Total 94/100

All code compiles and runs: +20

AM 213A HW3

Joseph Moore

Winter 2022

Part 1

a)

SVD Error 30/30

- The first ten singular values are as follows,

$$\begin{aligned}
\sigma_1 &= 663180.202318 \\
\sigma_2 &= 85706.595735 \\
\sigma_3 &= 62129.025680 \\
\sigma_4 &= 34664.633004 \\
\sigma_5 &= 31861.792296 \\
\sigma_6 &= 21872.721620 \\
\sigma_7 &= 19628.442780 \\
\sigma_8 &= 18434.937653 \\
\sigma_9 &= 13693.815446 \\
\sigma_{10} &= 12815.208252.
\end{aligned}$$

The k^{th} singular values are as follows,

$$\begin{aligned}
\sigma_{20} &= 7528.024652 \\
\sigma_{40} &= 5489.124664 \\
\sigma_{80} &= 3948.779979 \\
\sigma_{160} &= 2668.223578 \\
\sigma_{320} &= 1515.865932 \\
\sigma_{640} &= 821.893126 \\
\sigma_{1280} &= 513.568032 \\
\sigma_{2560} &= 179.115035
\end{aligned}$$

The very last singular value is $\sigma_{3355} = 16.724645$

- The matrix of singular values corresponds to the level of image compression depicted below.

$\Sigma_{\sigma_{20}}$ creates the image

$\Sigma_{\sigma_{40}}$ creates the image

$\Sigma_{\sigma_{80}}$ creates the image

$\Sigma_{\sigma_{160}}$ creates the image

$\Sigma_{\sigma_{320}}$ creates the image

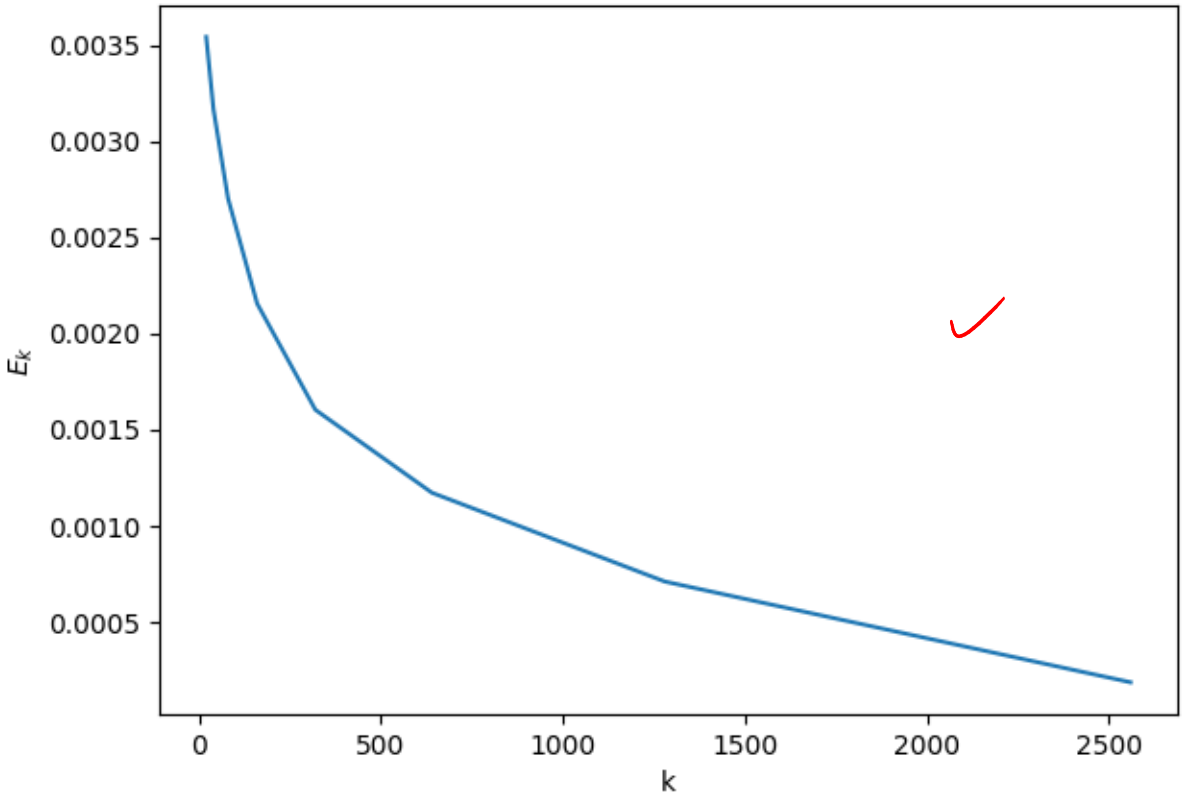
$\Sigma_{\sigma_{640}}$ creates the image ✓

