Emile Goulard
uID: u1244855
CS3200 - Assignment 9 Report
12/9/2022

Question 1

a) **After running “Eigenface_Assignment.m” and changing the for loop at line 38 to iterate 3 times, these were the produced Eigenfaces:**

Figure 1 - 1st Eigenface

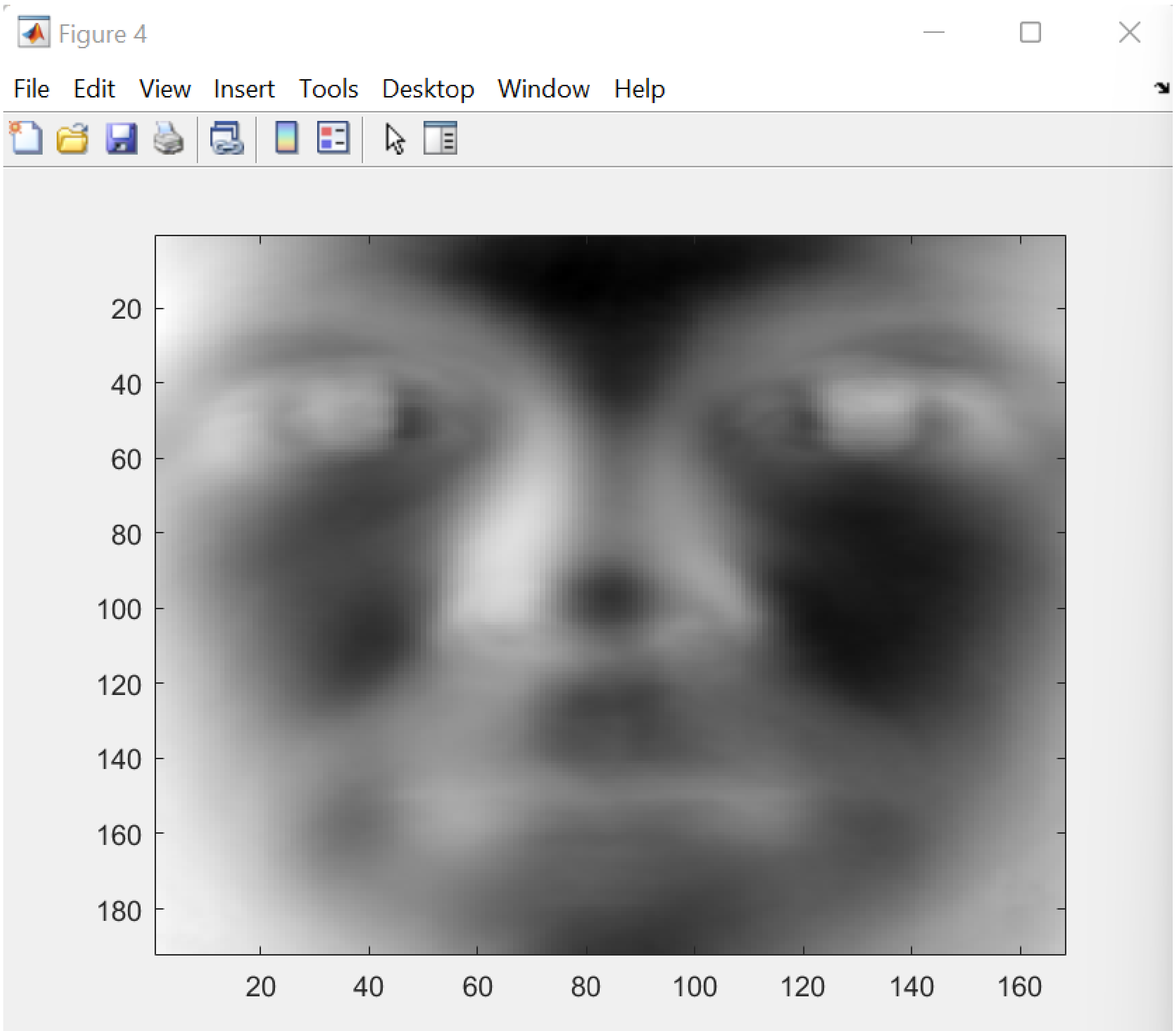


Figure 2 - 2nd Eigenface

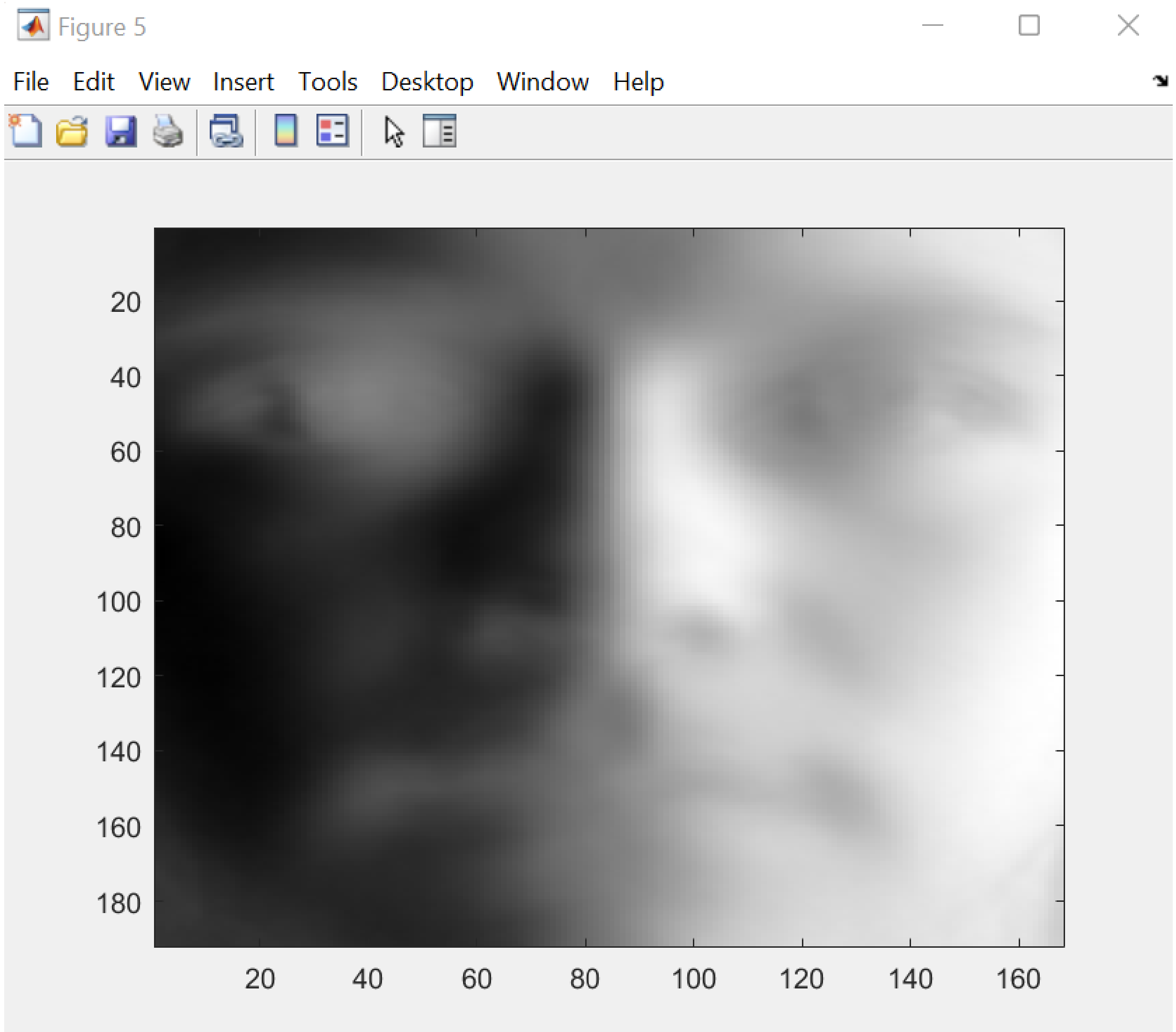
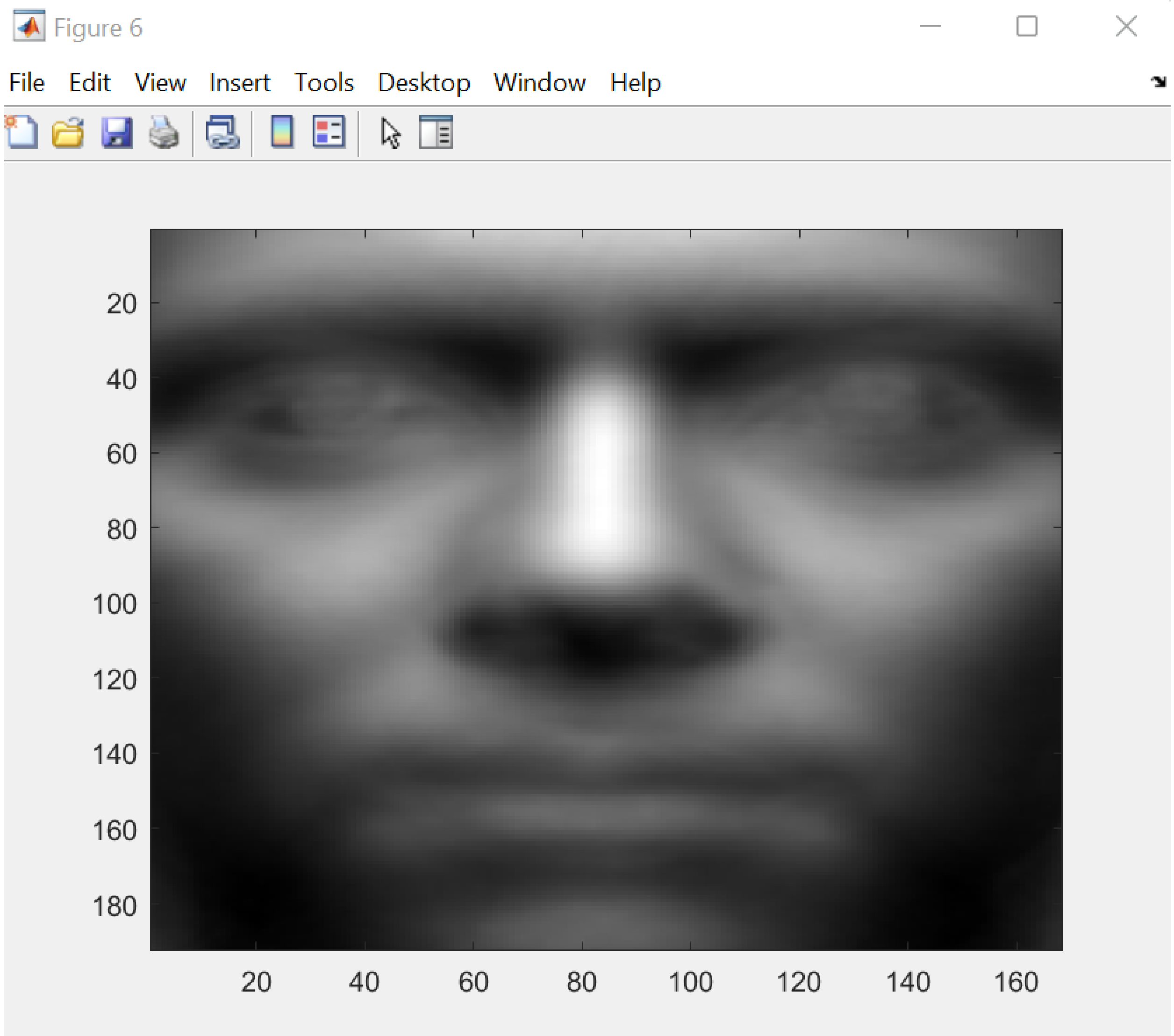


