PHY407 Lab 5: Fourier Transforms

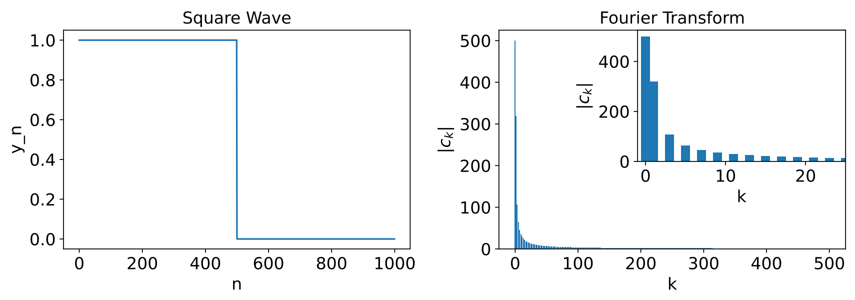
Work Distribution:
Emma Jarvis: Q3, Q4
Lisa Nasu-Yu: Q1, Q2

October 16, 2021

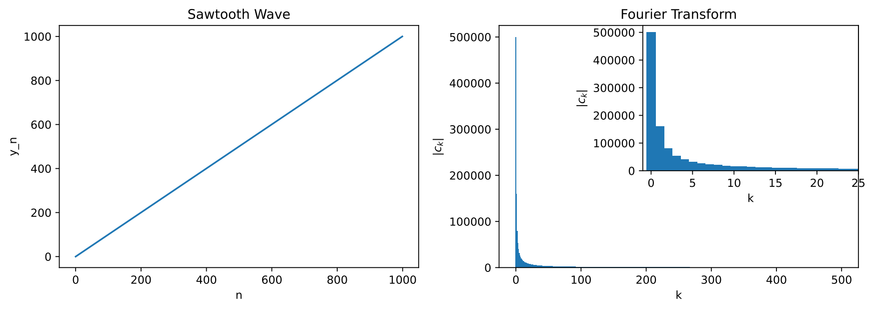
1 Various Examples of Fourier Transforms

1.a Newman 7.1

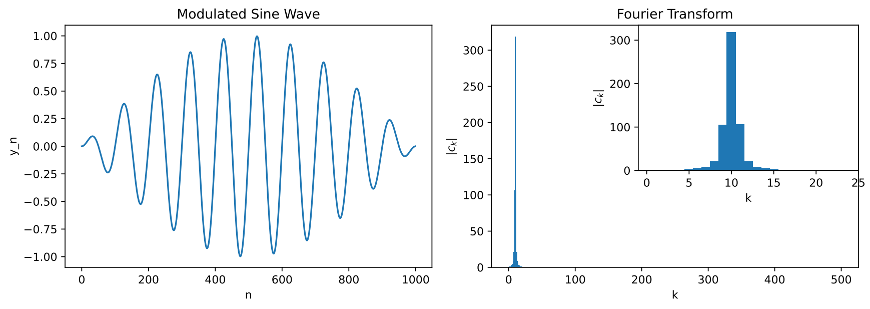
The discrete Fourier transform was applied to various periodic functions.



(a)



(b)



(c)

Figure 1: Coefficients in the discrete Fourier transforms of 3 periodic functions: square wave (Figure a), sawtooth wave (Figure b), and modulated sine wave (Figure c). N=1000 samples were taken for each function within a range of 1 period. The inset plots show a cropped view in the range of k less than 25, to better depict the details in the densest regions of the plot.

1.b

We computed the intensity from a diffraction grating using a fast Fourier transform as described in Newman Exercise 7.8. Our choice of N=1000 sampling points gave a good approximation of the grating transmission function, with Fig. 2 showing a strong constructive interference at x=0m, 2 peaks of weaker constructive interference on either side (1st order diffraction), and destructive interference otherwise.

