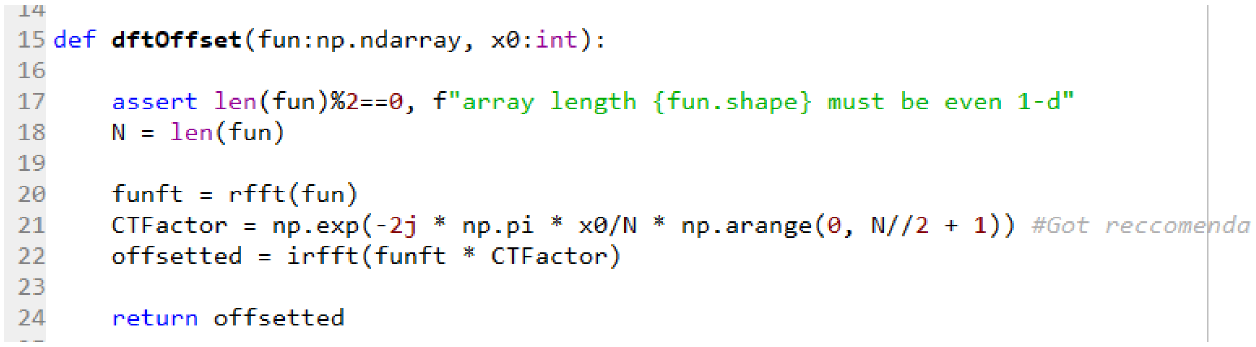
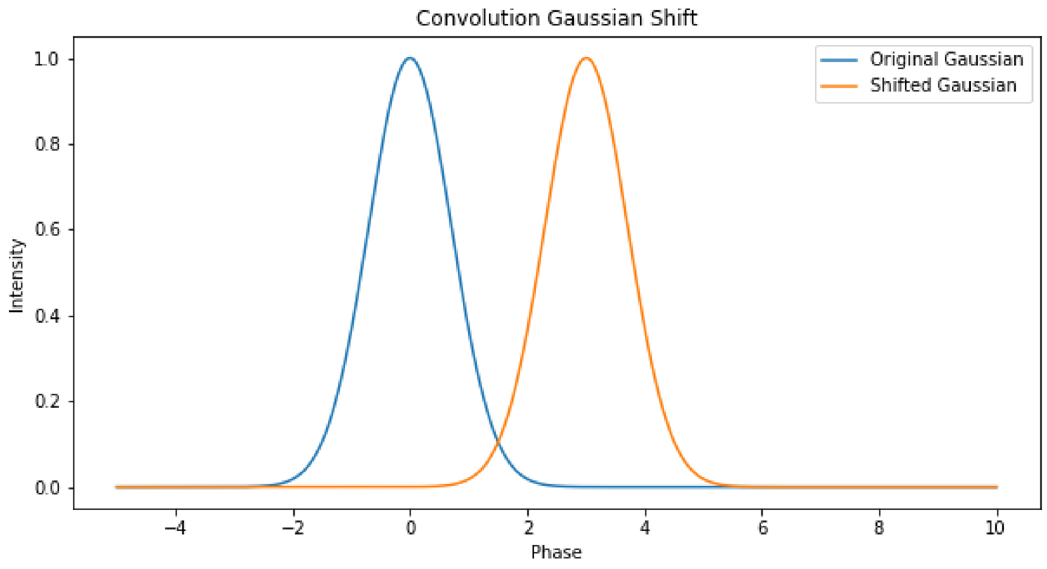
Problem 1)

In the hopes of continuously shifting using convolution, we employ the following function:



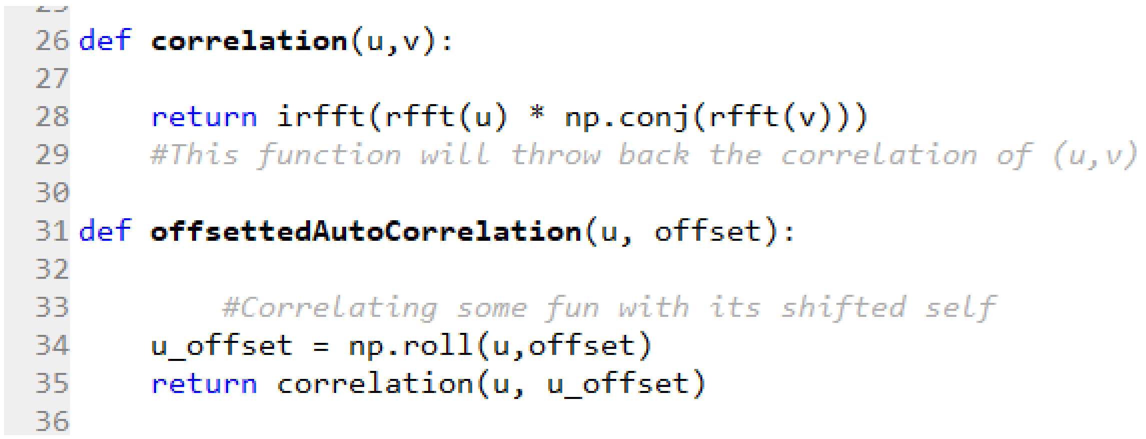
```
def dftOffset(fun:np.ndarray, x0:int):

    assert len(fun)%2==0, f"array length {fun.shape} must be even 1-d"
    N = len(fun)

    funft = rfft(fun)
    CTFactor = np.exp(-2j * np.pi * x0/N * np.arange(0, N//2 + 1)) #Got reccomenda
    offsetted = irfft(funft * CTFactor)

    return offsetted
```



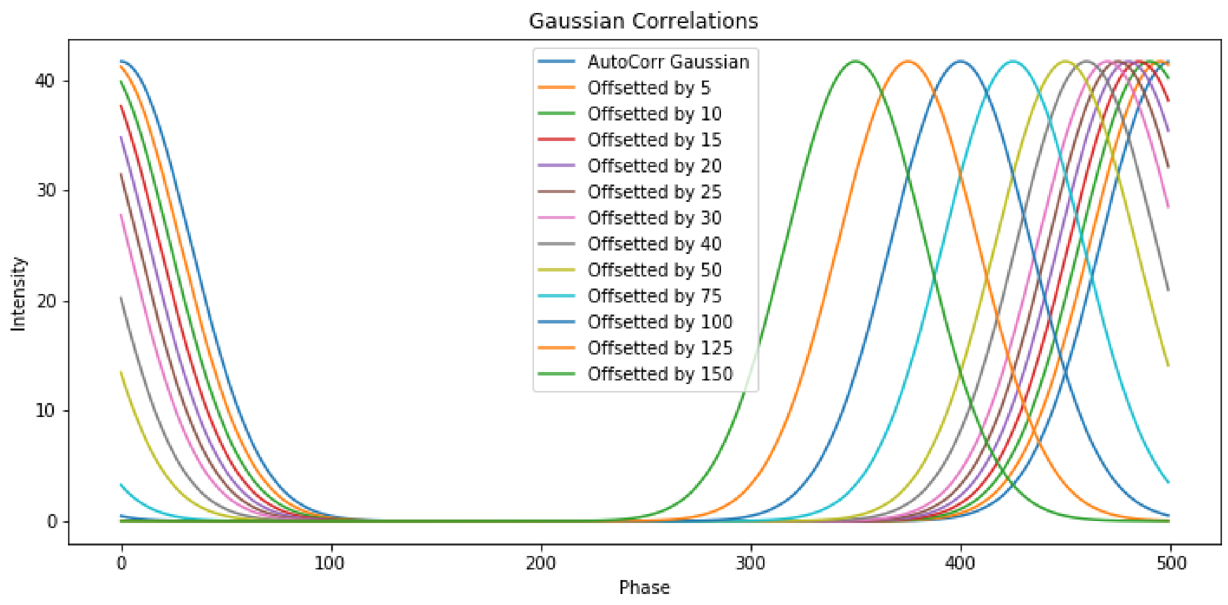
Problem 2)

(a) Since the convolution of two Gaussians is a gaussian and they're even functions, the correlation matches the convolution. Subsequently, the correlation of a gaussian with an offset version of itself is the convolution of a gaussian then offset by negative that amount. A gaussian's correlation to itself is a gaussian.

