Parallel Image Restoration

EECE 5698 Final Project Report

Cooper Loughlin & Daniel Silver

April 18, 2019

1 Introduction

Synthetic degradation of images is commonly used to assess different restoration algorithms. In our project we use synthetic degradation to assess a specific restoration function. This restoration function is dependent on a hyperparameter λ, that will be selected based on testing different values for λ and selecting the one that restores the image the best. However, one problem with this is that it is very computationally expensive to run the cross-validation that is necessary to identify this value. Parallelism helps solve this problem by splitting up the work among multiple workers so that computations can occur more quickly and therefore more values of λ can be tested. Our project seeks to analyze the speed of finding such a λ in different levels of parallelism in addition to the quality of the restored images based on the power of the input noise.

2 Image Restoration

Image restoration is the task of obtaining high quality images from degraded observations. Digital images are generally subject to distortion such that the resulting image has unwanted artifacts that are not present in the original scene. Two common artifacts are additive noise and blur. Noise refers to random variations in the image that may result form, for example, the electronics of the camera. Blur, on the other hand, is a spatial distortion that may be due to several factors including, focus, motion, or atmospheric effects. Image restoration is of great importance in many fields such as astronomy, medicine, security, and remote sensing. As a result, many restoration techniques have been developed for various scenarios.

2.1 Degradation Model

A primary consideration in image restoration is how best to model the degradation process. In this work, we consider a simple degradation model whereby a degraded image observation is comprised of a clean image which has been blurred and had noise added. The model is summarized by Figure 1.

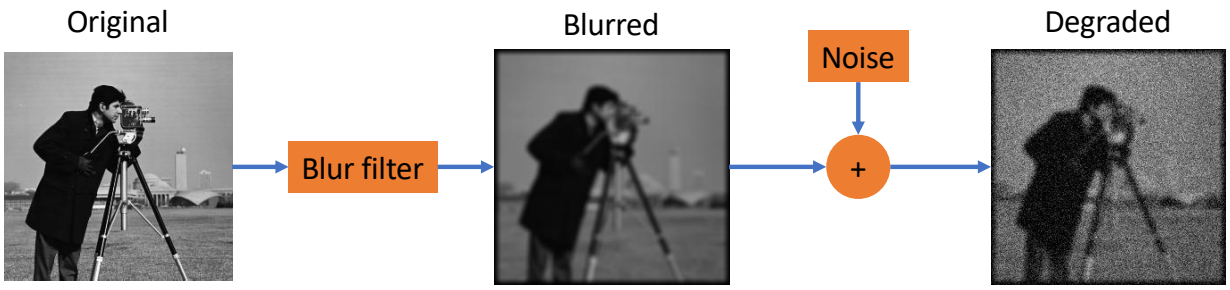

Figure 1: The image degradation model considered.

The image blur can be modeled as a linear convolution between the clean image and a blur filter, defined by an impulse response. The degraded observation can thus be written

$$y[m,n] = x[m,n] * h[m,n] + n[m,n] \tag{1}$$

where $y[m,n]$ is the degraded image, $x[m,n]$ the clean original, and $n[m,n]$ is noise. The quantity $x[m,n]$ denotes the pixel value at coordinate $[m,n]$ of an image x with M rows and N columns. Each element of noise $n[i,j] \sim \mathcal{N}(0,\sigma^2)$, where σ^2 is the noise power and $\mathcal{N}$ denotes the univariate Gaussian distribution. The degradation model can alternatively be expressed in the frequency domain as

$$Y(\omega_1,\omega_2) = X(\omega_1,\omega_2)H(\omega_1,\omega_2) + N(\omega_1,\omega_2) \tag{2}$$

where ω_1, ω_2 denote spatial frequency components. The quantity $X(\omega_1,\omega_2)$ is obtained as the Discrete Fourier Transform (DFT) of the image x. The blur filter h in this model is considered to be a fixed deterministic quantity, while the noise n is random.

