Vinayak Ravichandran

Github: @vravich01

Email: vinayak1ravichandran@gmail.com

Image Processing & Lower Rank SVD

Section 1: Introduction

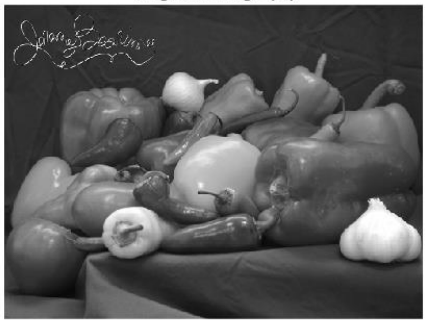
In this analysis, noisy and blurry images were reconstructed to improve quality as an attempt to recover the original image. In *Section 2*, gaussian noise was added to the original image and a lower rank truncation of the noisy image was computed to improve quality. In *Section 3*, a blurring operation was performed on a vector representation of an original image. Then, gaussian noise was added to the blurred image. Again, the lower rank truncation will be used to improve image quality; the accuracy, or lack thereof, of a naïve solution is also explored.

Section 2: Denoising Images

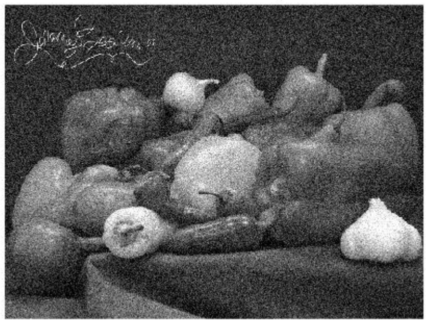
2.1) Methods

2.1.ab) The first 100 singular values ($\sigma_1, \sigma_2, \sigma_3, \dots, \sigma_n$) of the two provided images, shown below, A (original) and A_n (noisy) were plotted:

Original Image (A)



Noisy Image (A_n)

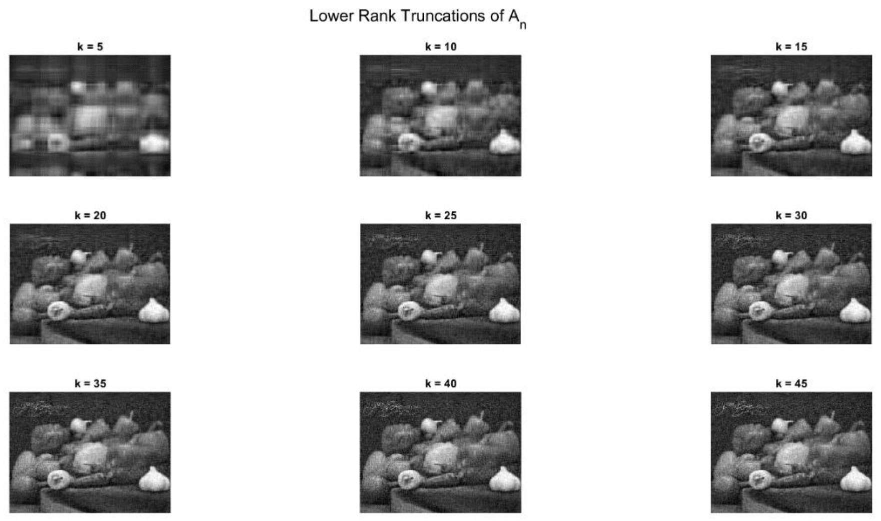


2.1.cd) Then, a rank-k truncation (A_k) of the SVD of A_n was computed for $k = 5, 10, 15, \dots, 45$. The nine corresponding images are shown below:

Vinayak Ravichandran

Github: @vravich01

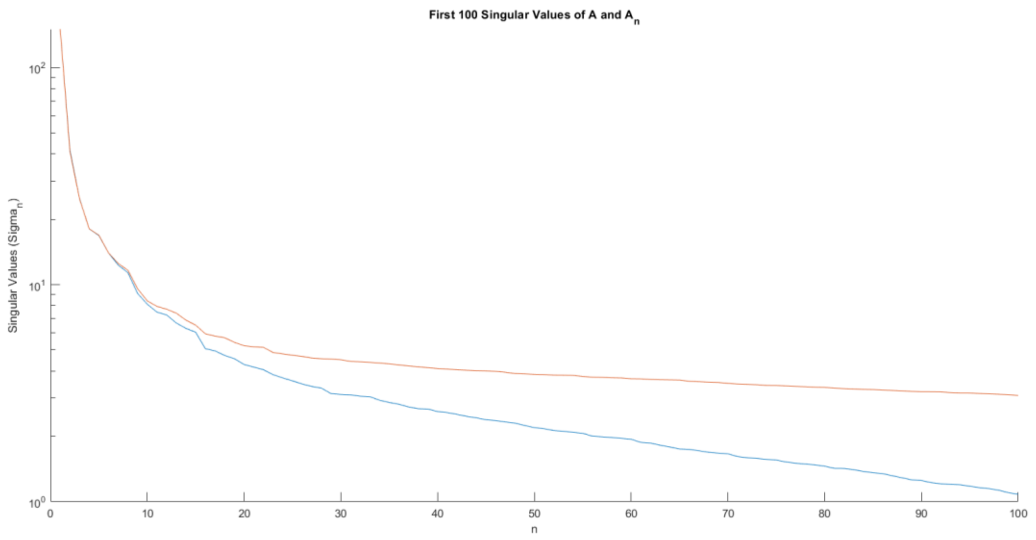
Email: vinayak1ravichandran@gmail.com



Then, the relative error of each truncation was plotted.

2.2) Results

2.2.ab) The plot of the first 100 singular values (A: blue, A_n: red):

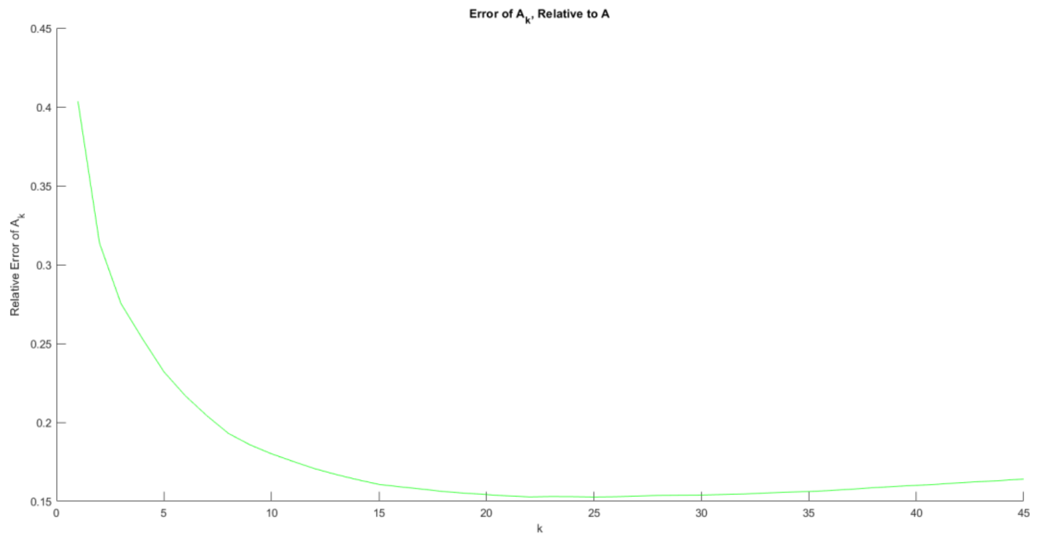


2.2.cd) The plot of the relative errors of A_k:

Vinayak Ravichandran

Github: @vravich01

Email: vinayak1ravichandran@gmail.com



2.3) Conclusion

From a simple eye test. A_{35} or A_{40} appear to be the most like an original image. However, the plot of relative errors reveal that the rank-20 or rank-25 truncation has a lower error. It is important to note that relative error was calculated using the Frobenius norm as follows:

$$Err_{Rel} = \frac{||A_k - A||_F}{||A||_F}$$

Therefore, it is possible to yield a lower relative error despite having a “worse” image since the Frobenius norm is not an element-wise operation, its value depends on the entire matrix.