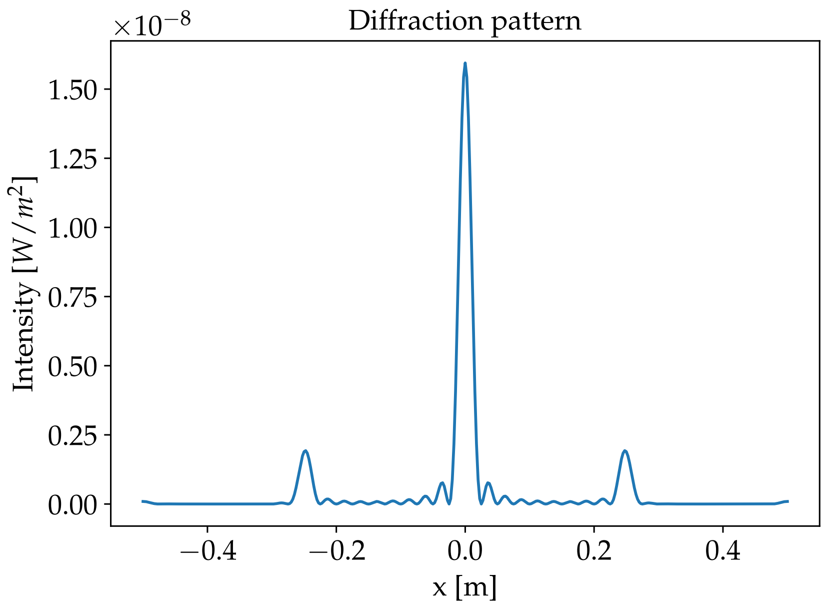


Figure 2: Diffraction grating pattern computed by fast Fourier transform, for N=1000 sampling points.

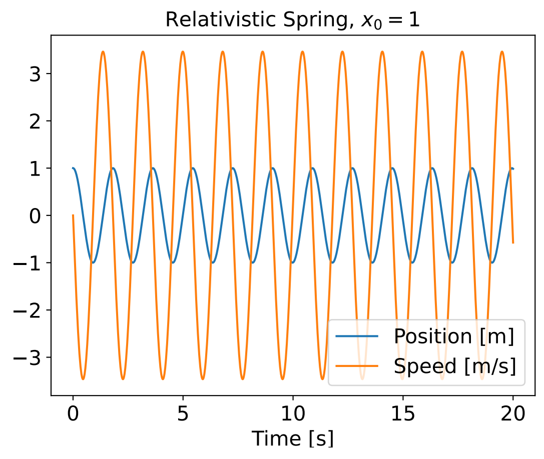
2 Fourier Transforms in the Time Domain

2.a Re-visiting the Relativistic Spring

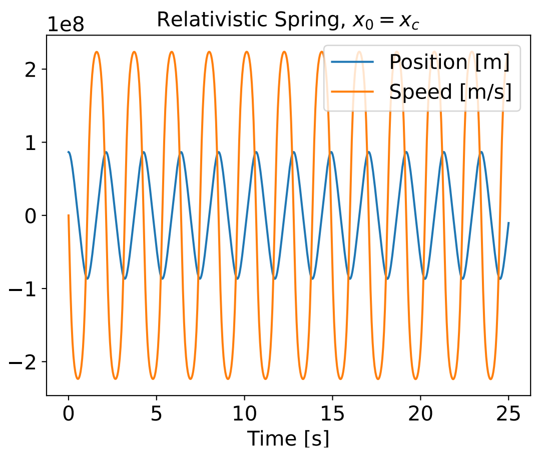
We simulated the velocity and positions of a relativistic spring using the Euler-Cromer method, where the acceleration was given by Eq. 1.

$$\frac{dv}{dt} = -\frac{k}{m}x(1 - v^2/c^2)^{3/2} \tag{1}$$

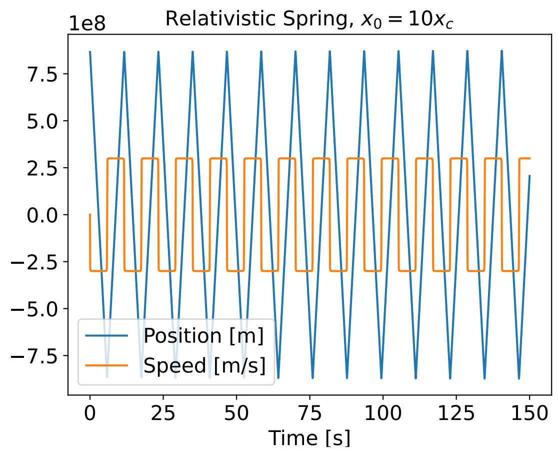
Our choices of total time and time step to obtain stable results for the 3 initial positions of x_0 =1, x_c, $10x_c$ in metres were t=20s, 25s, 150s, and dt=0.00001s. The total simulation times were chosen to give at least 10 periods in each case. x_c, the initial position for a relativistic spring to reach a maximum speed of c, the speed of light, was calculated in lab 3 as $x_c = \frac{1}{2\sqrt{3}}c$.



