PHY407 Lab5

Genevieve Beauregard (1004556045)
Anna Shirkalina (1003334448)

Oct 2020

Question 1

a)

We load the data from `sunspots.txt` into arrays and plot it as a function of time in Fig 1. We see that there are about 4 oscillations every 500 months (count the number of peaks between 0 and 500 months for example). Thus we postulate the period is around 125 months, or a frequency of $f = 1/125 = 0.008$/month.

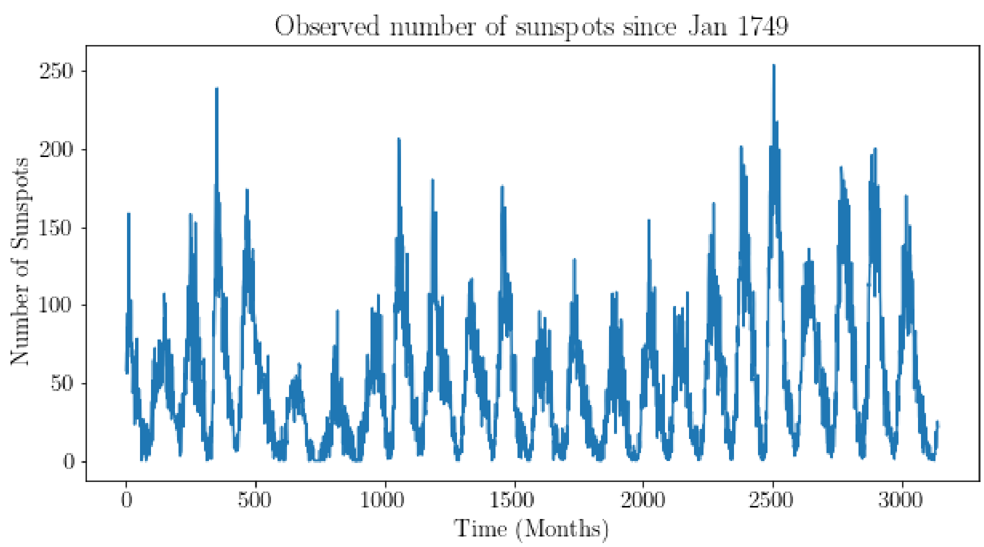

Figure 1: The number of observed sunspots as a function of time.

We then take its fourier transform using `numpy.fft.rfft` to observe for peaks in the power spectrum. In this case, we use `np.fft.rfftfreq` to create the frequency axis. We can see the plot of the full power spectrum in Fig 2. Of course, we see the relevant information is around the origin so we consider the first 30 samples in Fig 3. We observe there is a peak around 24th sample, which occurs at $f = 0.00764$/month. Thus, the approximate frequency of the global sinuisodal pattern observed in our time series would be $f = 24$ /month.

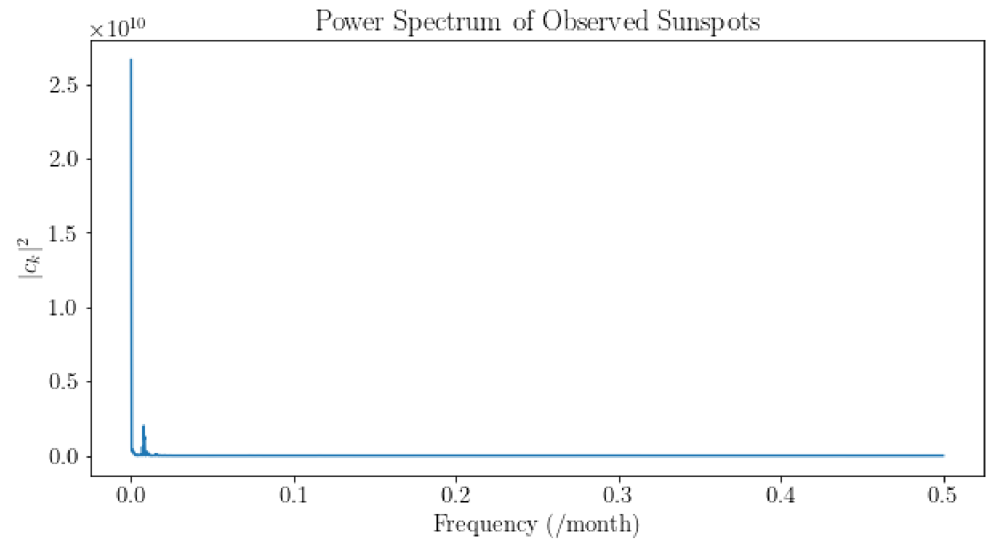

Figure 2: The full power spectrum zoomed of `sunspots.txt`.

