Step 1: Problem Identification and Statement

The aim of this assignment is to analyze the relationship between the EMG signal and the grip force, and to derive a mathematical model to infer the grip force from the muscle activity.

The assignment requires the following tasks from the student:

1. Task 1:
    - Load the recorded data
    - Apply a low pass filter with a cutoff frequency of 200 Hz for both the EMG signal and the force signal.
    - Crop the first 500 samples of the force and EMG data as well as the last 250 data points.
    - Normalize the force signal such that the minimum value is zero and the maximum value is 100.
    - Plot the EMG signal (in Volts) and normalized force (in percentage of MVC) against the time axis in two subplots
2. Task 2:
    - Develop a method for automatic **identification of portions (segments)** corresponding to one trial of grasping activity. These segments must have an equal number of data points.
3. Task 3:
    - Determine the **average** of all these segments (force and muscle signals). You must plot the average segment for the force and muscle data over time, for the duration of one trial. Create a plot for the grip force (y-axis) against the muscle activity (x-axis). Comment on this graph.
4. Task 4:
    - Obtain the **best fit function** to represent the variation of EMG activity versus the force Superimpose the fit function on the plot of the EMG activity/force. Analyze the results

Step 2: Gathering Information

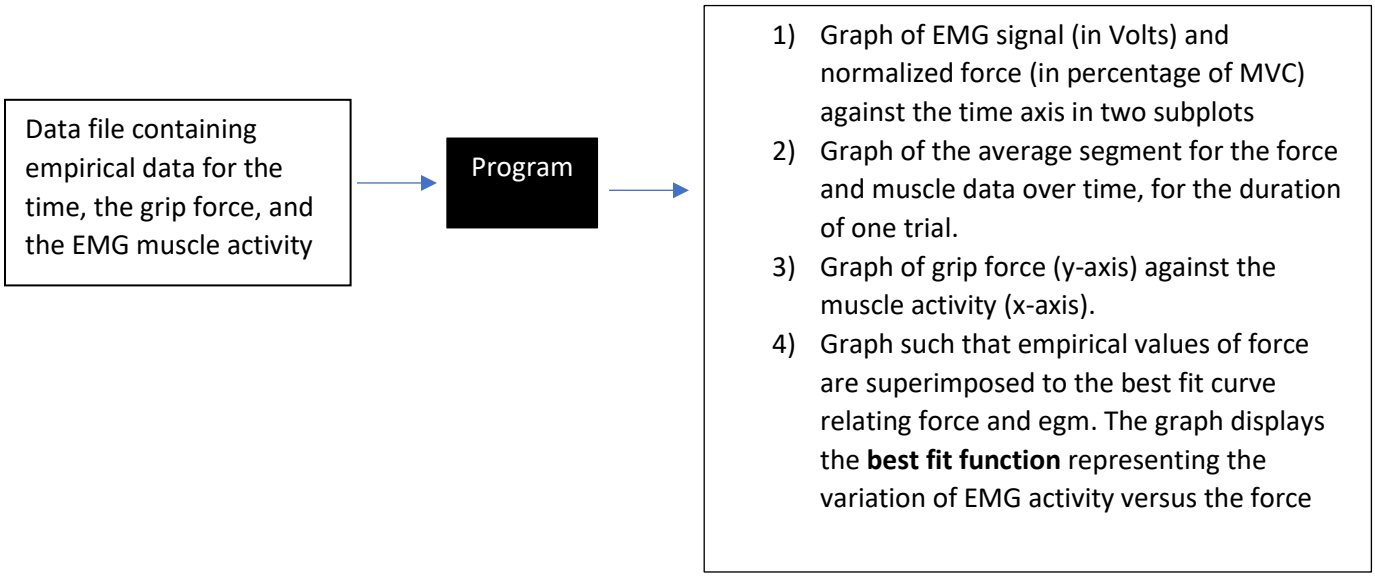
One important research about hand-arm system is the information about **hand grip force** and **pressing force** on a tool handle. One popular solution for measuring grip force involves attaching force sensors to the handled tool. However, this is uncomfortable because the force sensors interfere with the handle construction. An alternative solution is to record the electrical activity produced by muscles, electromyography (EMG). It has been assumed **that EMG signal are proportional to muscle tension** responsible for palm grip. This solution has a significant advantage when comparing to force sensors, it can be used with regular tool without interfering in the handle.

Electromyography is a technique connected with receiving, recording, and examining of myoelectric signals. These signals are coming into being thanks to physiological changes which are taking place in muscular fibers. During relaxation, a muscle shows no electric relevant activity so that the electric line is straight. During the contraction, potential of motor units deviates the EMG line.