(a)

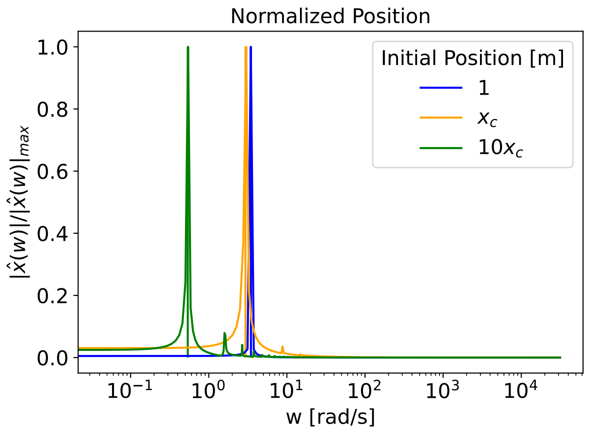


(b)

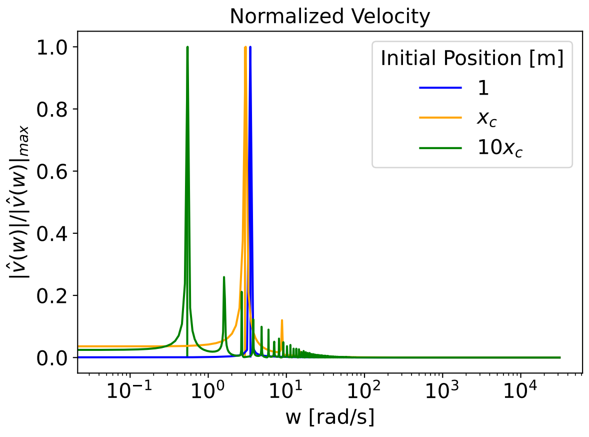


(c)

Figure 3: Simulated motion for a relativistic spring with from various initial positions, expressed in m, and initial velocity 0.



(a)



(b)

Figure 4: Fourier transforms of position and speed of a relativistic spring. The period estimated by the by the Euler-Cromer method is plotted as a vertical line on Figure a.

The spectrum for the Fourier transform of position peaks a different angular frequencies. The most relativistic case, $x_0 = 10x_c$ has more peaks than the other two cases. This is as we expect, as the more relativistic case diverges from the simple harmonic oscillations of a classical model. The spectra in Fig. 4b and Fig. 4a peak at nearly the same frequencies, but at higher amplitudes for the velocity than the position.

The vertical lines, representing the period computed using Gaussian quadrature, in Fig. 4b and Fig. 4a, match their corresponding characteristic frequencies, indicating that both methods give estimates.

2.b Filtering an Audio Recording

For Q2b, we low pass filter the sound from the file, graviteaTime.wav. The output file is attached in the submission, named *output_file.wav*.

