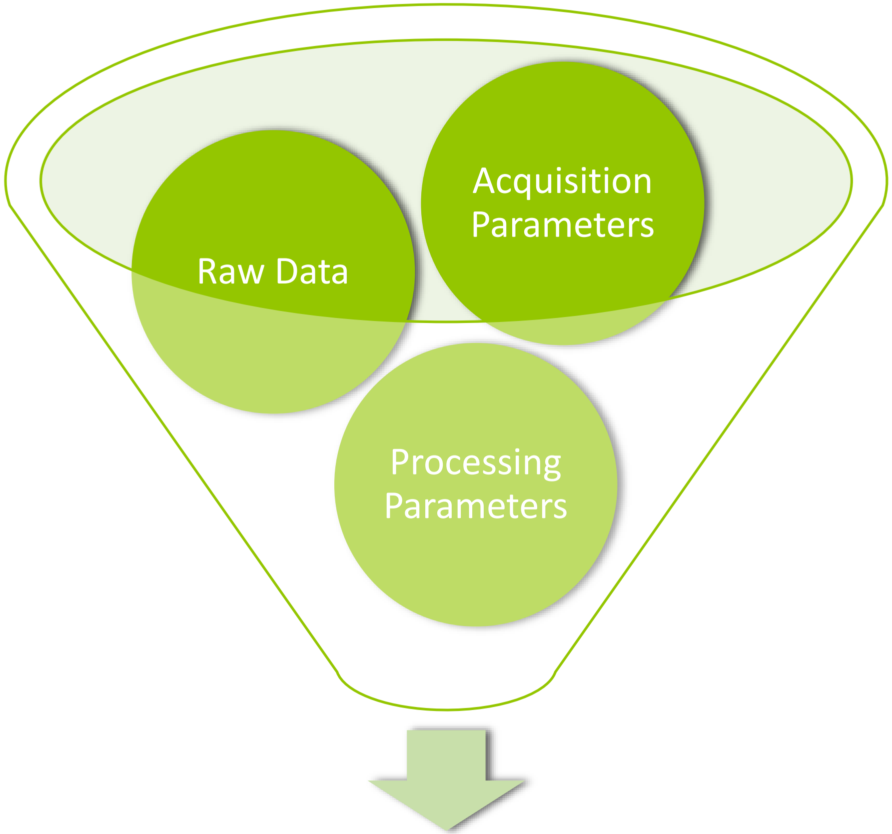
Nervous Activity Analyzer
Biological Signals

