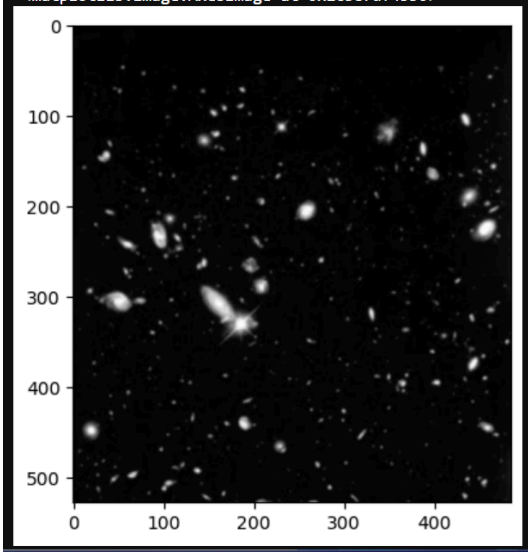
Homework 1
Zacci Oduneye
Student #: 002-48-7453

Question 1 Breakdown

- Question 1 stated that I must use 2 different low pass filters so for my first filter I decided to use the gaussian filter provided by scikit-image.

```
hw1_img_gaussian = filters.gaussian(hw1_img)
print(hw1_img_gaussian.shape)
plt.imshow(hw1_img_gaussian, cmap="gray")
```

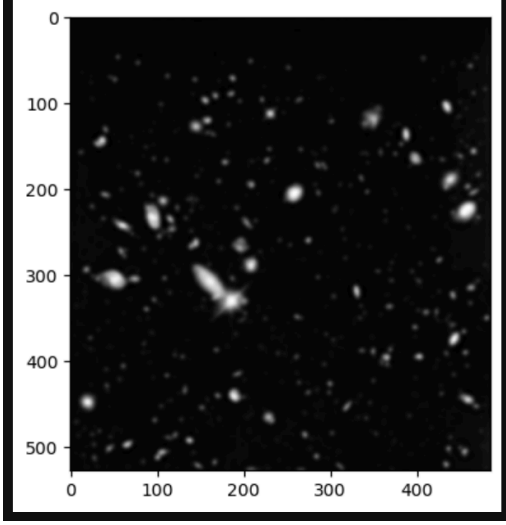
```
(528, 485)
[5]:
<matplotlib.image.AxesImage at 0x1c587df4550>
```



- For the next filter I used a butterworth function also provided by scikit-image.

```
hw1_img_bw = filters.butterworth(hw1_img, high_pass=False, cutoff_frequency_ratio=
plt.imshow(hw1_img_bw, cmap="gray")
```

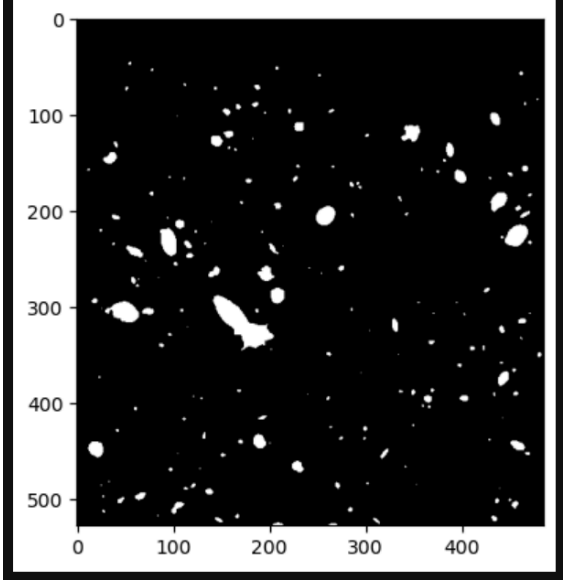
```
[6]:
<matplotlib.image.AxesImage at 0x1c587e8e820>
```



- After the filters I decided to apply the gaussian filtered image to a threshold function that is also from scikit-image. I then took the result of that and applied to the gaussian image itself and made a new image.

```
thresh = filters.threshold_otsu(hw1_img_gaussian)
hw1_img_thresh = hw1_img_gaussian > thresh
plt.imshow(hw1_img_thresh, cmap="gray")
```

```
[7]:
<matplotlib.image.AxesImage at 0x1c58990b7c0>
```

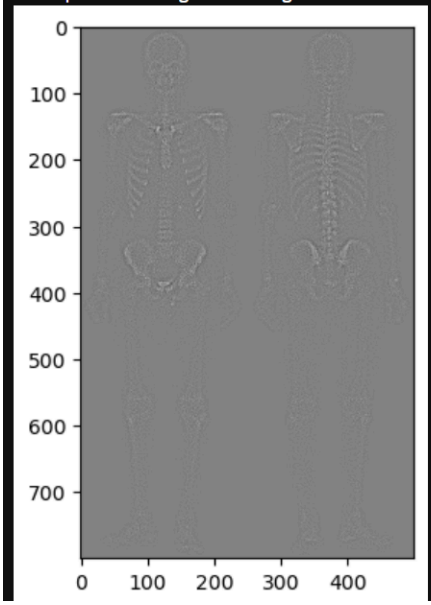


Question 2 Breakdown

- The first thing I did was take a Laplacian of the image.

```
hw2_img_laplacian = filters.laplace(hw2_img)
plt.imshow(hw2_img_laplacian, cmap="gray")
```