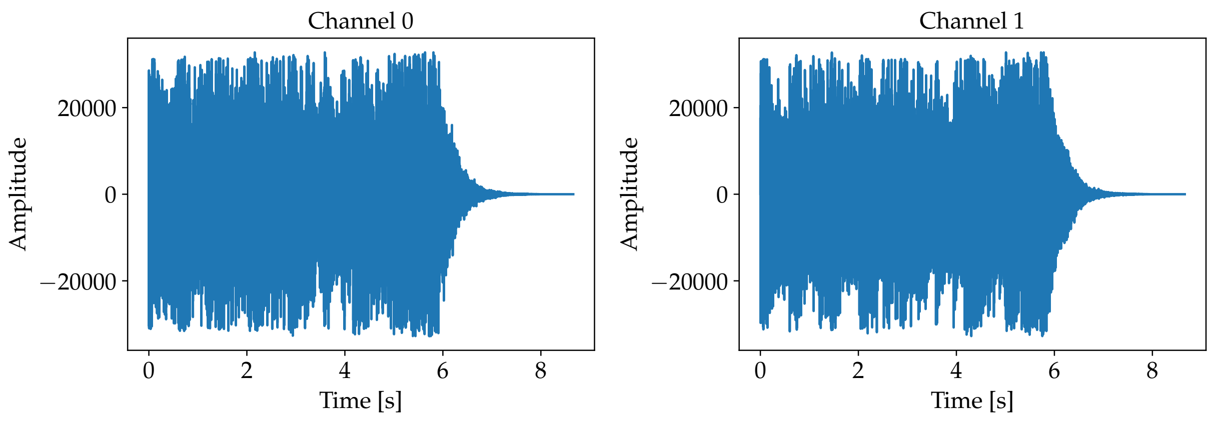
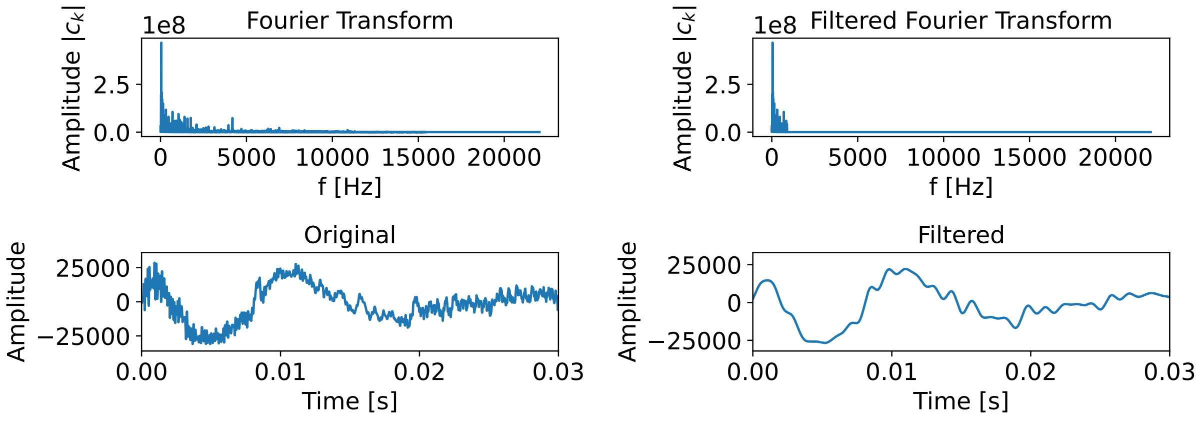
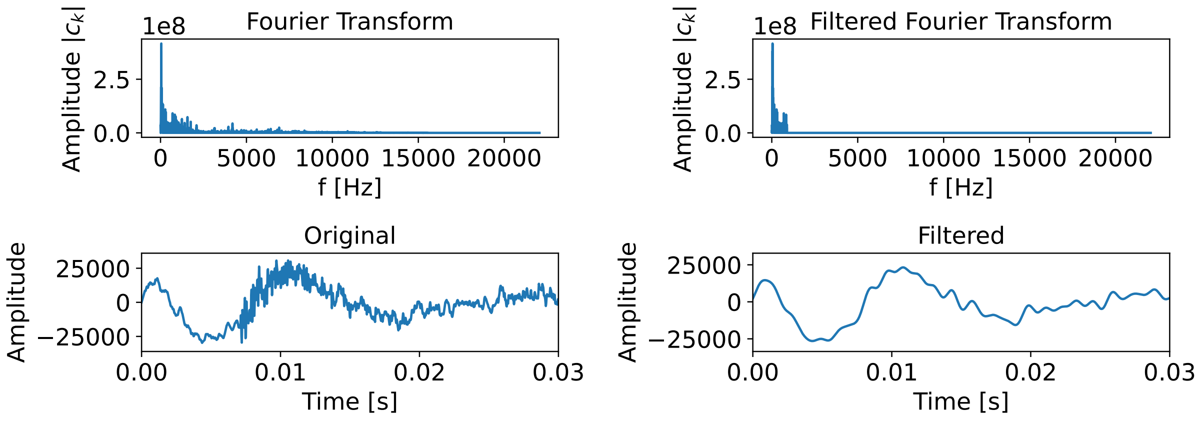


Figure 5: Signal from each channel of the audio file graviteaTime.wav.