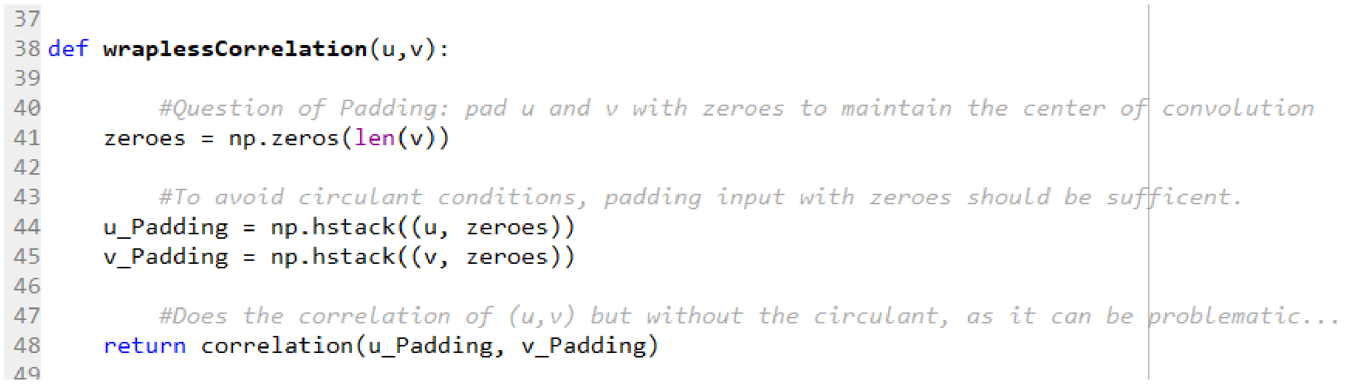
(b) Now, we're interested in calculating the correlation of a shifted Gaussian with itself and comparing it with the initial Gaussian:



```python
def correlation(u,v):

    return irfft(rfft(u) * np.conj(rfft(v)))
    #This function will throw back the correlation of (u,v)

def offsettedAutoCorrelation(u, offset):

        #Correlating some fun with its shifted self
    u_offset = np.roll(u,offset)
    return correlation(u, u_offset)

```



Problem 3)



```python

def wraplessCorrelation(u,v):

        #Question of Padding: pad u and v with zeroes to maintain the center of convolution
    zeroes = np.zeros(len(v))

        #To avoid circulant conditions, padding input with zeroes should be sufficent.
    u_Padding = np.hstack((u, zeroes))
    v_Padding = np.hstack((v, zeroes))

        #Does the correlation of (u,v) but without the circulant, as it can be problematic...
    return correlation(u_Padding, v_Padding)

```

Q4)

(a) let's demonstrate that

$$\sum_{x=0}^{N-1} \exp[-2\pi i k x/N] = \frac{1-\exp[-2\pi i k]}{1-\exp[-2\pi i k/N]}$$

$\hookrightarrow$ $$\left(1-\exp[-2\pi i k/N]\right)\sum_{x=0}^{N-1} \exp[-2\pi i k x/N] = 1-\exp[-2\pi i k] =$$

$$= \sum_{x=0}^{N-1} \exp[-2\pi i k x/N] - \sum_{x=0}^{N-1} \exp[-2\pi i k(x+1)/N]$$

Every term except $x = \begin{cases} 0 \\ N-1 \end{cases}$ cancel. This results in:

$\exp[0] = 1$ and $\exp[-2\pi i k \frac{N}{N}] = e^{-2\pi i k}$

$$\Rightarrow \left(1-e^{-2\pi i k/N}\right)\cdot\sum_{x=0}^{N-1} \exp[-2\pi i k x/N] = 1-e^{-2\pi i k}$$ ✓

(b) let's show that this approaches N as k approaches zero, and it is zero for any integer K (not a multiple of N)... {L'Hôpital's}

As $k \to 0$, $$\lim_{k\to 0} \frac{1-\exp[-2\pi i k]}{1-\exp[-2\pi i k/N]} = \lim_{k\to 0} \frac{\frac{d}{dk}\left(1-\exp[-2\pi i k]\right)}{\frac{d}{dk}\left(1-\exp[-2\pi i k/N]\right)}$$

$$= \lim_{k\to 0}\left(\frac{2\pi i}{2\pi i/N}\right)\left(\frac{\exp[-2\pi i k]}{\exp[-2\pi i k/N]}\right) = N$$

$\rightarrow$ It must be $=$ zero for non-multiples of N as $(K \% N) \neq 0$

(c)

$$\sin(x) = \frac{e^{ix} - e^{-ix}}{2i} \rightsquigarrow \text{let's scale this arbitrarily:}$$

