In [281]:

```
# import python modules
import pyedflib
import numpy as np
import sys
import os
import pandas as pd
from scipy import signal
from scipy import interpolate
from scipy import integrate
from scipy import stats
from matplotlib import pyplot as plt
%matplotlib widget
```

In [249]:

```
# define custom functions
def butter_bandpass(x,fl,fh,fs,order):
    fnyq = fs/2
    fl = fl/fnyq
    fh = fh/fnyq
    sos = signal.butter(N=order,Wn=[fl,fh],btype='bandpass',output='sos')
    xf = signal.sosfiltfilt(sos,x)
    return(xf)
# }
def butter_lowpass(x,fl,fs,order):
    fnyq = fs/2
    fl = fl/fnyq
    sos = signal.butter(N=order,Wn=fl,btype='lowpass',output='sos')
    xf = signal.sosfiltfilt(sos,x)
    return(xf)
# }
def butter_highpass(x,fh,fs,order):
    fnyq = fs/2
    fh = fh/fnyq
    sos = signal.butter(N=order,Wn=fh,btype='highpass',output='sos')
    xf = signal.sosfiltfilt(sos,x)
    return(xf)
# }
def load_eegdata(fname):
    print("Loading eeg data from {} ...".format(fname))
    with pyedflib.EdfReader(fname) as f:
        header = f.getHeader()
        filedur = f.getFileDuration()
        nsamples = f.getNSamples()
        nchannels = f.signals_in_file-3
        prefilter = f.getPrefilter(0)
        samplefreqs = f.getSampleFrequencies()[0:-3]
        channels = f.getSignalLabels()[0:-3]
        datetime = f.getStartdatetime()
        dd = np.zeros((nsamples[0],nchannels))
        for i in np.arange(nchannels):
            dd[:,i] = f.readSignal(i)
#       }
#   }
    print("EEG data loading successfull.")
    print("nChannels: {}".format(nchannels))
    print("EEG channel names: {}".format(channels))
    print("EEG data samples: {}".format(nsamples))
    print("File Header: {}".format(header))
    print("Data shape: {}".format(dd.shape))
    return(dd,samplefreqs,channels)
# }

def eegplot(dd,fs,channels):
    nsamples = dd.shape[0]
    nchannels = dd.shape[1]
    t = np.arange(0,nsamples)/fs
    fh, ah = plt.subplots(nrows=nchannels, sharex=True,squeeze = True)
    for i in np.arange(0,nchannels):
        ah[i].plot(t,dd[:,i])
        ah[i].spines['right'].set_visible(False)
        ah[i].spines['top'].set_visible(False)
        ah[i].spines['bottom'].set_visible(False)
        ah[i].set_title(channels[i])
#   }
    ah[-1].spines['bottom'].set_visible(True)
    fh.tight_layout()
    fh.subplots_adjust(top=0.9)
    return(fh,ah)
# }

def eegpower(dd,fs,freqbands,epochdur):
#     power spectral density at different epochs using Welch's method
    nchannels = dd.shape[1]
    nsamples = dd.shape[0]
    nsamplesepoch = int(epochdur*fs)
    nperseg = int(nsamplesepoch*0.2)
    nepochs = int(np.floor(nsamples/(epochdur*fs)))
    samples_omit = nsamples-(nepochs*nsamplesepoch)
    print("nepochs: {}".format(nepochs))
    print("nsamples: {}".format(nsamples))
    print("nsamplesepoch: {}".format(nsamplesepoch))
    print("samples omitted {}".format(samples_omit))
    dd2 = dd[0:nepochs*nsamplesepoch,:]
    print(dd.shape)
    dd2.resize((nsamplesepoch,nepochs,nchannels))
    print(dd2.shape)
    powers = np.zeros((nepochs,nchannels,len(freqbands)))
    print(features.shape)
    freqkeys = list(freqbands.keys())
    for i in range(0,nepochs): # epochs
        for j in range(0,nchannels): # channels
            for k in range(0,len(freqkeys)):
                x = dd2[:,i,j]
                freqband = list(freqbands[freqkeys[k]].values())
                xf = butter_bandpass(x,*freqband,fs,order=2) # bandpass filter the signal
                f,pxx = scipy.signal.welch(xf,fs=fs,nperseg=nperseg)
                freq_res = np.mean(np.diff(f))
                ifreqband = [np.where(f > freq)[0][0]-1 for freq in freqband]
                f = f[ifreqband[0]:ifreqband[1]]
                powers[i,j,k] = integrate.simps(pxx[ifreqband[0]:ifreqband[1]],dx=freq_res) # absolute power
                _,pxxf = scipy.signal.welch(x,fs=fs,nperseg=nperseg) # compute periodagram on the whole unfiltered signal
                powers[i,j,k] = powers[i,j,k]/integrate.simps(pxxf,dx=freq_res) # compute relative power
#                  print("freqency resolution: {}".format(freq_res))
#                  fh = plt.figure()
#                  ah = fh.add_subplot(111)
#                  ah.plot(f,pxx)
#               }
#           }
#       }
    return(powers)
```

