Data Systems

Project Report

Student Name: Marella MORAD

Student ID: 103076428

Date: 31/10/20

Self-Assessment Details

The following table provides my self-assessment for my individual contributions to the project.

	Few (0-49)	**Minor** (50-59)	**Important** (60-69)	**Major** (70-79)	**Major & Leadership** (80-100)
Self-Assessment (**please tick**)					✓

Declaration

I declare that this report is my individual work. I have not copied from any other student's work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part of this submission been written for me by another person.

Signature: Marella

Design Tasks Details

Project tasks

1. **Project A – Heart Rate Monitor**
   For this project, the main requirement is to process the supplied ECG data samples and produce the heart rate in bpm, beats per minute. The required software needs to fulfil the quest of accurately keeping track of the ECG data samples. Analysis of the samples for peaks would allow for the calculation of the heart rate. To do this, there are several steps that the program needs to address.
   - Plotting the provided ECG data samples.
   - Finding R-peaks and their x-axis coordinates (indices) that will be used in determining the time for a one heart cycle.
   - Finding the time it takes to complete one waveform, which is the difference between two consecutive R-peaks). This can also be referred to as the period of the wave.
   - Calculating the heart rate in seconds first then converting the outcome into bpm and displaying that to the user.
2. **Project B – Heart Rate Analyser**
   The project problem for part B is very similar to part A. This is because the analysis would be done on the same 8 samples of ECG data. In terms of requirements there are two main requirements:
   - Finding the wave attributes, that are the P-wave, Q-peak, R-peak, S-peak and the T-wave and labelling them on the plot.
   - Finding the time intervals, that are the PR-interval, the QT-interval, the QRS-complex and comparing these values to the given normal range values and displaying the outcomes in the command window.

Background research

The project problem is mainly about examining the 8 ECG data samples for either heart rate values or wave analysis. Therefore, it is crucial to understand what ECG waveforms are. An ECG waveform, also known as an EKG or an electrocardiogram, is a graphical representation of the heart's strength and timing (E. Klabunde 2012). The main attributes of an ECG waveform are the P-wave, the QRS complex and the T-wave (see figure 1). For project A, only the R-peaks matter. This is because the heart rate depends only upon the time difference between two consecutive R-peaks (ECG Heart Rate Calculator 2018). For project B, as per the requirements, all the 5 deflections (P-wave, Q-peak, R-peak, S-peak and T-wave) need to be determined and from there the other three intervals the (PR, QRS and QT) labelled on figure 1, calculated as well.

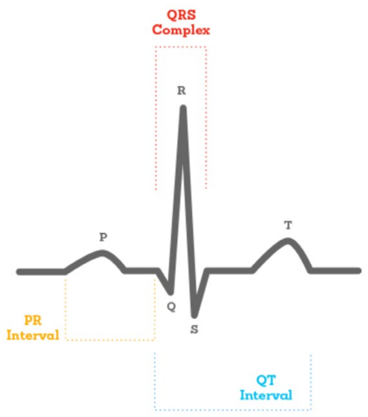


Figure 1: an ECG waveform with the main attributes labeled on it (Alivecor 2020)

After understating the basics of a simple ECG waveform, it is important to be able to apply this knowledge to real ECG samples. The 8 ECG samples provided were recorded using an ECG sensor and then digitised to 360 samples/second (J. Zheng, 2020). However, while recording, the sensors picked some other frequencies that led to noise in the 8 samples. This noise does not affect the functionality of project A, however, it does affect the determination of the required peaks and so the intervals as well (Project B). In order to get the most accurate results possible from the samples, filtering of the samples is required. There are multiple types of filters, however, the most commonly used ones are the low pass filter and the high pass filter. In general, the main functionality of a filter is to remove a specified range of frequencies. Besides, a 'pass' filter works such that there is a specific range of frequencies that it will allow to pass and therefore block the other part of the signal (Keim 2019). In MATLAB, there is a variation of filters that could work on the 8 samples that require analysis, such filters are, filtfilt(), filter(), butter() and lowpass() (Mathworks 2020). All these filters require a normalized cutoff frequency, that ranges between 0 and 1. This value can be determined after performing a Fourier Transformation on the signal to transform it to frequencies in Hz instead of time in seconds (Week 8 Lab) and then examining the frequencies to determine what range causes the noise. The outcomes are then used to determine the cutoff frequencies. Another way of determining the cutoff frequency is to by experimenting a range of values on the signal and then choosing the one that best serves the purpose. After filtering is applied the signal looks much smoother as can be seen in figure 2.

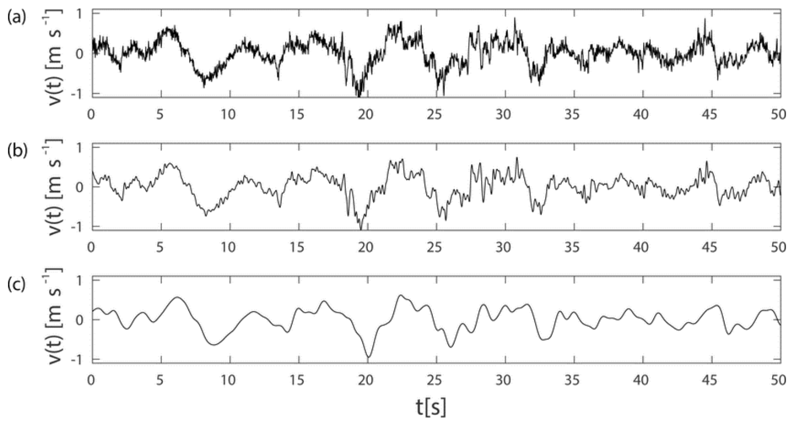


Figure 2: the difference between filtered (Bottom) and non-filtered signal (Top) (European Geosciences Union 2017)

Moving on, MATLAB makes it easy to determine the peaks of a signal using the findpeaks() function. The function's properties can be specified to determine the peaks in the most accurate way possible. One property that was helpful to this project is the MinPeakProminence. First of all, "the prominence of a peak measures how much the peak stands out due to its intrinsic height and its location relative to other peaks" (Mathworks, 2020) see figure 3. Therefore, determining a suitable value for the MinPeakProminence helped to find the R-peaks only.