2.2 Restoration Algorithm

Given the model of (1), the task of image restoration is to retrieve the image x given y. In general, perfect reconstruction of y is impossible due to the presence of noise. Thus, at best, it is only possible to recover an estimate $\hat{x}$ of the original image. Designing an algorithm to estimate $\hat{x}$ depends on what assumptions are made about the parameters of the model and what data is available. In this work, it is assumed that the blur filter h is fixed but unknown and that N training images pairs (y_i, x_i) are available. The image restoration technique presented in this work is twofold. First, the blur filter h is estimated using the training images. Then, the regularized inverse filter of h is used in image restoration. Regularization is necessary to reduce the effects of noise on the restored image.

Using the quantities (y_i, x_i), the blur filter impulse response h may estimated in the following way. In the frequency domain, an estimate of the filter frequency response using

the training image pair (y_i, x_i) may be written

$$\hat{H}_i(\omega_1, \omega_2) = \frac{Y_i(\omega_1, \omega_2)}{X_i(\omega_1, \omega_2)} \tag{3}$$

$$= \frac{X_i(\omega_1, \omega_2)H(\omega_1, \omega_2) + N_i(\omega_1, \omega_2)}{X_i(\omega_1, \omega_2)} \tag{4}$$

$$= H(\omega_1, \omega_2) + \frac{N_i(\omega_1, \omega_2)}{X_i(\omega_1, \omega_2)}. \tag{5}$$

Taking an average over the all the training samples results in the estimate

$$\hat{H} = \frac{1}{N}\sum_{i=1}^{N} \hat{H}_i \tag{6}$$

where the arguments (ω_1, ω_2) have been dropped for brevity. Since the noise is zero mean, the expected value

$$\mathrm{E}[\hat{H}] = \frac{1}{N}\sum_{i=1}^{N} \mathrm{E}[\hat{H}_i] \tag{7}$$

$$= \frac{1}{N}\sum_{i=1}^{N} \mathrm{E}\Big[H + \frac{N_i}{X_i}\Big] \tag{8}$$

$$= H + \frac{1}{N}\sum_{i=1}^{N} \frac{N_i}{X_i} \tag{9}$$

$$= H, \tag{10}$$

thus the estimate is unbiased.

Given the unbiased estimate of the blur filter, one way to restore an image is to simply apply the inverse filter to a degraded image in the frequency domain as

$$\hat{X} = \frac{Y}{H} \tag{11}$$

$$= X + \frac{N}{H} \tag{12}$$

and obtain the restored as the inverse DFT $\hat{x}$ of $\hat{X}$. Due to the presence of noise, however, this will result in a very poor estimate of x. Specifically, estimates obtained in this way have extremely high variance. Figure 2 shows an image restored in this way. Clearly, the restoration is useless due to the extreme levels of noise.

An intuitive reason for the high degree of noise in the restoration follows from the frequency response of H and the frequency content of the Gaussian noise N. The DFT of blur type filters tend to be large at low frequencies and very small (almost 0) at high frequencies. As a result, the inverse $1/H$ will have very high DFT values at high frequencies. On the other hand, the Gaussian noise N will have roughly the same frequency content at all frequencies on average. The term N/H in (11) will thus be extremely large, dominating X.

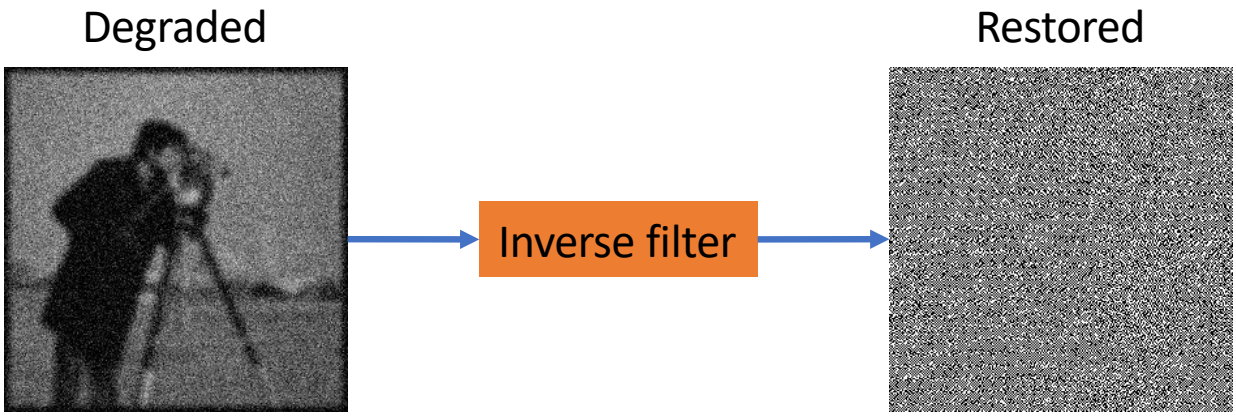

Figure 2: A noisy blurred image restored by an inverse filter with no regularization.