EMG data is collected through the Delsys Bagnoli EMG system along with a flex sensor attached to a soft body. The EMG electrode and force sensor are connected to the data acquisition board which records the EMG and grip force signals (in the form of voltage) and store them into a file. The file stores the time stamp, the grip force, and the EMG muscle activity as three columns of data in the same order. The data

is collected at a sampling rate of 1 kHz for a duration of 30 seconds. The recorded data is stored in the data file “Data.mat”. The recorded data contains a total of 31 trials of press/release of the hand

Input Output Diagram



Step 3: Algorithm Explanation

Task 2

To identify the segments the following procedure is applied:

- The second column of the matrix, which contains values for force, is smoothed two times. This enables software to recognize maximum and minimum correctly ignoring negligible peaks or variations of the curve.
- Findpeaks is used on the force-time graph to find the number of peaks. We expect the number of peaks to be 31. The number of peaks identifies the number of cycles
- Findpeaks used on the negative of the force-time graph to find the time such that force is minimum (which is the same as x-values of local minimums). The positions of local minimum are the starting and ending points of the cycles. Therefore, the positions of minimum will be used to access each cycle in task 3 in order to find the average segment of force over time.
- The number of elements in a cycle should be constant. Hence was calculated as follows:

$$Domain \ = \ position \ of \ last \ minimum - position \ of \ first \ minimum$$

$$SegmentSize \ = \ domain \ / \ number \ of \ Peaks$$

- Number of elements for segment is found as decimal value. Therefore, $SegmentSize$ must be casted to integer $SegSizeInt$

Task 3

To find average force segment the following algorithm is applied:

- Create matrix AverageSegment ($SegSizeInt$,3) which will store an average cycle of force and average cycle of EGM. There are $SegSizeInt$ element in each segment, hence there should be $SegSizeInt$ element in the average segment for force too
- For each position i of average cycle, starting from i=0 which is first position,
  for each segment,
  find the force value in position i and store it into vector `ForceAti`.
  the average of elements in `ForceAti` is the average value of force for position i of average cycle. Hence average is stored into AverageSegment(i+1,2)
- The same procedure is used to find average cycle for EGM. Average values of EGM for each position are stored in the third column of AverageSegment

- Since the egm-time curve is very different from the force-time curve, data elaboration of the egm signal is necessary to plot the graph of force against egm. Therefore, to make the egm data comparable with the force data, we decided to model the egm data. This is done by firstly taking the absolute value of the third column of AverageSegment containing average segment, and secondly by fitting a curve using the gauss model. As a result, model of egm-time data can be compared to force-time data in graph 7.

Pseudocode

Task 1

```
Assign 200 to PassFreq
Assign 1000 to Rate
Assign 500 to CancTop
Assign 250 to CancBott

Read and store data from('Data.mat')
Assign the length of timestamps to readings
Assign readings - CancTop - CancBott to DataNum

Define matrix as a matrix of zeros with readings as row number, 3 as
column number
Define MatrixFilter as matrix of zeros with readings as row number, 3
as column number
Assign size of matrix to rows and cols
For i=1, until i = rows, increment i at each iteration:
     Assign timestamps(i,1) to matrix(i,1)
     Assign Force_signal(i,1)to matrix(i,2)
     Assign EMG_signal(i,1)to matrix(i,3)
End for loop
Assign lowpass of matrix(:,2),given PassFreq and Rate to MatrixFilter
(:,2)
Assign lowpass of matrix(:,3),given PassFreq and Rate to MatrixFilter
(:,3)
For i=1, until i = rows, increment i at each iteration:
  Assign timestamps(i,1)to MatrixFilter(i,1)
End for loop
```

```
For i=1, until i = CancTop, increment i at each iteration
     Assign [] to MatrixFilter (i,:)
End for loop
Assign length of MatrixFilter to rows


Define MatrixValid as matrix of zeros with DataNum as row number, 3 as
column number

For i=1, until i = DataNum, increment i by stepsize of 1 at each
iteration:
Assign MatrixFilter(i,1) to MatrixValid(i,1)
Assign MatrixFilter(i,2) to MatrixValid(i,2)
Assign MatrixFilter(i,3) to MatrixValid(i,3)
end for loop


Define MatrixFinal as matrix of zeros with DataNum as row number, 3 as
column number
Assign MatrixValid(:,1) to MatrixFinal(:,1)
Assign MatrixValid(:,3); to MatrixFinal(:,3)
Normalize MatrixValid(:,2) in range [0,100] and assign the normalized
result to
MatrixFinal(:,2)


Define MatFinCopy as matrix of zeros with DataNum as row number, 3 as
column number
Assign MatrixFinal(:,1) to MatFinCopy(:,1)
Assign MatrixFinal(:,2) to MatFinCopy(:,2)
Assign MatrixFinal(:,3) to MatFinCopy(:,3)

subplot the first of two graphs assigning MatrixFinal (:,1) to x axis,
MatrixFinal (:,2) to y axis
Assign 'Graph 1: Normalized Force and time relationship' to title
Assign 'time/s' to label of x axis
Assign 'Force/N' to label of y axis

subplot the second of two graphs assigning MatrixFinal (:,1) to x
axis, MatrixFinal (:,3) to y axis
Assign 'Graph 2: EGM and time relationship' to title
Assign 'time/s' to label of x axis
Assign 'EGM/V' to label of y axis
```