(a) Channel 0



(b) Channel 1

Figure 6: In each of Figure a) and Figure b), the bottom left plot depicts the signal from the original graviteaTime.wav file; the top left plot depicts the absolute value of the coefficients of its Fourier transform; the top right depicts the filtered coefficients of the Fourier transform, where all signals corresponding to a frequency greater than 880Hz are set to zero; the bottom right plot depicts the filtered signals in the time domain. Figure a) shows these plots for channel 0, while Figure b) shows channel 1. The plots are cropped to focus on the signals in in the range of 0-0.03s in the time domain.

3 Analysis of Sea Level Pressure in the Southern Hemisphere for Summer 2015

3.a

We began by loading the `SLP.txt, lon.txt` and `times.txt` files and making a contour plot contour plot to read this data. The x axis is the longitude in degrees, the y axis is the time in days since Jan. 1 2015. Figure 7 depicts this plot. This data corresponds to a latitude of 50 °S.

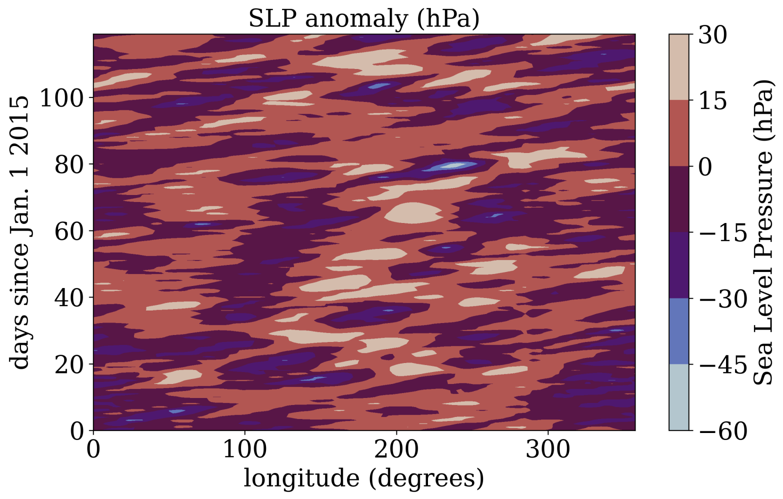
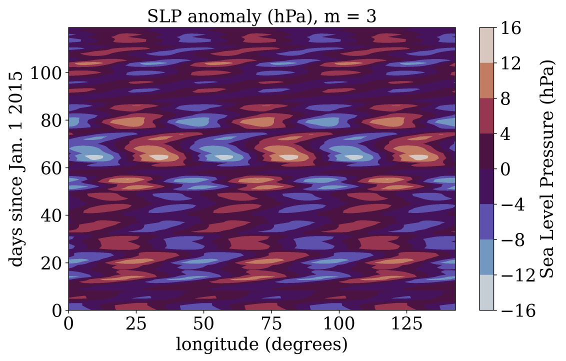
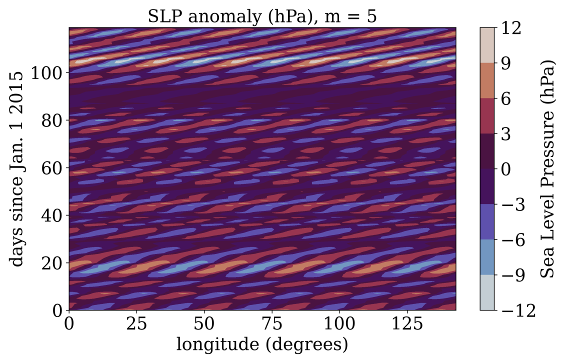


Figure 7: Time evolution of the SLP at a latitude of 50°S for the first 120 days of 2015.