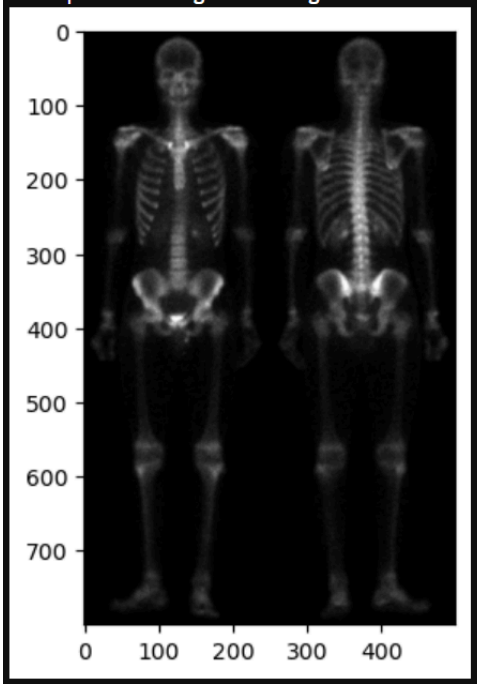
```
[10]:
<matplotlib.image.AxesImage at 0x1c5898c9610>
```



- I then sharpened the image by adding the laplacian image to the original image.

```
hw2_img_sharpened = hw2_img + hw2_img_laplacian
plt.imshow(hw2_img_sharpened, cmap="gray")
```

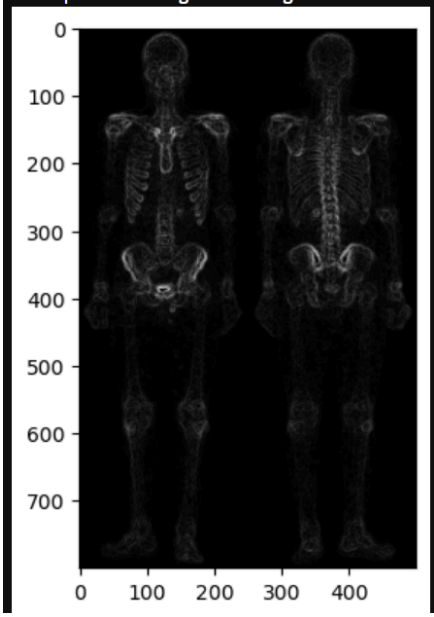
```
[11]:

<matplotlib.image.AxesImage at 0x1c589ac6b80>
```



- I then filter the original image using a sobel filter.

```
hw2_img_sobel = filters.sobel(hw2_img)
plt.imshow(hw2_img_sobel, cmap="gray")
```

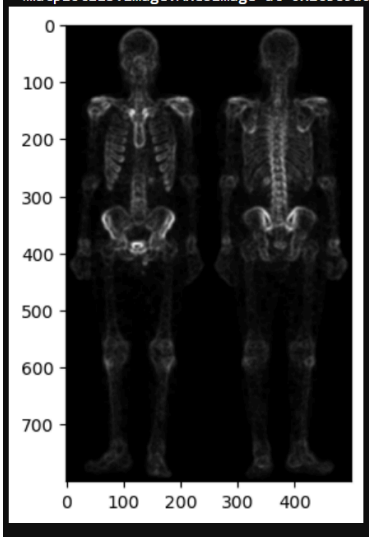
```
[12]:

<matplotlib.image.AxesImage at 0x1c5899c1670>
```



- I then smoothed the image.

```
hw2_img_smoothed = filters.rank.mean(util.img_as_ubyte(hw2_img_sobel), footprint=m
plt.imshow(hw2_img_smoothed, cmap="gray")
```

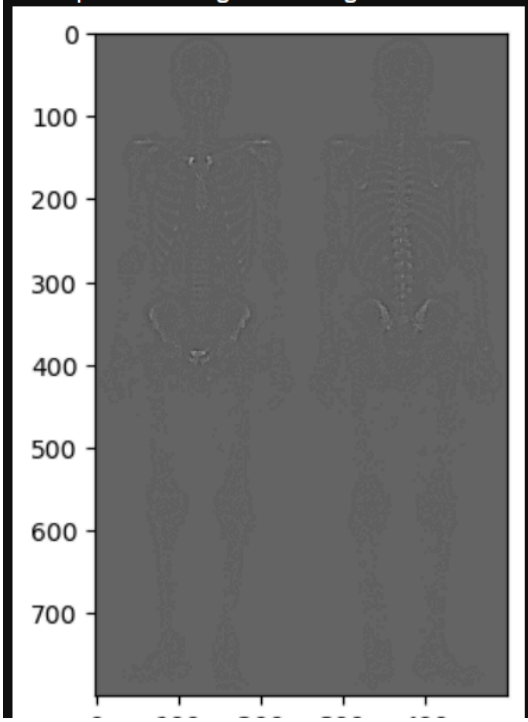
```
[13]:
<matplotlib.image.AxesImage at 0x1c589a6cdf0>
```



