HomeAssignement2_blankv2

July 12, 2019

1 Home assignement: Stochastic models

This tutorial illustrates the manipulation of stochastic models and signals.

1.1 Part 1: Processing and spectral analysis of Local Field Potentials

Local Field Potentials (LFPs) designate the "low" frequency part of the electrical potential in the extracellular medium of the brain, typically below 200Hz. It reflects many network processes, and is assumed to be generated, to a large extended, by post-synaptic currents entering the neurons.

Let us load some exemplary data recoded from the macaque hippocampus that you can dowload from https://owncloud.tuebingen.mpg.de/index.php/s/REc82bzpCetG27T (update the dataPath variable below before loading).

In [1]:
```
# load local field potential
dataPath = 'C:\\Users\\mitch\\MBsource\\scripts\\spcourse'
from scipy import io
import numpy as np
import matplotlib.pyplot as plt

lfpdat = io.loadmat(dataPath+"\\LFPdat.mat")

# Define LFP time series
LFP = lfpdat['LFP'][:,0]
# Define sampling period dt (sampling frequency is 660Hz)
dt = lfpdat['dt'][:,0]
t = np.arange(LFP.shape[0])*dt

# we plot only the begining of the recording
plt.plot(t[0:1000],LFP[0:1000])
plt.title('LFP time course')
plt.xlabel('time (s)')
plt.show()
```

```
<Figure size 640x480 with 1 Axes>
```

We want to estimate the Power Spectral Density of the LFP signal. We use the scipy.signal.welch function for this purpose. **Please refer to the spectral analysis tutorial on Ilias (tutorial 2) for the use of this function.**

Important: plot all periodograms with a physical frequency axis (Hz)

Question 1: - (a) The "nperseg" controls the width (number of points of the segments) used by the algorithm to compute a single DFT (with the FFT algorithm). We want to get a frequency resolution of approximately 1Hz, how should we choose (as an order of magnitude) the "nperseg" parameter? Plot the welch periodogram of the LFP signal. What are the dominant frequencies (in Hz) in this signal. - (b) The time resoved power in the so called gamma band (typically between 60 and 120Hz) of the LFP has been reported to reflect the amount of recurrent activity in the network, and is related to the BOLD signal measured in fMRI (see part 3). Perform the following operations to extract this time resolved power: - filter the LFP in the [60,120Hz] band to obtain signal g (use scipy.signal.filtfilt to apply a zero phase filter), - take the square of g, g^2, - low pass filter g^2 below $60Hz$ to get g_{pow} the time resolved gamma power. For each signal (LFP,g ,g^2,g_{pow}), estimate the PSD with the Welch periodogram and plot it. - (c) Given that the periodogram reflects (to some extent, see lecture) the expected squared modulus of the Fourier transform of signal realisations, justify qualitatively the shape taken by the successive PSDs (why particular frequencies appear or are preserved/removed...). - (d) Plot the square root of g_{pow} and check visually it follows the enveloppe of oscillations in g correctly. - (e) Based on the above periodogram of g_{pow}, justify that downsampling it by taking every 5th point is reasonable (hint: reason as if the downsampled signal corresponds to directly sampling the corresponding continuous signal, compute the new sampling frequency and the Nyquist frequency to justify). - (e) Downsample g_{pow} as explained above, to obtain g_d and compute an estimate of its PSD (using welch), with the same frequency resolution. Which window size should be taken now in the welch function? Based on comparing the PSD of g_d with the PSD of g_{pow}, is the downsampling well behaved? Justify qualitatively why the chosen frequency was appropriate.

1.2 Part 2: Estimation of auto and crosscorrelation functions

Given a single realisation of an observed time series (i.e. on single "trial"), to estimate the quantities γ_{XX} and γ_{XY} defined in the lecture (for X,Y jointly stationary processes), we essentially replace the expectation $\mathbb{E}$ in the formulas by an average over time $\frac{1}{T}\sum_{t=1}^{T}$. This time of estimate is implemented correctly in the *statsmodels* library, with *statsmodels.tsa.statstools.acovf* and *ccovf* (note the naming covariance differs from the term we used in the lecture, but they designate the same object).