Figure 3 - 3rd Eigenface



All these Eigenfaces differ slightly because they all are the average of the faces. A majority of the faces have similar nose structure, mouth expressions, and eyes that are equal distance from each other in each image. I also noticed the stretch marks by the mouth in both Figure 1 and 3, but not so much Figure 2. The major difference between each image is the contrast of light and dark areas. They vary because I think each image focuses on particular aspects of one face from the matrix to create its Eigenface. Lastly, as a result, the Eigenfaces take aspects of nose structure, mouth expression, cheeks, and eyes to produce different images, yet you can pick out which images do resemble the faces in the matrix.

b) For this, I took a picture of myself and then put it inside my working directory provided in the Assignment10.zip file. Make sure to uncomment the line that uses the 'My_Face' file before running the program. This is the original image (called 'My_Face') I took of myself for reference:

Below are the results of running the program using the 'My_Face.jpeg' file:

Figure 1 - My Face at 425 Eigenfaces

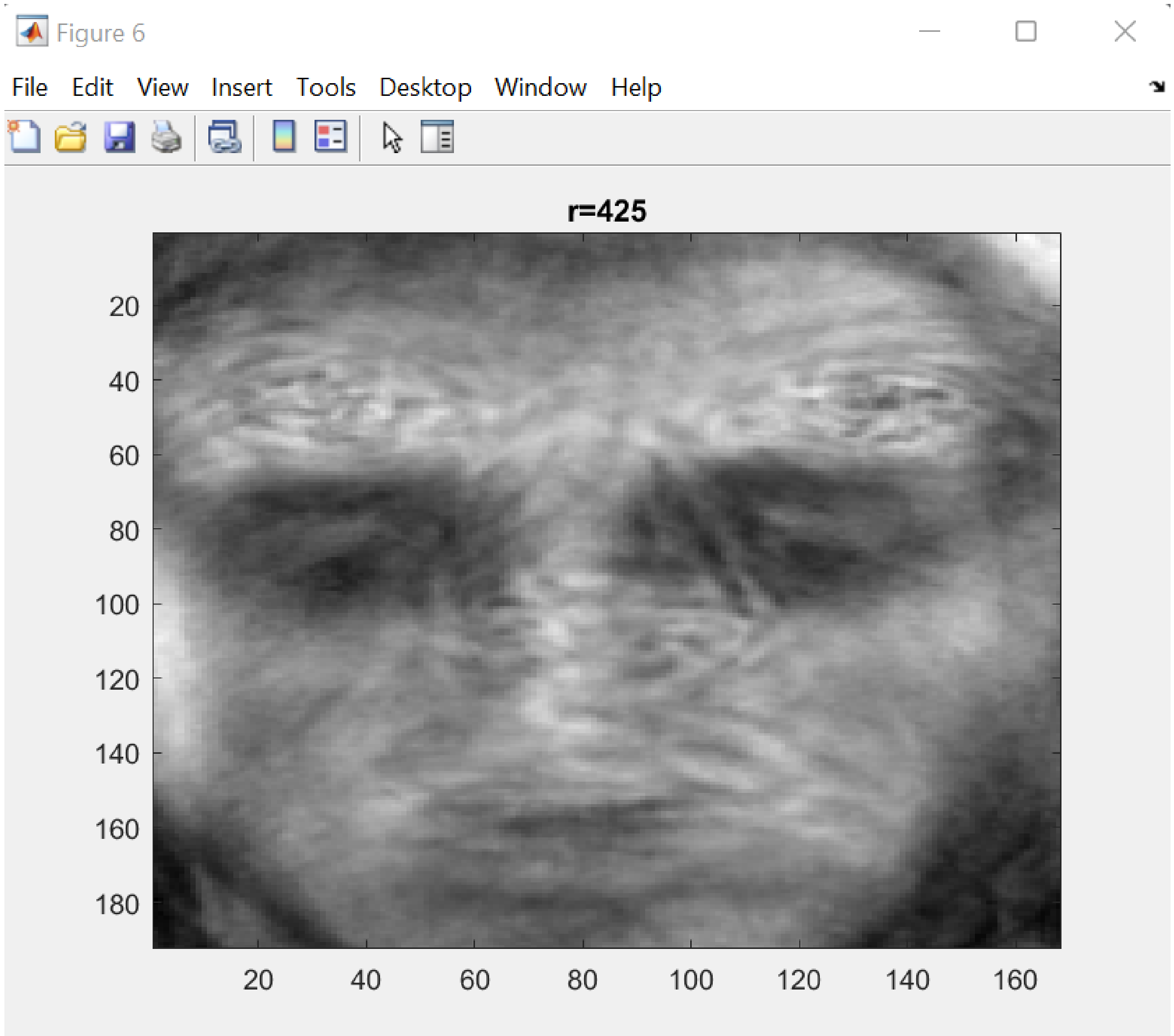


Figure 2 - My Face at 1050 Eigenfaces

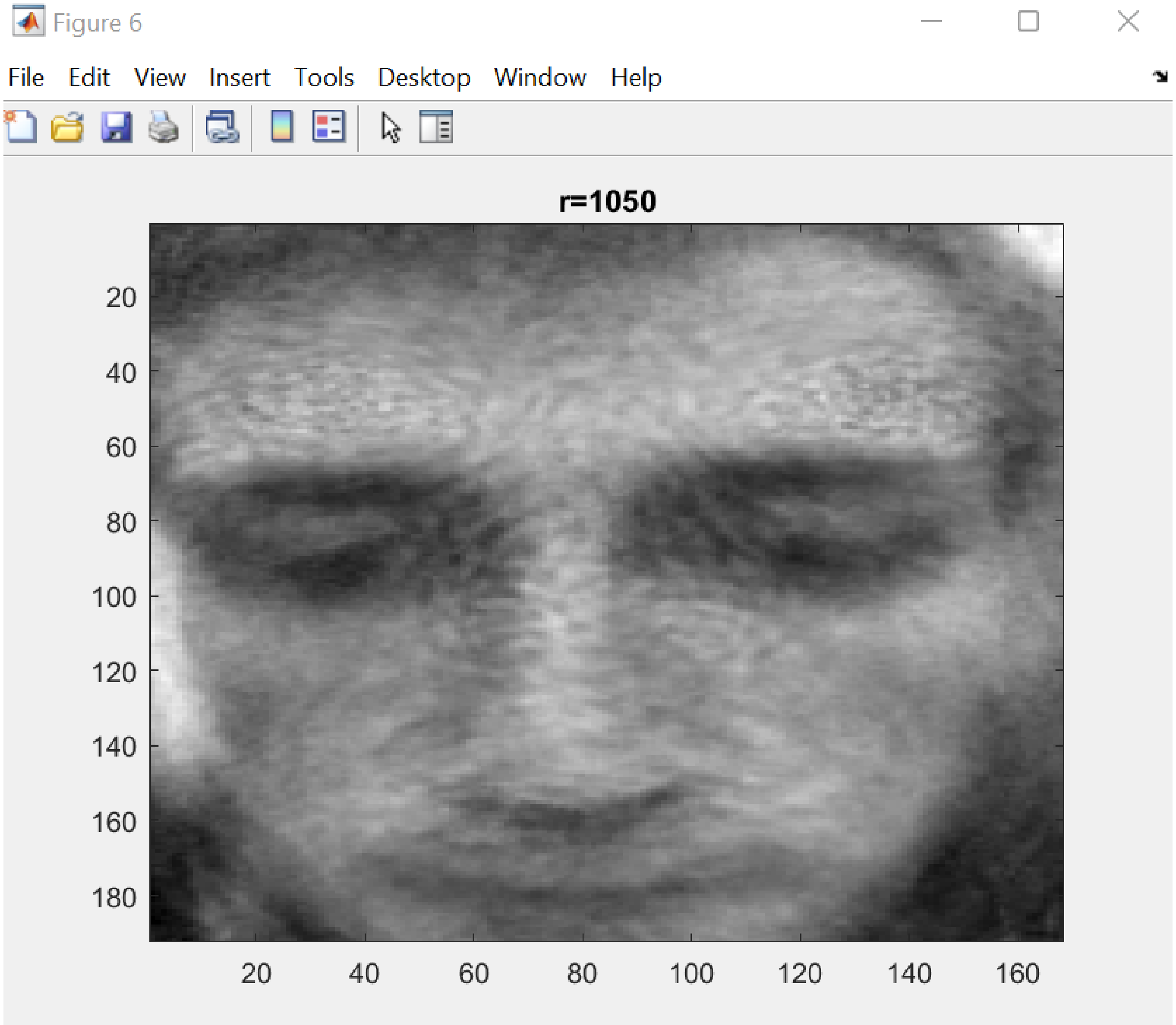
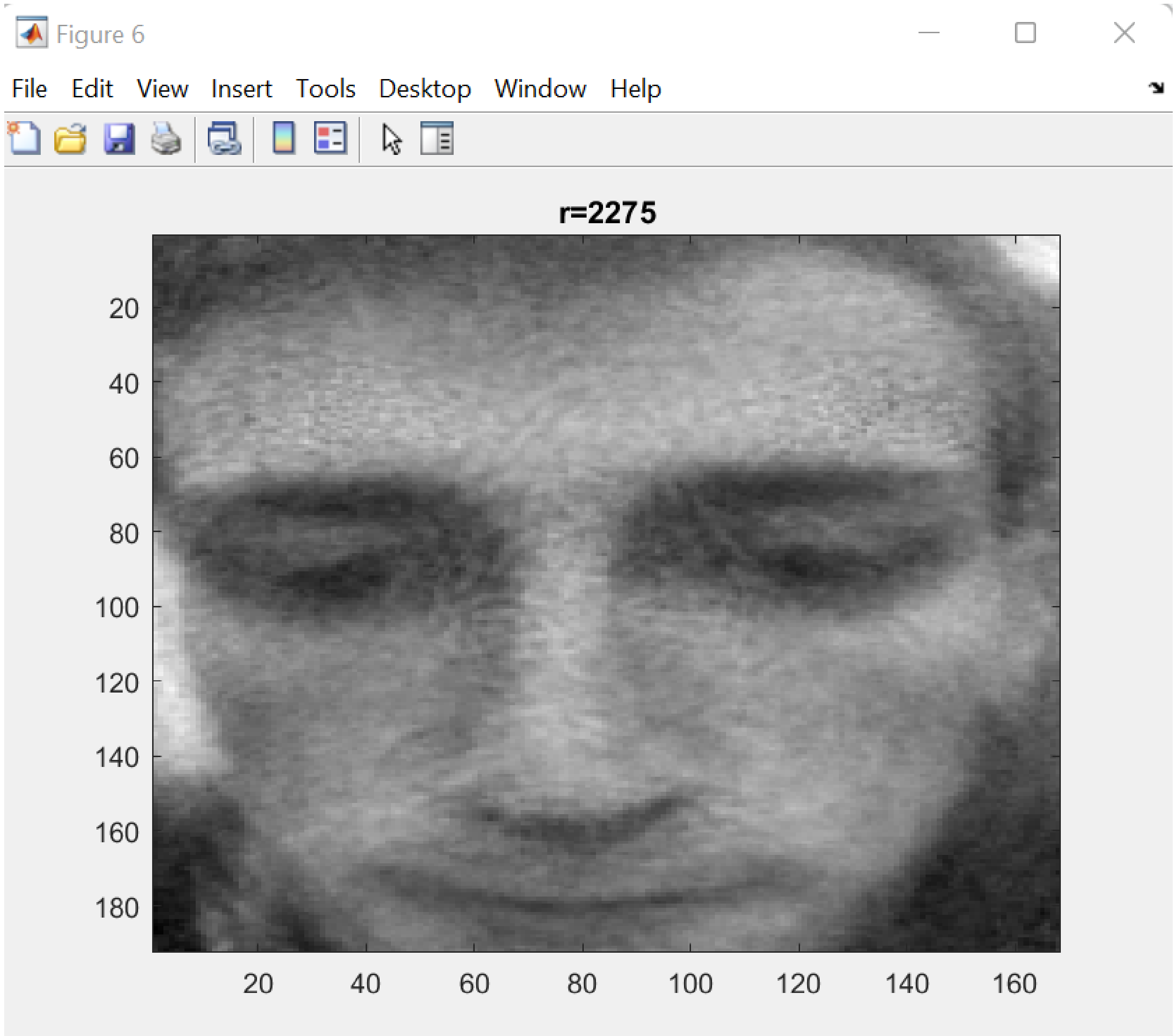


Figure 3 - My Face at 2275 Eigenfaces



As you can see in Figure 1, around 400-450 Eigenfaces are required to create a somewhat recognizable image like the original image. The following Figures 2 and 3 show a more complete image that resemble the original much better.

---

c) For this, I took a picture of myself and then put it inside my working directory provided in the Assignment10.zip file. Make sure to uncomment the line that uses the 'My_Face_Rotation' file before running the program. This is the original image (called 'My_Face_Rotation') I took of myself for reference:

Below are the results of running the program using the ‘My_Face_Rotation.jpeg’ file:

Figure 1 - My Rotated Face at 475 Eigenfaces

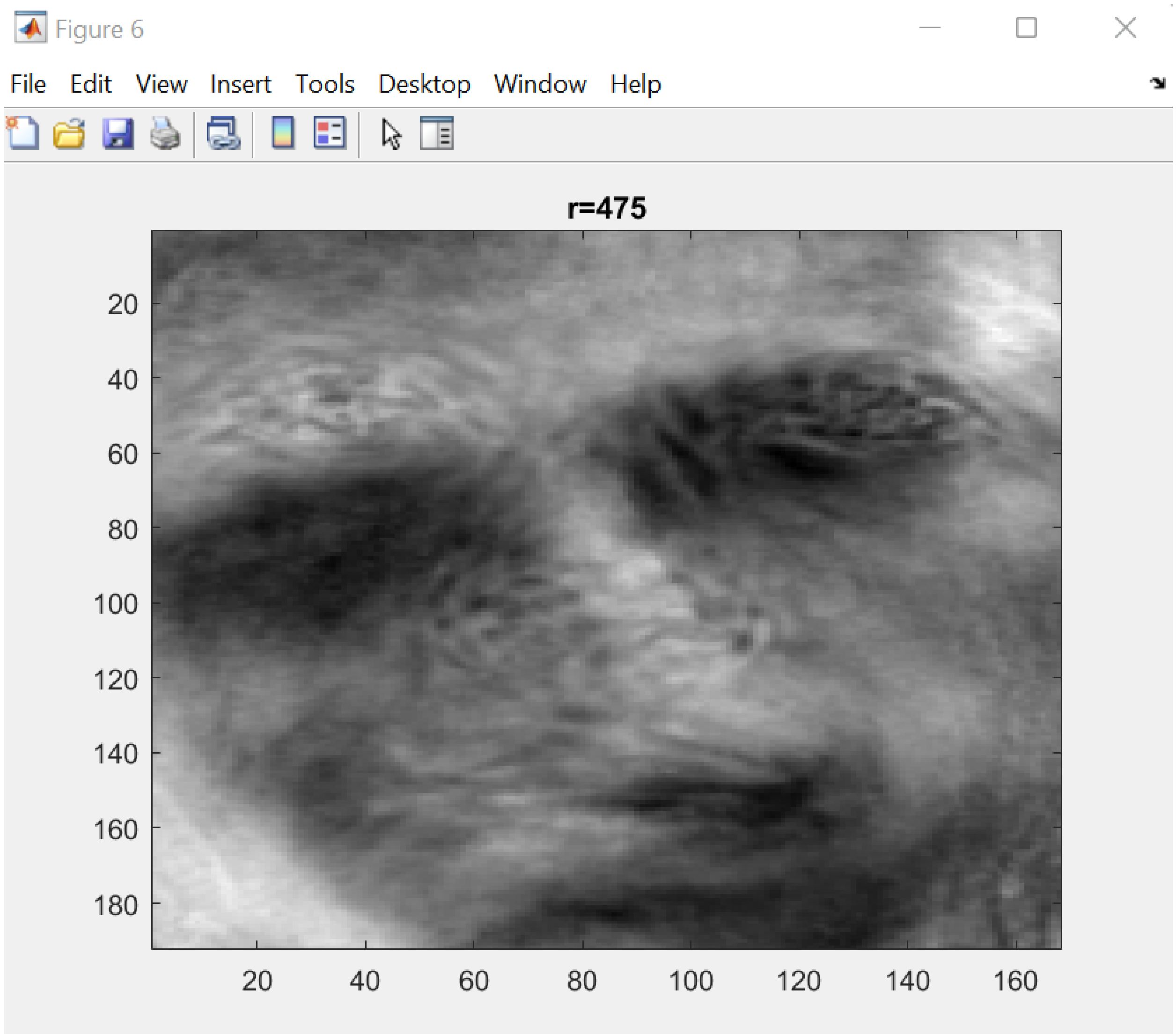


Figure 2 - My Rotated Face at 1025 Eigenfaces

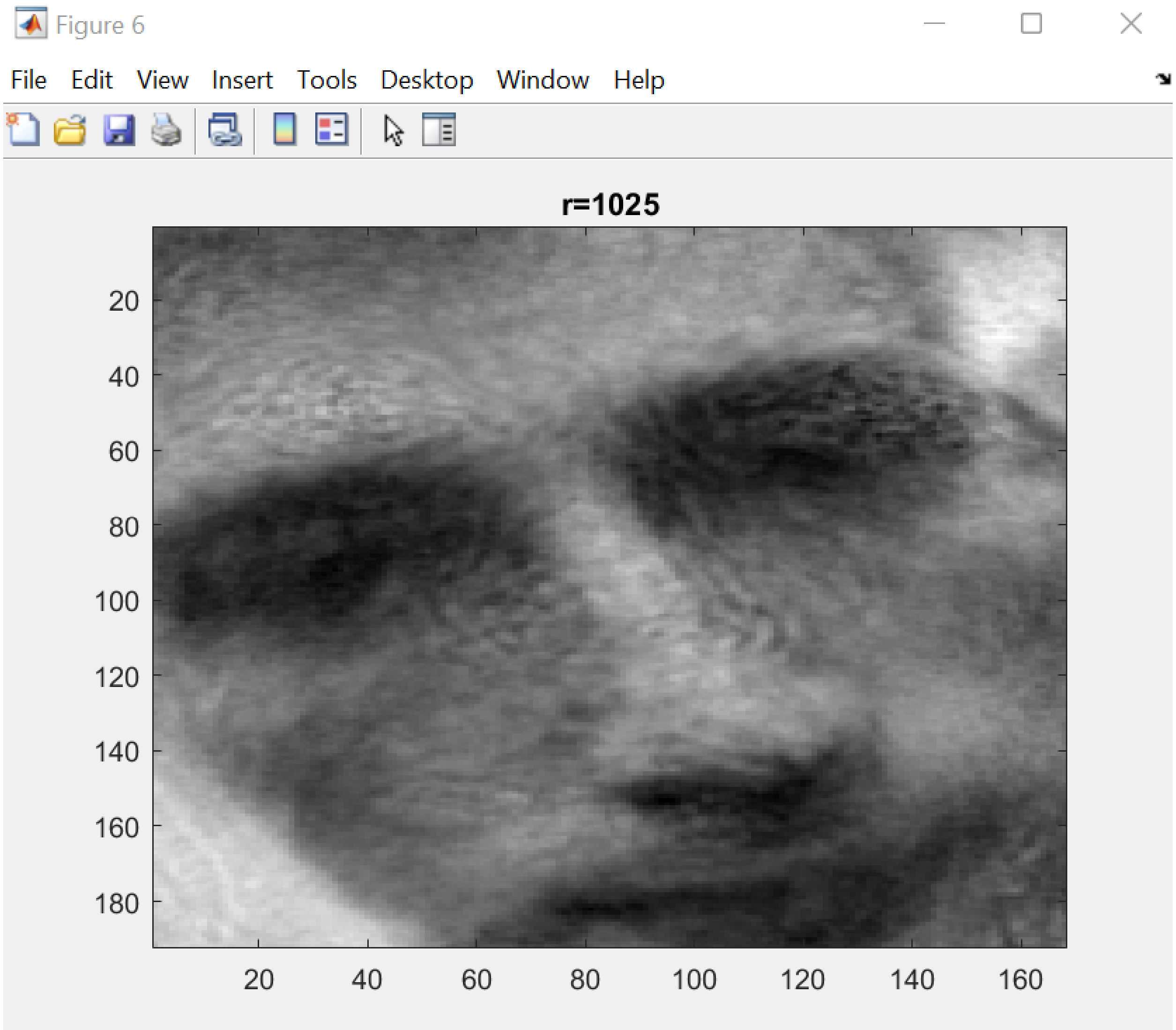
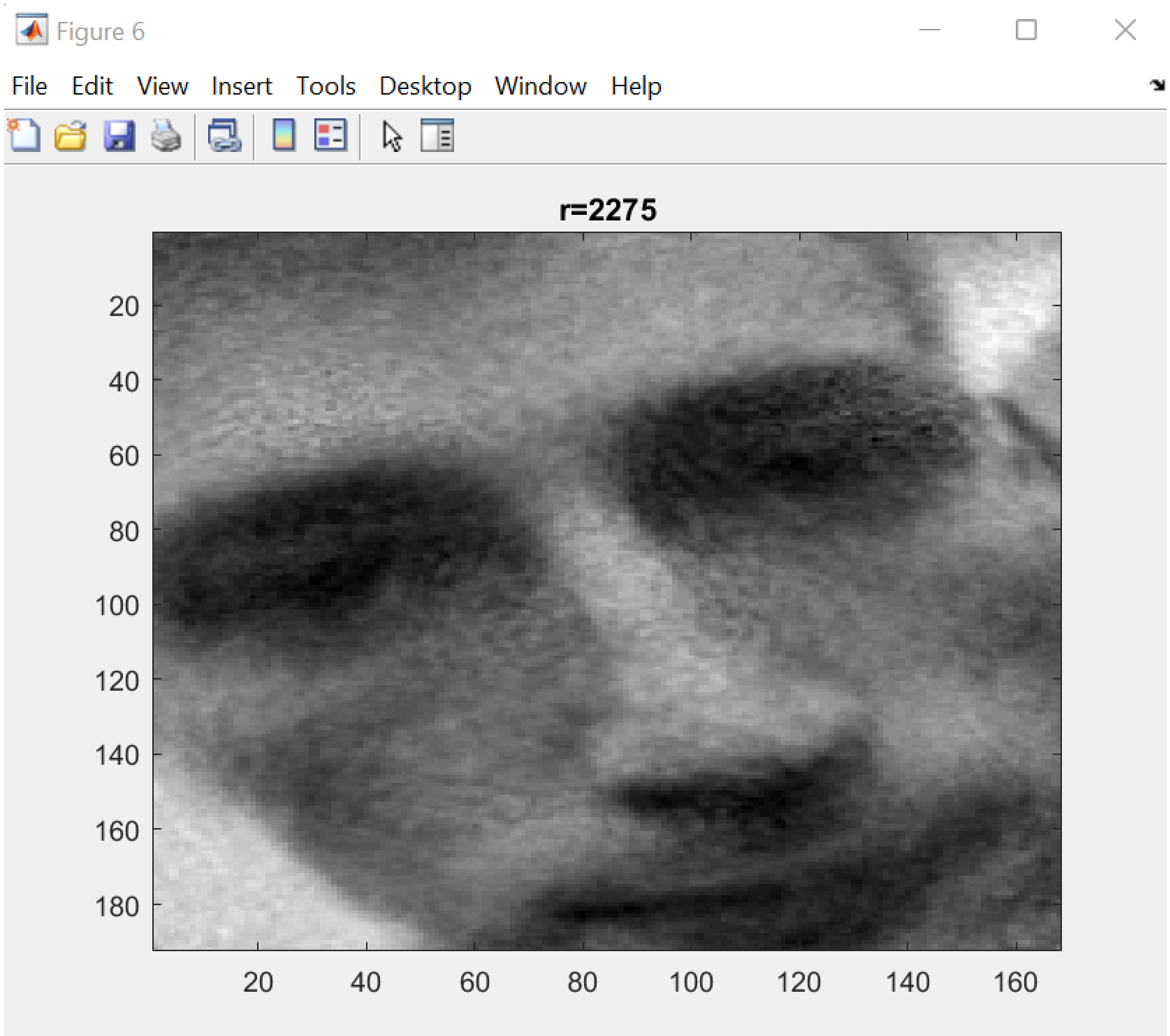


Figure 3 - My Rotated Face at 2275 Eigenfaces



As you can see in Figure 1, around 450-475 Eigenfaces are required to create a somewhat recognizable image like the original image. The following Figures 2 and 3 show a more complete image that resemble the original much better. During alignment, I noticed that the Eigenfaces produce a slightly muddier image compared to using my photo with center alignment. I think this is because the program works to identify areas where the eyes, nose, mouth, etc. would be but fails to perfectly capture it because of the skewed input image. Therefore, it produces an image that is less accurate and has more noise.

d) For this, I took a picture of myself and then put it inside my working directory provided in the Assignment10.zip file. Make sure to uncomment the line that uses the 'Dog_Image' file before running the program. These were the results:

Figure 1 - Dog Image with 475 Eigenfaces

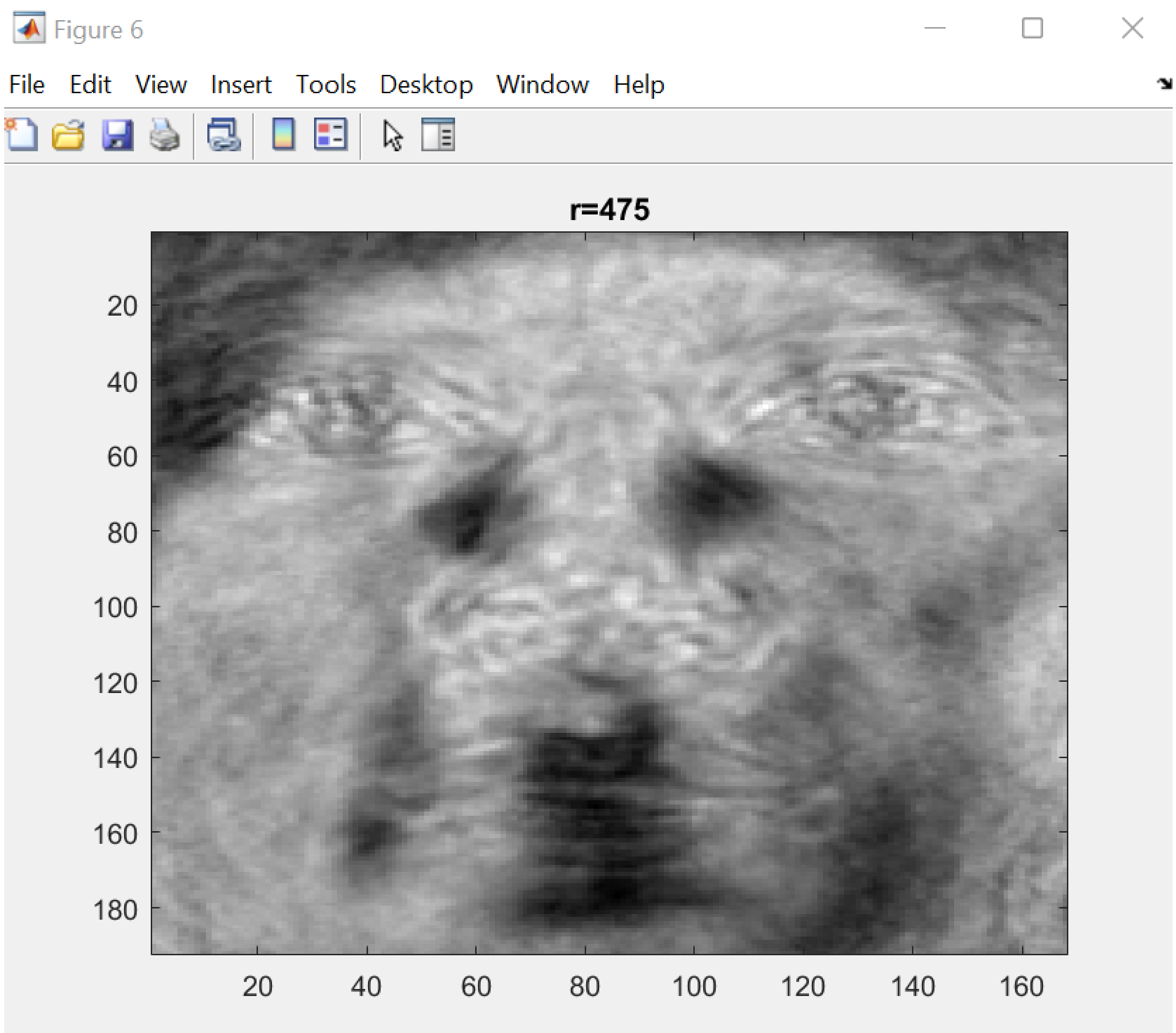


Figure 2 - Dog Image with 925 Eigenfaces

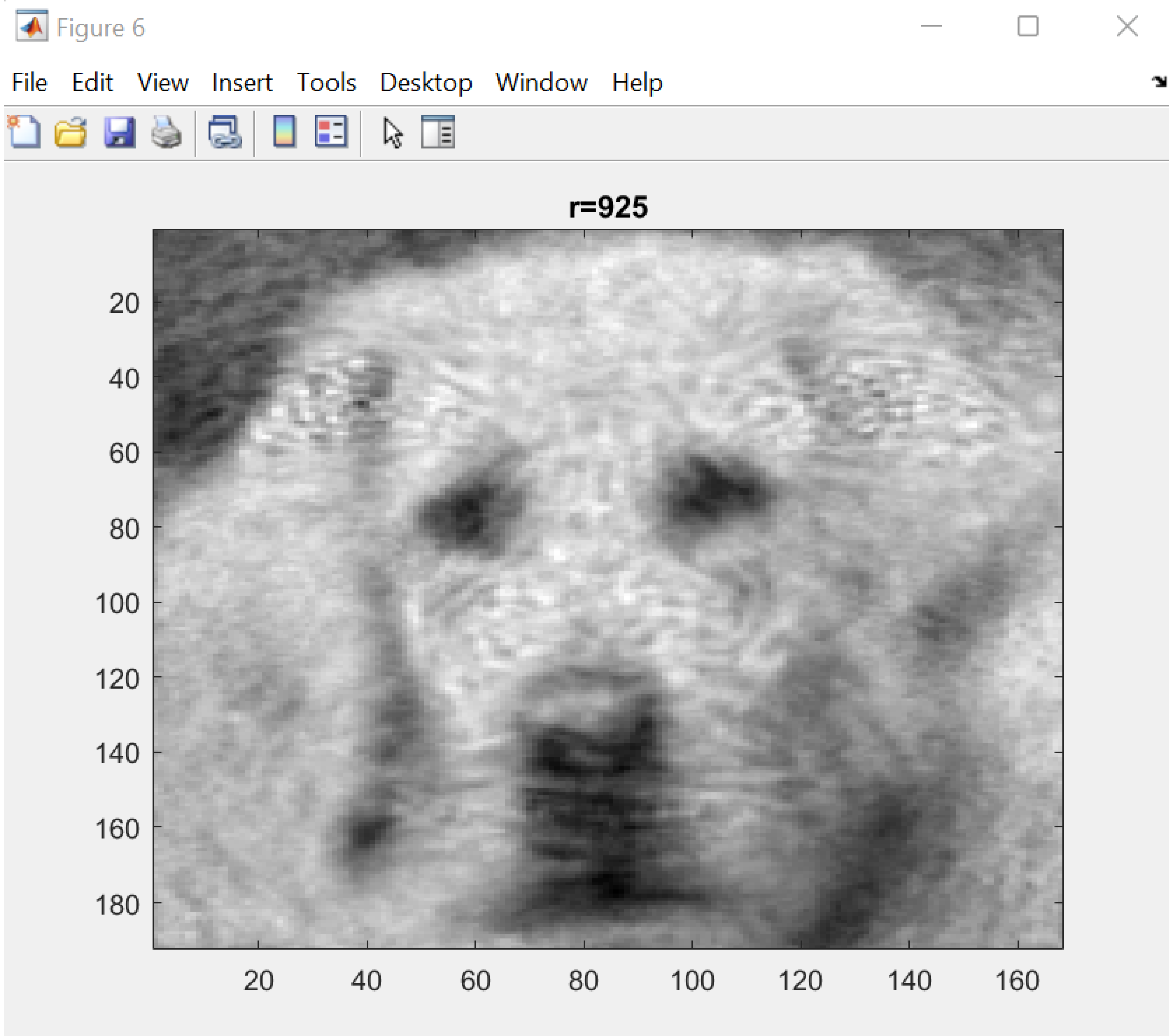
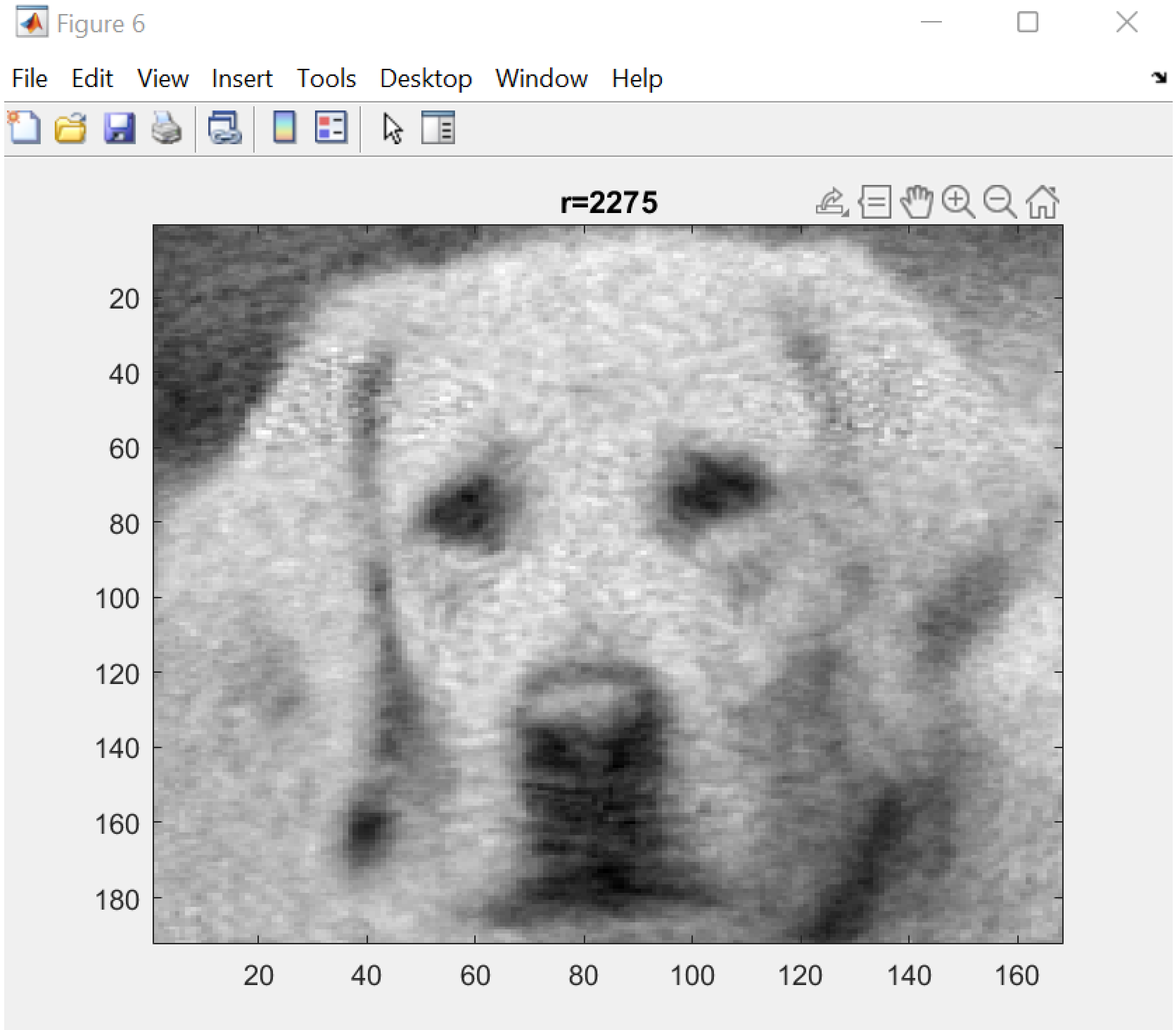


Figure 3 - Dog Image with 2275 Eigenfaces



In Figure 1, the image didn't start to resemble a dog until 400 Eigenfaces and thus I captured the moment at 475 which is fairly similar. I think the reason why the image does produce a dog image is because it doesn't actually reconstruct the image but it does sum across the weights of the eigenvectors to produce an image similar to its input and provide detail where lighter and darker areas appear.

---

f) **For this part, I used the code inside the for loop to plot the eigenface and used it in a section (lines 43 - 61) to compute the eigenfaces for each singular value. Below are the results of running the code:**