- I then took the product of the laplacian image and the smoothed image to make a mask.

```
hw2_img_mask = hw2_img_laplacian * hw2_img_smoothed
plt.imshow(hw2_img_mask, cmap="gray")
```

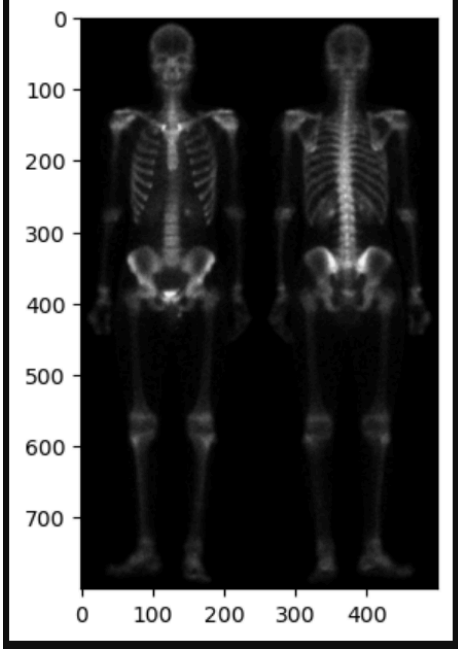
```
[14]:
<matplotlib.image.AxesImage at 0x1c589b2f430>
```



- I then sharpened the mask image with the original image to make another sharpened image.

```
hw2_img_sharpened2 = hw2_img + hw2_img_mask
plt.imshow(hw2_img_sharpened2, cmap="gray")
```

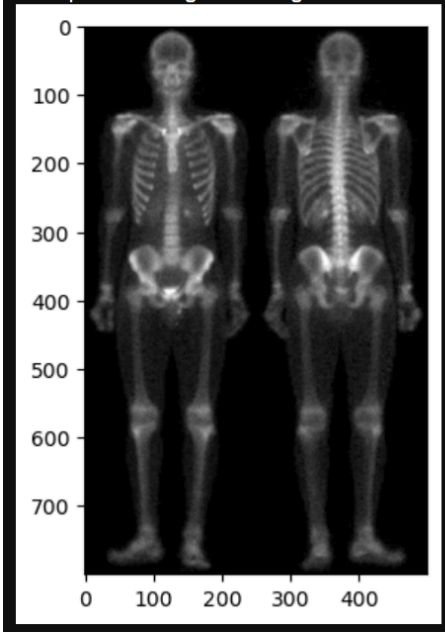
```
[15]:

<matplotlib.image.AxesImage at 0x1c589ba0d30>
```



- Finally I just took the power law transformation of the previously sharpened image.

```
hw2_img_power = exposure.adjust_gamma(hw2_img_sharp_rescaled, gamma=0.6)
plt.imshow(hw2_img_power, cmap="gray")
```

```
[17]:

<matplotlib.image.AxesImage at 0x1c589c25280>
```

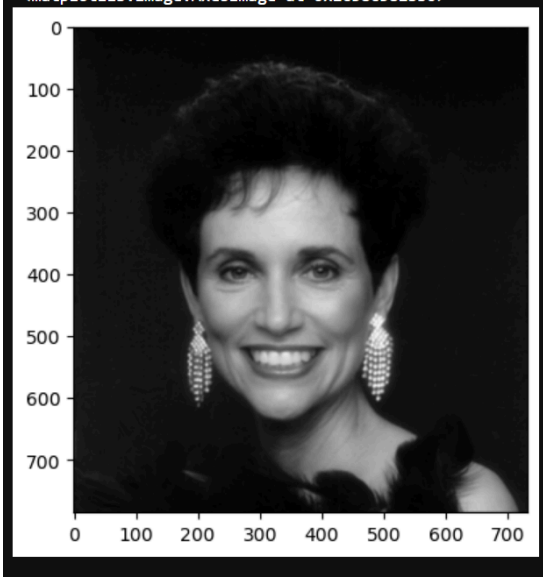


Question 3

- The first thing I did was created a low pass butterworth filter function because the scikit-image function would not work with a cutoff frequency of 140 or 120.
- I then used the newly created function to do a cutoff frequency of 140

```
hw3_img_bw140 = butterworth_highpass_filter(hw3_img, 140, 2)
plt.imshow(hw3_img_bw140, cmap="gray")
```

```
[21]:

<matplotlib.image.AxesImage at 0x1c58c582bb0>
```



- Finally I used the newly created function to do a cutoff frequency of 120

```
hw3_img_bw120 = butterworth_highpass_filter(hw3_img, 120, 2)
plt.imshow(hw3_img_bw120, cmap="gray")
```

[22]:

```
<matplotlib.image.AxesImage at 0x1c58c6102e0>
```