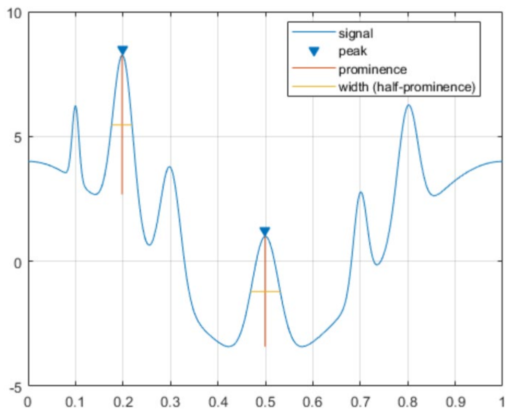


Figure 3: peak prominence illustration (Mathworks, 2020)

Final design solution

1. Project A

The final design solution for this project included a few steps. First of all, I performed filtering on the data sample in use using a lowpass() filter with a cutoff frequency of 0.01 that I determined by experiment on the samples. I chose this specific filter because it was the most efficient one out of the other ones I mentioned before. Also, because it fulfilled the purpose that I needed it for and was also easier to use than the other ones. The result of filtering sample 2 is shown in figure 4.

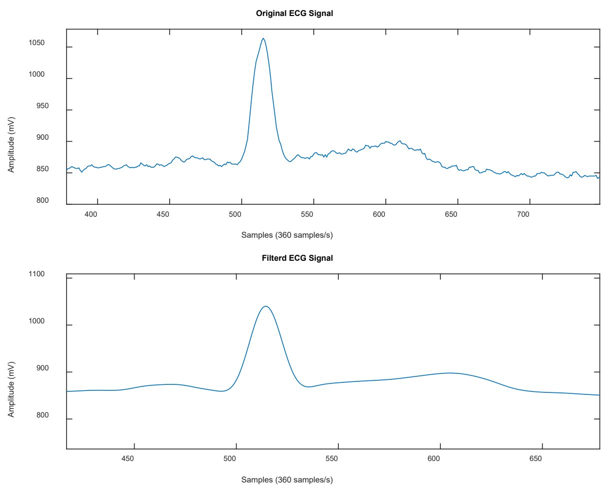


Figure 4: sample 2 filtered using a lowpass filter (M Morad 2020)

Even though filtering was not required for project A, I decided to use it to improve the accuracy of my results as in some of the samples, such as sample 1, there was a deflection in the R-peak that led to having more than one R-peak and a very short time difference between them and hence affecting the heart rate of the entire sample (see figure 5).

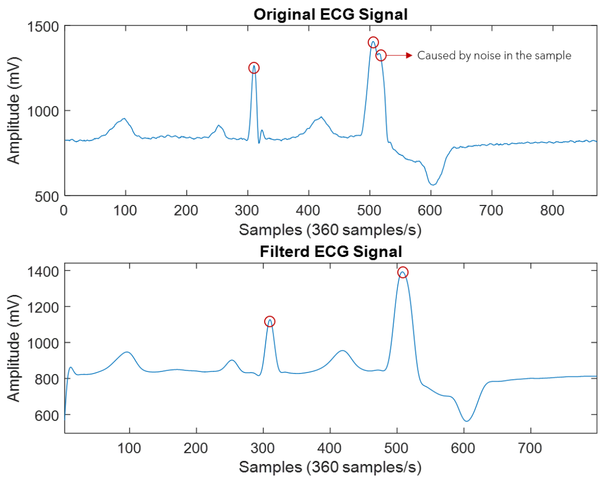


Figure 5: sample 1 unfiltered showing the faulty second R-peak (M Morad 2020)

After filtering the data samples, I used the MinPeakProminence property of the findpeaks() command in MATLAB to determine the R-peaks only. Before finding about this property, my approach was to find a y-limit that excludes all the other peaks and leaves only the R-peaks to be detected by the findpeaks() command as shown in figure 6.

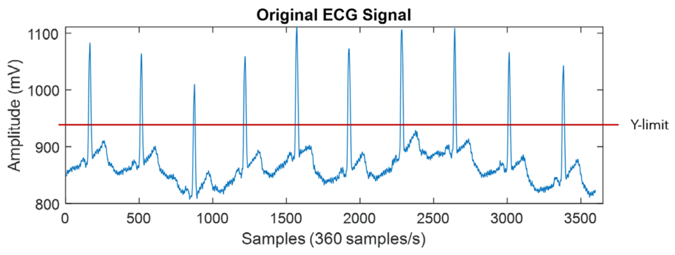
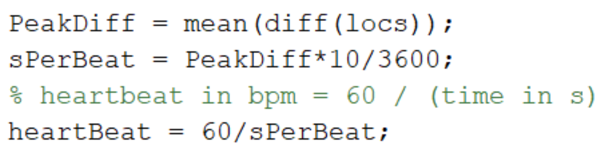


Figure 6: y-limit to find the R-peaks only (data considered is the one above the y-limit) (M Morad 2020)

This method required different y-limits for different samples, while the min prominence method worked dynamically for all samples. As shown in figure 3, I had to experiment on the data samples to find the largest small peak prominence so that I set my minimum prominence to be greater than that. This value turned out to be 140 for project A.

After the peaks have been determined, I calculated the difference between them using the diff() command in MATLAB. In the early stages of my design, I did not notice that the difference between the peaks that I was getting was in terms of samples and not seconds. However, after analysing the results, discussing with my colleagues and comparing the heart rate values that I got with the normal values, I was able to spot the error. Therefore, I converted the time difference into seconds using the sample rate of 360 samples/s and from there into bpm using the formulas shown in figure 7 (ECG Heart Rate Calculator 2018).



```
PeakDiff = mean(diff(locs));
sPerBeat = PeakDiff*10/3600;
% heartbeat in bpm = 60 / (time in s)
heartBeat = 60/sPerBeat;
```

Figure 7: formulas to calculate heartrate in bpm (M Morad 2020)

For this project and project B, I decided to keep my x-axis in the unit of samples because this meant that the x-coordinates of the points I will be finding will be integers which will work perfectly fine with the rest of my code.

Considering that I designed my project based on sample 1, I was expecting a large number of refinements that would need to be done for it to be as dynamic as possible. Fortunately, for this part of the project, there were only minor changes that needed to be done. These included the changing of the min prominence value that was eventually decided to be 140 and the filtering cutoff frequency (later set to 0.01). Other than that, everything worked just fine.

The outcomes of project A are shown in table 1. Please note that I have run the program multiple times to get all the values and then combined the results into table 1.