In [6]:

```
# load eeg data
datapath_eeg = "/home/anup/goofy/myprojects/eeg/"
fname_eeg = "eeg data.edf"
fullname_eeg = os.path.join(datapath_eeg,fname_eeg)
eeg,fs,channels = load_eegdata(fullname_eeg)
```

```
Loading eeg data from /home/anup/goofy/myprojects/eeg/eeg data.edf ...
EEG data loading successfull.
nChannels: 44
EEG channel names: ['EEG Fp1-Ref', 'EEG F7-Ref', 'EEG T3-Ref', 'EEG T5-Ref', 'EEG O1-Ref', 'EEG F3-Ref', 'EEG C3-Ref', 'EEG P3-Ref', 'EEG A1-Ref', 'EEG Fz-Ref', 'EEG Cz-Ref', 'EEG Fp2-Ref', 'EEG F8-Ref', 'EEG T4-Ref', 'EEG T6-Ref', 'EEG O2-Ref', 'EEG F4-Ref', 'EEG C4-Ref', 'EEG P4-Ref', 'EEG A2-Ref', 'EEG Fpz-Ref', 'EEG Pz-Ref', 'EEG LT EMG1-Ref', 'EEG LT EMG2-Ref', 'EEG RT EMG1-Ref', 'EEG RT EMG2-Ref', 'EEG LT RESP-Ref', 'EEG RT RESP-Ref', 'EEG X7-Ref', 'EEG X8-Ref', 'EEG X9-Ref', 'EEG X10-Ref', 'EEG X11-Ref', 'EEG X12-Ref', 'EEG X13-Ref', 'EEG X14-Ref', 'EEG X15-Ref', 'EEG X16-Ref', 'EEG ECG-Ref', 'EEG X18-Ref', 'EEG DC1-Ref', 'EEG DC2-Ref', 'EEG DC3-Ref', 'EEG DC4-Ref']
EEG data samples: [7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792
 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792
 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792
 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792
 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792 7233792
 7233792 7233792]
File Header: {'technician': '', 'recording_additional': '', 'patientname': 'Xxxx Xxxxxxxxxxxx', 'patient_additional': '', 'patientcode': 'xxxxxxx', 'equipment': 'Exported with Persyst EEGSuite', 'admincode': '', 'gender': '', 'startdate': datetime.datetime(2017, 7, 6, 11, 43, 40), 'birthdate': '01 jan 2017'}
Data shape: (7233792, 44)
```

In [20]:

```
# visualize eeg channels
print(fs)
fh,ah = eegplot(eeg[1:,0:3],fs[0],channels[0:3])
plt.show()
```

```
[256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256
 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256
 256 256 256 256 256 256 256 256]
```