by Jonas, Robel, Maria

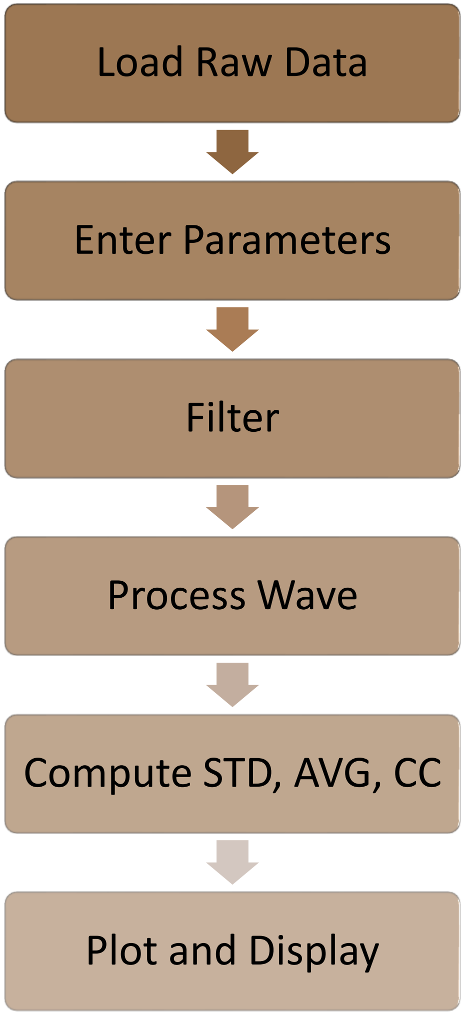
3-June-2014

INPUTS & OUTPUTS

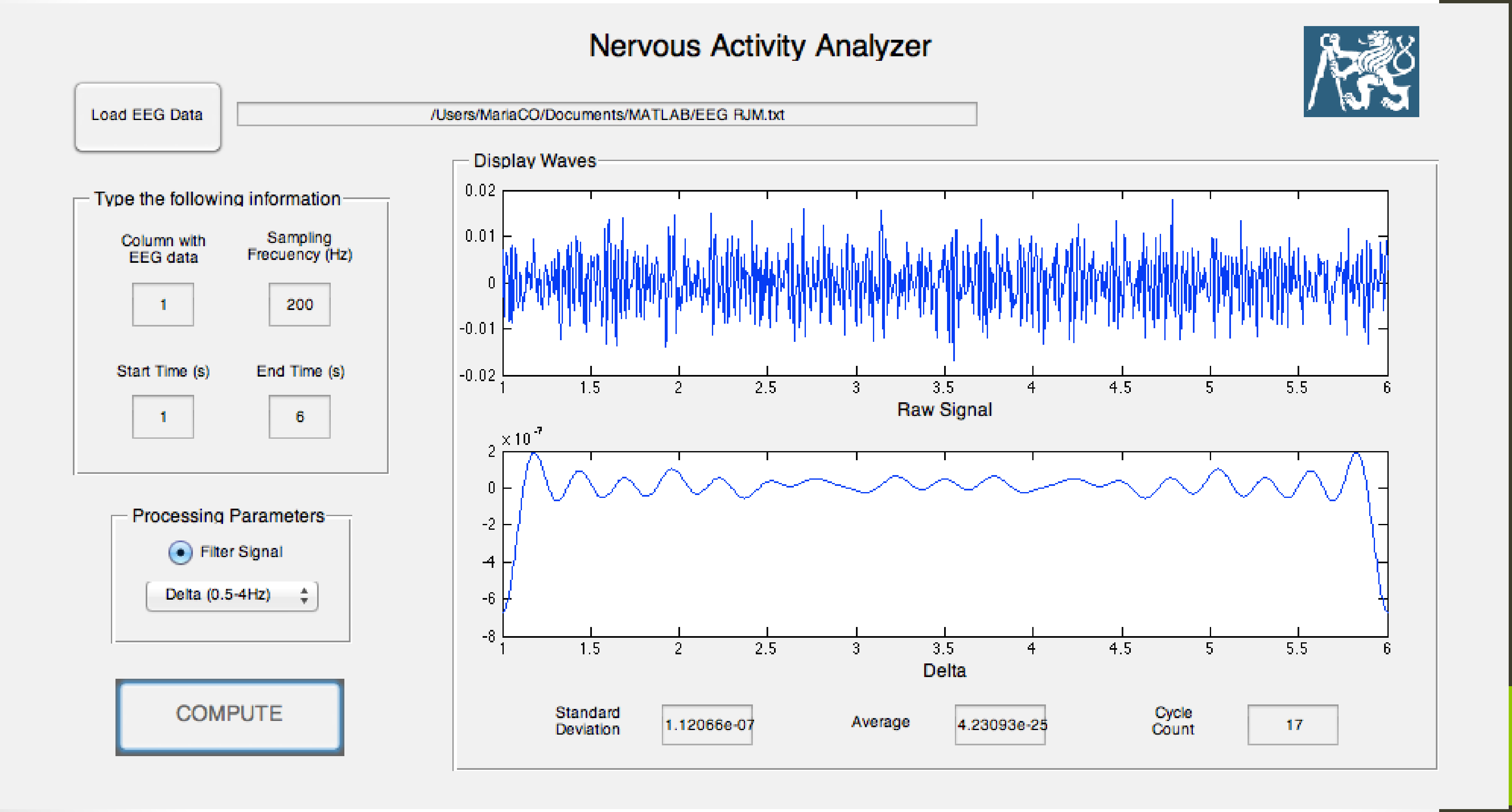


Computed Signal

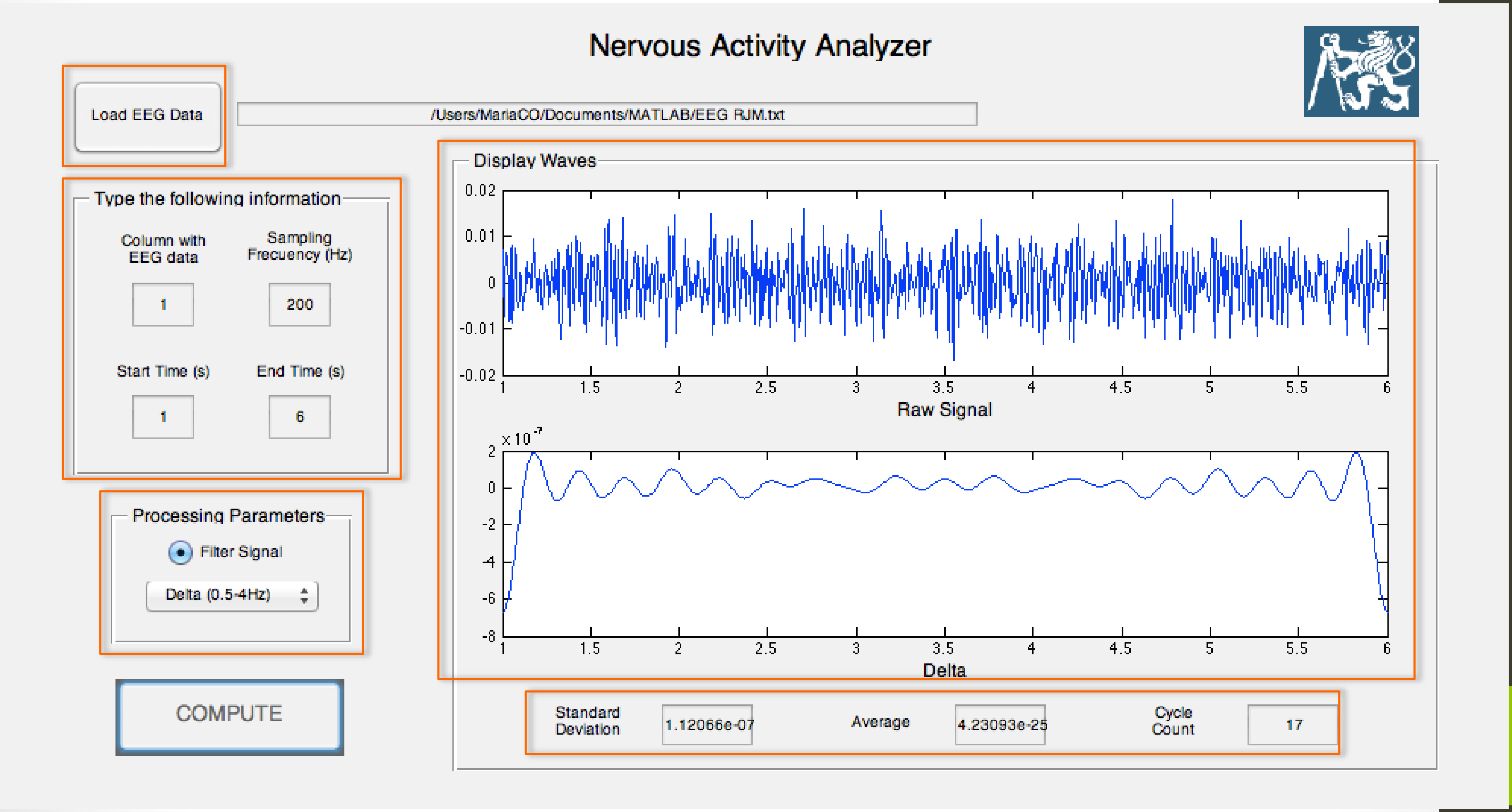
WORKFLOW


Load Raw Data
Enter Parameters
Filter
Process Wave
Compute STD, AVG, CC
Plot and Display

Graphical User Interface - GUI



Graphical User Interface - GUI



Load Raw Data

- [filename,filepath]=uigetfile({'*.*','All Files'},'Select EEG File');
- fullpathname=strcat(filepath, filename);
- set(handles.text_path, 'String', fullpathname)
- handles.rawdata1=load(fullpathname)