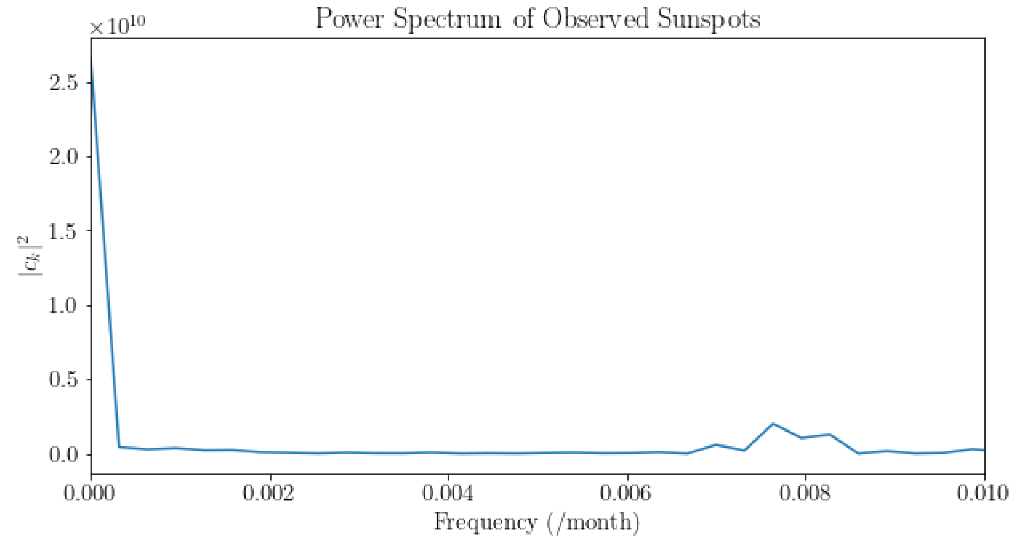

Figure 3: The power spectrum of `sunspots.txt` zoomed in.

b)

We first plot the time series of the Dow Jones index. We can see the plot in Fig 4. We can see a global pattern with many microfluctuations.

We take its fourier transform using `numpy.fft.rfft`. We then filter the higher frequency 'suboscillations' (the fluctuations we observe that not part of the overall pattern) by setting all but the first 10 % of the fourier coefficients to be zero. We do this once more but this time only preserving the first 2%. Taking the inverse fourier transform as instructed, we then our recovered (but filtered) time series against the original time series. Our plot can be see in Fig 5. We see that the time series recovered from the filtered coefficients are significantly smoother, yet still demonstrate the global trend, with the 2% filter being the smoothest.

Setting the fourier coefficients to be zero essentially removes the higher frequency components of the graph. When we recover the time series through an inverse fourier transform, we expect a time series with fluctuations of those frequencies to be removed. In this case, we remove the higher frequency components, only keeping the lower frequency oscillations pertinent to the overall trend of the data.