Task 2

```
cancel the plotted figure
Define MatrixSmooth as matrix of zeros with DataNum as row number, 3
as column number
```

```
Define MatrixSuperSmooth as matrix of zeros with DataNum as row
number, 3 as column number

Assign MatFinCopy(:,1) to MatrixSmooth(:,1)
Apply smoothdata to MatFinCopy(:,2) and assign the smoothed result to
MatrixSmooth(:,2)


Apply smoothdata to MatFinCopy(:,3) and assign the smoothed result to
MatrixSmooth(:,3)
Assign MatFinCopy(:,1) to MatrixSuperSmooth(:,1)
Apply smoothdata to MatrixSmooth(:,2) and assign the smoothed result to
MatrixSuperSmooth(:,2)
Apply smoothdata to MatrixSmooth(:,3) and assign the smoothed result to
MatrixSuperSmooth(:,3)

plot graph by assigning MatrixFinal (:,1) to x-axis and MatrixFinal (:,2) to
y-axis
continue drawing on this figure
plot graph by assigning MatrixSuperSmooth (:,1) to x-axis and
MatrixSuperSmooth (:,2) to y-axis, the line should be blue

Assign 'Time/s' to label x-axis
Assign 'Force (%)' to label y-axis
Assign 'Normalized Force curve', 'Smoothed Force curve' to legend
'Normalized Force curve' refers to first plotted curve 'Smoothed Force curve'
to second plotted curve
Stop drawing on this figure

Apply findpeaks to MatrixSuperSmooth (:,2) and set minimum peak as 16.6.
Store y coordinate of peak values in vector Max
Store x coordinate of peak values in vector idxMax
Store peak width values in vector Width
Find length of vector Max, assign length to PeakNumber
Apply findpeaks to the negative of MatrixSuperSmooth (:,2)
Store y coordinate of peak values in vector Min
Store x coordinate of peak values in vector idxMin


Find length of vector idxMin and assign it to ComputedMin
Assign idxMin(2) to RangeStart
Assign idxMin(ComputedMin - 1) to RangeEnd
Assign (RangeEnd-RangeStart) / PeakNumber to SegmentSize
Cast SegmentSize to 16 bit signed integer to SegSizeInt

Define SegStartPnt as vector of zeros of length PeakNumber
For i=0, until i= PeakNumber-1, increment i in steps of 1 at each iteration
      Assign RangeStart + i * SegSizeInt to SegStartPnt (i+1)
End for loop
```