Alternatively, regularization can be applied to minimize the effect of noise on the restored image. The regularized image restoration may be written

$$\hat{X} = \frac{\hat{H}^*}{\hat{H}\hat{H}^* + \lambda} Y \tag{13}$$

$$= \frac{\hat{H}\hat{H}^*}{\hat{H}\hat{H}^* + \lambda} X + \frac{\hat{H}^*}{\hat{H}\hat{H}^* + \lambda} N \tag{14}$$

where the superscript $*$ denotes complex conjugate and $\lambda > 0$ is the regularization parameter. For $\lambda = 0$, the regularized inverse filter is equivalent to the non-regularized inverse filter. As λ grows, the estimate approaches 0. For λ in between, the estimate $\hat{X}$ contains a slightly biased term related to X and a noise term that has been reduced in magnitude. An image restored using regularization is shown in Figure 3. The full restoration model is shown in Figure 4.

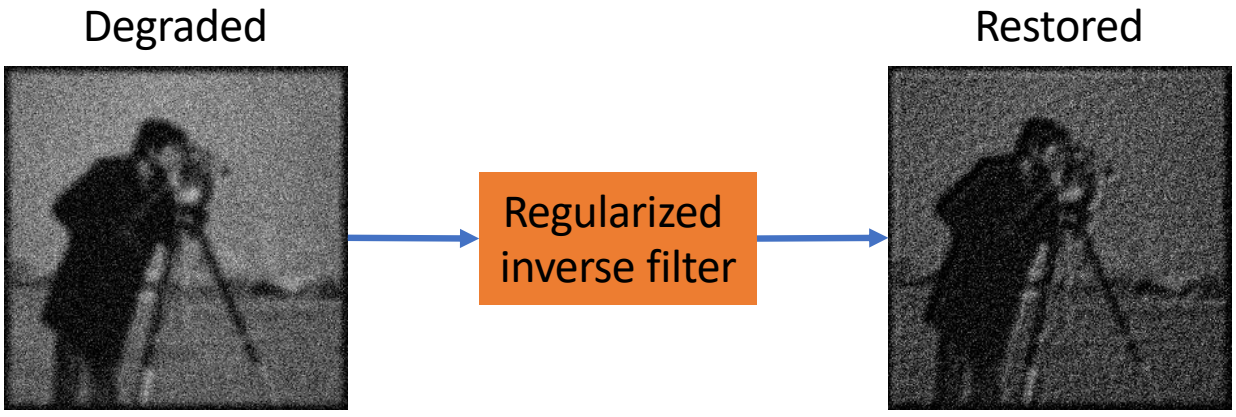

Figure 3: A noisy blurred image restored by a regularized inverse filter.

