In [20]:
```
import os
n = '22'
totalN = 22
exportData = 'Data\\Project\\'
motionImport = 'C:\\Users\\OndrejSpetko\\Desktop\\School\\MED7\\HRV-tracker\\Breathing\\motiondetection\\motiondetection\\Out\\'
sensorImport = 'C:\\Users\\OndrejSpetko\\Desktop\\School\\MED7\\HRV-tracker\\Breathing\\SerialListener\\Data\\User test\\'
```

In [2]:
```
#Motion file
from numpy import genfromtxt

motionRawData = genfromtxt(motionImport+n+'.txt', delimiter=',')
motionRawData.shape[0]
```

Out[2]:
```
3609
```

In [4]:
```
#Sensor file
FSR_IR = genfromtxt(sensorImport+n+'.txt', delimiter=',')
FSR_IR.shape[0]
```

Out[4]:
```
1256
```

In [5]:
```
#Match sizes
import numpy as np
import math

while motionRawData.shape[0] > FSR_IR.shape[0] :
    every = math.ceil(motionRawData.shape[0]/(motionRawData.shape[0] - FSR_IR.shape[0]))
    motionRawData = np.delete(motionRawData, list(range(0, motionRawData.shape[0], every)), axis=0)
motionRawData.shape[0]
```

Out[5]:
```
1256
```

In [6]:
```
#Time data
timeData = motionRawData[:, 0]
```

In [7]:
```
# Motion data
from scipy import signal
temp = motionRawData[:, 1]
motionData = signal.savgol_filter(temp, 151, 3)
motionData = np.interp(motionData, (motionData.min(), motionData.max()), (0, 10))
#motionDataNorm
```

In [8]:

```
# Normalize FSR data
FSRData = np.array(FSR_IR[:, 0])
FSRData = signal.savgol_filter(FSRData, 151, 3)
FSRData = signal.savgol_filter(FSRData, 151, 3)
FSRData = np.interp(FSRData, (FSRData.min(), FSRData.max()), (0, 10))
#FSRDataNorm.shape
```

In [9]:

```
# Normalize IR data
IRData = np.array(FSR_IR[:, 1])
IRData = signal.savgol_filter(IRData, 151, 3)
IRData = signal.savgol_filter(IRData, 151, 3)
IRData = np.interp(IRData, (IRData.min(), IRData.max()), (0, 10))
#IRDataNorm.shape
```

In [10]:

```
#Scatter plot
import plotly as py
import plotly.graph_objs as go
import plotly.io as pio

# Create traces
trace0 = go.Scatter(
    x = timeData,
    y = motionData,
    mode = 'lines',
    name = 'Motion'
)
trace1 = go.Scatter(
    x = timeData,
    y = FSRData,
    mode = 'lines',
    name = 'FSR'
)
trace2 = go.Scatter(
    x = timeData,
    y = IRData,
    mode = 'lines',
    name = 'IR'
)

#data = [trace0, trace1, trace2]
data = [trace0, trace1, trace2]

# Plot and embed in ipython notebook!
#pio.write_image(fig, exportImage+'fig1.png')
py.offline.plot(data, filename='Scatter.html')
```

Out[10]:

```
'file://C:\\Users\\OndrejSpetko\\Desktop\\School\\MED7\\HRV-tracker\\Breathin
g\\PostProcessing\\Python\\Scatter.html'
```

In [11]:

```
#Correlation coefficient
#np.corrcoef(motionData, FSRData), np.corrcoef(motionData, IRData), np.corrcoe
f(FSRData, IRData)
```

In [12]:

```
#Bar plot correlation
corr1 = np.corrcoef(motionData, FSRData)[0][1]
corr2 = np.corrcoef(motionData, IRData)[0][1]
corr3 = np.corrcoef(FSRData, IRData)[0][1]
data2 = [go.Bar(
            x=['Motion vs. FSR', 'Motion vs. IR', 'IR vs. FSR'],
            y=[corr1, corr2, corr3],
            text=[corr1, corr2, corr3],
            textposition = 'auto',
)]

layout = go.Layout(
    title='Correlation Coefficient of Sensors',
)

fig = go.Figure(data=data2, layout=layout)
py.offline.plot(fig, filename='Bar.html')
```