Task 3

```
cancel the plotted figure
Define AverageSegment as matrix of zeros with SegSizeInt as row number, 3
as column number
```

```
Define AvgSegABS as matrix of zeros with SegSizeInt as row number, 3 as
column number

For For i=1, until i= SegSizeInt
      Assign MatrixFinal((RangeStart+i),1) to AverageSegment(i,1)
End for loop

Define ForceAti as vector of zeros of length PeakNumber
Define EMGAti as vector of zeros of length PeakNumber

For i=0, until i= SegSizeInt-1, increment i in steps of 1 at each iteration
            For k=1, until k = PeakNumber
                  Assign SegStartPnt(k))+I to row
                  Assign MatrixFinal(row,2) to ForceAti(k)
                  Assign MatrixFinal(row,3) to EMGAti (k)
            End for loop

      Find mean of vector ForceAti at assign mean to AverageForcei
      Assign AverageForcei to AverageSegment((i+1),2)
      Find mean of vector EMGAti and assign mean to AverageEMGAti
      Assign AverageEMGAti to AverageSegment((i+1),3)
End for loop

Apply normalize in range [0,100] to AverageSegment(:,2) and assign result to
AverageSegment(:,2)
Define AvgSegment as matrix of zeros with SegSizeInt as row number, 3 as
column number
Assign AverageSegment(:,1) to AvgSegABS(:,1)
Apply normalize in range [0,100] to AverageSegment(:,2) and assign result to
AvgSegABS (:,2)
Apply absolute value to AverageSegment(:,3)and apply result to AvgSegABS(:,3)

Apply findpeaks to AvgSegABS(:,2) and set 99.9 as minimum height of peaks
Store y-values of peaks in ForceMax
Store x-values of peaks in FrceMxPos
Apply findpeaks to AvgSegABS(:,3) and set 0.25 as minimum height of peaks
Store y-values of peaks in EMGMax
Store x-values of peaks in EMGMxPos

Define timestamps as vector of zeros of length EMGMxPos
Define egm as vector of zeros of length EMGMxPos
Define force as vector of zeros of length EMGMxPos

For i=1, until i = EMGMxPos, increment i in steps of 1
      Assign AverageSegment(i,1) to timeStamps(i)
      Assign AverageSegment(i,2) to force(i)
End for loop


subplot the first of four graphs in a column assigning AvgSegABS(:,1) to x
axis, AvgSegABS(:,2) to y axis
Assign 'Graph 4: Average Normalized Force and Time Relationship' to title
Assign 'time/s' to label of x axis
Assign 'Force/(%)' to label of y axis
```

subplot the second of two graphs in a column assigning MatrixFinal (:,1) to x axis, MatrixFinal (:,2) to y axis
Assign 'Graph 5: Absolute Average EMG and Time Relationship' to title
Assign 'time/s' to label of x axis
Assign 'EMG/v' to label of y axis

Apply fit using gauss1 method given AvgSegABS(:,1)+Shift as x-variable and AvgSegABS(:,3) as y-variable
Subplot the third of two graphs in a column assigning MatrixFinale(:,1) to X-axis, assign MatrixFinal (:,2) to y-axis, plot the model GaussEGM on the Same plot.
Assign 'Graph 6: Shifted Absolute Average EGM Model' to title
Assign 'time/s' to label of x axis
Assign 'EGM/v' to label of y axis
Apply coeffvalues to GaussEGM, and store coefficients into ecoeff
Assign ecoeff(1) *exp(-((timeStamps-ecoeff(2))/ecoeff(3)).^2)to egm

subplot the fourth of four graphs in a column assigning egm to x axis, force to y axis
Assign 'Graph 7: Force and EGM Relationship' to title
Assign 'EGM /V' to label of x axis
Assign 'Force (%)' to label of y axis

Task 4

Cancel previously plotted figure

Apply polyfit given egm as x-axis, force as y-axis, degree of polynomial equal to 2. Store model in m
Apply polyval given m and egm and store polynomial in p

Plot graph given egm as x-axis, force as y-axis. Curve should be formatted as'*'. Superimpose graph given egm as x-axis, p as y-axis. Curve should be formatted as'-');

Assign 'Graph 8: Empirical Data and Solution Comparison' to title
Assign 'EGM /V' to label of x axis
Assign 'Force (%)' to label of y axis
Assign 'Empirical values of Force', 'Model Prediction of Force' to legend 'Empirical values of Force' refers to first plotted curve 'Model Prediction of Force' to second plotted curve
Draw a grid on the graph
Assign to s 'y =(% float value 1, 1 decimal place precision) x^2 + (% float value2, 1 decimal place precision) x + (%float value 3, 1 decimal place precision)', given coefficients m(1),m(2),m(3)
Display s as text in position (0.035,60)
Apply rsquare to independent variable force, polynomial p. store result into output variables r2 rmse
Display as text in position (0.035,70) message 'R2 = ', r2 converted to character, ' RMSE = ', rmse converted to character