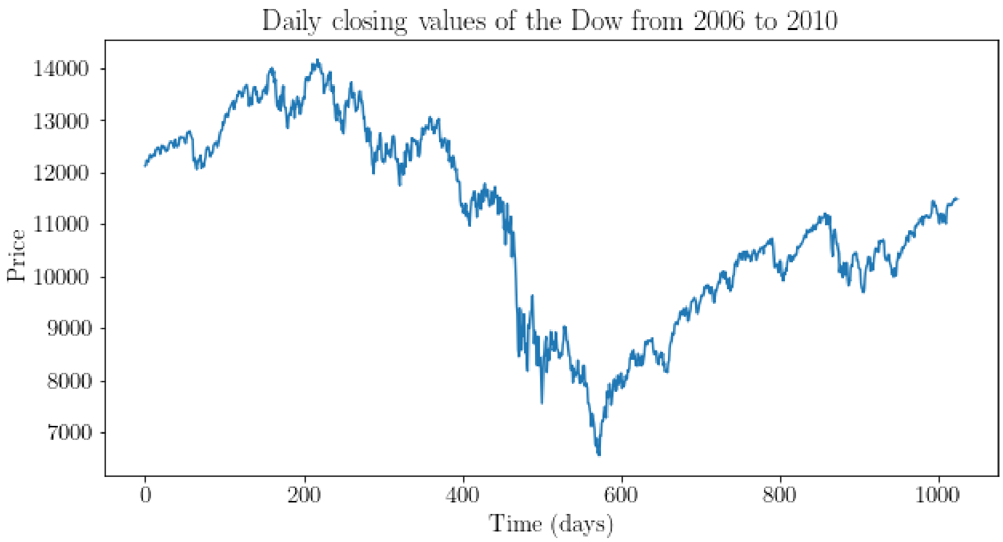

Figure 4: The Dow Jones Index closing price from 2006 to 2010.

c)

i)

We do this again data from `dow2.txt`, using the `numpy.fft.rfft` routine, only keeping the first 2% of fourier coefficients. We can see the plot of this against the original data in Fig 6. We can see the effects of the Gibb's Phenomenon, with the ends of the two functions trying to meet in the filtered data.

ii)

We repeat what was done in part (i) but we instead utilise the discrete cosine transformation routine (and its inverse) provided by the textbook. We plot the result against the original data in Fig 7. The ends of the filtered function are no longer forced to be periodic.

d)

i)

We plot the waveform of the piano and trumpet sound, obtained from their respective data sets. We can see their time series for the first 10,000 samples in Fig 8 and 9.

We use `numpy` fast fourier transform routines to obtain the power spectrum. For the frequency axis we use `fft.fftfreq`. We plot the power spectrum for

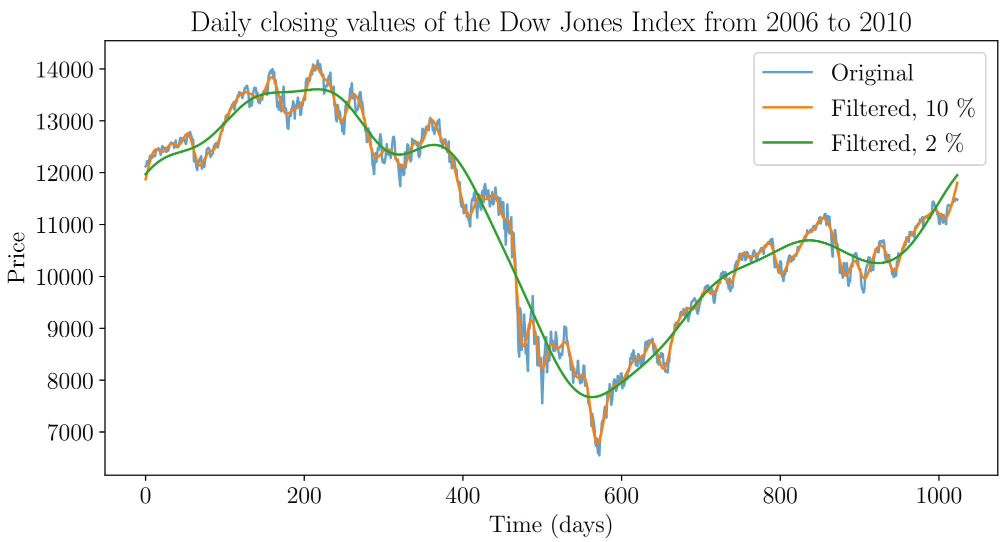

Figure 5: The Dow Jones Index closing price from 2006 to 2010, against the filtered data.