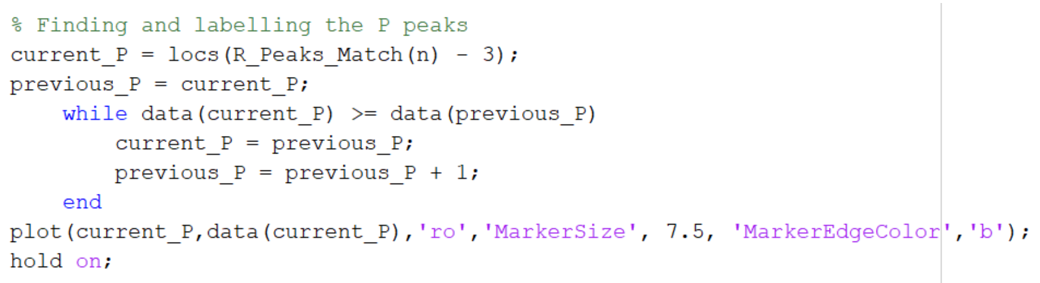
Table 1: Project A outcomes – Heart rates in bpm (M Morad 2020)

Sample No.	**Heart rate (bpm)**
Sample 1	65.08
Sample 2	60.46
Sample 3	90.46
Sample 4	89.97
Sample 5	49.35
Sample 6	53.32
Sample 7	84.81
Sample 8	109.16

2. Project B

The final design of this project also included a set of steps to be able to present the final product. To start with, I applied filtering to the data samples, the same type of filtering used for project A (lowpass filter). However, in this project, the cutoff frequency was different. Same as in project A, the value was determined upon the experiment of numbers. Eventually, I was able to find a number that suited the project and delivered an outcome that I am satisfied with.

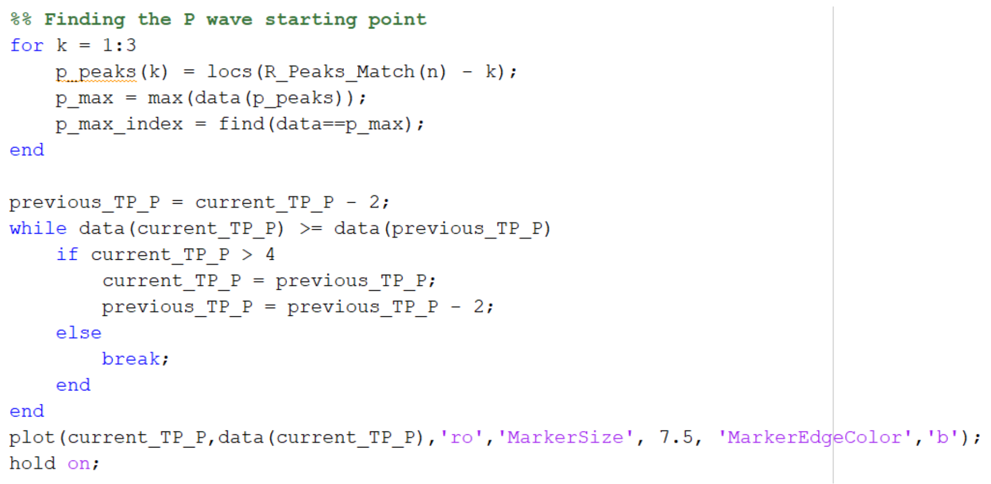
Next, I used the findpeaks command to find and store all peaks in the data sample and then do the same thing for the R-peaks only since they will be a reference point in determining the rest of the points (along with the MinPeakProminence property). In finding the Q-peaks and the S-peaks, I relied on the knowledge from the labs and implemented it to my samples. Furthermore, to find the starting point of the P-wave and the ending point of the T-wave, there were two methods that I considered. The first one was to count three peaks before the R peak for the P-wave and then subtracting a number from the result to be assigned as the starting point of the P-wave as shown in figure 8.



```
% Finding and labelling the P peaks
current_P = locs(R_Peaks_Match(n) - 3);
previous_P = current_P;
    while data(current_P) >= data(previous_P)
        current_P = previous_P;
        previous_P = previous_P + 1;
    end
plot(current_P,data(current_P),'ro','MarkerSize', 7.5, 'MarkerEdgeColor','b');
hold on;
```

Figure 8: 1st method to determine the P-wave (M Morad 2020)

The second method, which is more of a refinement of the first method, included finding the maximum peak out of the three previous peaks and then finding the turning point of the max and the will be designated as the P-wave's starting point (see figure 9).



```
%% Finding the P wave starting point
for k = 1:3
    p_peaks(k) = locs(R_Peaks_Match(n) - k);
    p_max = max(data(p_peaks));
    p_max_index = find(data==p_max);
end

previous_TP_P = current_TP_P - 2;
while data(current_TP_P) >= data(previous_TP_P)
    if current_TP_P > 4
        current_TP_P = previous_TP_P;
        previous_TP_P = previous_TP_P - 2;
    else
        break;
    end
end
plot(current_TP_P,data(current_TP_P),'ro','MarkerSize', 7.5, 'MarkerEdgeColor','b');
hold on;
```

Figure 9: 2nd method to determine the P-wave (M Morad 2020)

Eventually, I decided on using the second method since it provided me with a better outcome than the first one. *The difference between the two methods is shown in figure 10.* The same method was used to determine the endpoint of the T-wave.

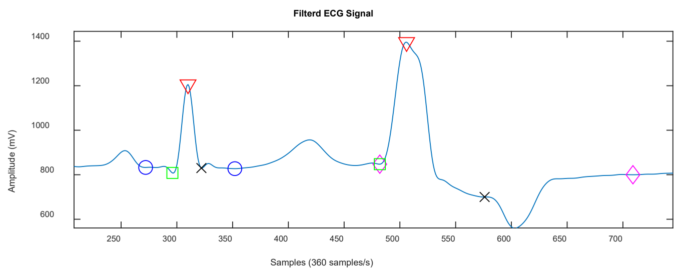


Figure 10a: 1st method outcomes using sample 1 (M Morad 2020)