$\Sigma_{\sigma_{1280}}$ creates the image

$\Sigma_{\sigma_{2560}}$ creates the image

$\Sigma_{\sigma_{3355}}$ is the all the singular values and thus is the original image

- $$E_{20} = \frac{||A - A_{\sigma_{20}}||_F}{mn} = 0.003543$$
$$E_{40} = \frac{||A - A_{\sigma_{40}}||_F}{mn} = 0.003166$$
$$E_{80} = \frac{||A - A_{\sigma_{80}}||_F}{mn} = 0.002703$$
$$E_{160} = \frac{||A - A_{\sigma_{160}}||_F}{mn} = 0.002156$$
$$E_{320} = \frac{||A - A_{\sigma_{320}}||_F}{mn} = 0.001604$$
$$E_{640} = \frac{||A - A_{\sigma_{640}}||_F}{mn} = 0.001173$$
$$E_{1280} = \frac{||A - A_{\sigma_{1280}}||_F}{mn} = 0.000711$$
$$E_{2560} = \frac{||A - A_{\sigma_{2560}}||_F}{mn} = 0.000187$$

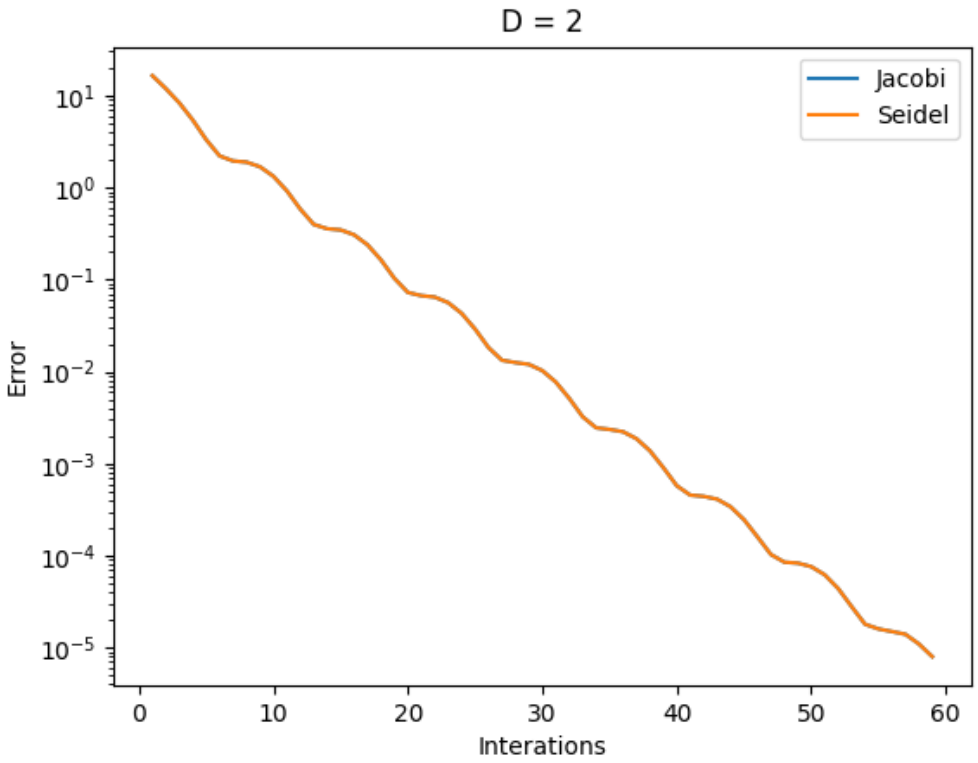


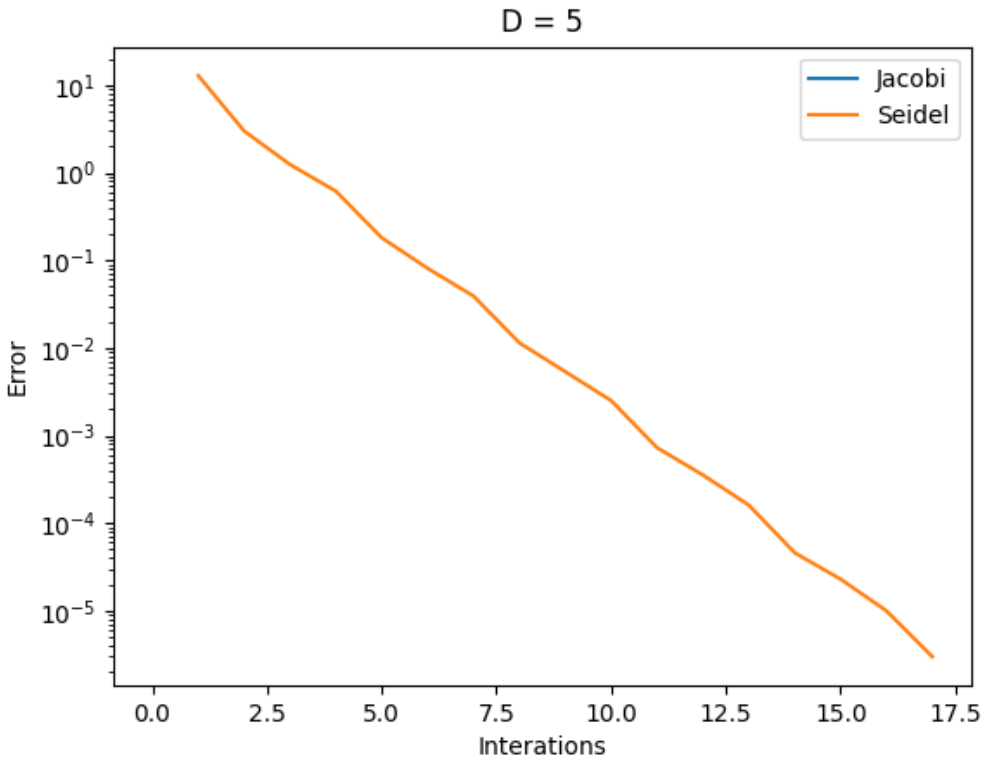
As the number of singular values increases the image becomes closer to the original. The error falls like $\frac{1}{k}$ and appears to asymptotically approach zero. This tells us that we can get most of the information we need from the first few singular values. The higher we up k the less information is added to our image. At $k = 1280$ the error is below 10^{-3}. However, from the graph we can tell that the error drops below 10^{-3} closer to $k = 1000$. With far less than half the total number of singular values we have a very small error.