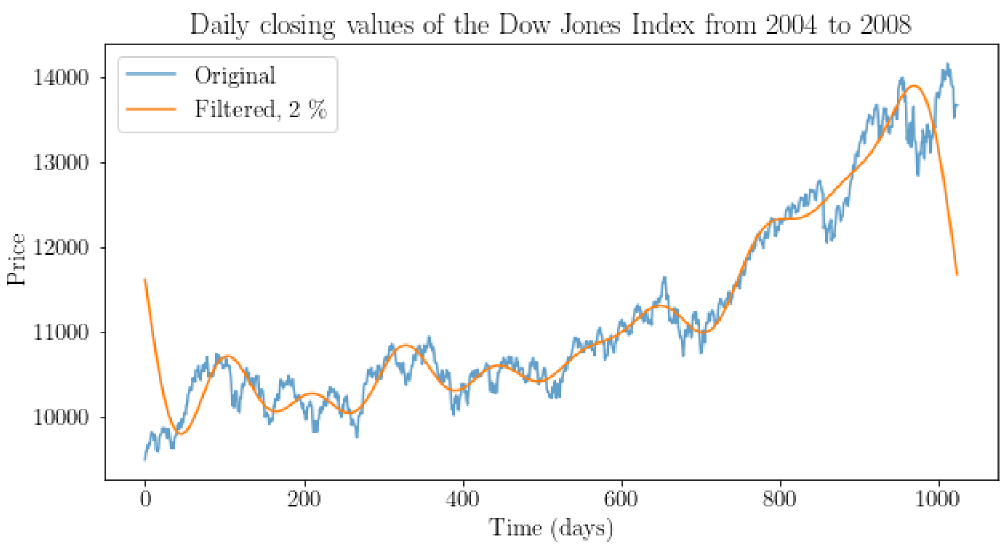

Figure 6: The Dow Jones Index closing price from 2004 to 2008, against the filtered data.

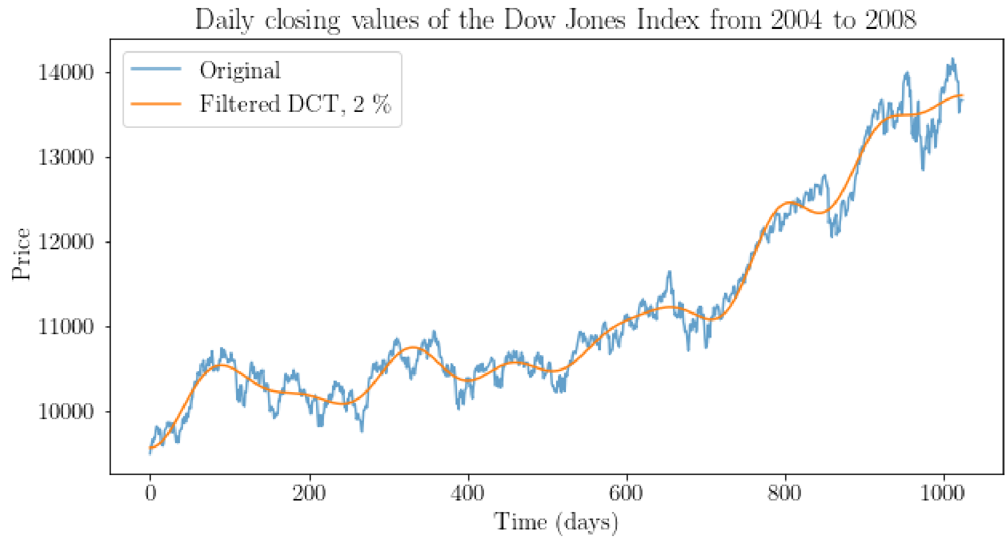

Figure 7: The Dow Jones Index closing price from 2004 to 2008, against the filtered data. Instead of a fourier transform, we used the discrete cosine transform to mitigate the Gibb's phenomenon that can be see in Fig 6
.