Figure 1 - Singular Value of 500 Eigenfaces

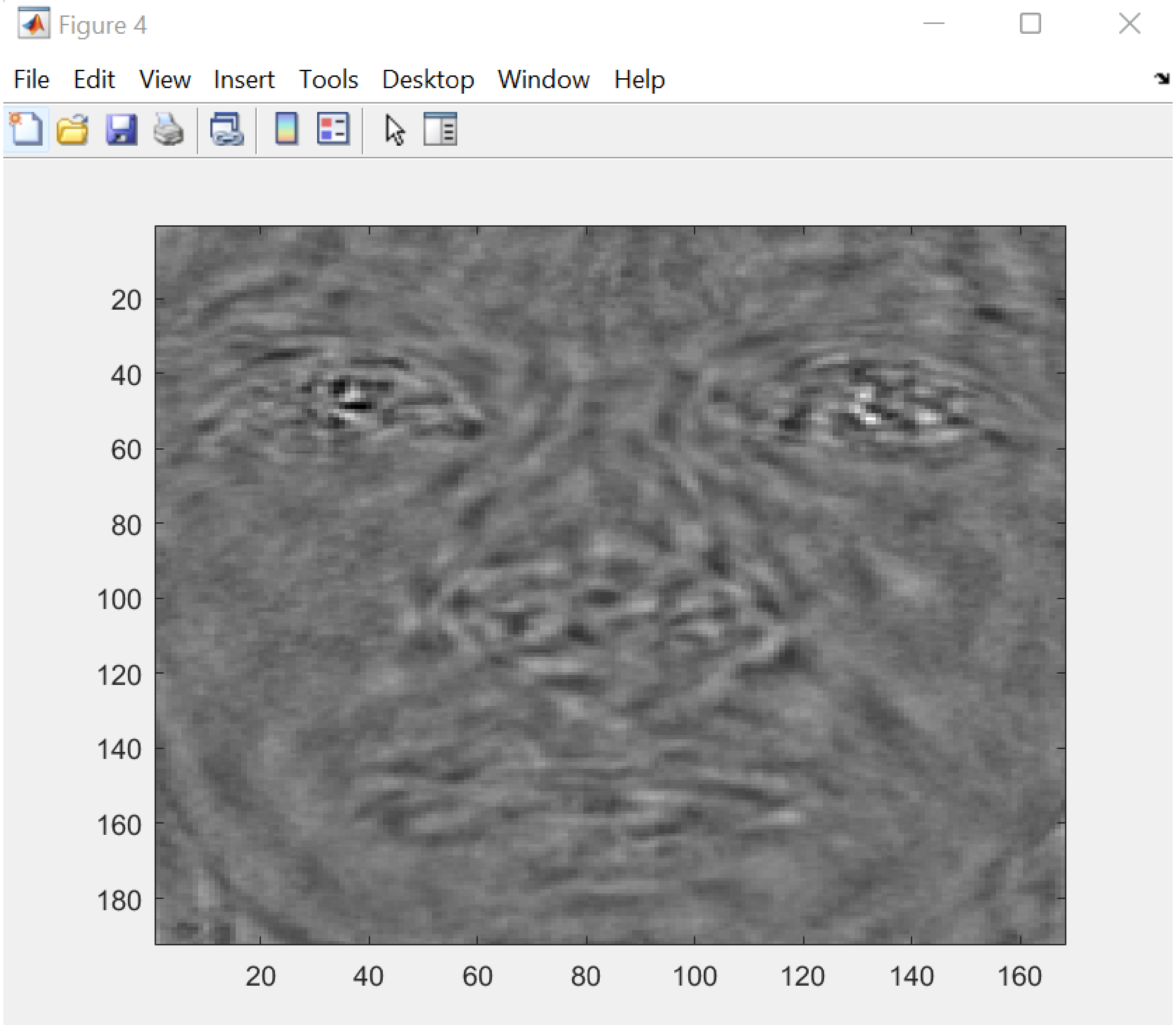


Figure 2 - Singular Value of 1000 Eigenfaces