At $k \approx 25$, the singular values of A and A_k begin to diverge so truncations greater than rank 25 will use more singular values that are increasingly different than from the singular values of the original. The approximate minimum relative error (0.1527) was observed for the rank-25 truncation, so it makes sense that large rank truncations are not very accurate. Conversely, the reason why a very low rank truncation is not very accurate could be due to the loss of image information when truncating to such a low rank. This could explain why the optimal lower rank truncation is not one which has a very low or high rank; it may be necessary for an approximate number of singular values to be used so that enough image information is retained, but using too many singular values retains too much of the gaussian noise as well.

Section 3: Deblurring Images

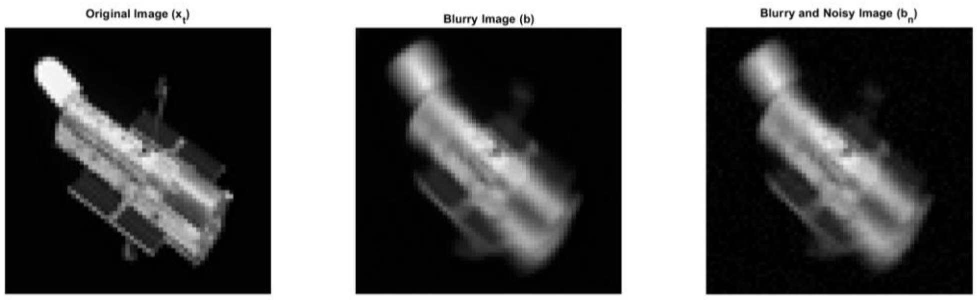
Vinayak Ravichandran

Github: @vravich01

Email: vinayak1ravichandran@gmail.com

3.1) Methods

3.1.ab) The provided images x_t (original image), b (burry image), b_n (blurry and noisy image) are shown below:



3.1.c) Note that b and b_n can be represented in terms of the blurring operator (A) and gaussian noise (η):

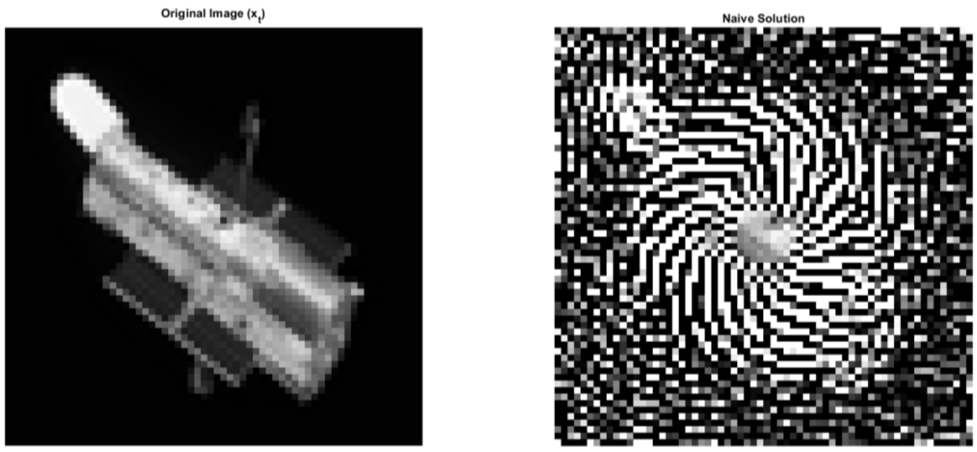
$$b = Ax$$

$$b_n = Ax + \eta$$

Note that b and b_n were computed by using x_t, so we expect to solve for x such that $x = x_t$. A naïve solution for processing b_n into its original would be to ignore the gaussian noise and simple apply the inverse of the blurring operator to solve for x as follows:

$$x_{naive} = A^{-1}b_n$$

The naïve solution, compared to the original, is shown below:



Vinayak Ravichandran

Github: @vravich01

Email: vinayak1ravichandran@gmail.com

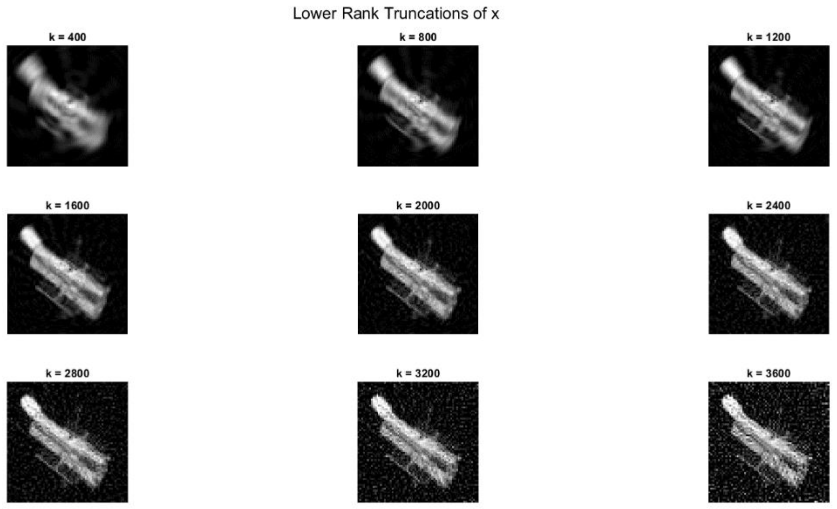
After computing the naïve solution, the singular values of the blurring matrix were plotted. Additionally, it is known that the relative error of b_n is 0.05.

3.1.ef) Then, the lower rank truncation of the blurring matrix was used to solve for x as follows:

$$A = U\Sigma V^T$$

$$x_k = V_k {\Sigma_k}^{-1} {U_k}^T b_n$$

For $k = 400, 800, 1200, \ldots, 3600$. The corresponding images are shown below:



The relative error of each instance of x_k was also plotted.

