Pattern Recognition

Assignment #1

SVD-EVD and Regression

Professor:
Hema A. Murthy

Nirav Bhavsar (CS17S016)
Sidharth Aggarwal (CS17S012)

Contents

Chapter 1

Singular Value Decomposition

We will perform Singular Value Decomposition on both square and rectangular images.
(a) by converting the image to grayscale.
(b) separately on each color bands.
(c) after concatenating the 8bit R,G,B channel to form a 24bit number.

1.1 SVD on Square Image by converting to Gray scale

The singular value decomposition of any $m \times n$ real or complex matrix $\mathbf{M}$ is a factorization of the form $\mathbf{U\Sigma V^*}$ where $\mathbf{U}$ and $\mathbf{V}$ are real or complex unitary matrix, $\mathbf{\Sigma}$ is a diagonal matrix with non-negative real numbers on the diagonal called Singular Values.

To compute SVD of a Matrix, we first find Eigen values $\mathbf{\Lambda}$ and Eigen vectors $\mathbf{X}$ of $\mathbf{A' * A}$, then eigen vector $\mathbf{X}$ would be $\mathbf{V}$ and $\mathbf{\Lambda}$ will be equal to $\mathbf{\Sigma^2}$. Then, to compute $\mathbf{U}$ we will calculate $\mathbf{A * V * \Sigma^{-1}}$

Now, we will convert our original image to gray scale image and perform SVD and reconstruct original image using $\mathbf{U\Sigma V^*}$.

Figure 1.1: Original Square image

Figure 1.2: Original Gray Scale & Reconstructed Square image

Now, we will select Top N singular values of $\mathbf{\Sigma}$ and reconstruct image along with it's corresponding error image with respect to original image.

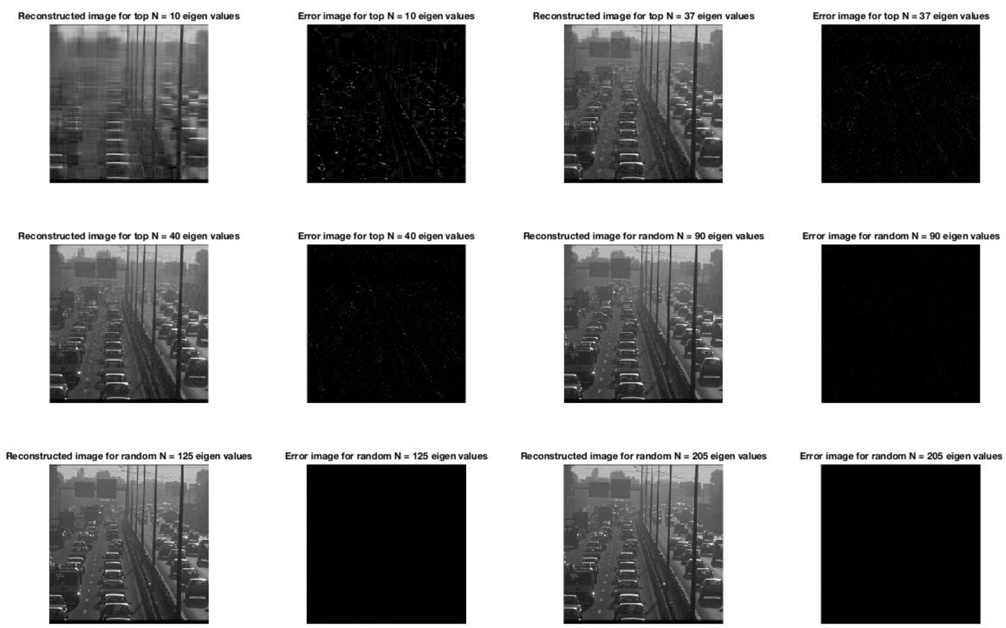

Figure 1.3: SVD on Gray Scale Square image

Below graph shows how Frobenius Norm of error image decreases as we select more top N singular values.