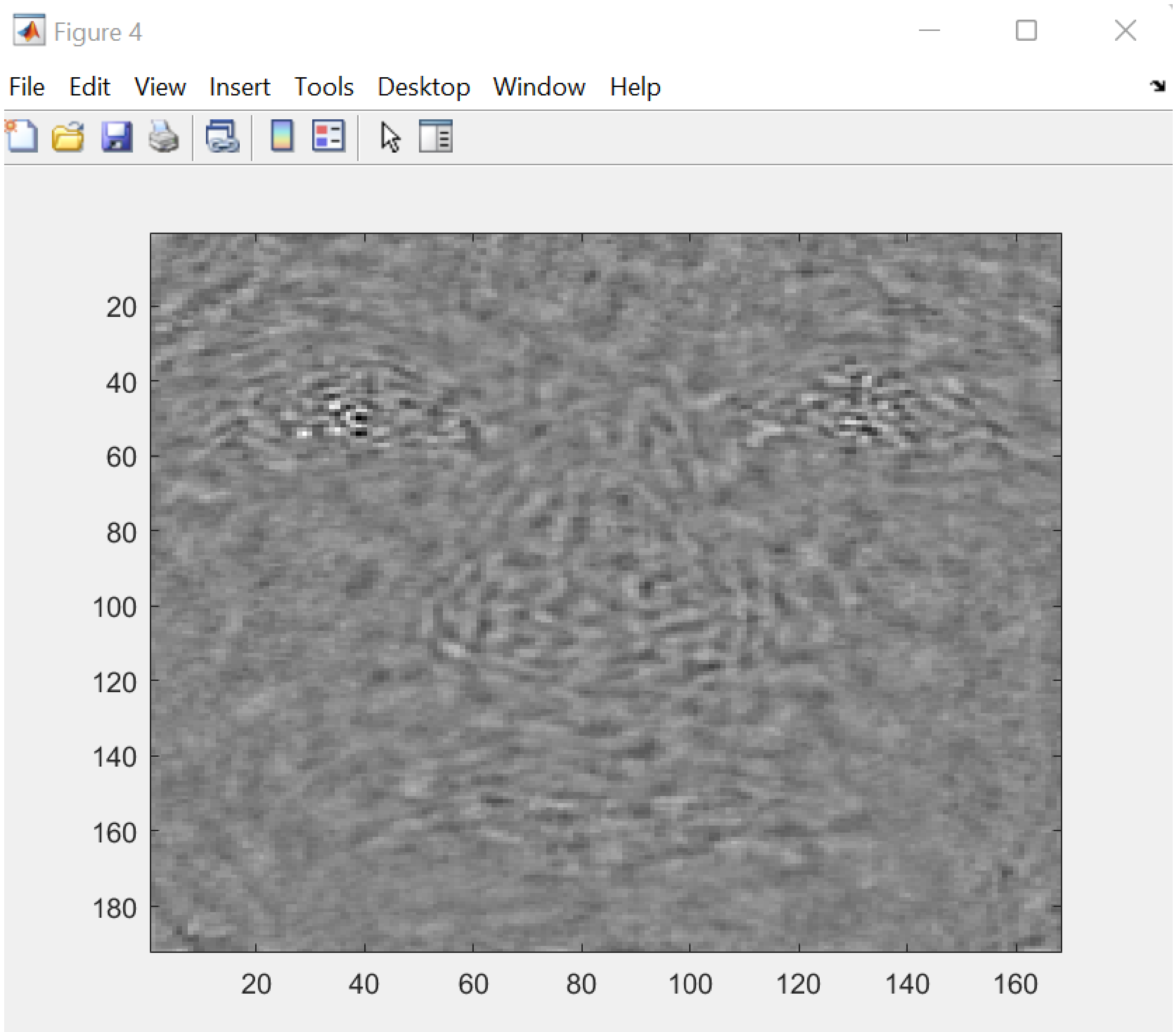


Figure 3 - Singular Value of 1500 Eigenfaces

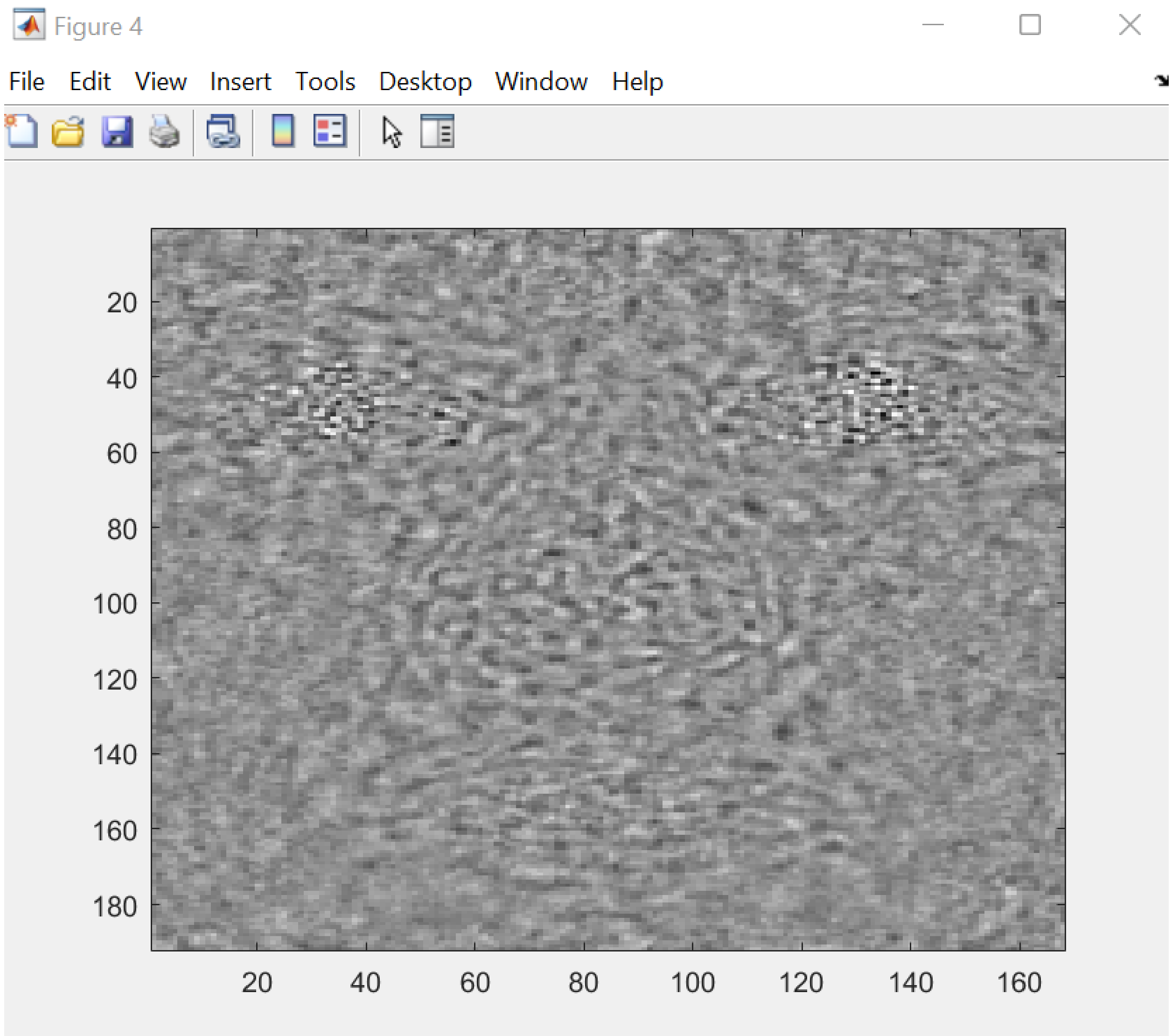
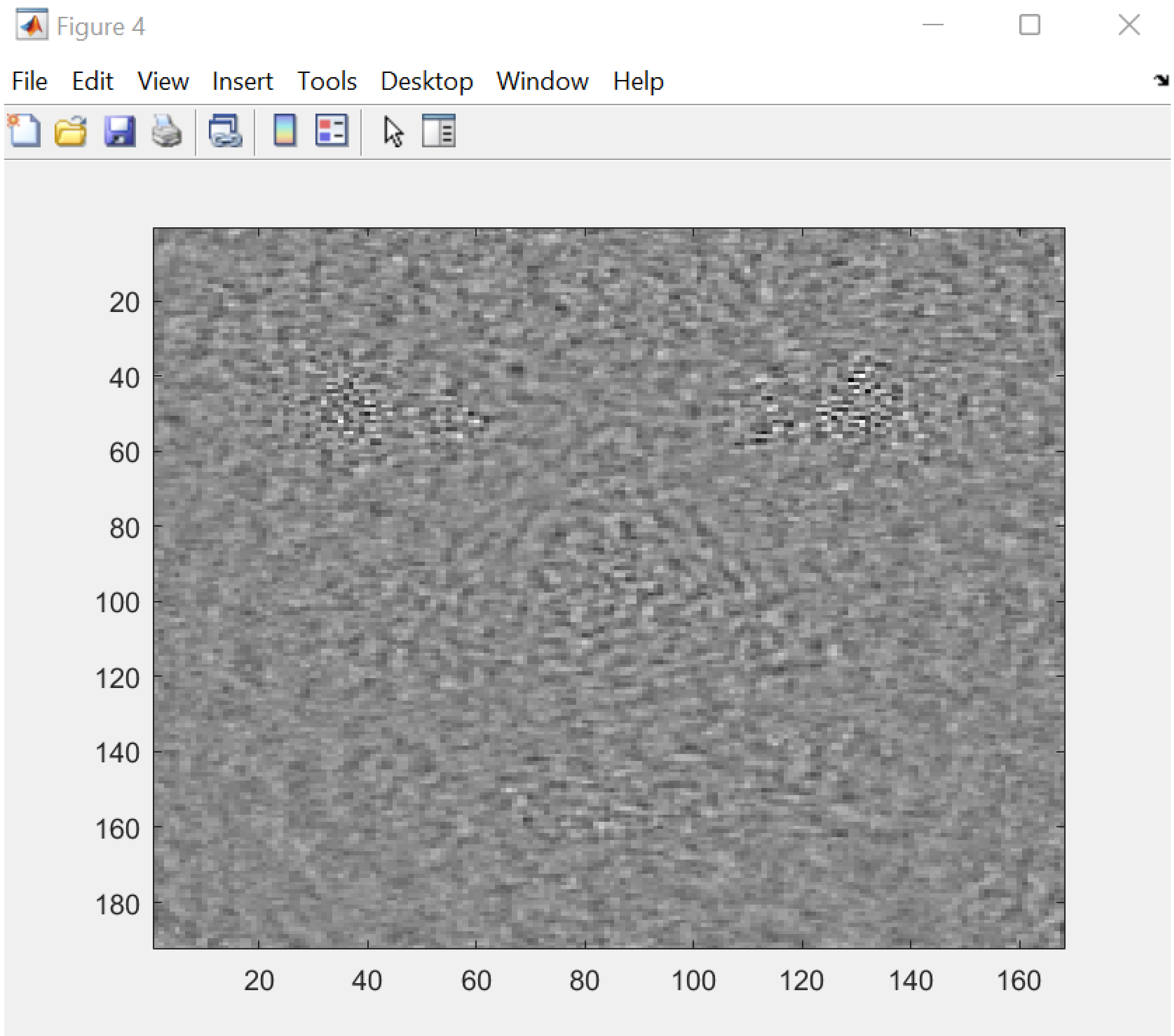