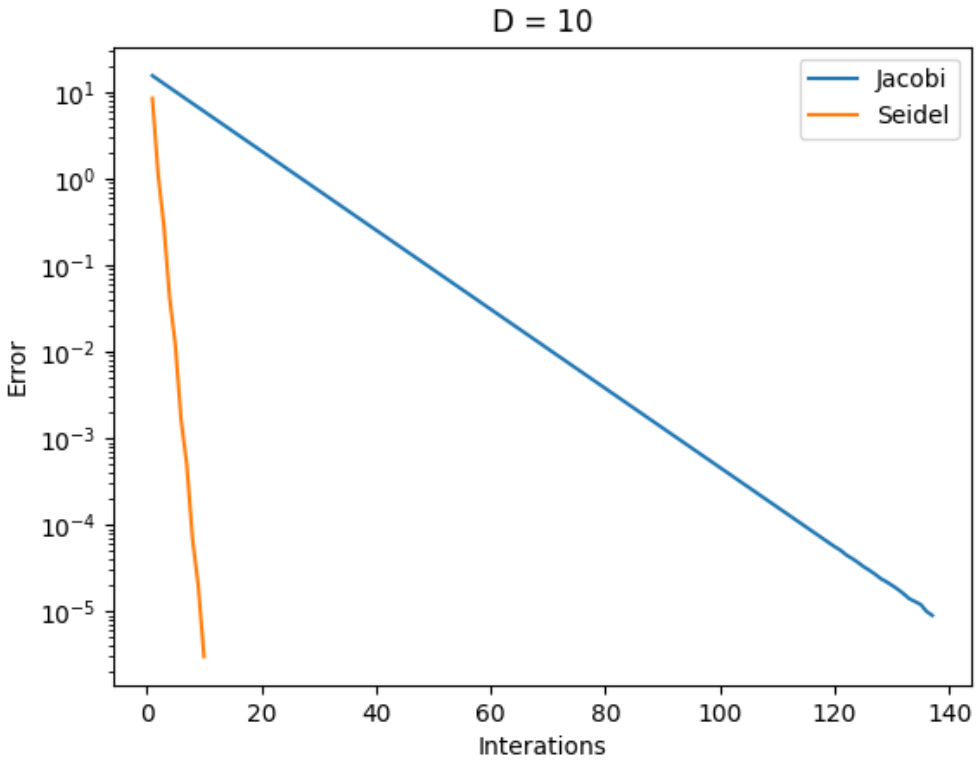
b.1)

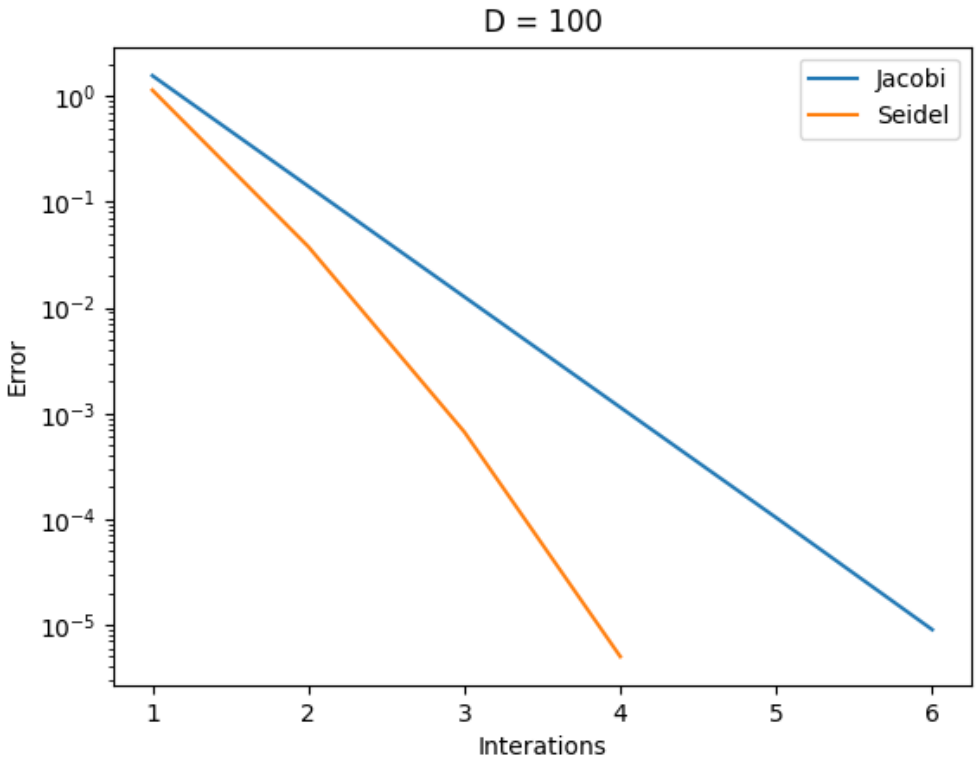
- Both the Gauss-Jacobi and the Gauss-Seidel use an iterative method to solve $Ax = b$ for x. The Gauss-Jacobi simply splits up A into its diagonal and remaining elements like $A = D + R$ and then follows $x^{k+1} = D^{-1}(b - Rx^k)$ until x converges to a result. Gauss-Jacobi converges when $\rho(D^{-1}R) < 1$. Gauss-Jacobi further splits R into upper and lower triangular matrices so that $A = L + D + U$. Gauss-Seidel follows $x^{k+1} = D^{-1}(b - Lx^{k+1} - Ux^k)$ and converges for any symmetric, positive definite matrix. Gauss-Seidel converges faster than Gauss-Jacobi when they both converge.