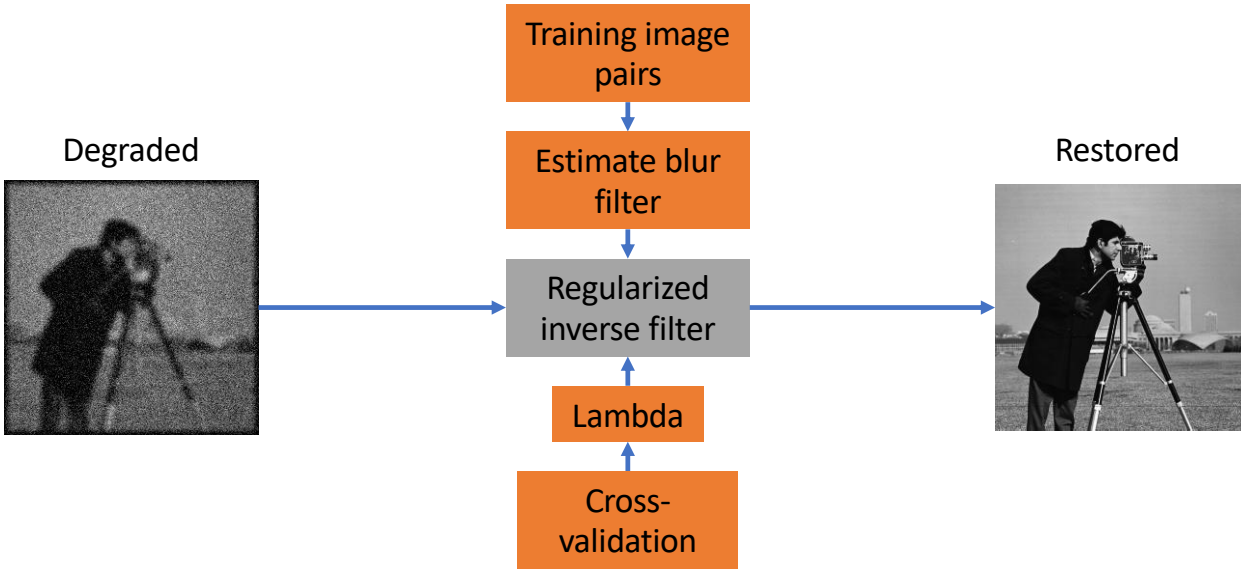

Figure 4: Block diagram of the image restoration model.

2.2.1 Choosing the Regularization Parameter

Regularization introduces a trade-off between bias and noise (variance) in the restored image. For small λ, there will be high variance, while large λ will result in high bias. Choosing an optimal λ may be done using cross-validation. Cross-validation is a means of estimating how well a certain model generalizes to unseen data. In this case, we are interested in measuring how well our restoration model generalizes to unseen images.

In this work, k-fold cross-validation is used to choose an optimal value of λ. Using this technique, the training samples are divided into k equal groups. A restoration model is then computed using all but one the k-th group. The residual sum of squares (RSS) is then computed on the group that was omitted. The RSS for the k-th group is defined as

$$\mathrm{RSS}_k(\lambda) = \sum_{i,k=1}^{N_k} ||\hat{x}_i - x_i||^2 \tag{15}$$

where N_k is the number of samples in each fold. The process is repeated for each of the remaining groups and the total RSS is computed as

$$\overline{\mathrm{RSS}} = \frac{1}{k} \sum \mathrm{RSS}_k. \tag{16}$$

Cross-validation is performed for various values of λ and the one resulting in the minimum RSS is chosen for the restoration algorithm.

3 Implementation

There were many small implementation decisions we had to make in order to get reliable, comprehensible, and easily computable results. We chose to use Apache Spark to implement

all of our algorithms because we wanted to access the level of parallelism on the time it takes for our restoration algorithm to run at small incremental changes in the level of parallelism. We chose Apache Spark and Python as this makes this process easy. We chose to store all RDDs in the as a tuple in the form of (index of image, numpy 28x28 image). This was so we could re-identify a particular image after shuffling occurs within an RDD. We maintained one RDD of original images, one of degraded images, and one of restored images.

One design decision we made was that we chose the MNIST dataset to use as the image set to degrade and restore. The MNIST data set is 60,000 black and white hand drawn images of size 28x28 pixels. This data set contains many small images in a single color dimension, which makes the restoration algorithm very easily computable, especially in parallel as we have many images with relatively few operations.We also chose to use a subsection of MNIST of the first 1,000 images. This is because the RDDs would have been too large otherwise and Apache Spark throws an error at very large datasets. The first that we chose to use the first 1,000 images means nothing as the ordering of MNIST images is random, so we should get roughly an even distribution of all digits, not that this should affect how our restoration algorithm works at all.

The image processing was done in the frequency domain, through the use of the fast Fourier transform (FFT). The compute the degraded images, the FFTs of both the clean images and blur filter were taken and multiplied together. The inverse FFT on the blurred images were then taken and noise was added. The blur filter was estimated by dividing the FFTs of the degraded and clean images. The restored images were similarly computed using FFTs of the degraded images and regularized inverse filter.

4 Validation

The restoration model can be validated by showing that the restored image is more similar to the original than the degraded. To validate the proposed restoration model, we synthetically generate a training set of images based on the degradation model. As a result, we are able to train the restoration model using a set of known pairs (x_i, y_i) of clean x_i and degraded y_i images. We use the common MNIST dataset of handwritten digits.