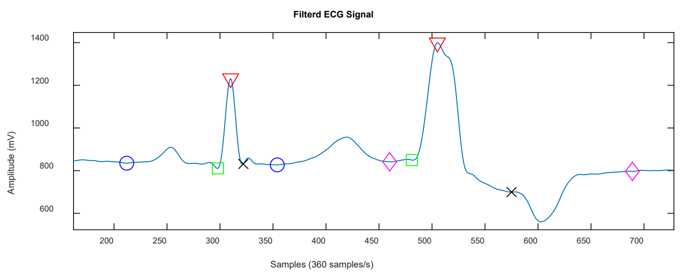
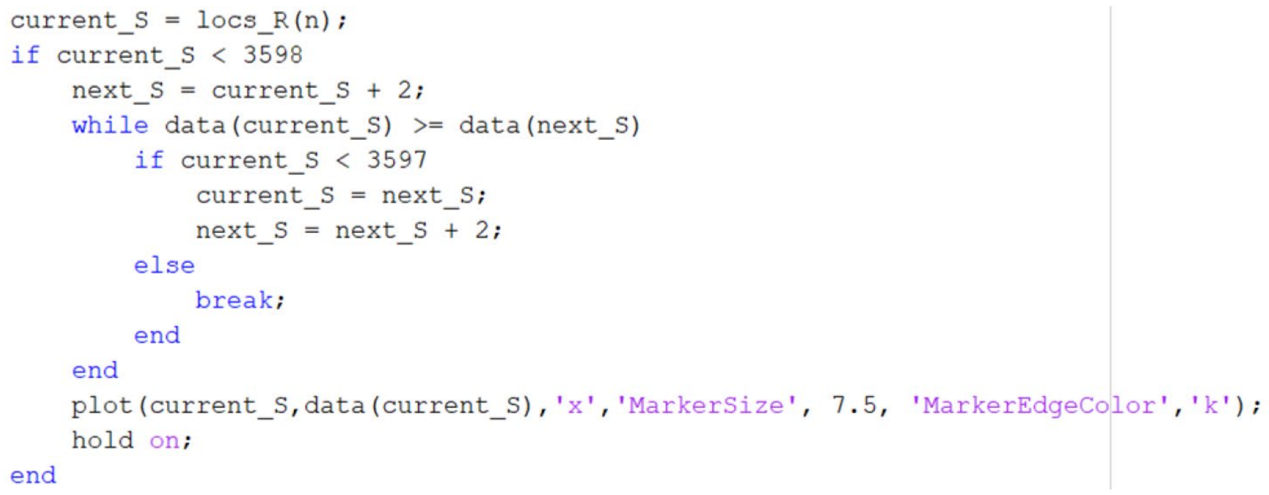


Figure 10b: 2nd method outcomes using sample 1 (M Morad 2020)

Moreover, the last step was to determine the time intervals, the PR, QT and the QRS complex duration. Calculating these values was not challenging since the more challenging part was already done, that was finding the points of interest. After calculating the intervals, I took the mean values of each one and this value was displayed to the user. Also, it was compared to the normal range values and the outcome was also displayed. The choice of using method 2 in the previous step improved the accuracy of the intervals and hence more of the time intervals were in the normal range.

Unlike project A, implementing the design of project B using sample 1 on the other samples, led to errors in the program due to incompatibility. The main error was that: *The index exceeded the number of elements.* I fixed this problem using conditional if statements as shown in figure 11 below.



```
current_S = locs_R(n);
if current_S < 3598
    next_S = current_S + 2;
    while data(current_S) >= data(next_S)
        if current_S < 3597
            current_S = next_S;
            next_S = next_S + 2;
        else
            break;
        end
    end
    plot(current_S,data(current_S),'x','MarkerSize', 7.5, 'MarkerEdgeColor','k');
    hold on;
end
```

Figure 11: the required adjustments to the code to resolve the debugging errors (M Morad 2020)

Upon experiment, I discovered that since `next _S = current_S + 2`, current_S cannot exceed 3598. This is because, if we take current_S as 3599, next_S will be 3601, which is over the indices of our data (3600), causing the error to occur. Besides, the second if statement, inside the while loop, if current_S < 3597 was also decided upon experiment. I noticed that since current_S is assigned to whatever the value is for next_S after the while condition is checked, current_S cannot exceed 3597. The case where it exceeds would result in an incompatibility error (index number greater than 3600).

I also faced the same error while finding the T-waves, however, this time the index number was related to the R-peaks, not the whole data.

Note: in finding the T-wave, and after experimenting on the data, I decided to count 4 peaks after the R-peak, instead of 3 peaks that I used in finding the p-waves as this turned out to be more suitable to this situation. Since I am looking for 4 peaks after the current R-peak, the index number for the R-peak needed to be less than or equal to the total number of R-peaks minus 4.

The outcomes of project B are shown in table 2. Please note that I have run the program multiple times to get all the values and then combined the results into table 2.

Table 2: Project B Command Window outcomes (M Morad 2020)

Sample No	PR-Interval (s)	QT-Interval (s)	QRS Complex Duration (s)
Sample 1	0.2300 - OUT	0.4086 – OUT	0.0928 - IN
Sample 2	0.1206 - IN	0.3628 – IN	0.1056 - OUT
Sample 3	0.1985 - IN	0.2900 – IN	0.0893 - IN

Sample 4	0.1613 - IN	0.1313 – IN	0.0589 - IN
Sample 5	0.0451 - OUT	0.3267 – IN	0.0847 - IN
Sample 6	0.0781 - OUT	0.3627 – IN	0.1586 - OUT
Sample 7	0.1688 - IN	0.3115 – IN	0.0968 - IN
Sample 8	0.0948 - OUT	0.3228 – IN	0.1272 - OUT

In terms of graphical representation, I chose sample 6 (see figure 12), to demonstrate since I have already shown the outcomes for sample 1 in figure 10b.

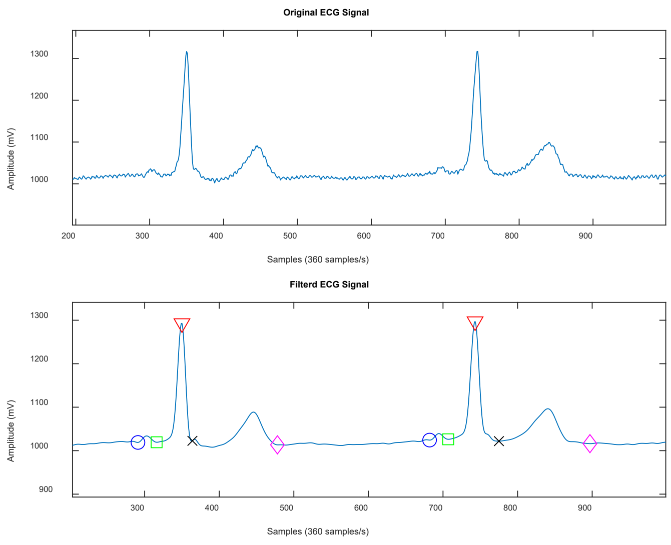


Figure 12: all the requested attributes labelled on the graph of sample 6 (M Morad 2020)

Conclusion and recommendation

In conclusion, I consider these two projects a great achievement for me. Not only because they are a part of the unit, but also because they have taught me a lot, either in terms of ECG knowledge or programming knowledge. I believe that both projects perform at a very good level. There could be some errors in the determination of the P-waves and the T-waves that I could not address. However, since no debugging errors have been encountered, I consider this project a success, having in mind that most results are within the acceptable range either for the heart rate or for the time intervals.