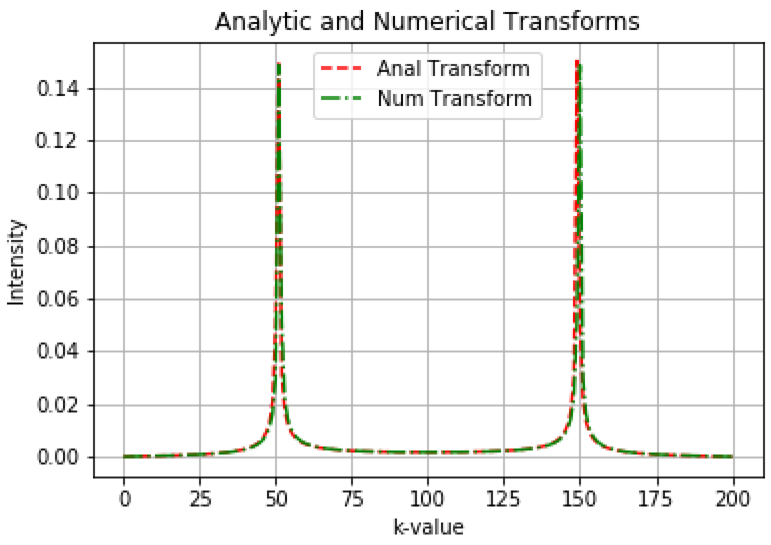
$$2i\sin\left(\frac{2\pi k' x}{N}\right) = e^{2\pi i k' x / N} - e^{-2\pi i k' x / N}$$

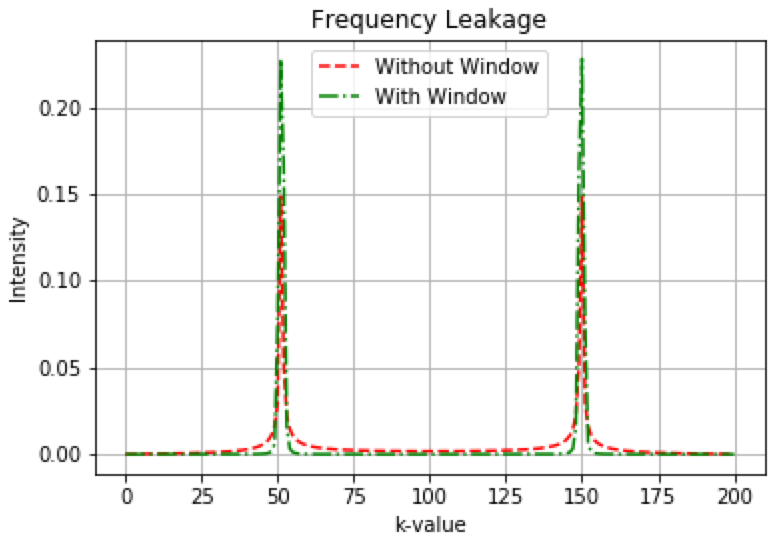
$$\hookrightarrow \text{DFT} \Rightarrow F(k) = \sum_{x=0}^{N-1} \exp\left[\frac{-2\pi i k x}{N}\right] \cdot \left(\exp\left[\frac{2\pi i k' x}{N}\right] - \exp\left[-\frac{2\pi i k' x}{N}\right]\right)$$

$$= \sum_{x=0}^{N-1} \exp\left[\frac{2\pi i (k'-k) x}{N}\right] + \sum_{x=0}^{N-1} \exp\left[-\frac{2\pi i (k'+k) x}{N}\right]$$

$$= \frac{1 - \exp[2\pi i (k'-k)]}{1 - \exp\left[\frac{2\pi i (k'-k)}{N}\right]} + \frac{1 - \exp[-2\pi i (k'+k)]}{1 - \exp\left[-\frac{2\pi i (k'+k)}{N}\right]}$$

Now, we may compare the analytic and numerical estimations of our transforms...





Clearly Less leaking with the window (sharper peaks falling to zero more quickly in k-space). The sharper corners associate with less spectral leakage; this happens as opposed to gradual decay as seen in the windowless peak.

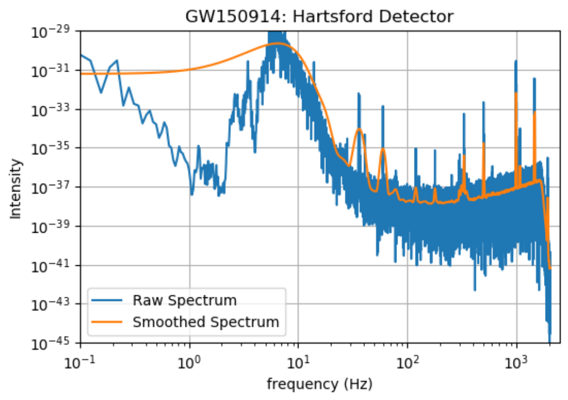
All code and comments can be found in the python code for Problem 4…