Install the library and check the code of the functions available at https://www.statsmodels.org/stable/tsa.html, note they call the *numpy.correlate* routine. Note statsmodels has a conda library available for installation if you use anaconda (see https://anaconda.org/anaconda/statsmodels).

```
In [30]: from statsmodels.tsa.stattools import acovf,ccovf
```

Question 2: - (a) Explain what does the "unbiased" option implemented by the acovf and ccovf routines. - (b) From reading the code (and the numpy routine "correlate" used in it), the routines return a one-sided function (set of lags start from zero). Given a vector for sample x and y, should we use ccovf(x,y) or ccovf(y,x) to estimate γ_{XY} as defined in the lecture **for the positive values of lag** τ? Justify. - (c) How can we use the acovf and ccovf routines to compute the auto- and cross-correlations to get the functions on an axis spanning both positive and negative

lags? Write the code to do that and justify with symmetries shown in the lecture. - (d) Generate a white noise realization η with 1000 i.i.d. Gaussian samples. Using the option "ubiased=False", estimate the autocorrelation function for both positive and negative lags (-999 to 999) and plot it. Compare the obtained estimate with the theoretical value. - (e) Repeat the analysis with the option "unbiased=True". What changes in the estimate and why? - (f) Compute a 10 samples-lagged version ξ of the above noise sequence η and adjust the length of the original sequence η so that both signals have the same length. Estimate the cross-correlation function $\gamma_{\eta\xi}$ (using "unbiased=False") between the two series, for both positive and negative lags, and plot it. Does the estimate correspond to the theory, justify?

1.3 Part 3: A LTI model of neurovascular coupling

Functional Magnetic Resonance Imaging measures the Blood Oxygen Level Dependent (BOLD) signal related to the concentration of oxyhemoglobin in the brain tissue and is used to assess functional brain activity.

While the link between neural activity and BOLD is complex, involving several non-linear phenomena, and still elusive, the fMRI community has since long used a linear time invariant system to model this relation. The impulse response of this model is called the Hemodynamic Response Function (HRF), and can be used to investiagte properties of neural activity based on fMRI measurements.

1.3.1 Reference HRF

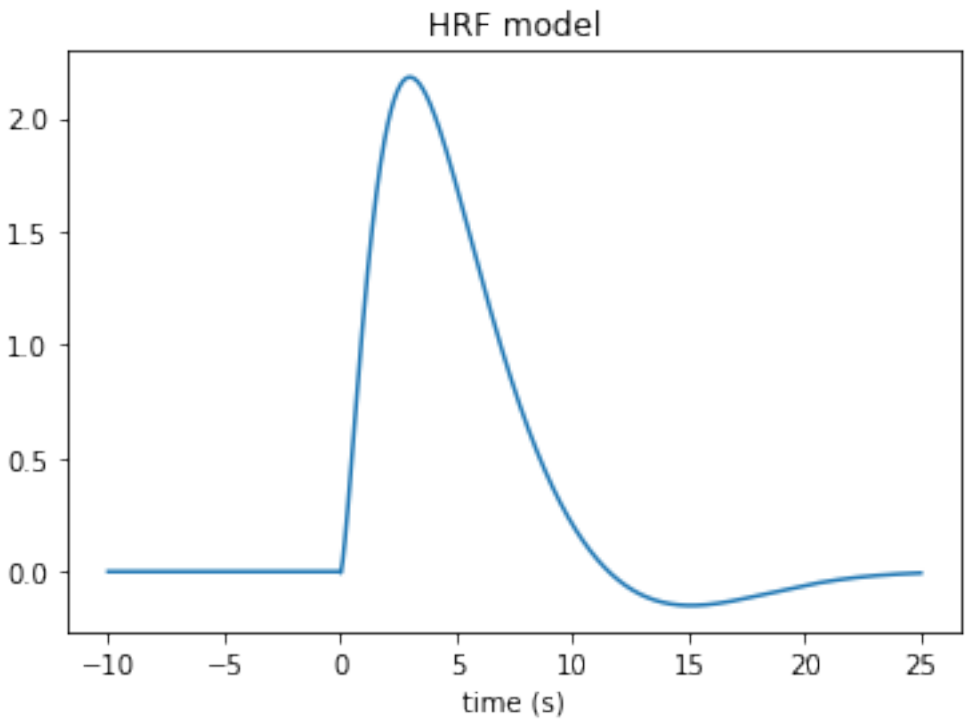
We first define a reference HRF in continuous time, similar to those used in popular software such as SPM (https://www.fil.ion.ucl.ac.uk/spm/).

In [58]:
```
import scipy.stats as stats

def hrf(t):
    pars = [2.5, 2.02, 14.5, 0.247]
    resp = pars[2]*stats.gamma.pdf(t,pars[0],scale = pars[1])-pars[3]*np.exp(-(t-25/2)
    resp[t<0] = 0
    return resp

t= np.arange(-10,25,.01)

plt.plot(t,hrf(t))
plt.title('HRF model')
plt.xlabel('time (s)')
plt.show()
```



Question 3: - Use the *hrf* function to sample the impulse response at the sampling frequency of the above downsampled signal g_d (approximately 132Hz). Note the impulse response is causal (values before $t = 0$ are zero). We can thus sample the hrf on an interval [0,25s]. Use this sampled HRF, that we will call h to compute an estimate y of the BOLD signal in response to a rectangular signal $x_T(t) = \begin{cases} 1, \text{ if } t \in [0, T] \\ 0, \text{otherwise} \end{cases}$ idealized signals x representing neural activity, for $T = .1s, 1s$ and $10s$. Comment (briefly) on the similarity between the shape of the HRF and the shape of the response y for different values of T (use properties of convolution seen in the lectures).