EEG RJM.txt

```
CH1 Input       alpha   beta    delta   theta   C5 -Gamma
-0.00994873     -9.43585E-006   4.98473E-005    4.37682E-005    0.000210477     0.000247318
-0.00567627     -0.000156764    -0.00062528     3.0851E-005     0.000148385     -0.00364706
-0.000350952    -0.000377901    -0.001375       -1.06451E-006   4.60298E-005    -0.00318385
0.00360107      -0.000531002    -0.00150483     -3.22119E-005   -4.39356E-005   0.00148266
0.00134277      -0.000565271    -0.000995866    -5.22267E-005   -9.71149E-005   0.00458012
-0.00401306     -0.000545968    -0.000391147    -6.76668E-005   -0.000132906    0.00300853
-0.00254822     -0.000552026    -0.000117415    -8.96792E-005   -0.000179997    -0.000607658
-0.000930786    -0.000576746    -5.38004E-005   -0.000118231    -0.000236956    -0.00182106
0.00521851      -0.000549408    0.000182988     -0.000142339    -0.000275852    0.000186972
0.00596619      -0.00040301     0.000826019     -0.000148582    -0.000265758    0.00297832
```

Enter Parameters

- handles.column1 = str2double(get(hObject,'String'));
- handles.fs1 = str2double(get(hObject,'String'));
- handles.t11 = str2double(get(hObject,'String'));
- handles.t21 = str2double(get(hObject,'String'));
- handles.Filter1 = (get(hObject,'Value'));
- handles.Wave1 = (get(hObject,'Value'));

Filter

- Filtering the EEG data by using FIR filters.
- It was filtered for the following:
  - 50Hz Powerline interference with a band stop filter
  - 0.5Hz Highpass and 90Hz Low pass filter
- FIR was chosen over IIR because:
  - FIR filters are inherently stable
  - FIR filters have a linear phase
  - FIR are robust to quantization

```
%FIR FILTERS
a = fir1(100,[f1n f2n], 'stop'); %50hz notch filter
y = filter(a,1,data1);
b = fir1(100,[f1l f2l], 'low'); %90Hz lowpass filter
x0 = filter(b,1,y);
c = fir1(100,[f1h f2h], 'high'); %0.5Hz highpass filter
x = filter(c,1,x0);
```

Specify values for Wn

- fir1 implements the classical method of windowed linear-phase FIR digital filter design.
- b = fir1(n,Wn,*'ftype'*)
- n specifies the order of the filter
- Wn = [f1 f2] is a two element vector between 0 and 1 where 1 corresponds on the Nyquist Frequency
- They were calculated using the following code

```
k = 0.003;
f1n = (50/(fs/2)) - k;
f2n = (50/(fs/2)) + k;
f1l = (90/(fs/2)) - k;
f2l = (90/(fs/2)) + k;
f1h = (0.5/(fs/2)) - k;
f2h = (0.5/(fs/2)) + k;
```

Process Wave

- Extraction of Delta, Theta, Alfa, Beta or Gamma wave using FFT & IFFT functions.
- 1. transform given signal to frequency domain using *FFT*
- 2. extract desired wave (frequency band) into blank vector using *switch*, based on user's selection

```
switch wave
case 1
   curve='Delta'
   B(round(m/2-0.5*m/fs):round(m/2-4*m/fs))=fur(round(m/2-0.5*m/fs):round(m/2-4*m/fs))
   B(round(m/2+0.5*m/fs):round(m/2+4*m/fs))=fur(round(m/2+0.5*m/fs):round(m/2+4*m/fs))
```

- 3. transform back into time domain using *IFFT*
- 4. create time vector based on sampling frequency and duration

```
t=t1:1/fs:t2
```

- 5. plot against the time vector

Compute STD, AVG, CC

- Standard deviation

```
Sdev=std(z)
```

- Average

```
Avg=mean(z)
```

- Cycle count

```
Cc=0
tr=0
for F = 2:m
       E=F-1;
       if tr==1
              if z(F)<z(E)
                     Cc=Cc+1
                     tr=-1
              end
       end
       if z(F)<z(E)
              tr=-1
       else
              tr=1
       end
end
```

Plot and Display

- plot(handles.plot1,t,data1)
- plot(handles.plot2,t,z)
-
- set(handles.Sel_wave,'String',curve)
- set(handles.STD1, 'String', STD)
- set(handles.AVG1, 'String', AVG)
- set(handles.CC1, 'String', CC)