Tested functionalities

- Graph1: Showing the data before and after applying the filter should demonstrate a difference in the noise level). Showing data before and after normalizing data should place force in range [0,100]
- Showing EGM data over time, filtered.
- Graph: 3: Plot superimposed graph of smoothed force curve and force curve. Smoothed force curve should be within the smoothed force curve. The position of peaks and local minimum should correspond. This is necessary for the correct identification of segments
- Our program is able to find the number of peaks which is the number of cycles. Hence the number of peaks will be displayed. We expect the number of peaks to be 31
- Vector SegStartPnt should be displayed. It is expected to store the starting points of the 30 cycles. The starting points should nearly correspond to the position of local minimum in force-time graph. They can't correspond precisely since $SegSizeInt$ was casted to integer from a decimal value to obtain a fixed integer number. This leads to the loss of some data points which should be negligible for the purpose of our analysis
- Graph 4: Plot average force segment over time. We expect a bell shape in range [0,100], since the cycle starts and finishes with a relaxed hand which applies a force of 0.
- Graph 5: Plot average EGM segment over time. We expect one peak which precedes the peak for force.
- Graph 6: Plot model of EGM segment over time. The fit line of EGM should be a bell shape resembling the curve of the average force segment. Position of peak of EGM and peak of force should be made equal to facilitate regression.
- Graph 7: Plot of Force against EGM. We expect a positive relationship between the two variables: as the EGM signal increases, the force impressed also increases.
- Graph 8: A quantitative relationship between force and EGM should be displayed on the graph, value for coefficient of determination should be displayed. Both empirical and predicted values of force should be displayed to enable the assessment of our model

Step 4: Code or implementation

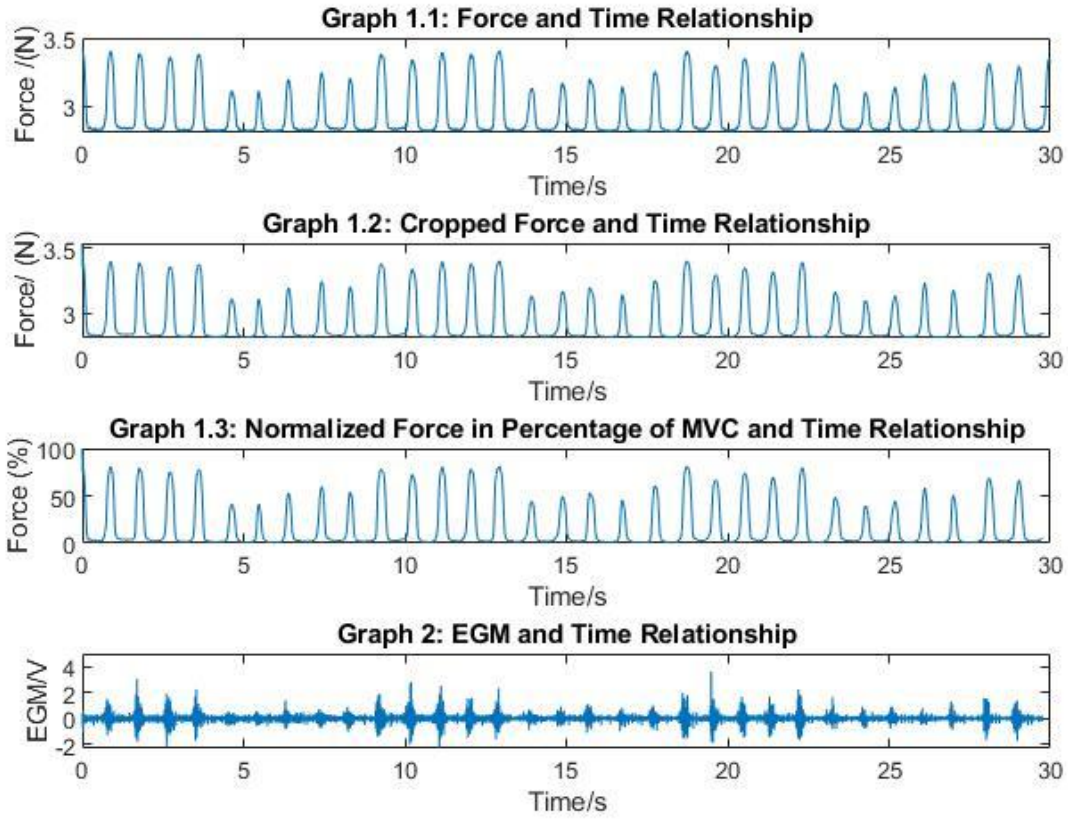
Our solution of the problem is presented in documents Task_1, Task_2, Task_3, Task_4. Which respectively include the sections 1, 2, 3 and 4 of the program. Script rsquare.m is used to find R^2 value in Task 4 to assess our relationship. The script belongs to Jered Wells (2020)

Step 5: Results and Analysis

The four parts of our program will be tested as shown in the command window below

```
Command Window
>> Task_1
>> Task_2
>> Task_3
>> Task_4
fx >>
```

Section 1:



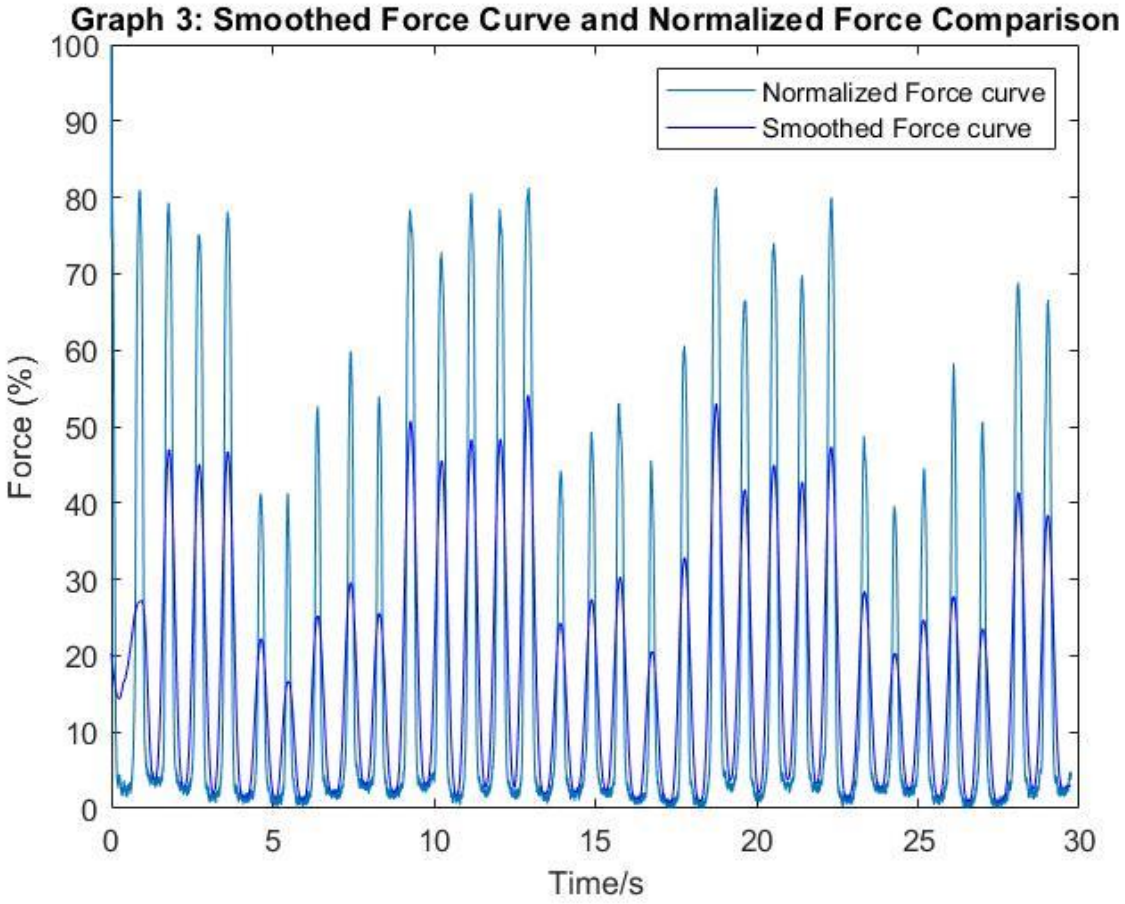
From Graph1, we infer that our program is able to read data from the given data file.

In graph 1.2 we notice that the first 500 values were not cut correctly.

In graph 1.3 The force is sucessfully normalized in the range [0,100]. However, we notice that the peaks have a maximum height of 80, and that Force = 100, only for x = 0, which is contradictory, since the hand is relaxed at the start of the cycle. This happened because we did not succeed in cropping the data correctly.

To **fix** this problem in section 2, the domain of our graph will be further restricted to include data points comprised between the positions of **first and last minimum** values of force; and the force will be normalized again in the range [0,100] so that peaks have the maximum height of 1.

Section 2:



In graph 3, we are able to see that the smoothed force curve is included in the normalized force curve. Therefore, the position of the maximum and minimum of smoothed and Normalized curve are equal and we can successfully apply the command findpeaks on the smoothed Force curve, in order to find the position of maximum and minimum values of the Normalized Force curve.

```
>> PeakNumber

PeakNumber =

     31
```

In graph 2, we count 31 peaks, indicating 31 cycles of compression of the hand. The number of peaks recognized by our program is 31, which is in line with our observation.