One thing that I could have done to improve my results would have been assigning legend titles to the markers on the plot in project B. This would have made it easier to identify the points labelled. Even though the points are in the order they are normally in, it would have been much simpler if they were labelled using legends. On the other hand, I think in project A, I was able to calculate the heart rate values in a very accurate way. Perhaps, I could use a more sophisticated filtering technology that would boost the accuracy of my results even further.

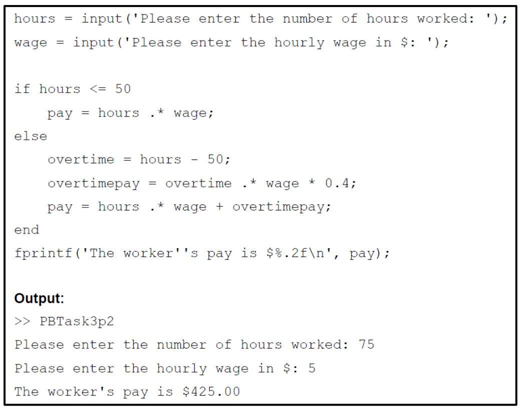
References

- ACLS Medical Training 2020b, *The Basics of ECG,* ACLS Medical Training, viewed 31 October, 2020, <https://www.aclsmedicaltraining.com/basics-of-ecg/>.
- Alivecor, 2020, *What is an ECG*, Alivecor.com, viewed 31 October, 2020, <https://www.alivecor.com/education/ecg.html#:~:text=Also%20known%20as%20an%20electrocardiogram,electrical%20activity%20in%20your%20heart.&text=This%20slowing%20signal%20appears%20as,beginning%20of%20the%20Q%20wave.>.
- E. Klabunde, R, 2012, *CV Physiology | Electrocardiogram (EKG, ECG)*, Cvphysiology.com, viewed 31 October, 2020, <https://www.cvphysiology.com/Arrhythmias/A009>.
- Jinchuan Zheng, 2020, *ENG10004 Digital and Data Systems Semester 2, 2020 (Project Guideline – Software Design)*, Swinburne Instructor, viewed on 15 July, 2020, <https://swinburne.instructure.com/courses/23326/files/9794619/download?download_frd=1>.
- Keim, R 2019, *What Is a Low Pass Filter? A Tutorial on the Basics of Passive RC Filters*, Allaboutcircuits.com, All About Circuits, viewed 30 September, 2020, <https://www.allaboutcircuits.com/technical-articles/low-pass-filter-tutorial-basics-passive-RC-filter/>.
- Mathworks 2020, *Digital Filter Design*, Mathworks.com, viewed 1 October, 2020, <https://au.mathworks.com/help/signal/filter-design.html>.
- ECG Heart Rate Calculator 2018, *ECG Heart Rate Calculator*, MDApp, viewed 2 November, 2020, <https://www.mdapp.co/ecg-heart-rate-calculator-483/>.
- Mathworks 2020, *Find Peaks in a Vector*, Mathworks.com, viewed 3 October, 2020, <https://au.mathworks.com/help/signal/ref/findpeaks.html>.

Reflective Journal

Week 3

This week I studied Relational and Logical Operators, Conditional Statements such as (if and switch) and Loops including the for and while loops. In learning that I was able to develop my MATLAB programming skills that I will need in the development of my project (Parts A & B). After watching the recorded lecture and attending the Lab session, I was able to successfully complete this week's Pass Borderline task as well as the first Pass Plus task. I then continued to do the last recommended task for the week, which was the first Credit task. I faced some challenges in solving the problems included in this task. I tackled the first problem and solved it by doing some research online as well as using the help commands that we were taught about in the previous weeks both lectures and labs. With the extra research I did, I gained more confidence is attempting at the second problem of this task. Nevertheless, this turned out to be more challenging than the previous one, and that was when I sought my peer's help and we discussed the problem and were able to see what went wrong and understand the functionality of the command that needed to be used. Eventually, the problem was solved, and all the functionalities of my program were providing the correct/logical outcomes. Overall, This week's materials are going to help me a lot in both Projects A and B, since they include the base knowledge of conditions along with the plots knowledge we gained from the previous two weeks that will be a solid part of our project (see figures 1R and 2R).



```
hours = input('Please enter the number of hours worked: ');
wage = input('Please enter the hourly wage in $: ');

if hours <= 50
    pay = hours .* wage;
else
    overtime = hours - 50;
    overtimepay = overtime .* wage * 0.4;
    pay = hours .* wage + overtimepay;
end
fprintf('The worker''s pay is $%.2f\n', pay);
```

Output:

```
>> PBTask3p2
Please enter the number of hours worked: 75
Please enter the hourly wage in $: 5
The worker's pay is $425.00
```

Figure 1R: Coding example using if statements

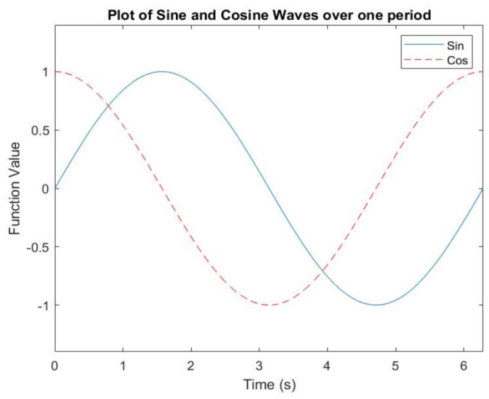
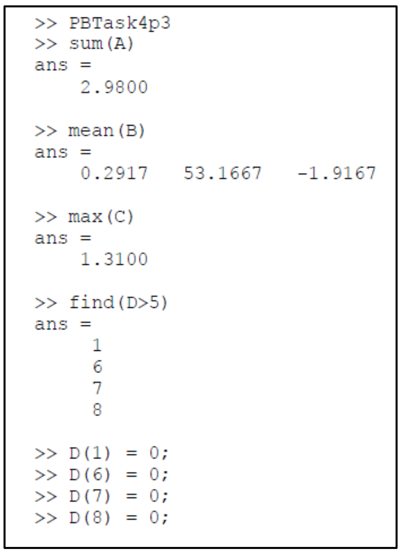