Out[12]:

```
'file://C:\\Users\\OndrejSpetko\\Desktop\\School\\MED7\\HRV-tracker\\Breathin
g\\PostProcessing\\Python\\Bar.html'
```

In [13]:

```
#Joining time + motion
temp = np.vstack((timeData, motionData))

size1 = temp[0].size
size2 = temp.size

temp = np.reshape(temp, size2, order='F')

temp = temp.reshape((size1, 2))
temp1 = temp
temp1.shape
```

Out[13]:

```
(1256, 2)
```

In [14]:

```
#joining FSR + IR
temp = np.vstack((FSRData, IRData))
size1 = temp[0].size
size2 = temp.size

temp = np.reshape(temp, size2, order='F')

temp = temp.reshape((size1, 2))
temp2 = temp
temp2.shape
```

Out[14]:

```
(1256, 2)
```

In [15]:

```
#Joining TimeMotion + FSRIR
final = np.hstack((temp1, temp2))
final, final.shape
```

Out[15]:

```
(array([[ 0.27698347,  0.17787944,  0.        , 10.        ],
        [ 0.31069183,  0.19216205,  0.09209894,  9.61185074],
        [ 0.38104832,  0.21048024,  0.18518009,  9.23447463],
        ...,
        [60.322742  ,  0.70283099,  1.28494056,  0.49277445],
        [60.35646   ,  0.78781354,  1.26721024,  0.53878921],
        [60.389397  ,  0.87809825,  1.25177358,  0.58708732]]), (1256, 4))
```

In [16]:

```
#Saving to file
if not os.path.exists(exportData+n):
    os.mkdir(exportData+n)
np.savetxt(exportData+n+'\\'+n+'.txt', final, delimiter=",", fmt='%s')
np.savetxt(exportData+n+'\\'+n+'C.txt', [['MF', 'MI', 'FI'], [corr1, corr2, co
rr3]], delimiter=",", fmt='%s')
```

In [21]:

```
#Average coef values
# coefFSR = 0.0
# coefIR = 0.0
# coefBoth = 0.0
# total = 0;
# for x in range(totalN):
#     n = str(x+1)
#     if os.path.exists(exportData+n):
#         temp = genfromtxt(exportData+n+'\\'+n+'C.txt', delimiter=',')
#         coefFSR += temp[1][0]
#         coefIR += temp[1][1]
#         coefBoth += temp[1][2]
#         total += 1
#     else:
#         print(n+" does not exist")
# coefFSR /= total
# coefIR /= total
# coefBoth /= total
# total, coefFSR, coefIR, coefBoth
```

```
11 does not exist
13 does not exist
```

Out[21]:

```
(20, 0.8004124875732593, 0.20311572904551775, 0.22608387608865596)
```

In [22]:

```
#Plot average coef values
#Bar plot correlation
corr1 = coefFSR
corr2 = coefIR
corr3 = coefBoth
data2 = [go.Bar(
            x=['Motion vs. FSR', 'Motion vs. IR', 'IR vs. FSR'],
            y=[corr1, corr2, corr3],
            text=[corr1, corr2, corr3],
            textposition = 'auto',
)]

layout = go.Layout(
    title='Mean Correlation Coefficient of Sensors across '+str(total)+' samples',
)

fig = go.Figure(data=data2, layout=layout)
py.offline.plot(fig, filename='Bar.html')
```

Out[22]:

```
'file://C:\\Users\\OndrejSpetko\\Desktop\\School\\MED7\\HRV-tracker\\Breathing\\PostProcessing\\Python\\Bar.html'
```