The synthetic degraded images y_i are obtained by blurring then adding noise to the clean images. The images are blurred using a Gaussian filter, whose impulse response is of the form

$$h[m, n] = A \exp\Big(-\frac{1}{2\sigma}(m^2 + n^2)\Big) \tag{17}$$

where A is a normalizing constant and σ determines the width of the body. For these experiments, $\sigma = 3$ was chosen with the size of the impulse response support limited to 15×15. Noise was additionally added by generating random samples from a Gaussian distribution with various amounts of power. The synthetic degraded images along with their corresponding clean images are used to estimate the blur filter and perform cross-validation in the restoration model.

The restoration model can be evaluated using metrics as well as visual inspection. Figure 5 shows the original, degraded, and restored images for various characters at various noise powers. The restored images are visually more similar to the originals than the degraded.

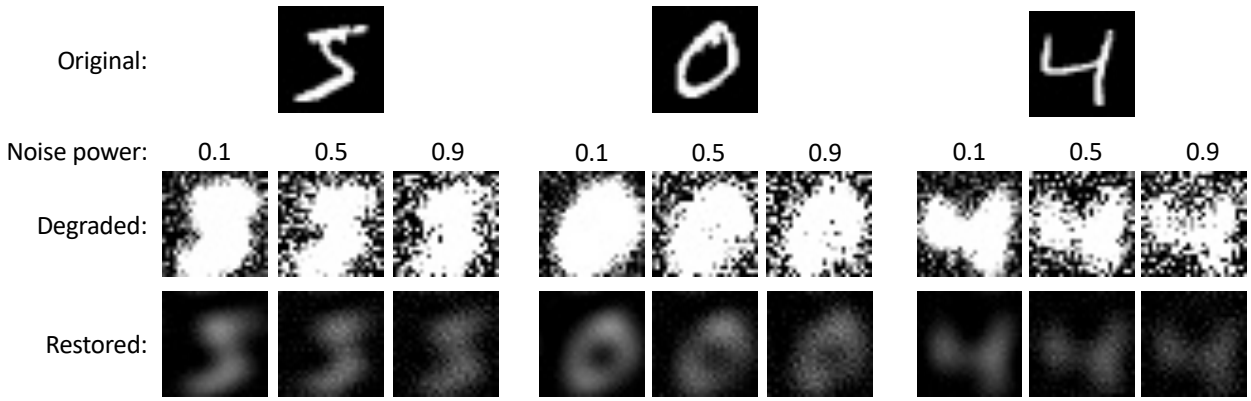

Figure 5: Original, degraded, and restored images for various noise powers.

Noise power	Distance clean to degraded	Distance clean to restored	ISNR (dB)
0.1	3202381.6	2704.6	61.4
0.2	3494171.5	2758.9	62.0
0.3	3808275.6	2799.1	62.6
0.4	4116680.6	2858.6	63.1
0.5	4423257.9	2909.4	63.6
0.6	4768672.4	2960.7	64.1
0.7	5119149.7	2991.1	64.6
0.8	5466767.3	3060.1	65.0
0.9	5873090.3	3097.0	65.5

Figure 6: Tabulation of the average distance and ISNR metrics for various noise powers.

Figure 6 details a set of metrics that validate the restoration model. The average distance from the original to the degraded is defined as

$$d_{\text{deg}} = \frac{1}{N}\sum_{i=1}^{N} ||y_i - x_i|| \tag{18}$$

and the average distance from the original to the restored as

$$d_{\text{res}} = \frac{1}{N}\sum_{i=1}^{N} ||y_i - \hat{x}_i||. \tag{19}$$

The increased signal-to-noise ratio ISNR is defined as

$$\text{ISNR} = 10\log_{10}\frac{d_{\text{deg}}^2}{d_{\text{res}}^2}. \tag{20}$$

For the image restoration to be valid, $d_{\text{res}} < d_{\text{deg}}$ and correspondingly ISNR > 0. For each of the noise levels tested, the restoration model produces a valid result.