Figure 2R: A plot that I computed for one of the portfolio tasks

Week 4

This week I learnt about user-defined functions as well as some data analysis commands. After watching the lecture and attending the online lab session, I started working on the assigned tasks for this week that were: Pass borderline task 4, Distinction task 1 and the High Distinction task 1. I was able to complete the PB task with no complications. However, I faced some difficulties trying to work out what commands to use in my distinction task. I was stuck on the task for almost three hours but then I decided to delete everything and start again because I knew it had something to do with using the same name for my variables (the ones inside the function and the ones I parsed them to the function). I ask a friend about it as well and we discussed it further and I was finally able to complete it. The next day I started working on my High distinction task. I still haven't finished it yet, but I've completed approximately %75 of it. This one required a little more research than the previous one and is also much more challenging. It also requires a solid understanding of the physics behind the projectile motion of a ball that is quite essential to the game. I believe the knowledge and all the new commands I learned and still learning about from this week's work will be quite essential for both parts A and B of the project (Heart Rate Monitor and heart rate analyser), especially the user-defined function that will ensure the efficiency of



```
>> PBTask4p3
>> sum(A)
ans =
    2.9800

>> mean(B)
ans =
    0.2917   53.1667   -1.9167

>> max(C)
ans =
    1.3100

>> find(D>5)
ans =
     1
     6
     7
     8

>> D(1) = 0;
>> D(6) = 0;
>> D(7) = 0;
>> D(8) = 0;
```

Figure 3R: Implementation of the data analysis commands covered this week

the code avoiding any unnecessary repetition of the code. In addition, the data analysis commands will also be handy in determining the most accurate values possible (see figure 3R).

Week 5

This week, I watched the lecture and attended the laboratory live session where I learnt about data electronics. In the first section, I learnt the difference between analog and digital signals that will be quite important in the working on Projects A and B (Note from week 9, after the change to project B, becoming fully Software, this material is no longer helpful to the project). I also learnt about Boolean algebra in terms of Boolean expressions, truth tables and logic gates. This information was quite important for finishing two of the recommended tasks for this week, the pass borderline task 5 and the pass plus task 2. In working on these two tasks, I learnt how input gets converted into binary expressions, 0s and 1s for the computer to understand it. As recommended, I also worked on the credit task, and was able to finish it. It required some research and a deeper understanding of the concept of the mean (average) of a set of data (smoother). Overall, this week's skills, even though they are no longer relevant to the projects, they helped me a lot in understanding the functionalities of input and output devices. For projects A and B, I continued my research, in which I learnt about how an ECG waveform is read, and what are the important attributes in it.

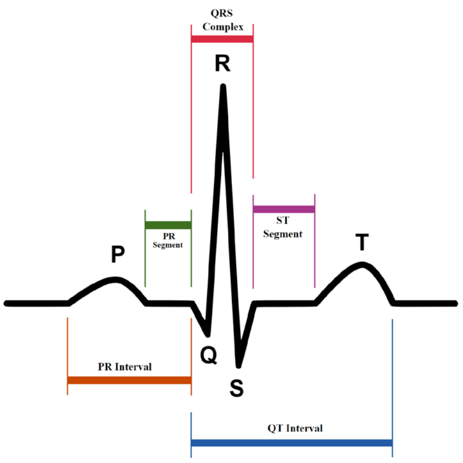


Figure 4R: Interpretation of one wave of an ECG signal. (Wikipedia Contributors, 2020)

Week 6

After learning about data electronics last week, I consolidated my knowledge with this week's lecture. I learnt about simplifying more complex Boolean expressions. I also learnt about number systems and converting between one and another. This was quite helpful as I used this knowledge to complete three of the portfolio tasks. The pass borderline task 6, the pass plus task 2 and the Distinction task 2. Also, learning about Simulink and drawing circuit gates using it, made the last part of the D2 task much simpler, and a confirmation that the connections are correct. After finishing these tasks, I started working on the High Distinction 2 task. Later this week I am planning to finish off the HD task so that I can use my mid-semester break to focus on continuing to understand projects A and B. Altogether, this week's materials fall under the same category as last week which is also no longer helpful for the projects. However, they still count as a big part of my learning in this unit. While working on the tasks mentioned earlier, I gained more confidence in working with graphs, customizing them, and adding labels to specific points (see figure 5R). Additionally, I worked on building my code from scratches, and breaking the task down into simpler parts. I also developed a very handy skill that is to test each individual part of my code before attempting to write the following part. This is quite useful for the projects since it will be very challenging to finish it in one sitting and so it is important to keep testing and refining.

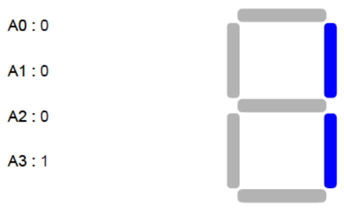


Figure 5R: customization of plots from the 2nd HD task

Week 7

During this week's Lab session, we were introduced to the new requirements of our project that became fully software based. After asking many questions and clarifying any points of confusion with the lab demonstrators. I continued my research on how ECGs are read and interpreted. With this assistance of the project guidelines that were full of information about ECGs, I was able to brainstorm my first idea of how project A would look like for me. The first thing I did was getting the samples uploaded to Canvas and testing the load function in MATLAB to see how it works and also to get an idea of how the samples look like by plotting them. Since it is already in the guidelines how to read an ECG and how to determine the heartrate, that is using the time for one cycle (between two R peaks), my research for this week was about two things. The first one was about how to convert time in millisecond to heartrate measured in bpm (beats per minute). This was not a hard task to do since the conversion is simple. I found online that the conversion is simply as follows:

$$Heartrate\ in\ bpm = \frac{60}{\frac{time\ of\ one\ cycle\ in\ ms}{1{,}000}} = \frac{60{,}000}{time\ of\ one\ cycle\ in\ ms}$$

This was a milestone for me, since it is a big part of project A. After that, I started looking into some filtering techniques that will provide me with smoother curves, especially around the last three data sets, where it gets a lot noisier. I found some methods online and I tried them, however, I decided to attend next week's lab session for a more detailed explanation about how filtering works and perhaps a more concise way of filtering.

Week 8

In this week's Lab session, we learnt about filtering techniques and the steps required to identify the noise frequencies in order to remove them using one of the filtering commands in MATLAB. We learnt about the FFT, which is a command that converts time waves into frequency waves. This command allows us to see which frequency acts as noise and which one is the actual frequency that we will rely upon for our analysis (see figure 6R). Note that I still do not have my own implementation of this command, that is why figure 6R is adapted from the lab notes for this week. After learning about these new commands, I did a trial program to see how I will manage filtering the 8 data samples we have been provided. Unfortunately, the program did not function as expected so I decided to wait until I watch Week 9's Lab session that will have a real example of filtering as mentioned by the tutors. I then dedicated my time to work on the progress presentation for both projects. Since we had project A's guidelines at an earlier time that Project B's, I had already done a little bit of research on that part in the previous weeks. Therefore, it was just a matter of arranging the information I had and designing an attractive and informative PowerPoint presentation. Later in the week, I started rehearsing my presentation, however, it turned out to be much longer than 5 minutes, that is the maximum length of the presentation. My plan for next week is to make my presentation more precise and straight to the point so I can deliver it in the assigned time interval.