In [435]:

```
# load glucose data
datapath_glu = "/home/anup/goofy/myprojects/eeg/"
fname_glu = "glucose data.xlsx"
fullname_glu = os.path.join(datapath_glu,fname_glu)
glu = pd.read_excel(fullname_glu)
print(glu)
f  = interpolate.interp1d(glu["epoch"],glu["glc_all"],kind='linear')
epochs_new = np.arange(3,169,1)
glu_new = f(epochs_new)
glu_intrp = pd.DataFrame({"epoch":epochs_new,"glu":glu_new})
fh = plt.figure()
ah = fh.add_subplot(111)
```

```
# ah.plot(epochs_new,glu_new,'o')
ah.plot(glu_intrp["epoch"],glu_intrp["glu"],'o')
ah.plot(glu["epoch"],glu["glc_all"],'o')
# ah.set_xlim([0,100
print(glu_intrp)
```

```
    epoch  glc_all
0       3      4.5
1      12      5.6
2      14      5.5
3      24      5.8
4      40      5.5
..    ...      ...
100   165      6.9
101   166      6.9
102   167      6.8
103   168      6.8
104   169      6.7

[105 rows x 2 columns]

    epoch       glu
0       3  4.500000
1       4  4.622222
2       5  4.744444
3       6  4.866667
4       7  4.988889
..    ...       ...
161   164  6.900000
162   165  6.900000
163   166  6.900000
164   167  6.800000
165   168  6.800000

[166 rows x 2 columns]
```

In [250]:

```
freqbands = {"delta":{"low":0.5,"high":4},"theta":{"low":4,"high":8},"alpha":{"low":8,"high":13},"beta":{"low":13,"high":30},\
            "gamma":{"low":30,"high":50},"total":{"low":0.5,"high":50}}
epochdur = 300
powers=eegpower(eeg,fs[0],freqbands,epochdur)
print(powers.shape)
```

```
nepochs: 94
nsamples: 7233792
nsamplesepoch: 76800
samples omitted 14592
(7233792, 44)
(76800, 94, 44)
(94, 44, 6)
(94, 44, 6)
```

In [372]:

```
# clip data to align the two datasets
# eeg starts at epoch 24
print(glu_intrp.shape)
glu_intrp2 = glu_intrp[(glu_intrp["epoch"]>23) & (glu_intrp["epoch"]<(24+94))].reset_index(drop=True)
# glu_intrp2["epoch"] = glu_intrp2["epoch"]-24
print(glu_intrp2.shape)
print(powers.shape)
# powers = powers[0,:,:]
```

```
(166, 2)
(94, 2)
(94, 44, 6)
```

In [436]:

```
plt.close('all')
fh = plt.figure()
ah = fh.add_subplot(111)
print(channels[10:20])
ah.plot(np.arange(0,powers.shape[0]),powers[:,10:22,:].mean(axis=1).mean(axis=1)*40-8)
ah.plot(glu_intrp2["epoch"]-24,(glu_intrp2["glu"]-glu_intrp2["glu"].mean())/glu_intrp2["glu"].std(),'-o')
# powers[20,:]
```

```
['EEG Cz-Ref', 'EEG Fp2-Ref', 'EEG F8-Ref', 'EEG T4-Ref', 'EEG T6-Ref', 'EEG O2-Ref', 'EEG F4-Ref', 'EEG C4-Ref', 'EEG P4-Ref', 'EEG A2-Ref']
```

Out[436]:

```
[<matplotlib.lines.Line2D at 0x7fc8d7ee3d68>]
```

In [475]:

```
# display correlation between power in EEG channels and glucose levels
plt.close('all')
x = glu_intrp2["glu"].to_numpy()
x = (x-x.mean())/x.std() # z-score data
fh, ah = plt.subplots(nrows=10, ncols=5,sharex=True,sharey=True,squeeze = True)
for i in range(0,10):
    for j in range(0,5):
        y = powers[:,i,j]
        y = (y-y.mean())/y.std() # z-score data
        ah[i,j].plot(x,y,'o',markersize=1,linewidth=1,color='red')
        ah[i,j].spines['right'].set_visible(False)
        ah[i,j].spines['top'].set_visible(False)
        ah[i,j].spines['bottom'].set_visible(False)
#        ah[i,j].set_title(channels[i])
```