After this, we extracted the component of SLP corresponding to the Fourier wavenumbers $m = 3$ and $m = 5$. To do this we began by taking the fast Fourier transform, `numpy.fft.rfft` of the SLP data. We then obtain the component of this Fourier transformed SLP for the $m = 3$ and $m = 5$ wavenumbers by slicing along the 3rd and 5th columns of this data. We then created two empty arrays of zeros; one for the wavenumber 3 and the other for the wavenumber 5. We set the 3rd column of the first array and 5th column of the second array to be the sliced Fourier transform of the SLP data. To turn this back into time/longitude space we did an inverse Fourier transform using `numpy.fft.irfft` on each array and plotted this as a contour plot with the time and longitude axis. Figure 8a and 8b depict these plots with wavenumbers of 3 and 5 respectively. In these plots, the `rfft` loses some of the longitude data, so the longitude only goes up to 150°rather than 360°.



(a) SLP corresponding to Fourier number $m = 3$



(b) SLP corresponding to Fourier number $m = 5$

Figure 8: Figure (a) depicts the time evolution of the SLP at a latitude of 50°S for the first 120 days of 2015 with the component of SLP corresponding to Fourier wavenumber $m = 3$. Figure (b) depicts the time evolution of the SLP at a latitude of 50°S for the first 120 days of 2015 with the component of SLP corresponding to Fourier wavenumber $m = 5$.

These plots depict the wave trains of the weather in the Southern hemisphere at a latitude of

50°with longitudinal wavenumbers of 3 and 5. This shows the patterns of cloudy weather and rain (low SLP) and fair and sunny weather (high SLP). For the wavenumber 3, these patterns travel more slowly than for the wavenumber of 5. There are more cycles for the wavenumber of 5 going from east to west than the SLP corresponding to a wavenumber 3. Since the theory of atmospheric physics predicts that shorter waves travel eastward (longitude going from 0°to 360°, or 150°as in this plot) faster than longer waves, this is consistent with what is seen since $m = 5$ corresponds to a shorter wavelength (since wavenumber is the reciprocal of the wavelength).

3.b

We then estimated the speed and direction of propagation for the case $m = 5$. The circumference of the latitude is given by Eq. 2 where r_E is the radius of Earth (6,365,000m) and θ is the latitude (50°S). Plugging in these values, we obtained a circumference of 38,591,379.26872892m.

$$C = 2\pi r_E \cos\theta \tag{2}$$

The speed of propagation was estimated from the graph. The wavelength is the inverse of the longitudinal wavenumber which is 1/5. Since the cycle repeats approximately every 25°in longitude (estimated from graph), the wavelength is $1/5 \cdot 25/360 \cdot C = 192,956,896.3436446$m. Since the period is about 80 days = 6912000s (estimated from the graph), the speed in m/s is approximately 30m/s easterly.

4 Newman 7.9: Image Deconvolution

4.a

We began by loading the `blur.txt` file containing a grid of values representing the brightness of a black and white photo. This photo was out of focus and was blurred using a Gaussian point spread function with a width of $\sigma = 25$. Using `imshow` we plotted this image as a density plot. This image is depicted in Fig. 9.

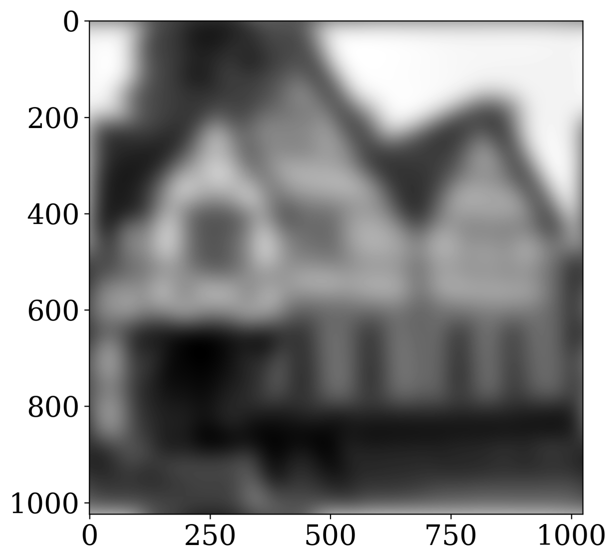


Figure 9: Blurry image obtained from `blur.txt` and plotted using `imshow`.