We can additionally validate that cross-validation results in a good regularization parameter. It is expected that a low λ will result in a model with high variance while a large λ will result in high bias. Both scenarios result in a high RSS when testing on new data. This quantity is estimated using cross-validation. A good λ is one that minimizes the estimated RSS. Additionally, it is expected that for higher noise power, λ should be larger. The intuition is that the high frequency response of the regularized inverse filter needs to be reduced more as the noise level increases. Figure 7 shows that cross-validation results in a clear minimum and that the optimal λ increases with noise power.

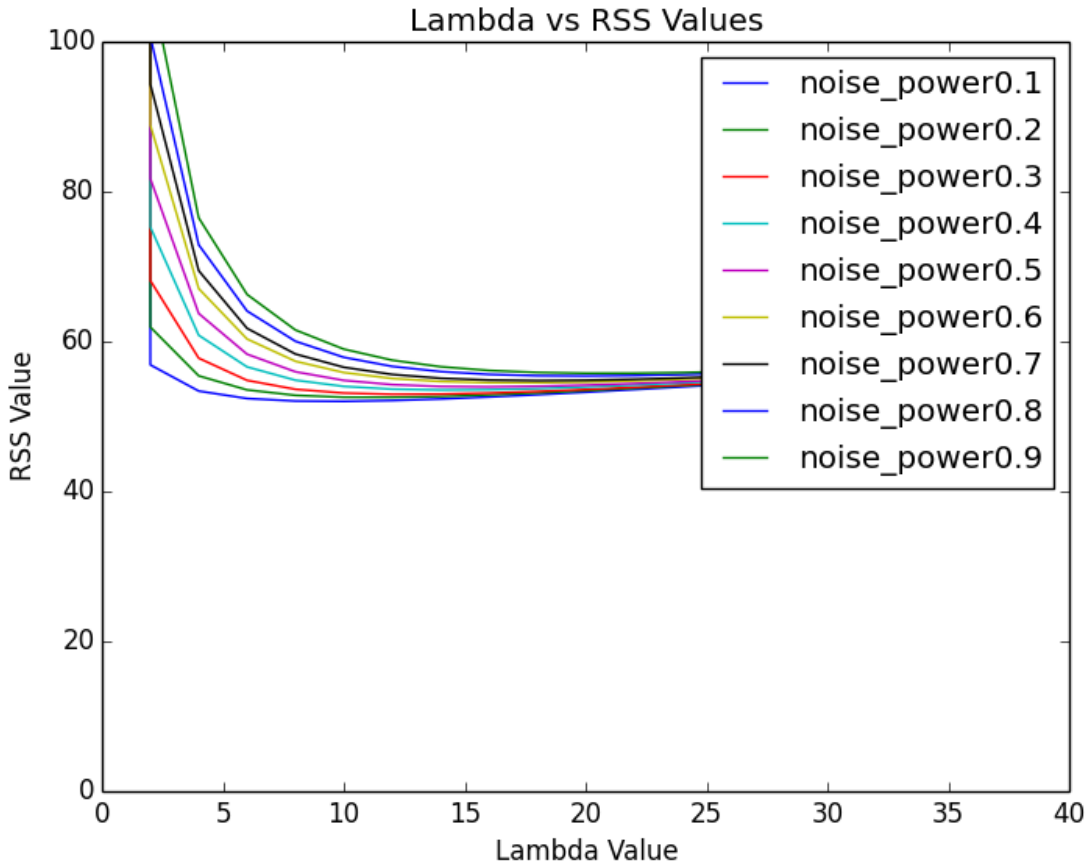

Figure 7: Average RSS values as a function of Lambda at different noise power levels.

5 Performance Analysis

We assessed our performance of our algorithm over the following levels of parallelism (1,2,3,4,5), where the parallelism of 1 is handled serially as it is only run on one compute node. The plot of computer nodes vs time for different noise levels is shown in Figure 8. As shown

