Practicalities of analysing biosignals

August 18, 2015

```
In [9]: import IPython.display as display
        import numpy as np
        import matplotlib.pyplot as plt
        import matplotlib as mpl
        %matplotlib inline
```

1 Practicalities of analysing biosignals

1.1 Dr. Emlyn Clay Head of Software Development Viditeck AG, Director of OpenVivo ltd.

2 Who am I?

- Undergraduate Pharmacology (˜2009) and doctorate of Pharamcology (2015)
- Developing software since I was 18, ˜10 years
- Web stuff (PHP), then VBscript, then MATLAB, then Python, then C . . . etc
- I spend most of my time programming and distributing biomedical equipment.

```
In [5]: display.Image(filename='images/emlyn.jpg')
```

Out[5]:

3 Committee member of PyData London!

- Monthly meetup for data science peoples using Python.
- ~1800 members, regular 200 people meetups.
- Yearly conference.
- Advocate the Python community.

```
In [22]: display.Image(filename='images/pydata_logo.png')
```

```
Out[22]:
```

4 Emlyn said ...

To: dgorissen@gmail.com
CC: "london@pydata.org" <london@pydata.org>
Subject: PyData London Call for Proposal - mind circulating?

Hello Dirk,

Do you mind forwarding the following call for proposal, below, for our PyData London Conference 2015 on

Thanks Dirk,

Emlyn

Hello Big Omegas,

...

5 Dirk said ...

Date: Fri, 8 May 2015 22:00:53 +0100
Subject: Re: PyData London Call for Proposal - mind circulating?
To: Emlyn Clay <eclay101@gmail.com>

On the condition you come do a talk ;) I have a slot week of 17 August :)

Dr. Dirk Gorissen
Research - Tech4Good - Flying Robots
Skype: dirk.gorissen
Mob: +44-7763-806-809
Twitter : https://twitter.com/dirkgor
LinkedIn: http://www.linkedin.com/in/dirkgorissen

Bribery!

6 Biosignals

Definitvely -

> A biosignal is any signal in living organisms that can be measured and monitored, contiually or intermittently. Some are bioelectrical, but it may refer to both electrical and non-electrical signals.

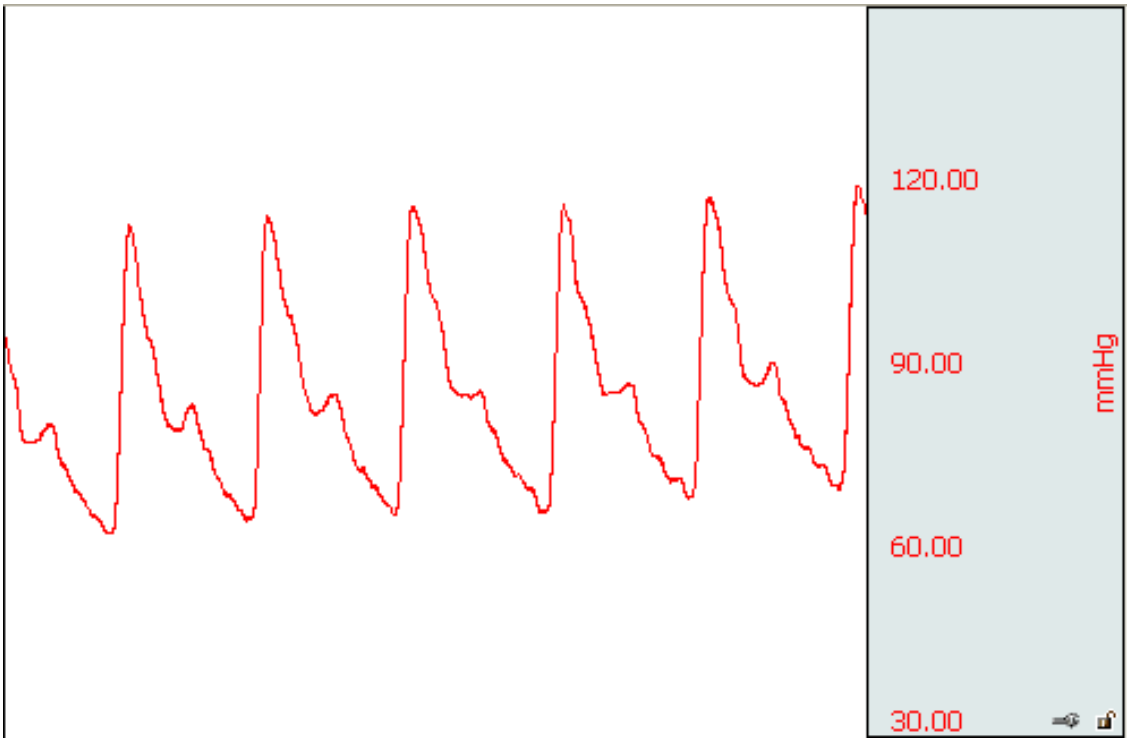
Analysing them can be used to assess:

- Healthy "normative" states
- Disease states
- Fundamental understanding

6.1 Blood pressure

```
In [10]: display.Image(filename='images/signal_blood-pressure.png')
```

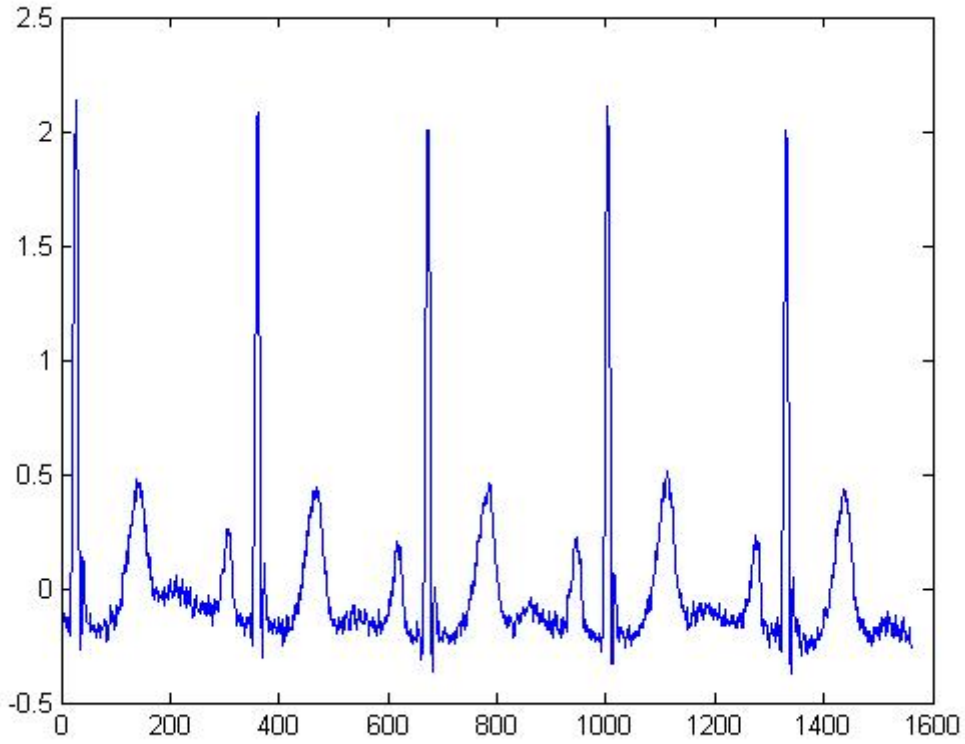
Out[10]:



6.2 ECG

```
In [12]: display.Image(filename='images/signal_ECG.jpeg')
```

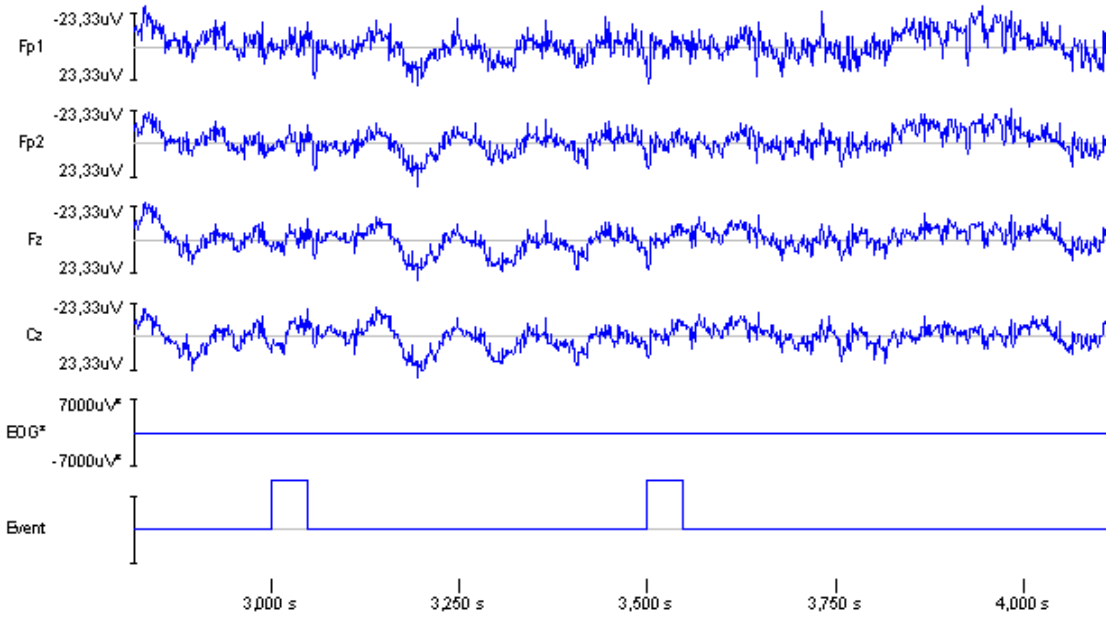
Out[12]:



6.3 EEG

In [13]: display.Image(filename='images/signal_EEG.png')

Out[13]:



7 What are we looking at today?

Practicalities.

- Record biosignals
- Storing them
- Process them
- Analyse them

8 Recording biosignals

Things you are going to need -

- Sensor
- Amplifier
- Analogue to Digital Convertor
- Storage media

8.1 Sensors

- A thing that attaches to the person,
- Possibly a box that is powering and conditioning the signal
- Something that connects to your amplifier

```
In [18]: display.Image(filename="images/sensors.png")
```

```
Out[18]:
```

8.2 Amplifier and data acquisition

Often they are in the same box.

```
In [19]: display.Image(filename="images/gusbamp.png")
```

```
Out[19]:
```

8.3 Storage media

Often it's a laptop or sometimes some embedded storage like a flash disk

```
In [20]: display.Image(filename="images/digital_storage.jpg")
```

```
Out[20]:
```

9 Salient points on equipment

Sensors + Calibrated + Sensitive, but not noisy

Amplifier/DAQ + Certified for use (CE, FDA) — not strictly necessary for hobbyist use, + *Don't connect hobby tech to mains! Use a battery.* + High raw sampling rate — oversampling. + Low noise, high input impedance.

Storage + Make sure it can handle your bandwidth + Get lots of it - one minute of ECG uncompressed, ˜120Mb (2 x 1024 x 60)

10 Storing Biosignals

Effectively; large vectors of doubles and some metadata about what we've recorded.

11 HDF5!

12 HDF5

- Portable
- Bindings to everything (C, Fortran, Python, Matlab, Java ...)
- Supports contiguous or chunked datasets
- Performant.

13 HDF5 in Python

```
In [3]: import h5py
        import numpy as np

        f = h5py.File("ecg.h5", "w")
        dset = f.create_dataset("ECG", (1024,), dtype='f')
```

14 HDF5 datasets

Store the raw signal in one dataset, store a processed signal in another.

```
In [6]: normal_ECG = f.create_dataset("normal_ECG", (65772,), dtype='f')
```

15 HDF5 attributes

You can store metadata right next to the data it describes! Yay!

```
In [8]: normal_ECG.attrs['species'] = 'human'
        normal_ECG.attrs['subject'] = 'emlyn'
        normal_ECG.attrs['date'] = '12/06/2007'
        normal_ECG.attrs['sex'] = 'Yes please'
```

16 Processing biosignals

Essentially, Digital Signal Processing — same rules apply. Biosignals tends to be continuous, periodic and complex waveforms.