1.3.2 LFP based model

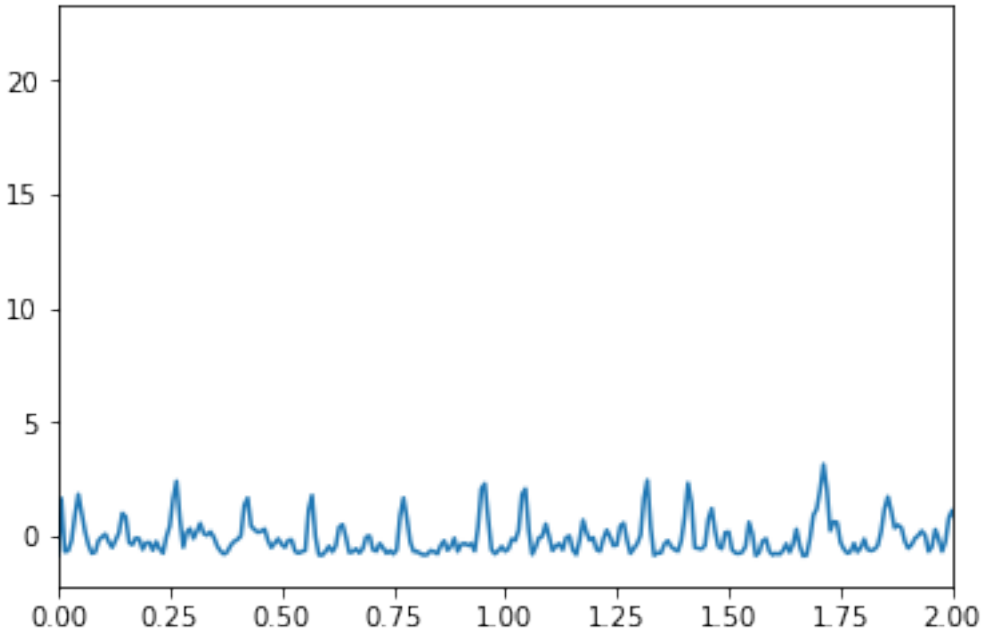
As an approximation for the neural activity, i.e. the input of the HRF-based model, we can take the time resolved gamma band power g_d computed above. You can check these papers for further details: https://www.ncbi.nlm.nih.gov/pubmed/11449264 https://www.jneurosci.org/content/32/4/1395

Let us reload the previously computed gamma power signal available at https://owncloud.tuebingen.mpg.de/index.php/s/3HN9jAmGW7miZ4t, sampled at approximately 132Hz (1/dt). We will also remove the mean and normalize the amplitude for simplicity. We will now call this normalized input signal x.

```
In [103]: gpow = np.load(dataPath+'\\gammaPower.npz')
          x = gpow['x']
          x = x -np.mean(x)
          x = x/np.std(x)
```

```
dt = gpow['dt']
t = np.arange(len(x))*dt
plt.plot(t,x)
plt.xlim([0,2])
```

```
Out[103]: (0, 2)
```



Question 4 - (a) Generate a simulated BOLD signal y by convolving the gamma power signal x by the sampled HRF h, and then adding a Gaussian white noise of variance 1. - (b) Compute and plot the autocovariance functions of x and y, and the cross-covariance γ_{xy} between them for a lag axis of [-40,40] **seconds** (normalize their amplitude to their respective maximum to allow comparison on the same graph). - (c) We aim at estimating the filter h based only on its input x and output y. Use the auto- and cross-correlation function estimates to build the matrix A and vector d seen in lecture 8 (identification of a FIR model, **check the updated slides on Ilias**). To construct A, you can check and use the *scipy.linalg.toeplitz* routine. For simplicity, choose the size of the maximum delay p that corresponds to the true h. - (d) Invert the system $Ab = d$ in order to get an estimate $\hat{h}$ of h. Is the fit qualitatively good? - (e) The original estimate was based on the full time course of the signals (600s). Quantify how varies the mean square error of the estimate of h when we estimate auto- and cross-correlation functions only from on initial segment of data of 30, 100, and 200s. Compare the performance of using biased ("unbiased=False") and unbiased estimates in these cases.