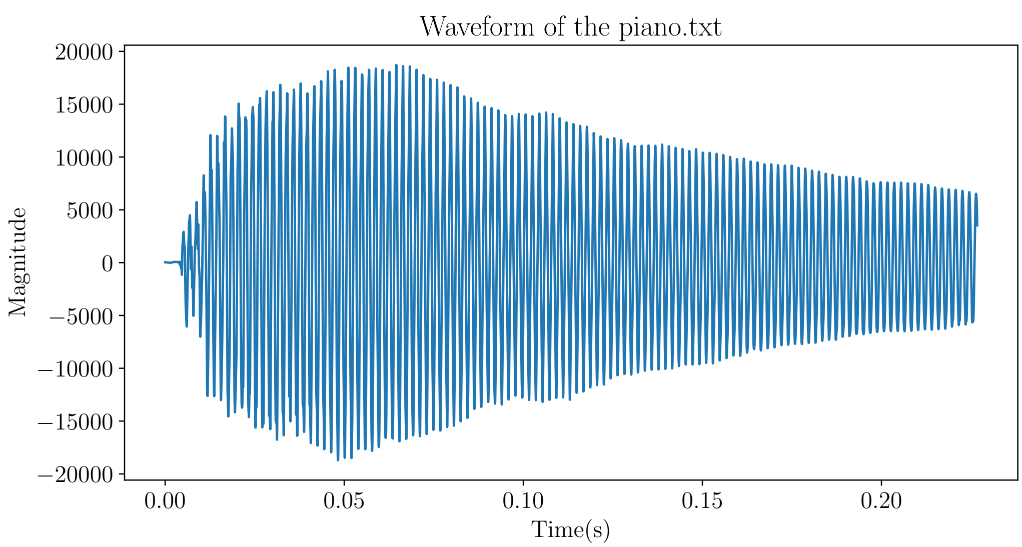

Figure 8: The time series of the piano
.

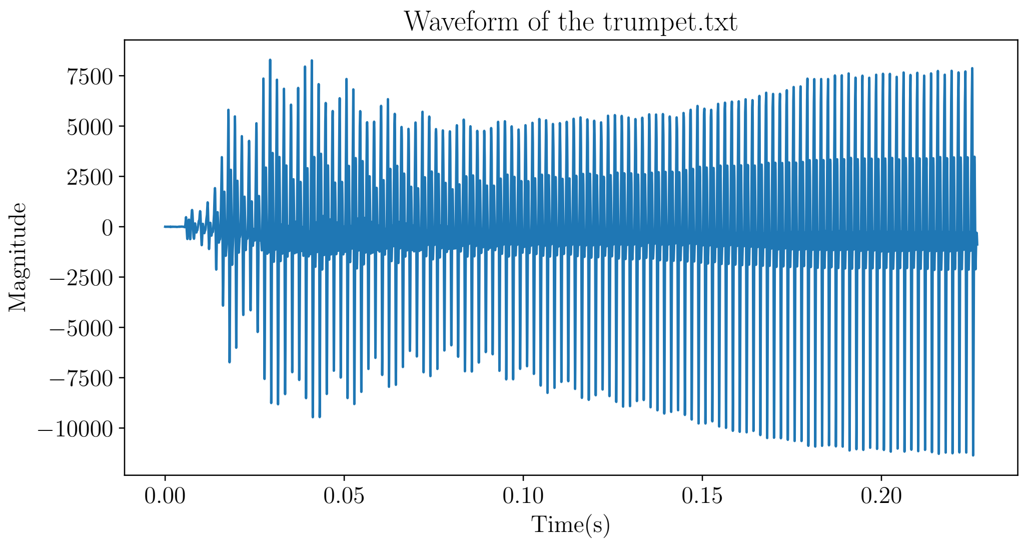

Figure 9: The time series of the trumpet

.

the positive frequencies (it is symmetric about the vertical axis anyways). Our plots can be see in Fig 10 and 11.

ii)

We can obtain the pitch of the sounds from the peaks of the waveforms – the first major peak is the fundamental frequency indicating the pitch [1].

The piano is relatively straightforward – there are few overtones and thus we can simply take the first major peak. We obtain the frequency at which the peak occurs to be 525.672 Hz.

To determine the trumpets frequency, we obtained the frequency at which the largest peak occurs: 1043.847 Hz. A quick observation of the graph tells us this is a the second fundamental, so the pitch for this sound would be $1043.847/2 = 521.9235$ Hz.

They are both fairly close to the the C5 (C is actually 523.25 Hz). [2].

