Physics 5730/7730, Spring 2022 Homework #4

Due Sunday March 27th

1. One of the most significant physics discoveries within the past decade was the first observation of gravitational waves[1] from a binary black hole merger by the LIGO scientific collaboration. In this exercise we will use tools introduced in class to extract the gravitational waveforms from the raw observational data.

   The time series data in files `GW150914_H.dat` and `GW150914_L.dat`[2] contain the strain signal from the first merger event. The observation was made simultaneously in Hanford (WA) and Livingston (LA). The files contain a single column of data, samples of the strain taken at 4096 Hz. The starting times are given in the file.

   Write a script **hw04a.py**, that reads the data, does the filtering and generates a plot, **hw04a.pdf**, showing the strain for both the Hanford and Livingston data. In the time coordinates correspondng to the data in the files, the time of the event is $t = 1126259462.44$ seconds. In your plot, display the horizontal axis in units of seconds and the vertical axis in units of noise standard deviation. Some hints:

   - Your code should assume the data files are in the same directory as **hw04a.py**. A good option for reading the files is **numpy.loadtxt()**.
   - Process the data from the Hanford and Livingston detectors separately, but in parallel.
   - **numpy.fft.rfft()** takes the fast Fourier transform (FFT) of a real function, returning it as an array. Calling **np.fft.rfftfreq(nt,dt)** (where **nt** is the number of points in the time series and **dt = 1/4096**) gets an array with the corresponding frequencies.
   - **SciPy**'s **signal.welch()**, applied to the raw data, provides an estimate of the power spectral density (PSD) of the data. It returns two arrays; the frequencies and the PSD evaluated at those frequencies. The frequencies will have units of Hz if you call **signal.welch()** with argument **fs=4096**, which is the sampling rate of the data.
   - Now, the arrays output by **signal.welch()** will be "sparsified". However, in order to use them we need arrays of the same length as the FFT output. Use scipy's **interp1d()** to create a function based on **signal.welch()**'s returned arrays, so that you can get interpolated values of the PSD at arbitrary frequencies.
   - Create a filter W as an array containing $1/\sqrt{\text{PSD}}$, being sure to evaluate W at all the frequencies involved in the FFT (this is why **interp1d()** is needed). Also mask out frequencies below about 25 Hz and above about 300 Hz (the range outside which LIGO is not expected to be sensitive to gravitational waves). Using $1/\sqrt{PSD}$ "whitens" the data (makes the power more evenly distributed across the frequency spectrum like white light). Apply the filter W to the frequency-domain signal.
   - Do an inverse FFT (**numpy.fft.irfft()**) to get back into the time domain. At this point, if you've done everything correctly the gravitational wave pulses should be visible roughly in the center of the whitened waveform.
   - As a last step, we want to
   - Estimate the background noise level by calculating the standard deviation of the filtered time series. *n.b.*: after filtering, the time series may have "edge" artifacts at the beginning and end points, so I would mask out/remove the first and last second of the time series when computing standard deviation. Your gravitational wave signals should be about 5–10 σ above the noise.
   - Now you're ready to use these signals to do astrophysics. Have fun!

[1]B.P. Abbott *et al.*, Phys. Rev. Lett. **116** 061102 (2016)
[2]Available from **www.gw-openscience.org**.