Jake Betenson
CS 3200 | u0624782

Assignment 6 – Singular Value Decomposition

I. Introduction

This experiment focused on the use of singular value decomposition (SVD) for solving ill-conditioned matrix systems and an image compression application. The experiment was performed using MATLAB R2020b on my personal machine using a 2nd generation Intel i5 processor (Ivy Bridge). Some code snippets, particularly image compression, are taken from lecture slides from the course.

II. SVD & Ill-Conditioned Matrix Systems

This experiment focuses on the used of SVD to solve poorly conditioned matrix systems. To get a baseline of results for comparison. Matlab's "\" function was used to solve the system $b = Hx$ where b is a column vector of $n \times 1$ where each entry is 1, H is a Hilbert matrix of $n \times n$ and $n \in \{5, 10, 15, 20, 25, 30\}$, and x is known to be an $n \times 1$ matrix with each entry being 1. Error was measured using the 2-norm. This experiment was repeated using SVD to solve for x. As well as another repetition with a filtering step on the SVD result with elements less than some tolerance, t, being set to 0. The results can be seen in Table 1 and 2. As n increases SVD tends to have a more favorable result by at least an order of magnitude. The filtering step can decrease the 2-norm measurement quite substantially depending on tolerance.

2-Norm Error		
Dimension (N x N)	**Matlab **	**SVD**
5	1.44E+03	1.44E+03
10	1.13E+07	1.13E+07
15	7.25E+08	6.17E+08
20	3.61E+09	2.31E+09
25	2.41E+10	2.31E+09
30	2.25E+09	4.16E+08

Table 1 – 2-norm error of the two main methods.

2-Norm Error of SVD with Filtering Step				
Dimension (N x N)	**t=1.0E-10**	**1.0E-12**	**1.0E-14**	**1.0E-16**
5	6.42E+02	6.42E+02	6.42E+02	6.42E+02
10	2.41E+04	2.41E+05	3.56E+06	3.56E+06
15	2.49E+04	1.58E+05	9.49E+06	9.36E+07
20	4.17E+04	2.22E+05	8.15E+06	5.58E+07
25	8.31E+04	4.02E+05	2.07E+06	6.71E+07
30	4.19E+04	8.19E+05	3.91E+06	7.60E+07

Table 2 – 2-norm error of SVD with filtering step.

III. SVD for Image Compression

This part of the experiment makes use of SVD to compress two provided images. One of a stuffed rabbit holding a sign and another of the tarantula nebula. The experiment takes the provided code snippets from the classroom and varies k to achieve suitable levels of compression. Comparisons are shown in Figures 1 and 2.

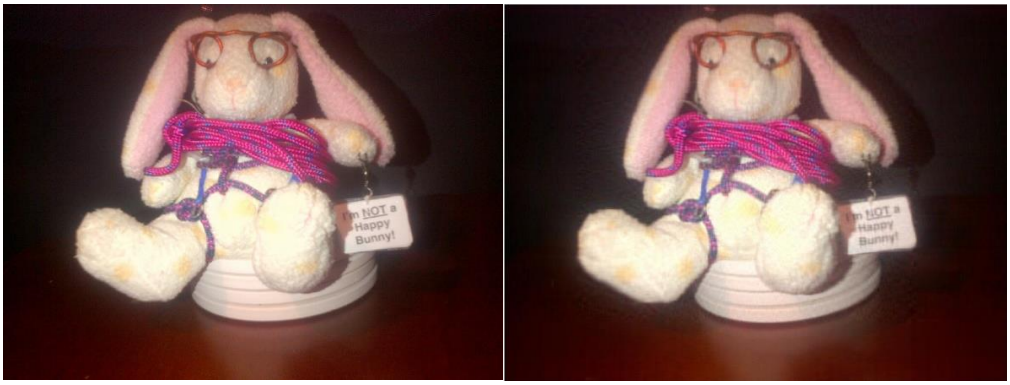

Fig. 1 – Original (left) vs Compressed (right) with readable signage.

SVD compression on the rabbit was able to go from 226 KB to 69 KB using $k = 60$. This is a compression ratio of $\frac{226}{69} = 3.28$. The SVD compression of the Tarantula nebula achieves file sizes of 136 KB to 82.2 KB using $k = 80$, with a ratio of $\frac{136}{82.2} = 1.65$. The standard for the compression of the tarantula nebula is a bit more nebulous (pun not intended), there are finer details that may be more important depending on application.

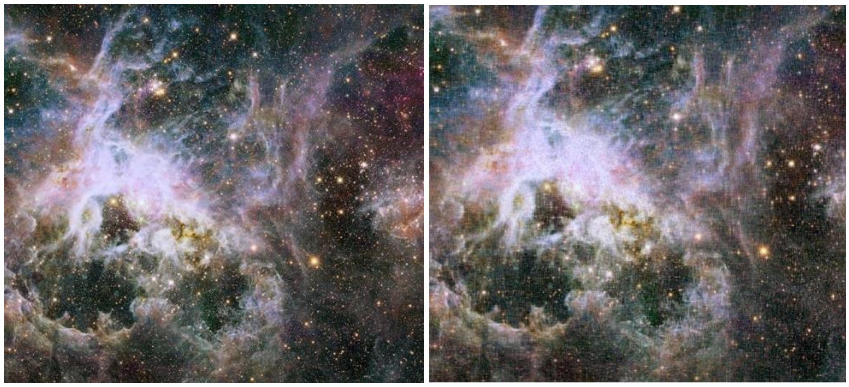

Fig. 2 – Original left. Nebula details preserved with compression.

IV. Conclusion

I have enjoyed each of the experiments so far, showing that computers are fallible and do make mistakes. It is our job to interpret results and show how far accuracy can be trusted. I always enjoy real world applications and the image compression was no exception.