AI for Medical Time Series

Lecture 3 exercises

March 6, 2024

Introduction

In this exercise set, you will explore concepts covered in the third lecture. You will practice how to transform time series data to the frequency domain. This week we will be working with electromyography (EMG) data and simulated data. EMG data can be found in file *'emg_healthy.npy'* which has the shape of (number of time points x number of channels) on Ilias. The sampling rate is 4000 Hz.

These exercises require the Numpy, SciPy, and Matplotlib libraries and Python. Please use **only** these libraries for this exercise set. Solutions with other packages will **not** be accepted, and you will not get any points. Please use comments to indicate which sub-task your are answering (# Exercise 1a, etc.). The exercise will be marked as PASSED if you get 18 / 24 points or more. Points are only awarded for exercises where your code produces the expected result, and where you provide comments describing what the code does.

If you have any questions please e-mail *pinar.goektepe@unibe.ch*. You should describe what you have done so far, and what issues you have encountered.

The exercises should be handed in by a group of two students. Copying code or solutions of individuals outside the group (e.g. submitting the code of other individuals as your own) will result in 0 points.

The solutions must be handed in via **ILIAS**. Deliver your submission as a compressed file (zip) containing one .py or jupyter notebook file. Please make sure to name the zip file as follows:

HW_homeworkNumber_GroupID_surname1_name1_surname2_name2.zip.

Deadline: 14:00, March 13

Exercises

1. **Implementing convolution** ....12 point

   (a) **Generating signal** .... 1 point
   Generate a 1-dimensional signal with 100 data points. All data points should be zero except the data points at the index between 50 and 70 ($mySignal$[50:70]). These data points should have the value of 1.

   (b) **Generating kernel** .... 1 point
   Generate a 1-dimensional kernel with size of 6. The values of the kernel should be [1. 0.8 0.6 0.4 0.2 0] (keep the order the same while generating the kernel).

   (c) **Implementing convolution** .... 8 point
   Implement your own Python function that takes a signal and a kernel as parameters and computes their convolution **without using built-in convolution functions** available in existing libraries (e.g. NumPy, Scipy, etc.) This function should return the output signal after the convolution. Call your convolution function with the signal and kernel you created before.

   Important : Solutions with any library's convolution function will not be accepted.

   (d) **Plotting signals** .... 1 point
   Plot the signal that you created before and after the convolution, as well as the kernel.

   (e) **Describing the effects of convolution** .... 1 point
   Describe how the convolution affects the signal that you created.

2. **Filtering EMG data via Fourier transformation** .... 12 points

   (a) **Importing raw data** .... 1 point
   Import the data file $'emg\_healthy.npy'$.

   (b) **Applying Fourier transformation** .... 3 points
   Apply a Fourier transformation to decompose data to its frequency components and plot the power spectrum. You are allowed to use Fourier transformation functions from SciPy library.

   (c) **Band-pass filtering via Fourier transform** .... 4 points
   Apply a high pass filter at 200 Hz and low pass filter at 400 Hz using the Fourier transformation. Plot the power spectrum after filtering.

   Important : Solutions filtering the data without using a Fourier transform will not be accepted.

   (d) **Transforming to the time domain from frequency domain** .... 2 points
   After the filtering via Fourier transform, go back to the time domain by computing the inverse Fourier transform of filtered data and plot reconstructed raw data. You are allowed to use inverse Fourier transformation functions from SciPy library.

(e) **Describing the effects of filtering** .... 2 points

Plot the first 300ms of the raw data before filtering and reconstructed data after filtering and describe the differences you observe.