In [ ]:

```

```

In [486]:

```
# find correlations using a linear regression model
rvalues = np.zeros((44,6))
pvalues = np.zeros((44,6))
x = glu_intrp2["glu"].to_numpy()
for i in range(0,44):
    for j in range(0,6):
        y = powers[:,i,j]
        slope, intercept, rvalues[i,j], pvalues[i,j], std_err = stats.linregress(x,y)
# convert results to a dataframe for easy extraction of specific values
rvalues = pd.DataFrame(data=rvalues,index = channels,columns=list(freqbands.keys()))
pvalues = pd.DataFrame(data=pvalues,index = channels,columns=list(freqbands.keys()))
# print(list(freqbands.keys()))
# print(pvalues.head)
```

In [625]:

```
plt.close('all')
fh = plt.figure()
ah = fh.add_subplot(111)
x= np.arange(-0.5,44,1)
y = np.arange(-0.5,6,1)
z = rvalues
# z = pvalues
print(x.shape,y.shape,z.shape)
ph = ah.pcolormesh(x,y,z.T,cmap='hot',vmin=-0.5,vmax=0.5)
ah.set_title("Correlation $(\itR)$ between EEG power and blood glucose levels\n")
ah.set_xlabel("EEG Channels",fontsize=16)
ah.set_ylabel("EEG frequency bands",fontsize=16)
yticks = list(freqbands.keys())
ah.set_yticks(np.arange(0,6))
ah.set_yticklabels(yticks)
ch = fh.colorbar(ph)
ch.set_label("Corerlation ($\itR$)")
print(yticks)
print(channels[24])
plt.tight_layout()
```

```
(45,) (7,) (44, 6)
['delta', 'theta', 'alpha', 'beta', 'gamma', 'total']
EEG RT EMG1-Ref
```

In [624]:

```
plt.close('all')
fh = plt.figure(figsize=(6,5))
ah = fh.add_subplot(111)
x= np.arange(-0.5,44,1)
y = np.arange(-0.5,6,1)
# z = rvalues*rvalues
z = pvalues
print(x.shape,y.shape,z.shape)
ph = ah.pcolormesh(x,y,z.T,cmap='hot',vmin=0,vmax=0.05)
ah.set_title("Significance for correlation $(\itR)$ between EEG power & blood glucose\n")
ah.set_xlabel("EEG Channels",fontsize=16)
ah.set_ylabel("EEG frequency bands",fontsize=16)
# ah.yaxis.set_label_coords(-0.1,1)
yticks = list(freqbands.keys())
ah.set_yticks(np.arange(0,6))
ah.set_yticklabels(yticks)
ch = fh.colorbar(ph)
ch.set_label("Significance ($\itP_{value}$)")
plt.tight_layout()
```

```
(45,) (7,) (44, 6)
```

In [614]:

```
plt.close('all')
glucose = glu_intrp2["glu"].to_numpy()
# glucose = (glucose - glucose.mean())/glucose.std()
channelid = 6
channelname = channels[channelid]
print(channelname)
gamma_power = powers[:,channelid,4]
gamma_power = (gamma_power - gamma_power.mean())/gamma_power.std()
```

```
t = glu_intrp2["epoch"].to_numpy()
# t = t-t[0]
t = t
fh = plt.figure()
ah = fh.add_subplot(111)
ah.plot(t,glucose,label="Blood glucose")
ah.plot(t,gamma_power,label= "".join((channelname," Gamma power")))
title = "".join(("Correlation between ", channelname, " Gamma power & blood glucose\n"))
ah.set_title(title)
ah.set_xlabel("Time (min)")
ah.set_ylabel("Z-score values",fontsize=18)
ah.legend()
```

```
EEG C3-Ref
```

```
Out[614]: <matplotlib.legend.Legend at 0x7fc88fcd2ac8>
```