4.b

After this, we created a program that creates an array with the same size as the photo (1024, 1024) that contains the samples drawn from the Gaussian f(x,y) given by Eqn. 3 using $\sigma = 25$ and x and y values ranging from -512 to 512 so that the image is centred at 0 because the Gaussian function is periodic. We made sure to plot the absolute value of this image because `imshow` cannot plot complex values.

$$f(x, y) = e^{-\frac{x^2+y^2}{2\sigma}} \tag{3}$$

We then plotted this image showing the Gaussian. Fig. 10 depicts this image we the sides going from 0 to 1024 to match the blurred image.

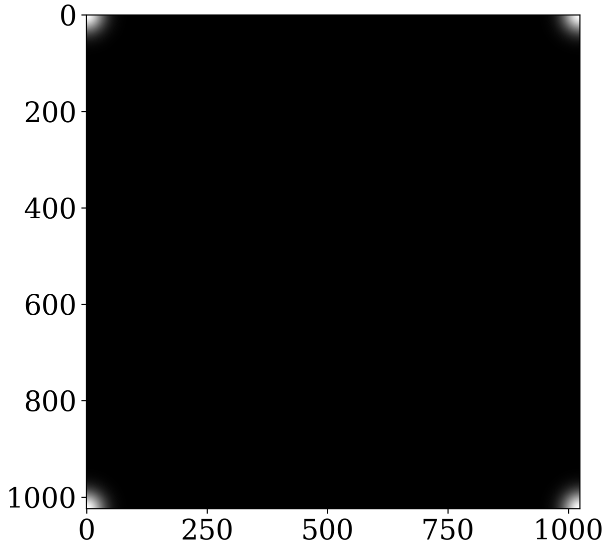


Figure 10: Visualization of the point spread function obtained using the Gaussian given in Eqn. 3. This was plotted using `imshow`.

4.c

We then wrote a program that uses this Gaussian to make the image unblurred. The first step of this program was to generate the image and the point spread function of the Gaussian. These were the arrays corresponding to the plots 9 and 10. After this, we used `numpy.fft.rfft2` to take the Fourier transform of each of these 2D arrays. We then divided the Fourier transform corresponding to the image by the Fourier transform of the point spread function. To ensure there is no divide by zero error when dividing by the Gaussian, we looped over each point and if the Gaussian at a certain point was below 10^{-5} we kept that point the same (just the point of the Fourier transformed image) without dividing by the Gaussian. If the value of the Gaussian was above 10^{-5}, we divided by the Fourier transformed Gaussian. Once we did this for each point of the image, we used `numpy.fft.irfft2` to take the inverse Fourier tranform of image to produce the unblurred image. Finally, we displayed this image using `imshow` by taking the absolute value of the inverse Fourier transform because `imshow` could not plot complex values. Fig. 11 depicts this unblurred image. It is still not perfectly sharp, but is much less blurry than the original image.

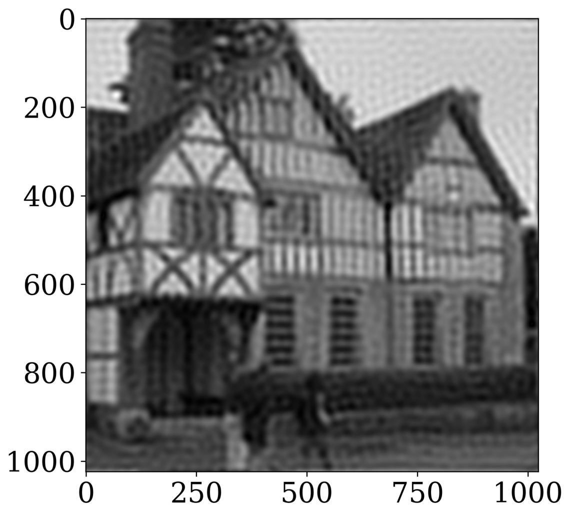


Figure 11: Unblurred image corresponding to Fig. 9. This blur was removed by computing the Fourier transform of the image 9 and the point spread function 10, dividing the image by the point spread function in Fourier space and then doing an inverse Fourier transform of this result. This image was plotted using `imshow`.