Figure 4 - Singular Value of 2000 Eigenfaces



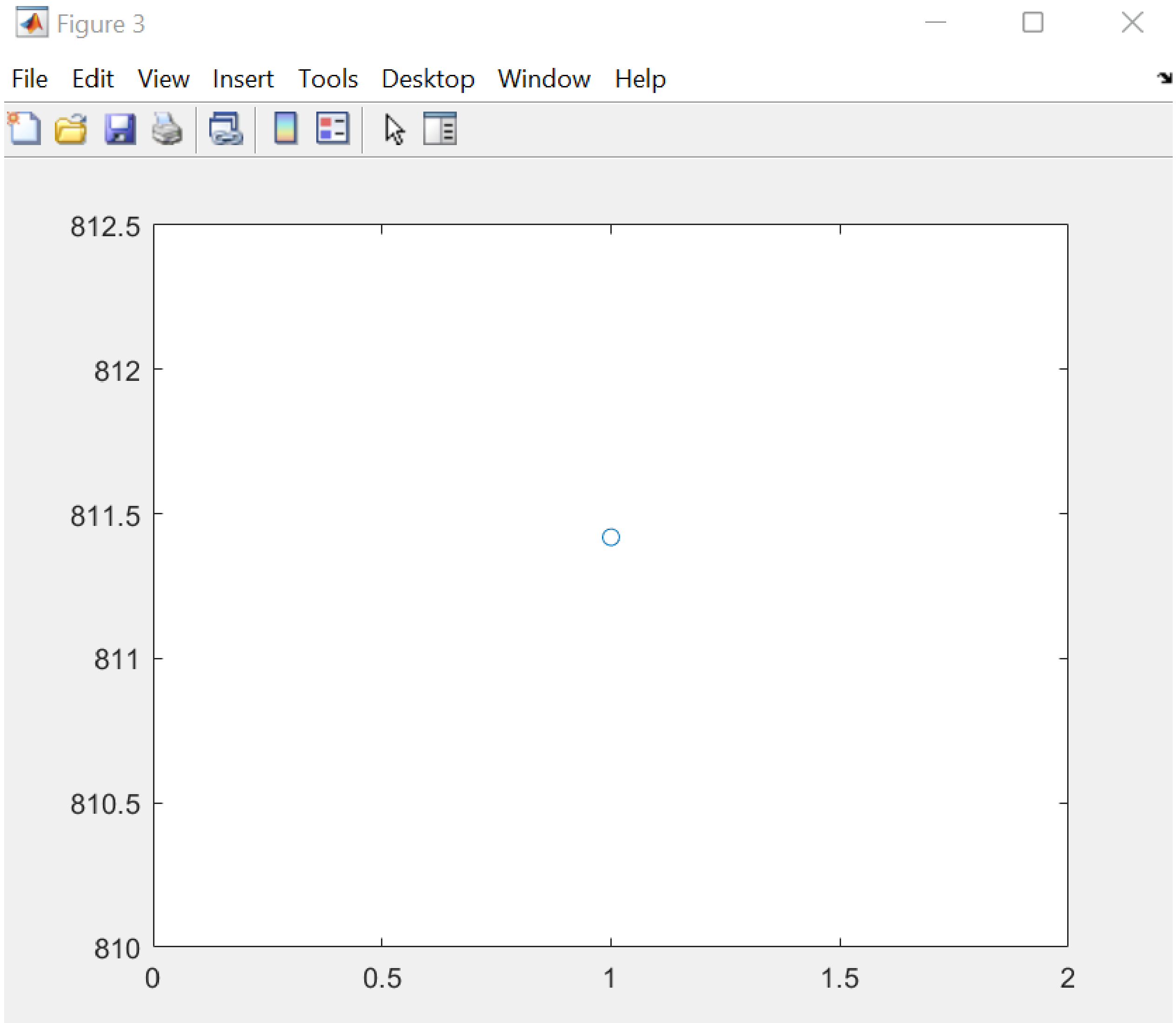
Based on the output shown in Figures 1 - 4, I can say that they do derive from the same averaged faces in the Figures in Part a; however, as the singular values increase, the more noise there is, and thus, the less accurate the Eigenfaces are. Compared to the first three Eigenfaces, Figure 1 displays the closest resemblance to those faces because I can still make out the cheeks, nose, and mouth where the shadows are located. While Figures 2 - 4 are filled with mostly noise, I find it still interesting that there is more noise in areas such as the eyes and mouth (and even a little noisy area where the nose is). So it's still possible to pinpoint where facial features would be at higher singular values.

For finding the median of the singular values, I inserted the

```
figure, plot(median(diag(S)),'-o')
```

at line 37 which resulted in:

Figure 5 - Median of Singular Values



Therefore the median of singular values is 811.5.

EXTRA CREDIT

Question 1

a) A Fractal is a shape that recursively contains detailed geometric structures at much smaller scales.
b) The term ‘fractal’ was coined by Benoît Mandelbrot in 1975. The term fractal originates from a latin word meaning ‘broken’ or ‘fractured’.