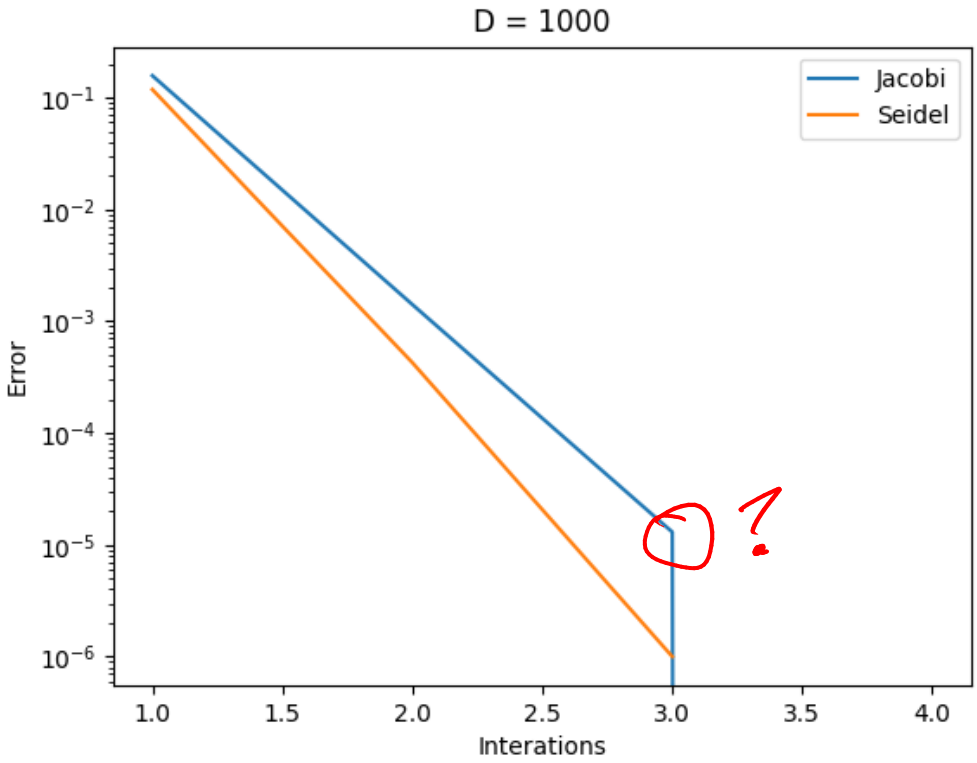
- The fist two plot for $D = 2$ and $D = 5$ have no Jacobi algorithm because it wouldn't converge.