Question 2

We load the data from `blur.txt` and use imshow to display the blurred image. We then define a function for Gaussian point spread at each point on the image. And then calculate the Fourier coefficients for the Gaussian point spread array

[1] `https://newt.phys.unsw.edu.au/jw/brassacoustics.html#natural`
[2] `https://pages.mtu.edu/~suits/notefreqs.html`

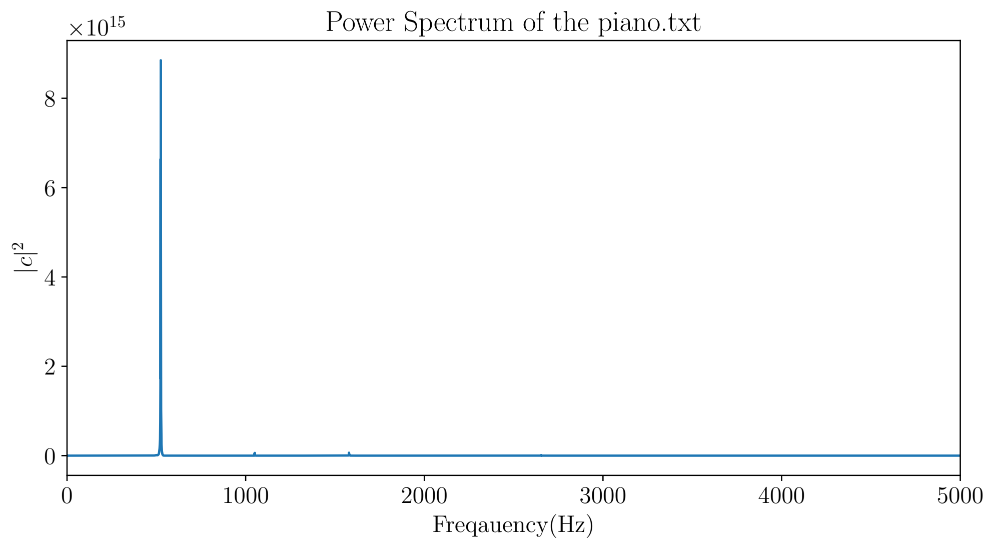

Figure 10: The power spectrum of the piano

.

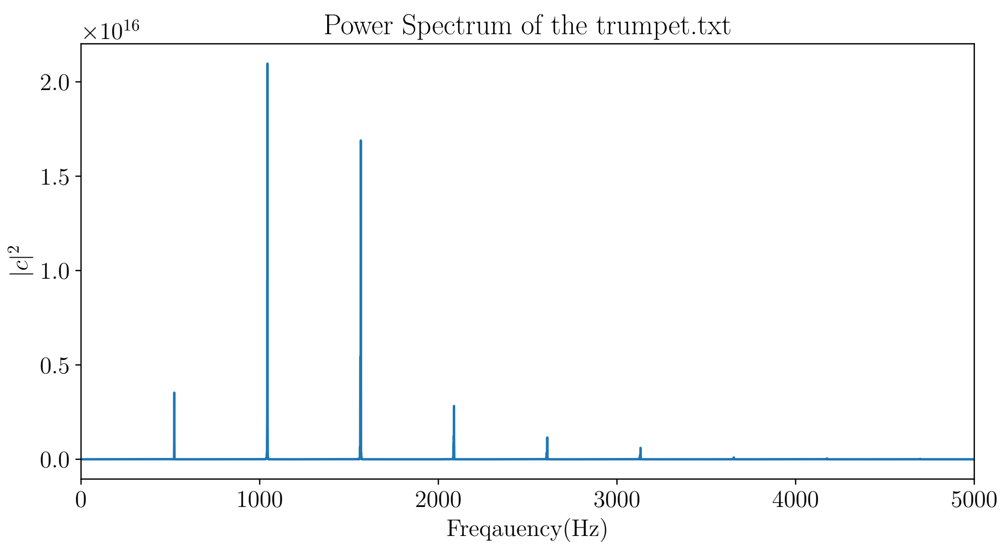

Figure 11: The power spectrum of the trumpet

.