c) To create a fractal using Newton's method, first you must find the function, the derivative of said function and some value 'a' that approximates the root of said function. Then, you calculate the derivative of a using the formula to find a closer approximation of the root:

$$a' = a - f(a) / f'(a)$$

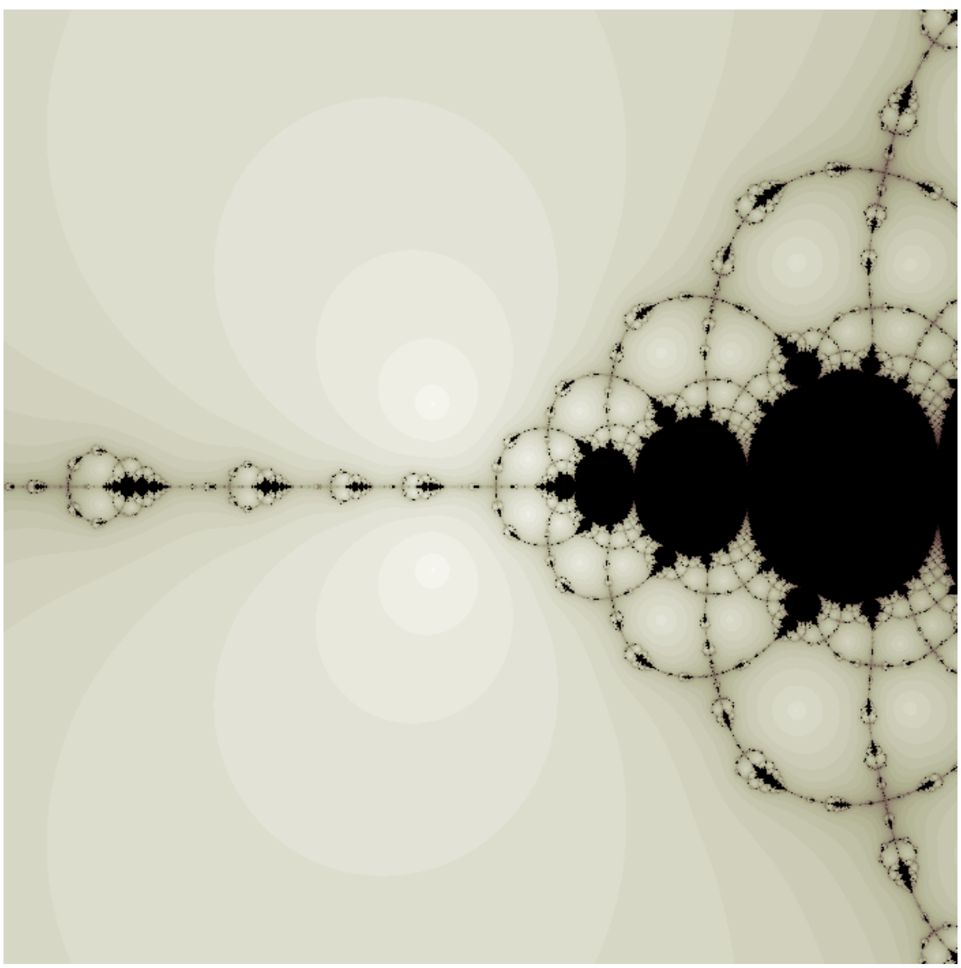
Lastly, you simply iterate finding the closer approximation of the root again and again until successive values are close enough to each other given some threshold value. This is determined by comparing to the actual value of 'r' using the formula:

$$f(r) = 0$$

Question 2

All Newton Fractal PNGs are located in the 'Newton Fractal' Folder

a) After looking at all the original 12 Newton Fractals, I like Newton Fractal #7 the best. It looks really cool. Here's the image for reference:



b) For this, I created three new fractals that can be viewed inside the same ‘Newton Fractal’ Folder. Inside the ‘NewtonFractal.m’ file, I added and edited three polynomials with different ranges, iterations, and thresholds. You can view these changes from line 172 - 190. These are Fractal numbers 13, 14, and 15 inside the folder:

Figure 1 - Newton Fractal 13

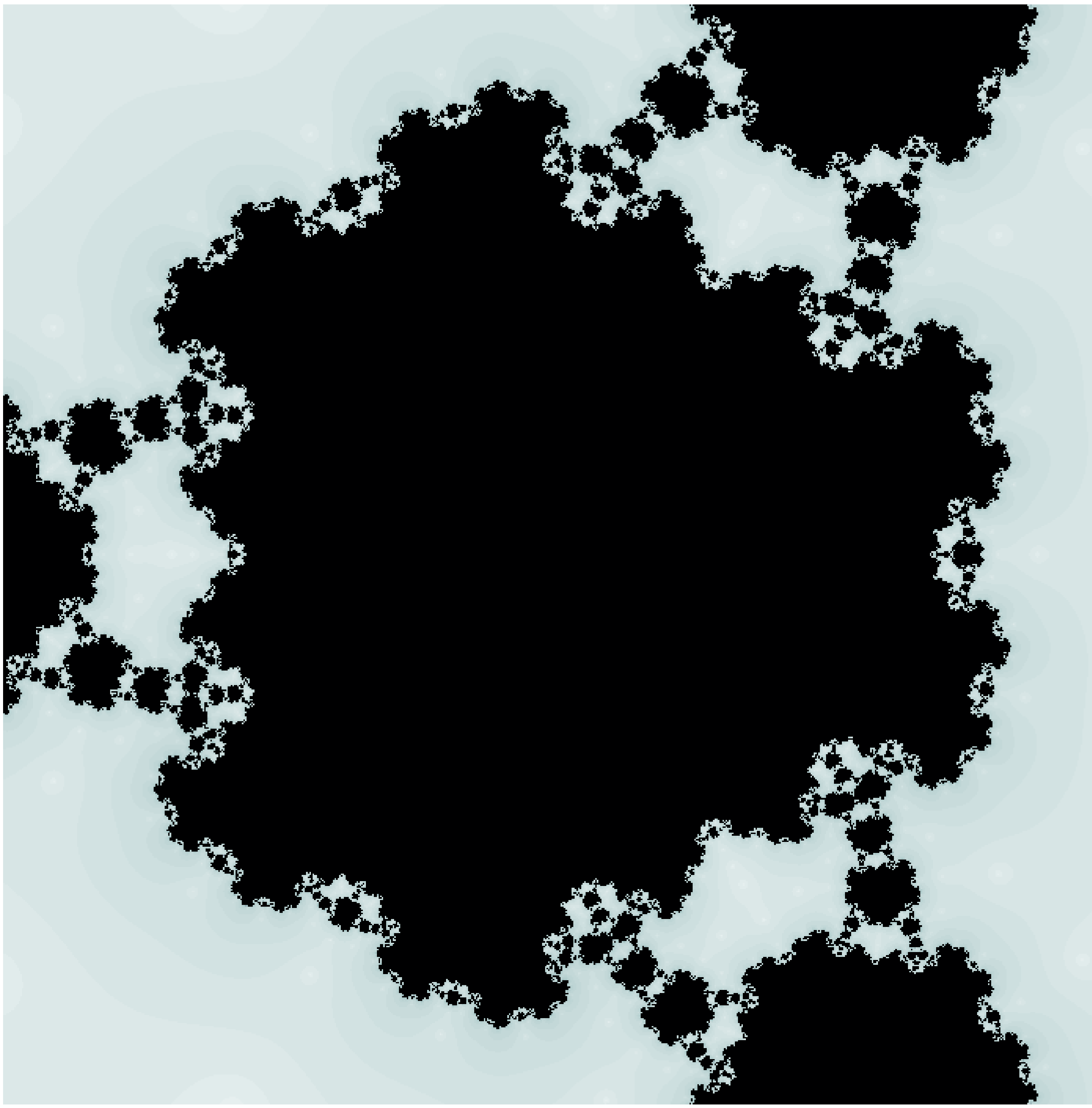


Figure 2 - Newton Fractal 14

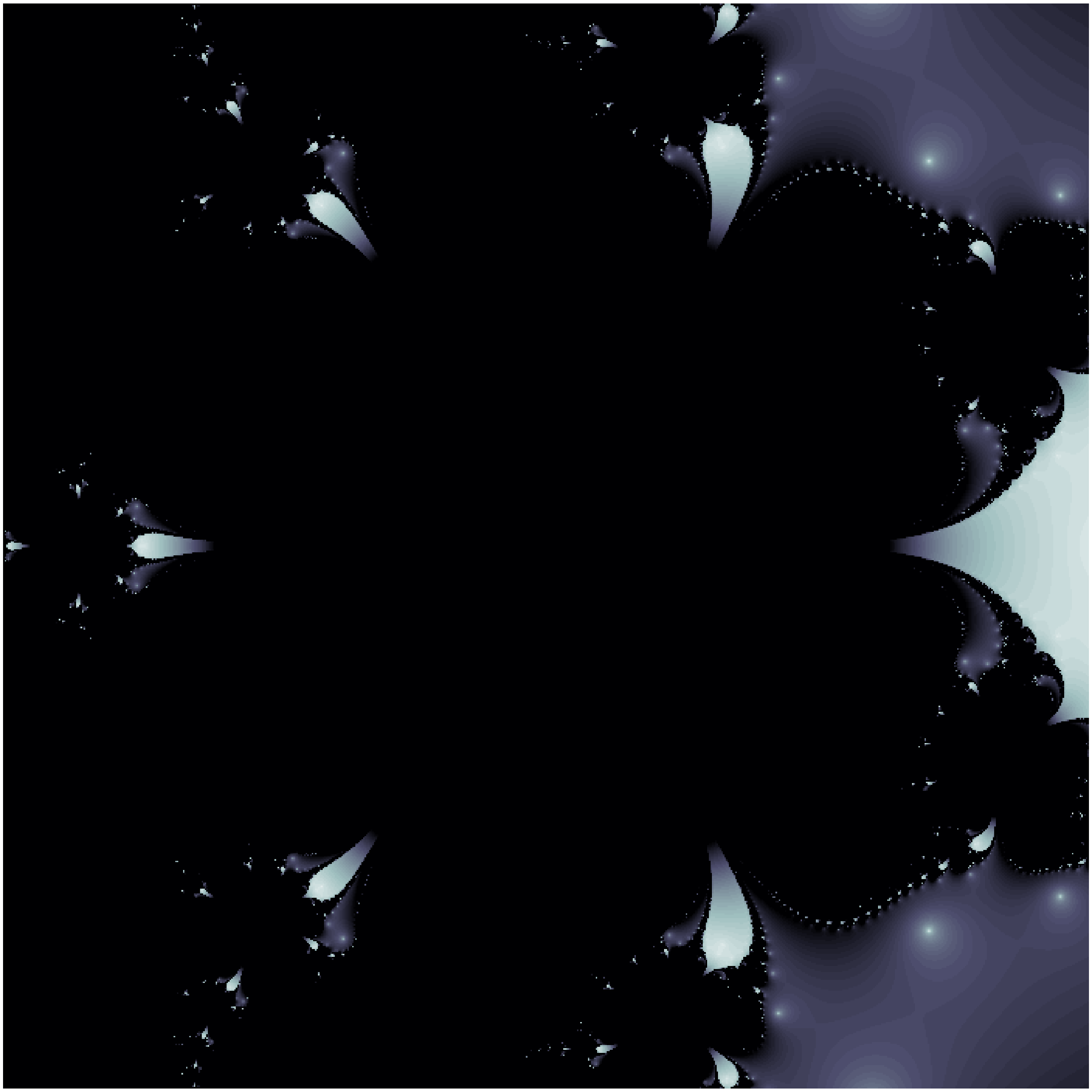


Figure 3 - Newton Fractal 15