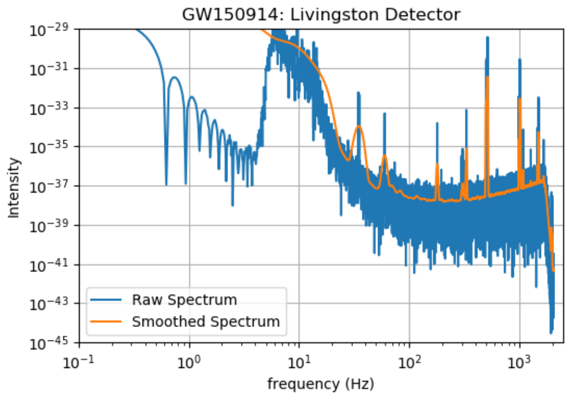
The FFT of this plot above will clearly be the same as the plot above.

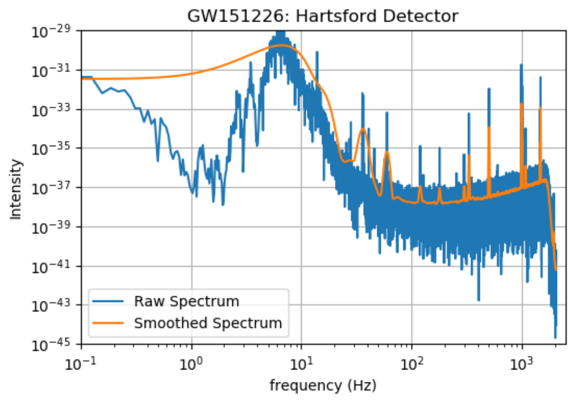
Problem 5)

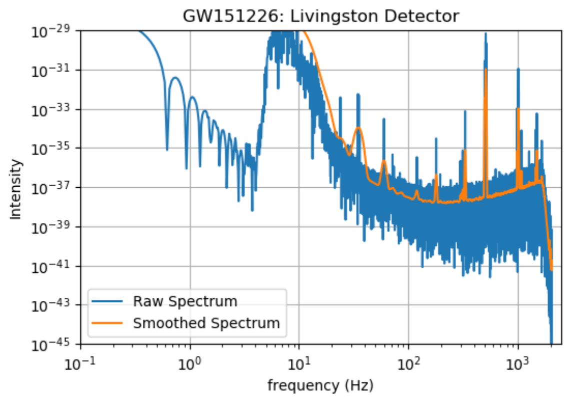
a)

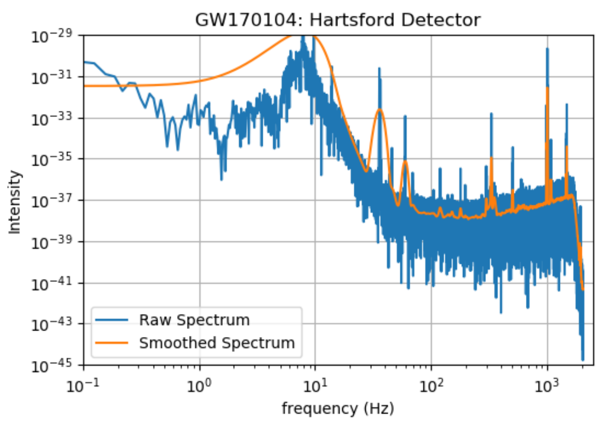
We wish to estimate the noise for each detector. Let’s calculate the smoothed power spectra first. We may use a turkey window:

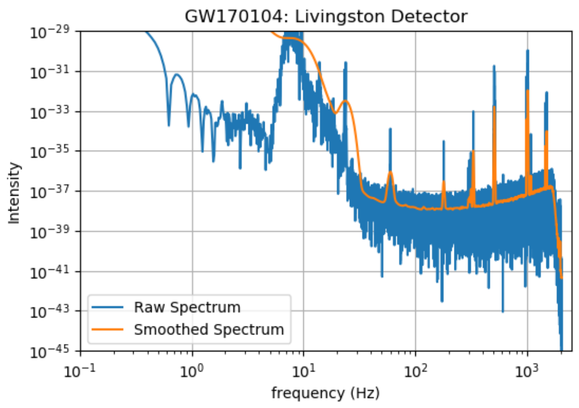


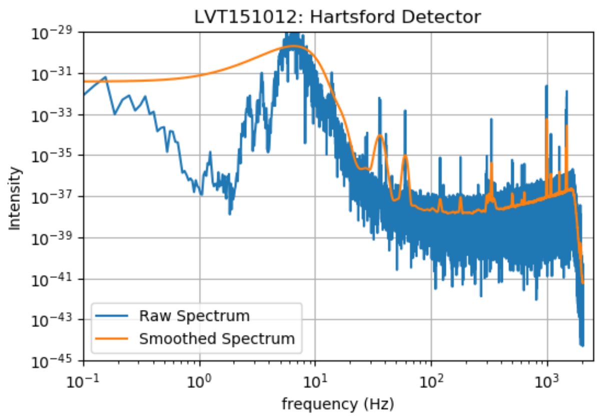


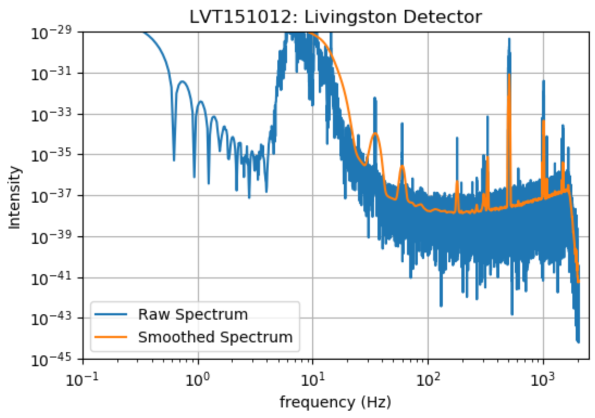





GW170104: Hartsford Detector
Intensity
frequency (Hz)
Raw Spectrum
Smoothed Spectrum


GW170104: Livingston Detector
Intensity
frequency (Hz)
Raw Spectrum
Smoothed Spectrum

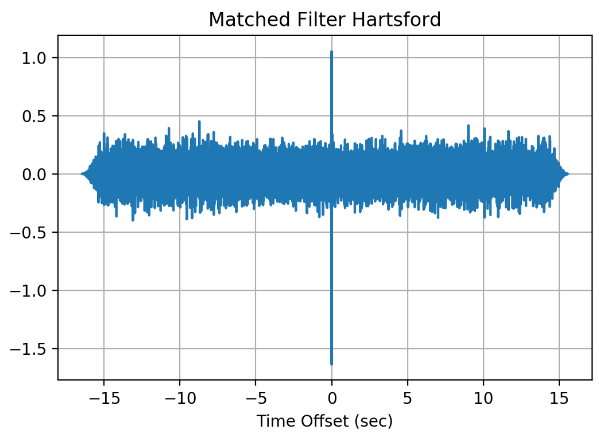

LVT151012: Hartsford Detector
Intensity
frequency (Hz)
Raw Spectrum
Smoothed Spectrum

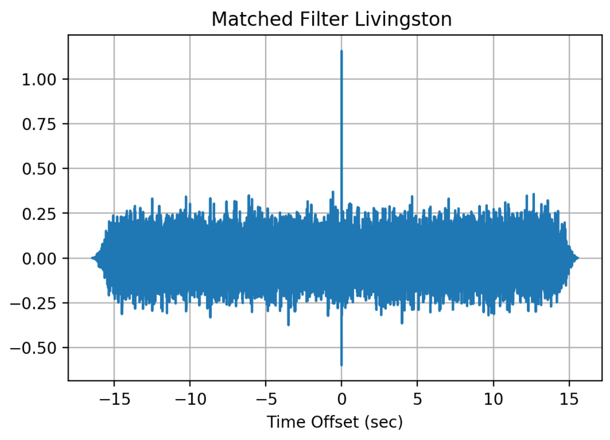

LVT151012: Livingston Detector
Intensity
frequency (Hz)
Raw Spectrum
Smoothed Spectrum

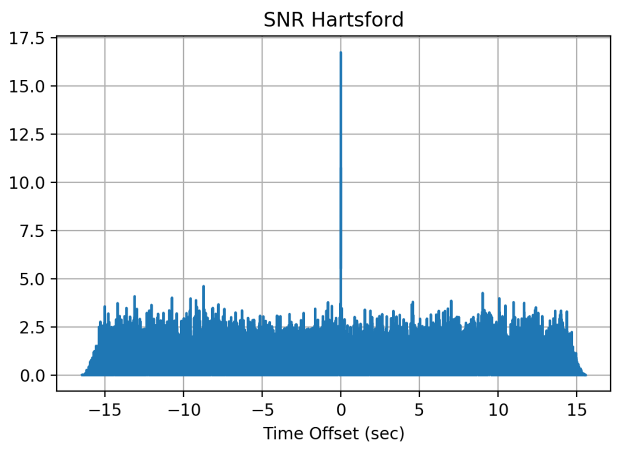
b+c)