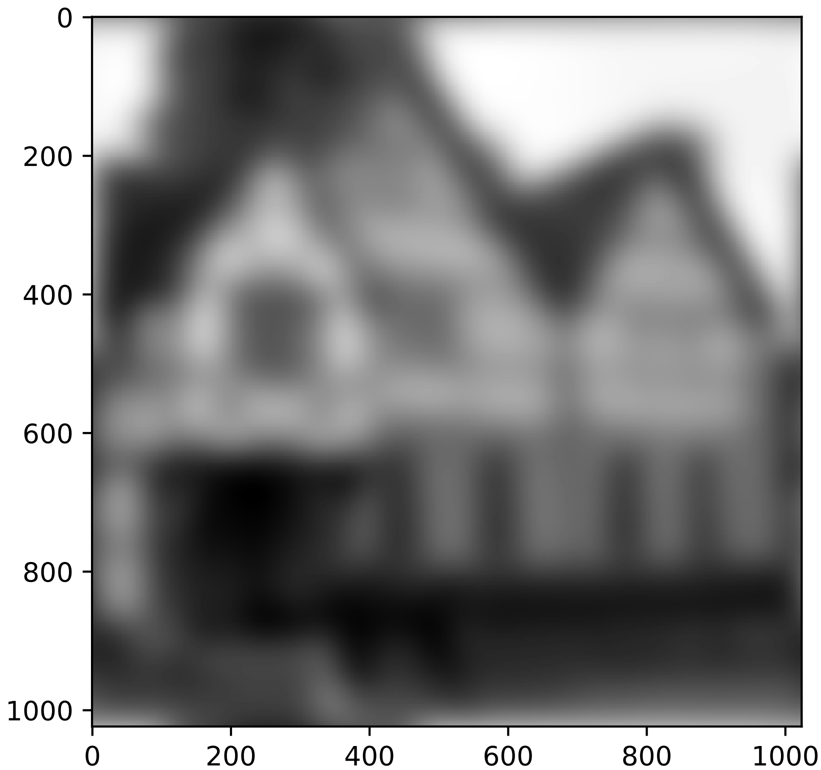

Figure 12: The blurry image

.

and for the blurred image. We then compute the Fourier coefficients for the unblurred image using equation $a_{nm} = \frac{b_{nm}}{f_{nm}NM}$, where n, m are the indicies of the rows, columns respectively. And N,M are the dimensions of the image. Our images can be see in Fig 12, 13 and 14.