D = 2
Jacobi
Seidel
Error
Interations
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
0
10
20
30
40
50
60


D = 5
Jacobi
Seidel
Error
Interations
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


D = 10
Jacobi
Seidel
Error
Interations
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
0
20
40
60
80
100
120
140


D = 100
Jacobi
Seidel
Error
Interations
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
1
2
3
4
5
6



- The Jacobi algorithm did not converge on the first two runs likely because our matrix failed the test $\rho(D^{-1}R) < 1$. The Seidel algorithm converged for all of the runs as all the matrices were symmetric and positive definite. The Seidel algorithm also converged much faster each time both algorithms converged. The difference, however, seemed to close as the D increased and Jacobi converged more easily.

- Setting $a_{ii} = i$ resulted in convergence for the Seidel algorithm but not for the Jacobi algorithm.

b.2)

- The Conjugate Gradient algorithm for $Ax = b$ can be thought of as the following steepest decent on a quadratic equation. Because a quadratic equation only has one minimum then we are guaranteed to approach that minimum as long as we follow a decent direction. Because we are choosing the steepest decent, which must be orthogonal to all of the previous iterations decent directions, we must find the minimum in m iterations.

- First we acknowledge that all $r_i^T r_j = 0$ for all $i \neq j$.

$$p_0 = r_0 \Rightarrow p_0^T r_0 = r_0^T r_0$$

$$p_1 = r_1 + \beta p_0 \Rightarrow p_1^T r_1 = (r_1^T + \beta p_0^T) r_1 = r_1^T r_1 + \beta p_0^T r_1 = r_1^T r_1 + \beta r_0^T r_1 = r_1^T r_1$$

$$p_2 = r_2 + \beta p_1 \Rightarrow p_2^T r_2 = (r_2^T + \beta p_1^T) r_2 = r_2^T r_2 + \beta p_1^T r_2 = r_2^T r_2 + \beta r_1^T r_2 = r_2^T r_2$$

Which continues like this showing that $p_k^T r_k = r_k^T r_k$. From this we see that α is the same.

$$E_{new} = r_k^T r_k = p_k^T r_k$$

We now show that β is the same for both.

$$r_{k+1} = r_k - \alpha A p_k \Rightarrow A p_k = r_k - r_{k+1}$$

$$\beta = -\frac{r_{k+1}^T y_k}{p_k^T y_k} = -\frac{r_{k+1}^T A p_k}{p_k^T A p_k} = -\frac{r_{k+1}^T (r_k - r_{k+1})}{p_k^T (r_k - r_{k+1})} = \frac{r_{k+1}^T r_{k+1}}{p_k^T r_k} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} = \frac{E_{new}}{E}$$

From all this it becomes trivial to show that the two algorithms are equivalent.

- It's clear that the Conjugate Gradient algorithm not only requires more iterations but also gets worse as the D increases. This, however, is not the same as computational time. It might very well be that CG has much faster iterations. Below is a table showing the number of iterations for each algorithm and D value.

	D = 2	D = 5	D = 10	D = 100	D = 1000
Jacobi	Does not converge	Does not converge	137	6	4
Seidel	59	17	10	4	3
Conjugate Gradient	1109	1707	3559	190429	19712896

- We can show that the eigenvalues of our matrix are $D - 1$ and $D + 9$ where D is the value of the diagonal elements. This means that the matrix is already relatively well conditioned.

- For the 10x10 matrix where $a_{ii} = i$ CG converges in 4474 iterations. The 100x100 diverges immediately. In fact, from testing I find that anything over 12 diverges immediately.

Unfortunately your implementation is wrong. For D=10 it actually converges in 2 iterations. Also way too far for $a_{ii}=i$.

-5