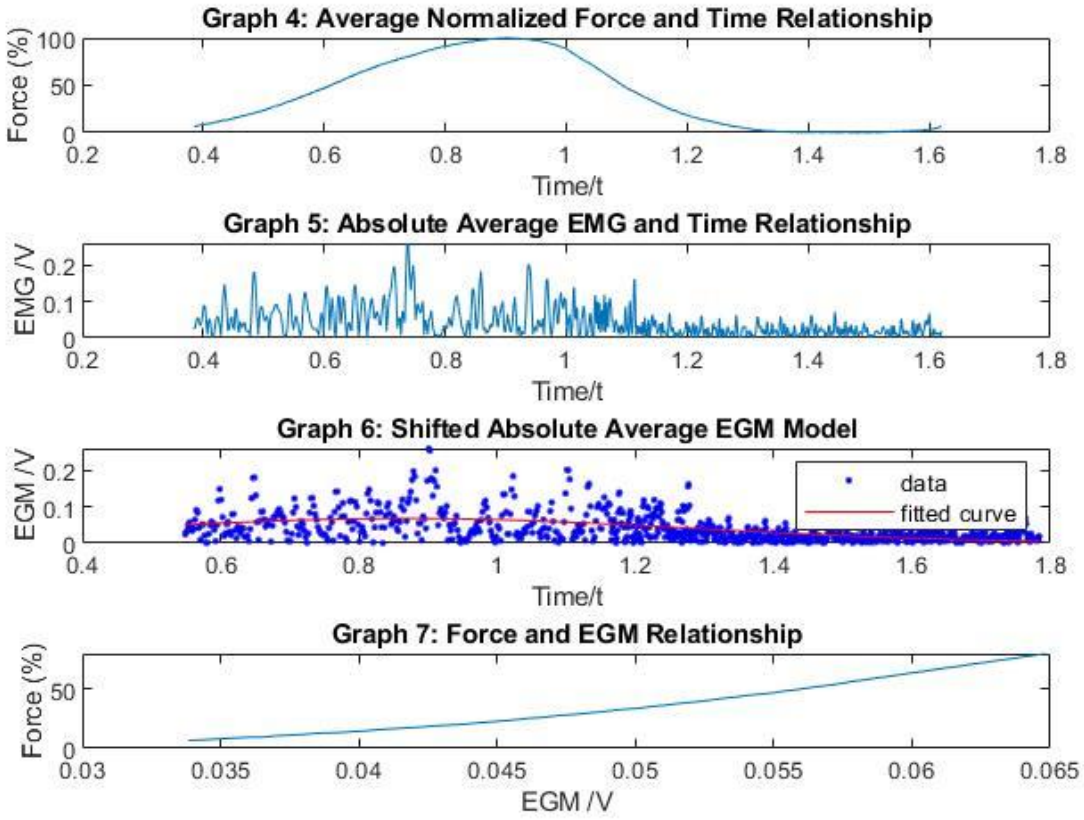
The starting point of each of our segments can be known by displaying the array SegStartPnt.

```
SegStartPnt =

  Columns 1 through 8

         192        1121        2050        2979        3908        4837        5766        6695

  Columns 9 through 16

        7624        8553        9482       10411       11340       12269       13198       14127

  Columns 17 through 24

       15056       15985       16914       17843       18772       19701       20630       21559

  Columns 25 through 31

       22488       23417       24346       25275       26204       27133       28062
```

```
idxMin =

         120
         192
         853
        1759
        2666
        3654
        4596
        5434
        6415
        7385
        8262
        9242
       10199
       11101
       11987
       12933
       13878
       14842
       15806
       16801
       17745
       18690
       19579
       20467
       21356
       22295
       23340
       24221
       25134
       26060
       27003
       28066
       29004
       29064
```

We can see that the starting points of segment almost correspond to the position of local minimum in force-time graph. The small discrepancy is justified since we required the number of data points of each cycle to be a constant integer. Position of first and Last minimum were discarded since they are incorrectly recognized by findpeak as local minimum. They could not signal the start of the cycle since the distance to the closest element was much smaller than $SegSizeInt$

Section 3:



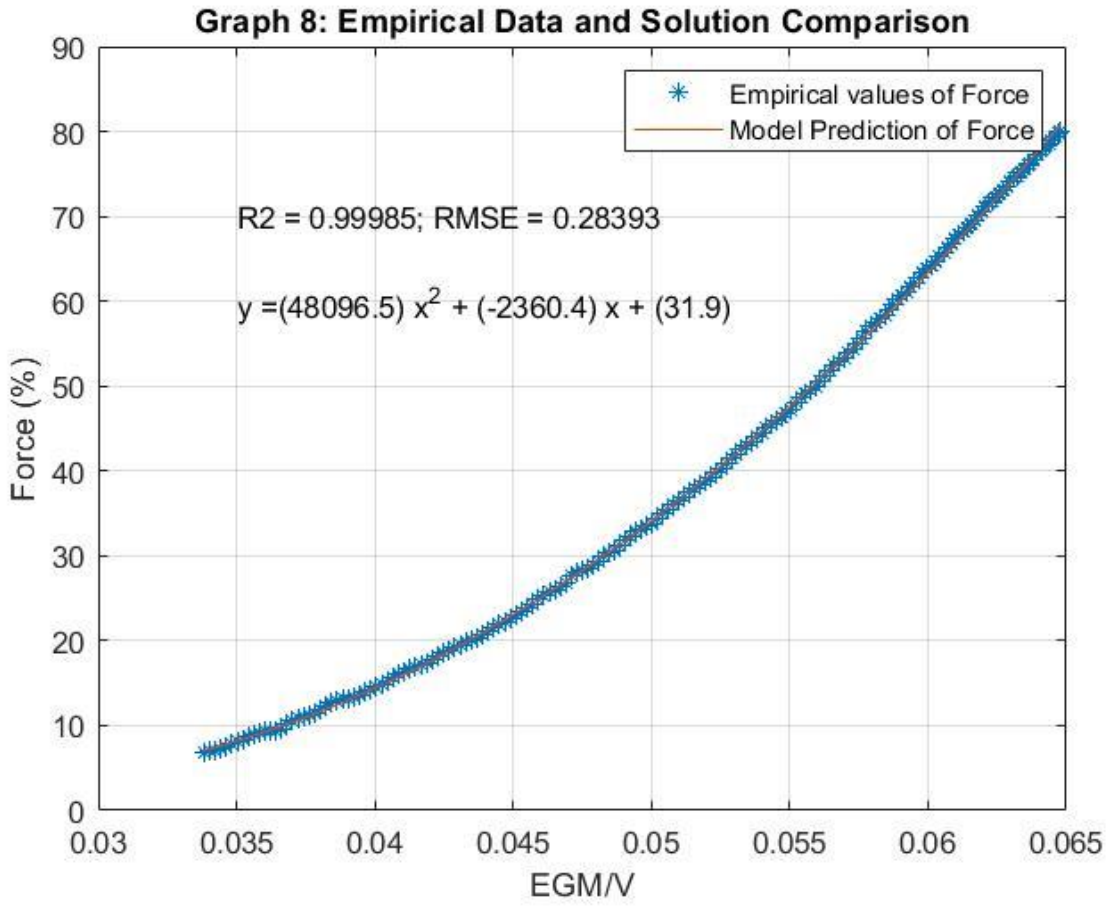
Graph 4: The average force for a segment is displayed. The force ranges from 0 to 100, hence it was correctly normalized. It displays the value of 0 at the start of the cycle, the maximum value of one in the middle of the cycle; and decreases to 0 at the end of the cycle. This is in line with the physical description of a hand which first loosely holds the sensor (Force = 0), then squeezes the sensor (F=1), to become loose again (F=0).

Graph 5: The peak of the EGM signal occurs before the peak of force. This is in line with our background: the EGM in a body is produced before the force manifests. Graph 5 shows the difference in nature of Force and EMG values. EGM fluctuates widely; and has many peaks.

Graph 6: The peak of our average EGM signal moved; and its position is now equal to the position of the peak of force, signaling that the EGM was correctly shifted. Our gauss model of EGM resembles in shape Graph 6, hence Force and EGM can now be compared

Graph 7: The relationship between Force and EGM signal is correctly displayed signaling that the data elaboration of the force an EMG is sufficient to enable a comparison of the two variables. There is a positive relationship between Force and EGM: as EGM increases the force impressed by the muscle also increases, which is in line with our background

Section 4:



This section of our program presents the following relationship between Force and EGM in Graph 8:

$$Force = 48096.5(EGM)^2 - 2360.4(EGM) + 31.9$$

Hence there is a quadratic relationship between the two variables: when the EGM doubles the Force quadruples.

The coefficient of determination $R^2 = 0.99985$ is very high. Therefore, we infer that our model is a good prediction for Force values. This can also be observed since our model coincides almost perfectly with the empirical values of the force. Hence, we could attempt now using our program to predict values of the force impressed by the forearm given EGM values for this individual.

Bibliography

Jered Wells (2020). R-square: The coefficient of determination (https://www.mathworks.com/matlabcentral/fileexchange/34492-r-square-the-coefficient-of-determination), MATLAB Central File Exchange. Retrieved December 15, 2020.