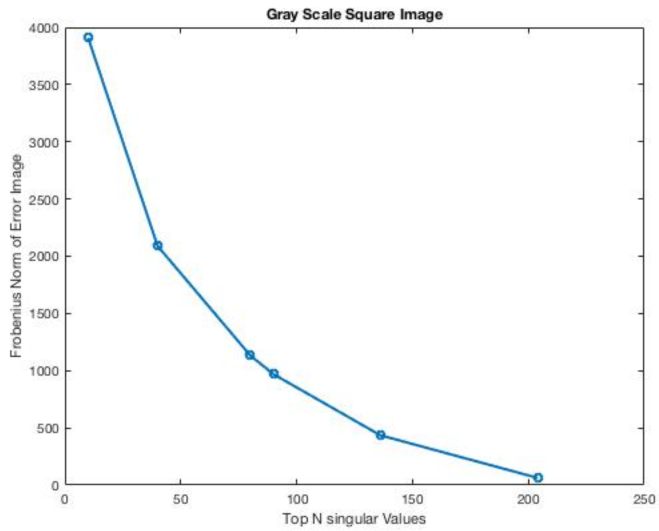

Figure 1.4: Graph of Frobenius Norm Vs Top N singular values of Gray scale square image

1.2 SVD on Square Image by seperating it's Red color band

Similarly, we extract red color band from original image and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

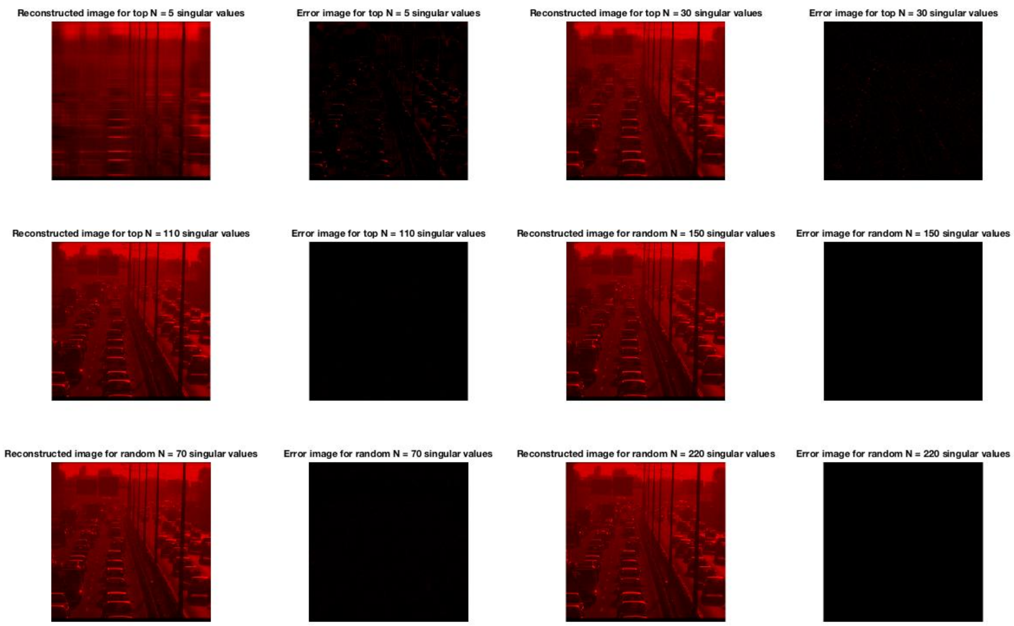

Figure 1.5: SVD on Red color band of Square image

1.3 SVD on Square Image by seperating it's Green color band

Similarly, we extract green color band from original image and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