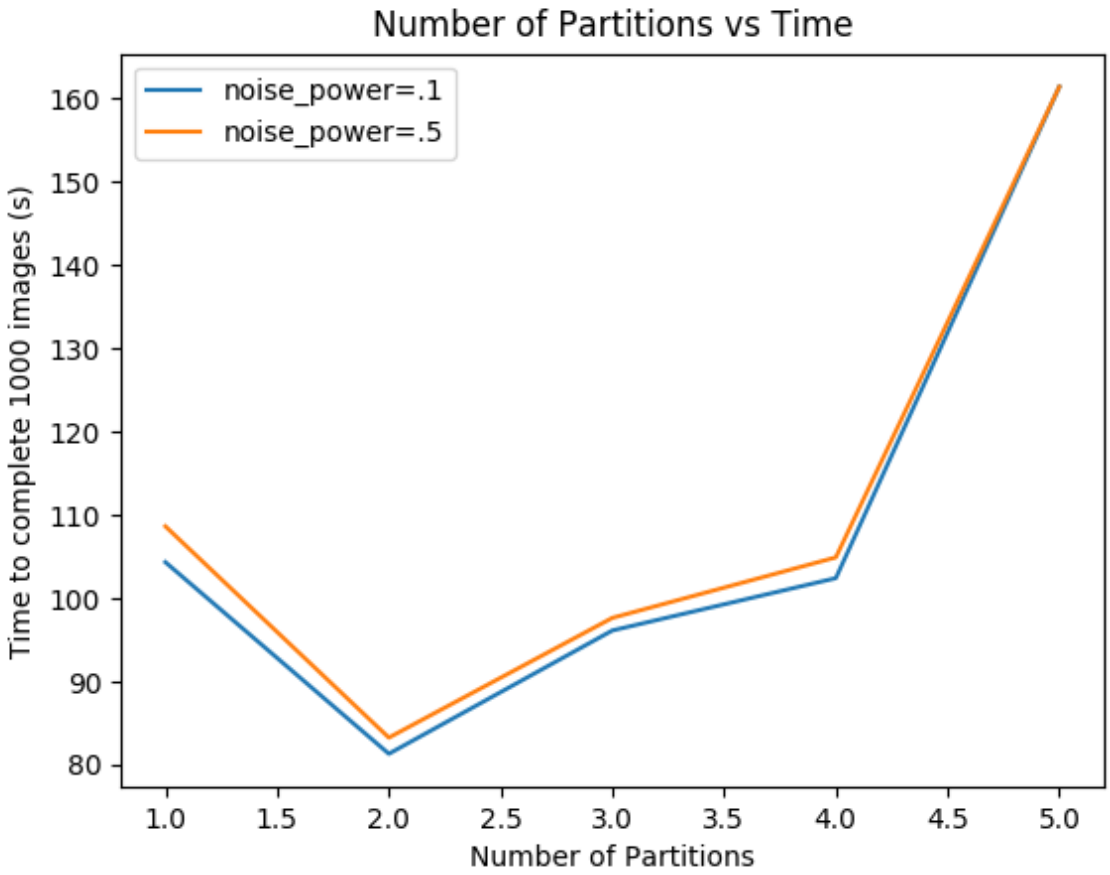

Figure 8: Speed of computation for different levels of parallelism.

here, the increase in speed due to parallelism is apparent in both cases of noise powers from transitioning from having the RDDs partitioned on 1 node, to having the RDDs partitioned to 2 nodes. The amount of processing does not depend on the level of noise in the degraded images; thus, the computation time is roughly equal for the various levels of noise, as expected. This shows that there is a noticeable speedup of over 20% just by adding 1 more node. However, adding more nodes than this actually leads to a decrease in speed. We attribute this to the added communication cost sending such large datasets between nodes for processing. The point at which the increase in communication cost outweighs the increase in computation cost, when trying to optimize for runtime, is at 3 partitions.

6 Conclusion

We successfully assessed the speed of our restoration algorithm as a function of the level of parallelism. This was made possible by selecting the number of compute nodes using Apache Spark. Parallelism helped us attain a faster speed to find an optimal λ to be used to restore the images with the best quality. We also analyzed the performance of the restoration

algorithm by comparing the restored images to the original, non-degraded images at different levels of degradation, controlled by the noise power level of the degradation.

Appendix A Author Contributions

A.1 Cooper

Degradation, Restoration, Creating Figures, Data Validation, Regularization

A.2 Daniel

Image collection/Formatting, Preprocessing, Image Output, CSV Data Output, Creating Graphs

Appendix B Code

All code used in this project can be found at https://github.com/SilverEngineered/Image-Restoration