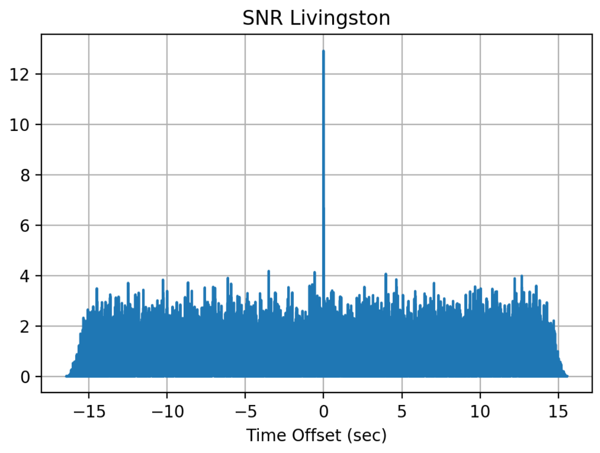
In order to find gravitational waves within these events' noise data, we must compute the matched filter. From this, we compute the SNR and approximate it as the RMS. We then pass a filter over each of these data points:

All code can be found in PSET6_Q5.py

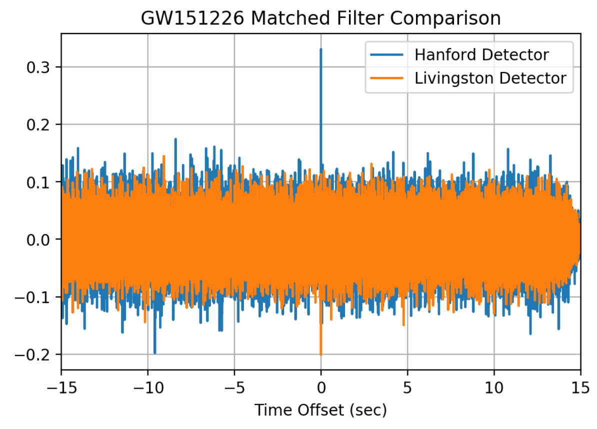


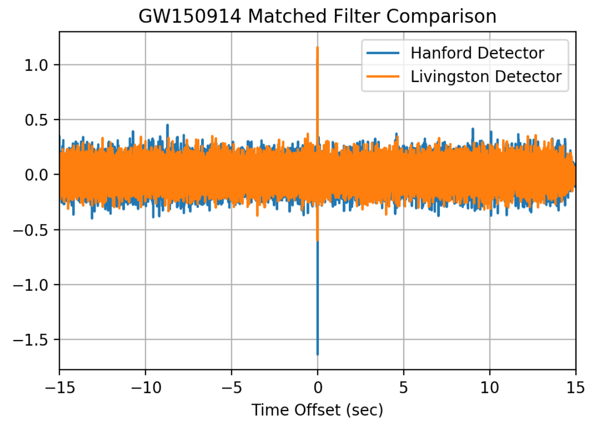


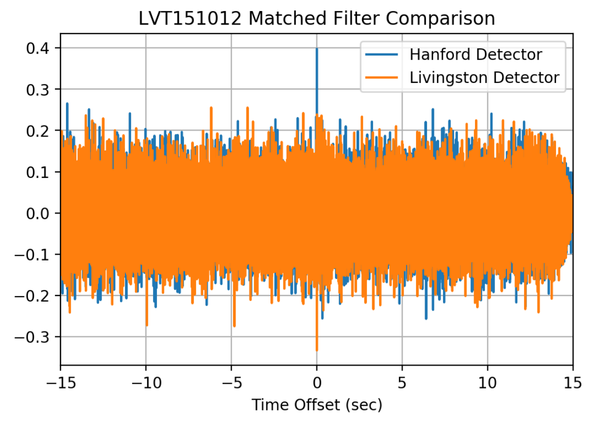


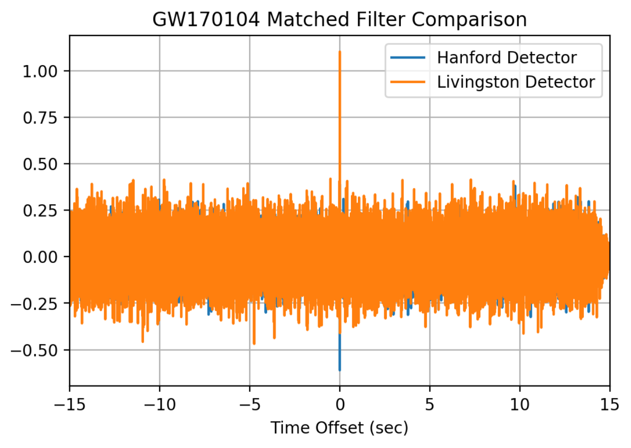