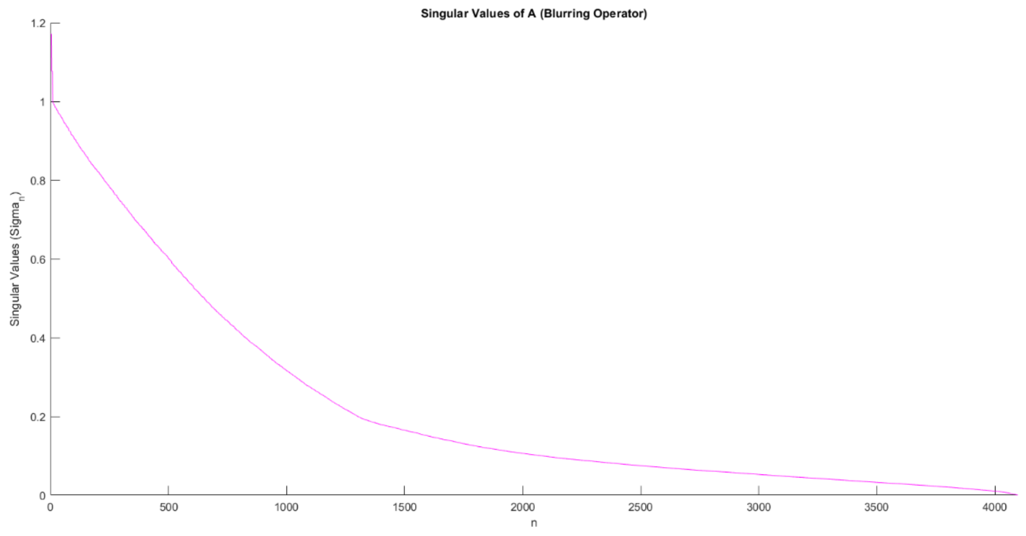
3.2) Results

3.2.a) The plot of the singular values of A, also note that $\kappa(A) = \frac{\sigma_1}{\sigma_n} \approx 5039$:

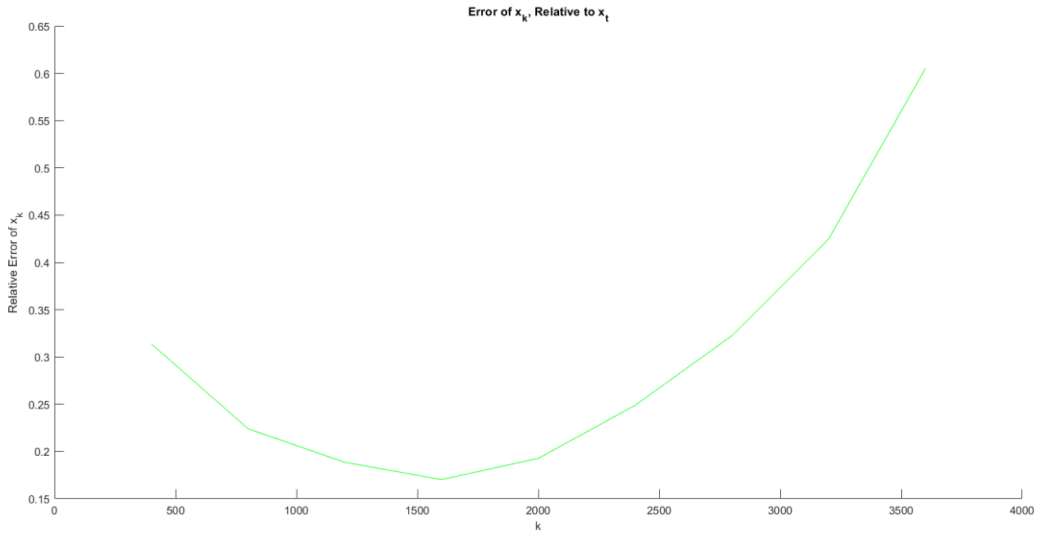
Vinayak Ravichandran

Github: @vravich01

Email: vinayak1ravichandran@gmail.com



3.2.c) The plot of the relative errors of x_k:



3.3) Conclusion

Since the relative error (in the 2-norm) of the blurry image and the blurred noisy image is 0.05, we know that there is a somewhat significant amount of noise added on. The naïve solution obviously results in some distortion of the original image, so it is rather inaccurate to directly ignore the effects of the gaussian blur. Computing a truncated inverse using the singular value decomposition proves to be more useful than the naïve solution.

Vinayak Ravichandran

Github: @vravich01

Email: vinayak1ravichandran@gmail.com

Also, note that the rank for which the relative error of x_k is minimized is $k = 1600$. This also happens to be the approximate number of greatly varying singular values of A; after $k = 1600$, the singular values tend to change in a nearly linear fashion with a low slope. It appears that after "capturing" the important image data stored in the first 1600 singular values, the remaining singular values only increase error.

Again, there seems to be a "sweet spot" at which the truncated image is optimized. It makes sense why a very low rank truncation would not work since a lot of the blurring cannot be undone with out those singular values. It also makes sense why a significant majority of the singular values are not used since then the program would attempt to unblur pixels that are just noise.