16.1 Basic a.ka. cleaning the signal

- Filtering noise
- Correct the baseline
- Smoothing

16.2 Advanced a.k.a feature detection

- Wavelet convolution
- Peak detection
- Morphology analysis

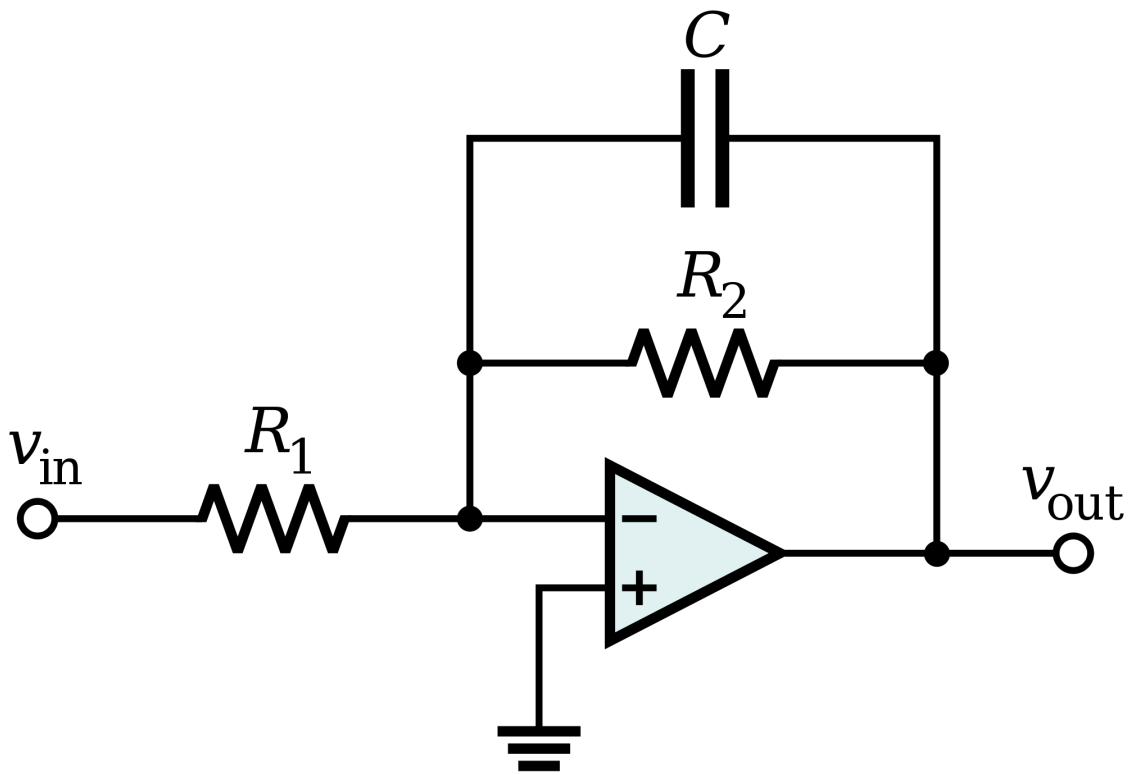
17 Filtering noise

The signal you are looking for is usually between a specific wavelength so you can apply bandpass filters to just focus on the region of interest.

. . . you can do this with analogue electronics

```
In [18]: display.Image(filename='images/lowpass_filter.png', width='400px')
```

Out[18]:

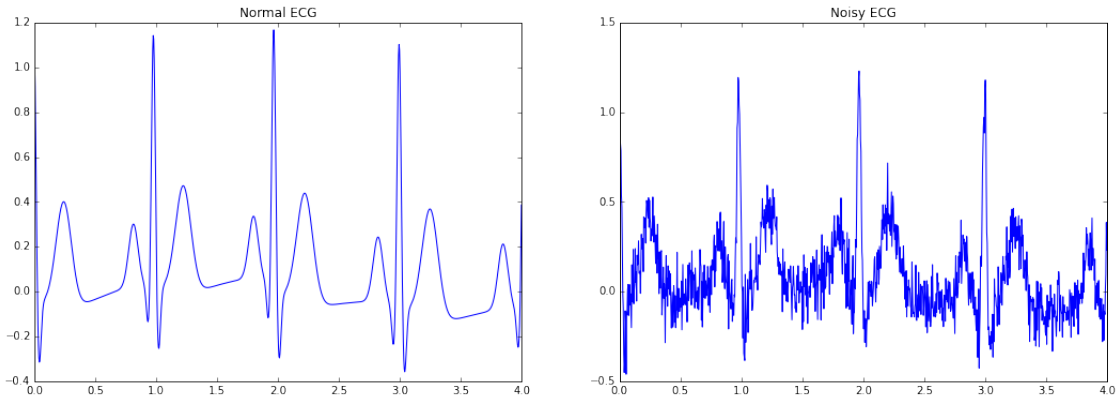


. . . but, then you have to worry about all sorts of compromises due to frequency responses and cut-off definition.