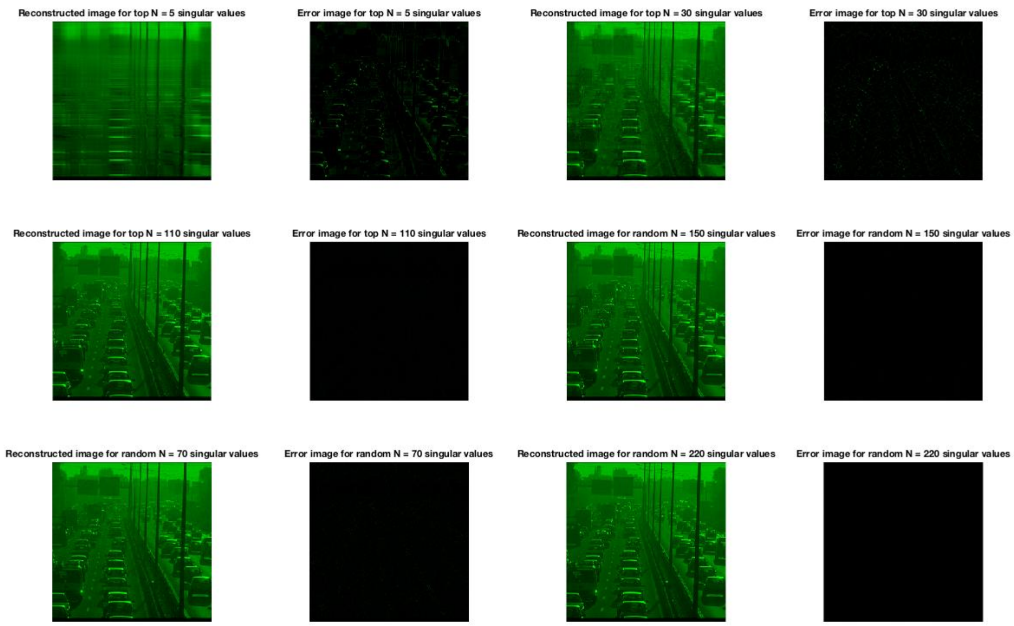

Figure 1.6: SVD on Green color band of Square image

1.4 SVD on Square Image by seperating it's Blue color band

Similarly, we extract blue color band from original image and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

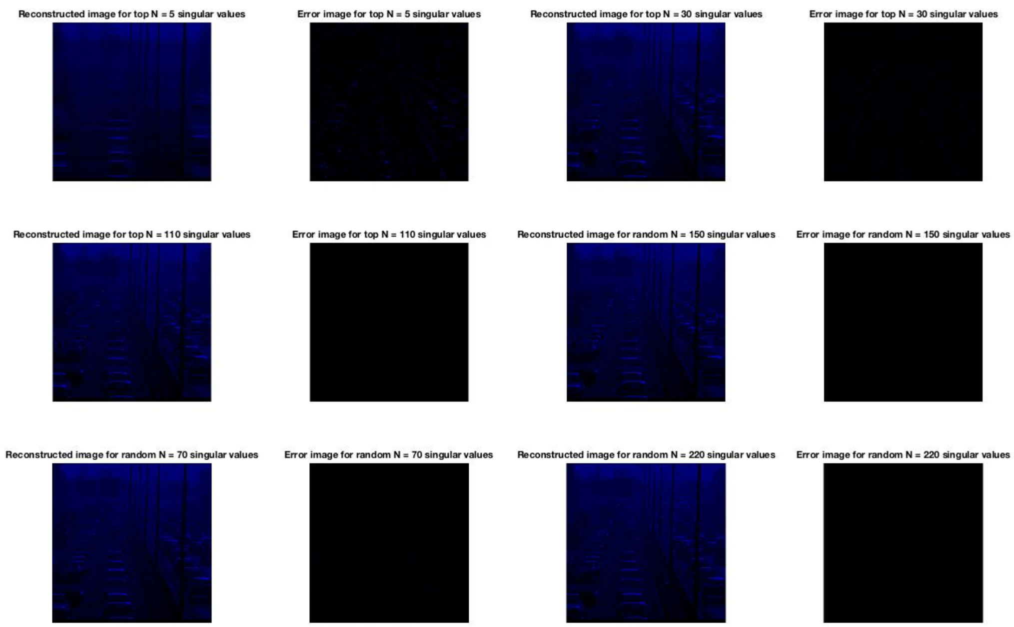

Figure 1.7: SVD on Blue color band of Square image

1.5 SVD on Square Image after concatenating the 8bit R,G,B channel as RGB to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as R then G then B and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