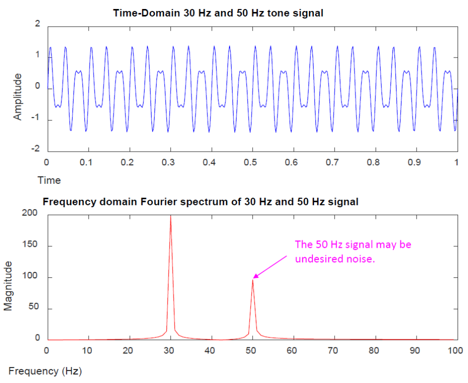


Figure 6R: demo of the FFT command adapted from week 8's lab notes

PROJECTS PARTS A & B PROGRESS PRESENTATION

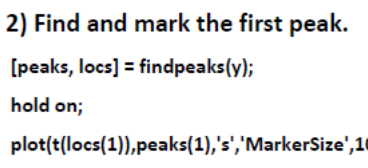
NAME: MARELLA MORAD

STUDENT ID: 103076428

Figure 7R: a snip of my progress presentation for the projects

Week 9

This week included a much more detailed approach into the filtering concept and how to apply it to our situation having some very noisy signals such as sample 8 of the ECG data. The lab session elaborated on the steps we learnt in last weeks lab in order to filter a specific signal. The examples the lab instructor went through were helpful in terms of understanding how to apply these techniques to our own project. We also learnt about point identification techniques using a command in MATLAB. This will be quite



2) Find and mark the first peak.

```
[peaks, locs] = findpeaks(y);
hold on;
plot(t(locs(1)),peaks(1),'s','MarkerSize',10)
```

Figure 8R: the peaks locs command, adapted from week 9's lab notes.

useful for Project B, since we are required to identify the peaks, Q, R and S of the ECG signal, see figure 4R for reference. It will also assist us in labelling the time intervals, PR, QT and the QRS. During my mid-semester break, I worked on an algorithm to find the x-axis coordinate of a peak on my plot of one of the ECG samples, since the findpeaks command returned only the y-coordinate which was not helpful in finding the time difference i.e. the heartbeat. In this algorithm I used a loop that went through the data set in use and found the index number for each peak value. During this week's lab session, I learnt a more efficient way that was using the [peaks locs] identification along with the findpeaks command, that returns both the x and y coordinates (ms and mV values). After studying this material, I continued working on my progress presentation which is due the end of this week. I was finally able to shorten it to a more precise presentation and record my narration to it.

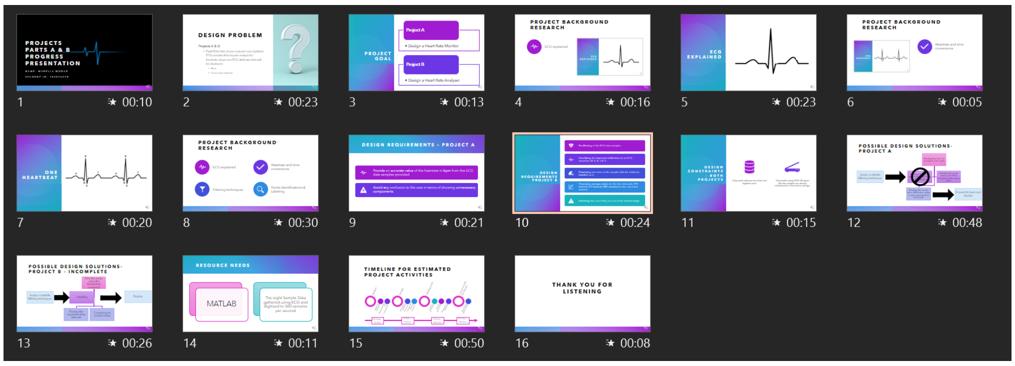
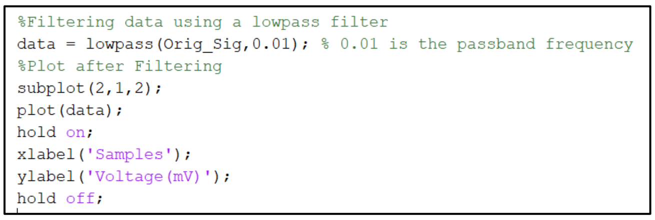


Figure 9R: the finished look of my progress presentation.

Week 10

During this week's Lab session, we were introduced to LabView and its functionalities. However, this software is irrelevant for the project, and so I am skipping it. In my own time, I worked on the filtering technique for project A. Even though in my timeline I indicated that project A should be done by the start of week 10, it had to go a bit overtime for the filtering to be complete. I tried to fit the filtering technique introduced in the lab session of last week into my code, however, it did not work out. Therefore, I did a little bit of research on other filtering functions that MATLAB provides us with and found the one that worked perfectly with my code. The filter that I used is called lowpass(), to filter the 8 data sets, see figure 10R. After that, I found out that the way I was calculating my heartrate was wrong, since I was not considering the digitizing rate, 360 samples/s. I used the outcomes of my first code, to figure out the factor that I needed to multiply by to get the time of one cycle in seconds. This turned out to be.



```
%Filtering data using a lowpass filter
data = lowpass(Orig_Sig,0.01); % 0.01 is the passband frequency
%Plot after Filtering
subplot(2,1,2);
plot(data);
hold on;
xlabel('Samples');
ylabel('Voltage(mV)');
hold off;
```

Figure 10R: lowpass() filter implemented in my code

```
sPerBeat = PeakDiff*10/3600
```

With the PeakDiff in samples, instead of seconds. After that I was able to calculate the heartrate for all the 8 samples and all of them produced values that were in the range. In my research I also learnt about the concept of prominence in graphs. This enabled me to get only the R peaks instead of any other irrelevant peaks and works dynamically on all samples. The fact that I had to work on project A limited my development time for project B to only one a half week. However, I think that with the filtering done, the other tasks of project B are simpler and less time consuming.

