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**

* 1. **Methods**

**2.1.ab)** The first 100 singular values () of the two provided images, shown below, (original) and (noisy) were plotted:

A picture containing timeline

Description automatically generated

**2.1.cd)** Then, a rank-k truncation () of the SVD of was computed for . The nine corresponding images are shown below:

Timeline

Description automatically generated

Then, the relative error of each truncation was plotted.

* 1. **Results**

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

Chart, line chart

Description automatically generated

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

Chart

Description automatically generated

* 1. **Conclusion**

From a simple eye test. or 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:

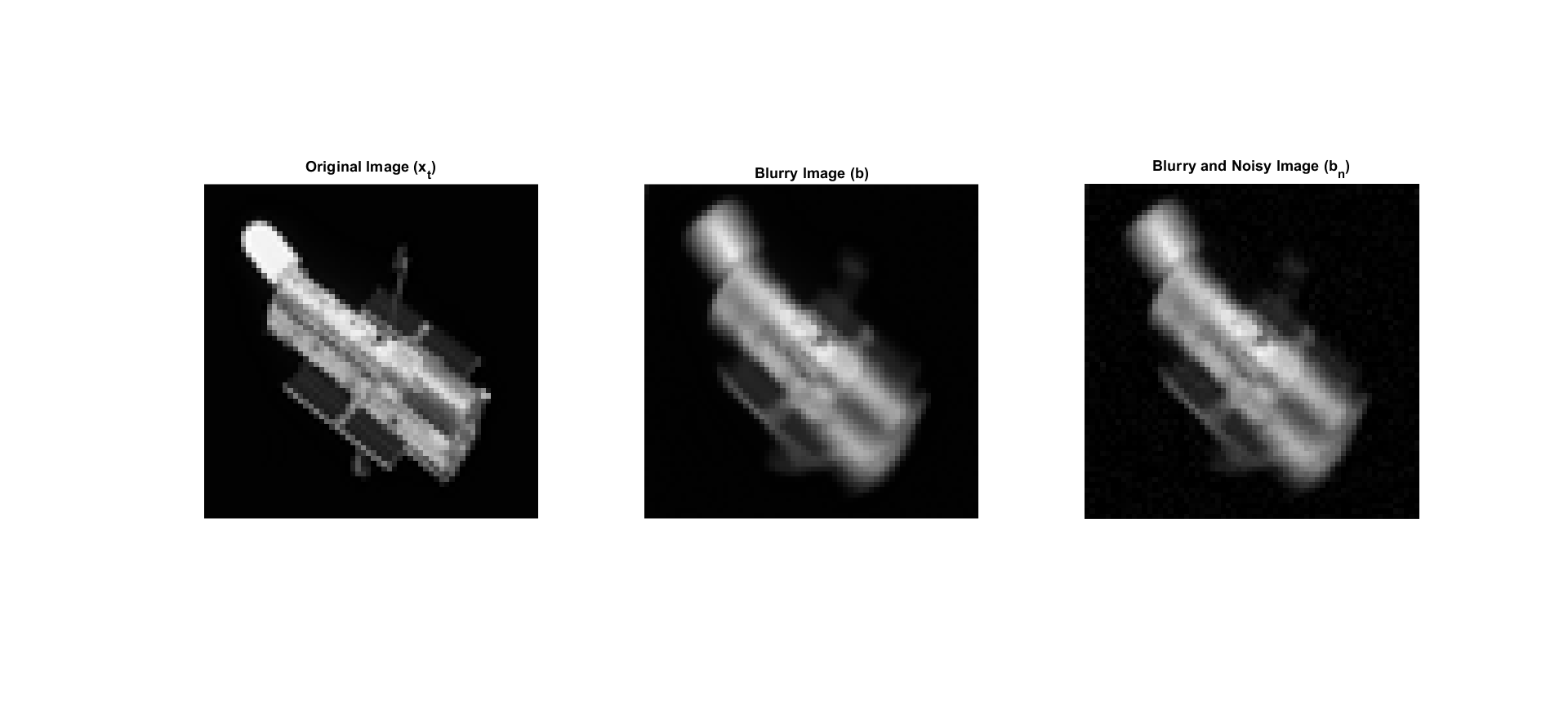
Therefore, it is possible to yield a lower relative error despite having a “worse” image since the Frobenius norm is not an element-wise operation, its value depends on the entire matrix.

At , the singular values of and 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**

* 1. **Methods**

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



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

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

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

A close up of a logo

Description automatically generated

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

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

For . The corresponding images are shown below: A picture containing timeline

Description automatically generated

The relative error of each instance of was also plotted.

* 1. **Results**

**3.2.a)** The plot of the singular values of , also note that :

Chart

Description automatically generated

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

Chart

Description automatically generated

* 1. **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.

Also, note that the rank for which the relative error of is minimized is . This also happens to be the approximate number of greatly varying singular values of ; after , 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.