Filtering noise with the FFT

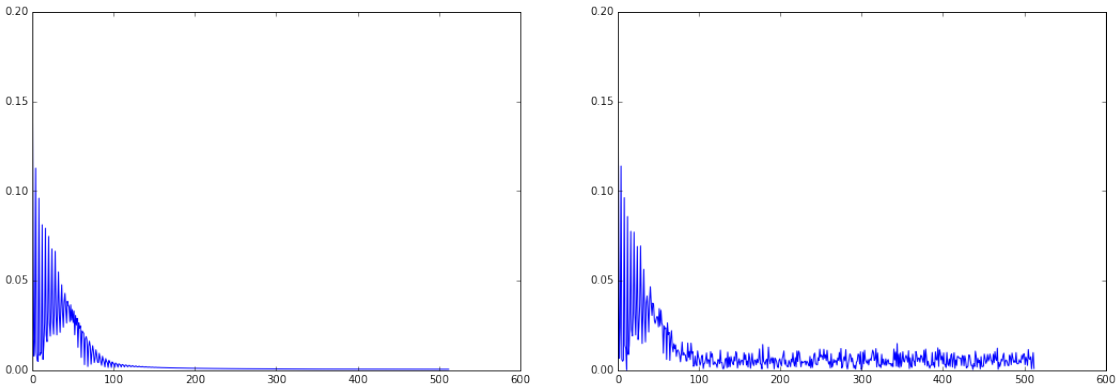
The grand conceit of Fourier's work was that all sinusoids can be described as a series of sinusoids superimposed on one another.

```
In [48]: # Make a signal
         samples = 1024
         time = np.loadtxt('data/ecgsyn.dat', usecols=(0,))
         ecg = np.loadtxt('data/ecgsyn.dat', usecols=(1,))
         noise = np.random.normal(0,0.1,samples)

         figure = plt.figure(figsize=(18, 6))
         ax1 = figure.add_subplot(121)
         ax2 = figure.add_subplot(122)

         ax1.set_title('Normal ECG')
         ax1.plot(time[:samples], ecg[:samples])

         ax2.set_title('Noisy ECG')
         ax2.plot(time[:samples], ecg[:samples]+noise)

Out[48]: [<matplotlib.lines.Line2D at 0x10c86fa50>]
```




```
In [49]: normal_ecg_fft = np.fft.fft(ecg[:samples])
         noisy_ecg_fft = np.fft.fft(ecg[:samples]+noise)

         figure = plt.figure(figsize=(18, 6))
         ax1 = figure.add_subplot(121)
         ax2 = figure.add_subplot(122)

         freqs = np.linspace(0.0, 1.0/(2.0*(1/samples)), samples/2)

         ax1.set_title('Normal ECG FFT')
         ax1.plot(freqs, 2.0/samples * np.abs(normal_ecg_fft[0:samples/2]))

         ax2.set_title('Noisy ECG FFT')
         ax2.plot(freqs, 2.0/samples * np.abs(noisy_ecg_fft[0:samples/2]))
```

```
Out[49]: [<matplotlib.lines.Line2D at 0x10e05b1d0>]
```



... zero the numbers of the frequencies you don't want and you filter them from the signal.

18 Correct a baseline

18.1 ... for quiet signals (ECG)

- Substract the modal value away from the signal

- FFT and filter lower frequency components

18.2 . . . for noisy signals (EEG)

- Substract the mean away from the signal
- Substract a window mean also.

19 Smoothing

Noise is often random so a moving-average moving-exponential window are good, but there are caveats. There is a tradeoff between:

- How smooth the signal is
- How precise the peaks (and troughs) are

. . . I often use the Savitzky-Golay smooth with biosignals because it favours maintaining the shape of the signal.

19.1 Intro. to Signal Processing:Smoothing - University of Maryland

```
In [51]: display.IFrame('http://terpconnect.umd.edu/~toh/spectrum/Smoothing.html', width="100%", height=
```

```
Out[51]: <IPython.lib.display.IFrame at 0x10b9ce590>
```