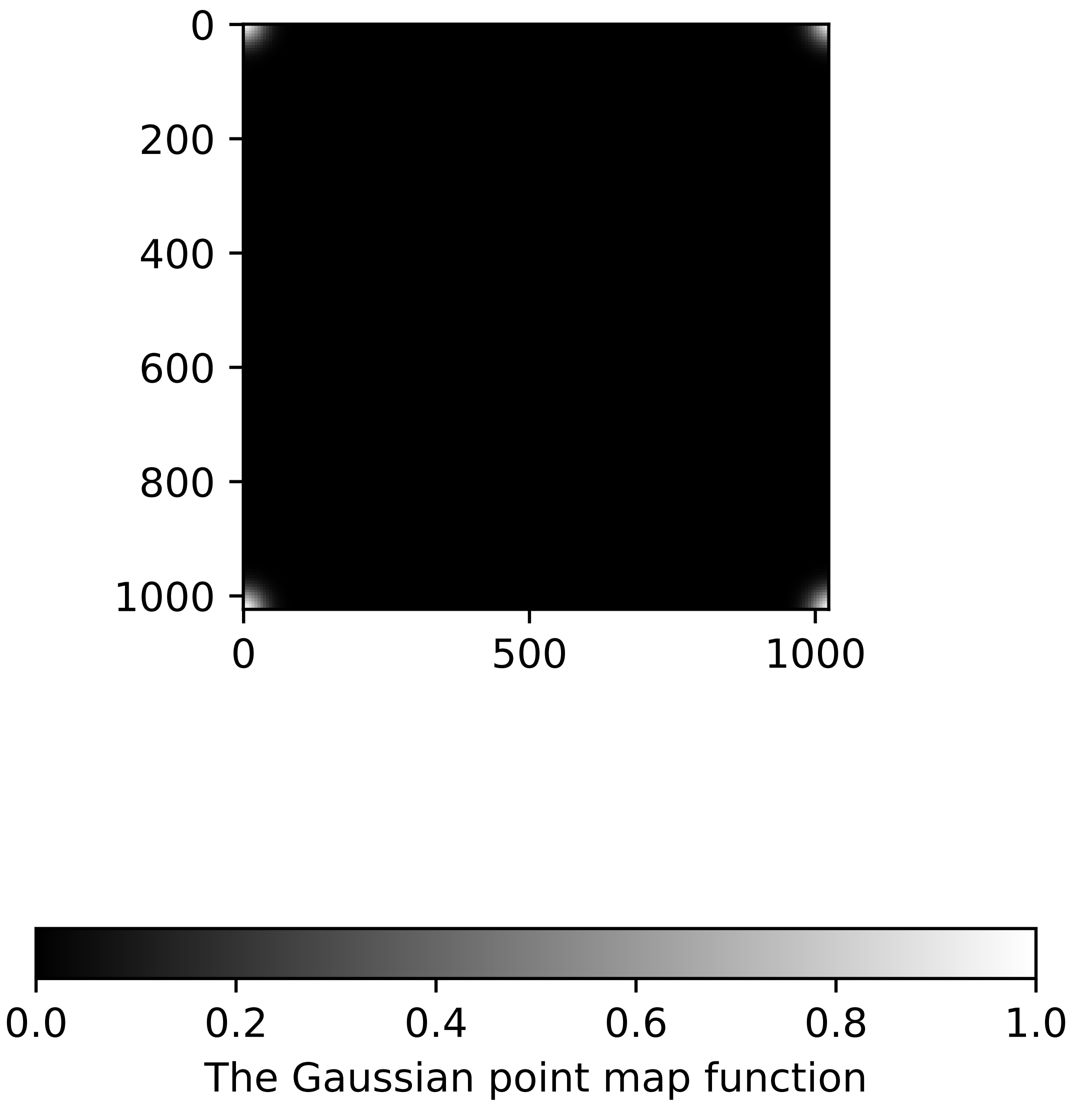

Figure 13: The Guassian point spread function

.

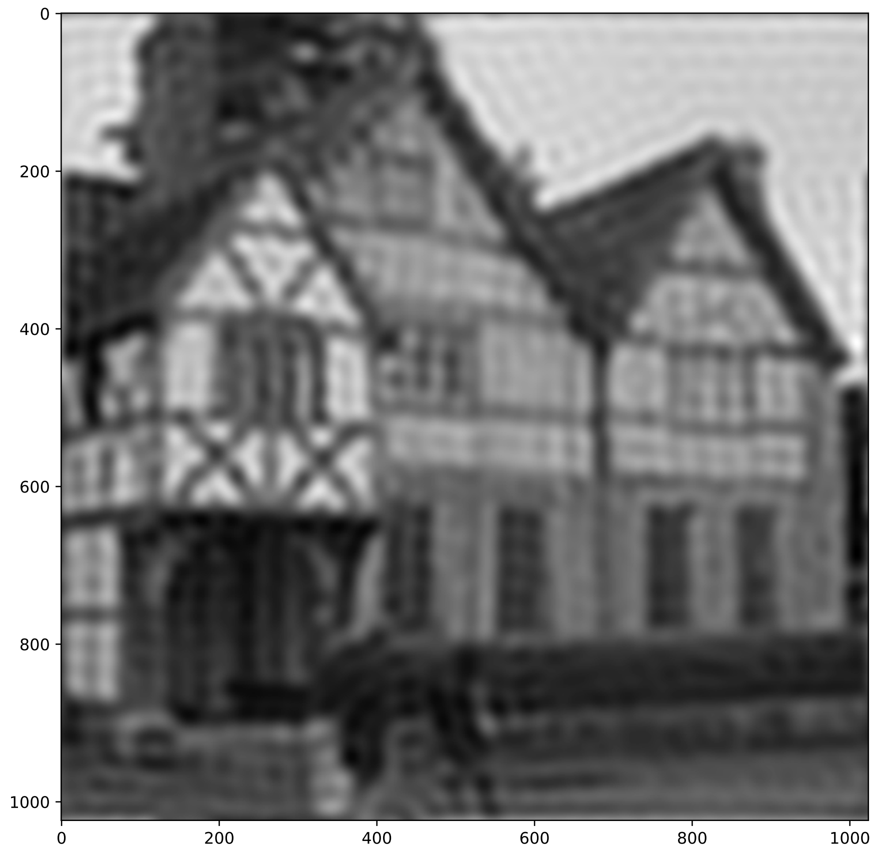

Figure 14: The unblurred photo

.