Week 11

This week's lab session did not have any new information, as it was just a Q&A session. I was able to clear all the concerns I had, especially about finding the start point of the P wave and the end point of the T wave. I learnt that the P wave, starts 3 peaks before the Q peak, and that the T wave ends, 3 peaks after the S peak. I was then able to implement this to my design for project B. I implemented the same filtering technique, lowpass() filter, however, this time I changed the passband frequency, because I wanted more peaks to be present in the wave to be able to determine the P and T waves' initial and end points. I changed the passband frequency to 0.05, instead of 0.01, which corresponds to 180 Hz, instead of 36 Hz. While a very smooth curve was very suitable for Project A, since we were only required to find the R-peaks, a less smooth curve is required for Project B. The difference is not huge between the two filtered data, as shown in figure 11R.

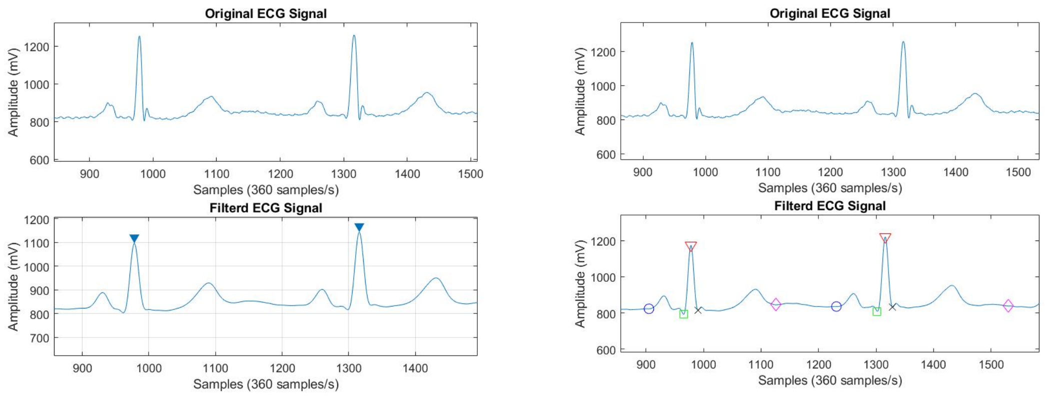
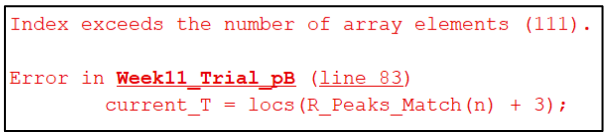


Figure 11R: lowpass filter with passband frequency 0.01 (left) and 0.05 (right)



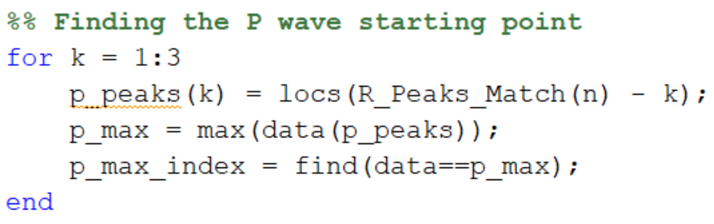
```
Index exceeds the number of array elements (111).

Error in Week11_Trial_pB (line 83)
        current_T = locs(R_Peaks_Match(n) + 3);
```

Figure 12R: error in samples adjustment

As suggested in the project guidelines I designed my program based on the first sample, which is also displayed in figure 11R. After that, I tested the other samples, and I had some issues in samples 4 and 6. These two samples ended on an R peak, with no signal for the S peak and the T wave to be determined. Therefore, I had to do some adjustments to the code. I was getting the error shown in figure 12R. Therefore, I decided to use if statements to limit the allowed index number used as I was eventually able to resolve this error. My plan for next week is to conduct further testing on the two projects in case they require any refinements and after that record the demonstration video as the due date for the code and video is on the Friday of next week.

Week 12

This week was the last week to work on the project. In the first three days of the week I finalised my project code for project B. After checking with my tutor, I realised that I was calculating the P-wave and the T-waves incorrectly. Therefore, I experimented on some values and re-designed my code. I found that the problem was with having a small number of peaks, and that I required more peaks to do this calculation. I increased my passband frequency for project B to 0.07 which corresponds to 252Hz. This increased the total number of peaks in my sample. For example sample 1 now has 143 peaks. After this modification, I designed a few new lines of code that would find three peaks before the Q peak (for the p-wave) (see figure 13R) and four peaks after the S peak (for the T wave). I am aware that the T-wave is 3 peaks after the s-peak not 4, however, 4 turned out to work better for my samples than three.



```
%% Finding the P wave starting point
for k = 1:3
    p_peaks(k) = locs(R_Peaks_Match(n) - k);
    p_max = max(data(p_peaks));
    p_max_index = find(data==p_max);
end
```

Figure 13R: finding the peaks for the determination of the p-wave's initial points

After finding these peaks, I went and implemented the same code that I used to find the Q peaks and the S peaks, to find the turning points that will mark the start of the P-wave and the end of the T-wave.

This method produced more accurate/reasonable results. After finishing off the project code, I looked for any parts of my code that needed explanation and I did that in a separate word document that I uploaded later with my code. Lastly, I practiced the script for my video, recorded it, edited the parts that needed editing and was able to fit everything in the 6 minutes limit we had. I then uploaded everything to canvas and started working on my report.

Reflection on my learning experience (Answer the following questions)

1) **What are the most important things you learnt from this project?**

   One of the most important things that I learnt is organisation of project activities. It was really helpful to have a schedule that allowed me to finish all the tasks before the deadline. I also learnt a lot about ECG and how these waves are interpreted, as well as how filtering works.

2) **Did you meet your project goals?**

   Yes, I met my project goals. I developed a program for project A that calculates the heartrate in BPM, and for project B, I was able to display all the requested attributes as well as calculating average values of the time intervals and comparing those to the normal range values.

3) **What parts of the project do you particularly like? Why?**

   I like project B more, because it challenged me to do some deeper research and get a deeper understanding of the project problem/requirements.

4) **What do you find particularly challenging?**

   There were few parts of project B that were challenging, such as filtering, however after some research I was able to pick a suitable and simple to operate filter for my projects. Another challenging aspect was figuring out how to limit the code to the parts of the graph that has data in it, for example some data samples did not have any S-waves at the end of the sample.

5) **What are the things that helped you most in this project?**

   The main help I got was from my tutors and the lab sessions they ran. Also, I got some help from discussions with my colleagues, as well as undertaking research whenever it was required.

6) **One thing I would like to improve upon is?**

   Time management, I learnt a lot in this project how important it is to have a schedule and I did have some sort of a schedule, however, I did not quite stick to it, so in the future, I will be more committed to my schedules.