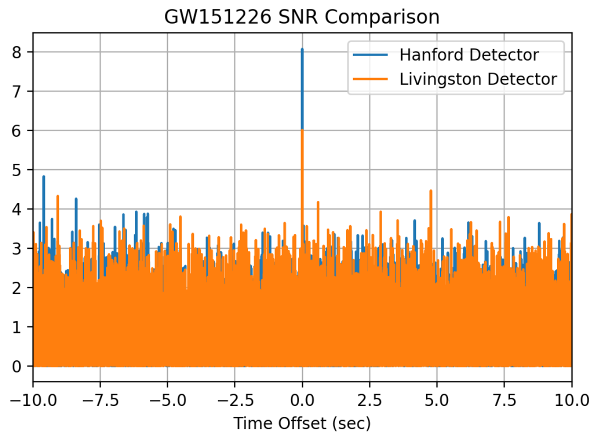
Below are the overlapped Matched Filter SNR for each individual Event:

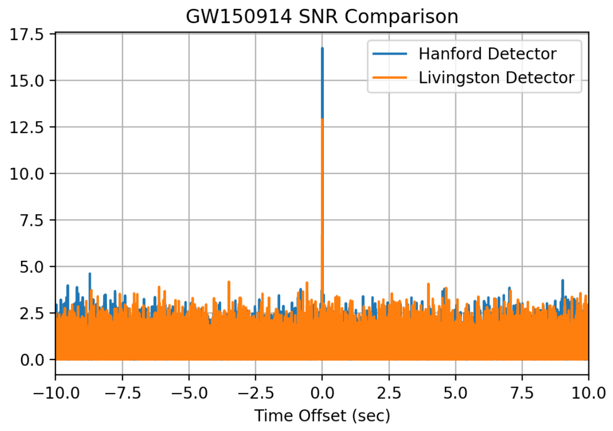


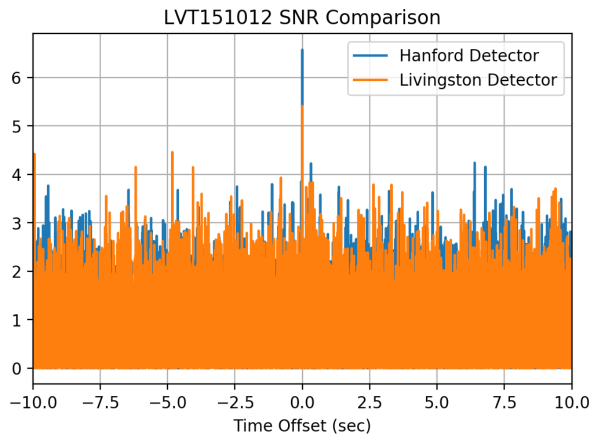


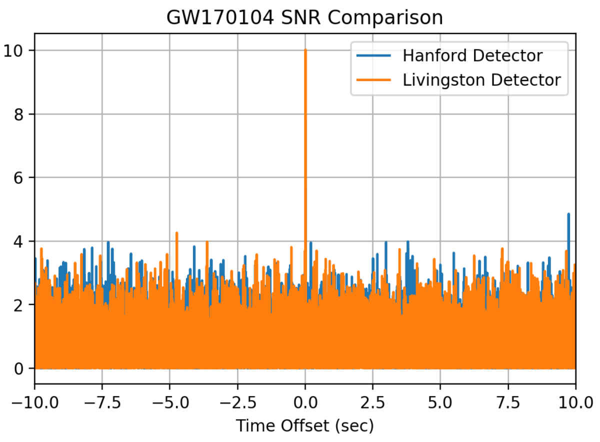












f)

We may find the difference in time via the SNR offsets:

```
GW150914 Time Difference in sec 0.00732421875
GW151226 Time Difference in sec 0.001220703125
LVT151012 Time Difference in sec 0.00048828125
GW170104 Time Difference in sec 0.003173828125
```