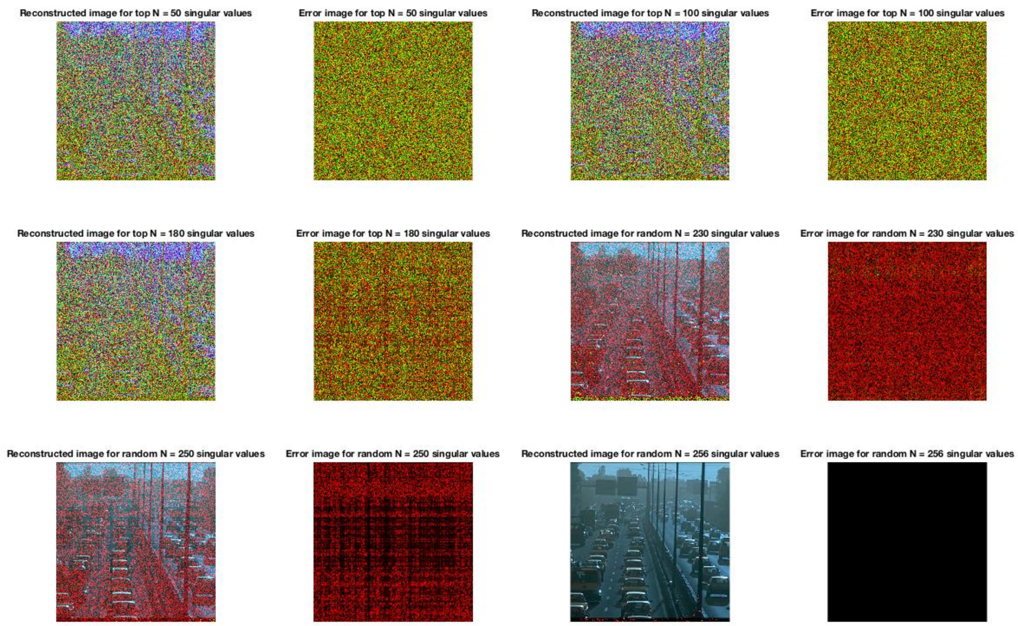

Figure 1.8: SVD on Square image concatenated at RGB

1.6 SVD on Square Image after concatenating the 8bit R,G,B channel as BRG to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as B then R then G and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

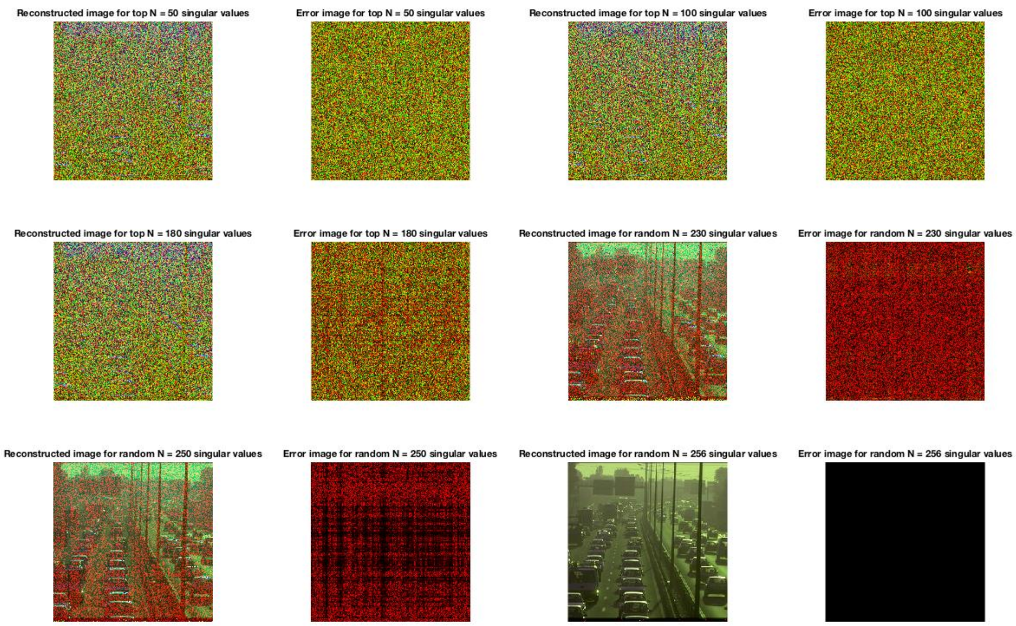

Figure 1.9: SVD on Square image concatenated at BRG

1.7 SVD on Square Image after concatenating the 8bit R,G,B channel as GBR to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as G then B then R and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

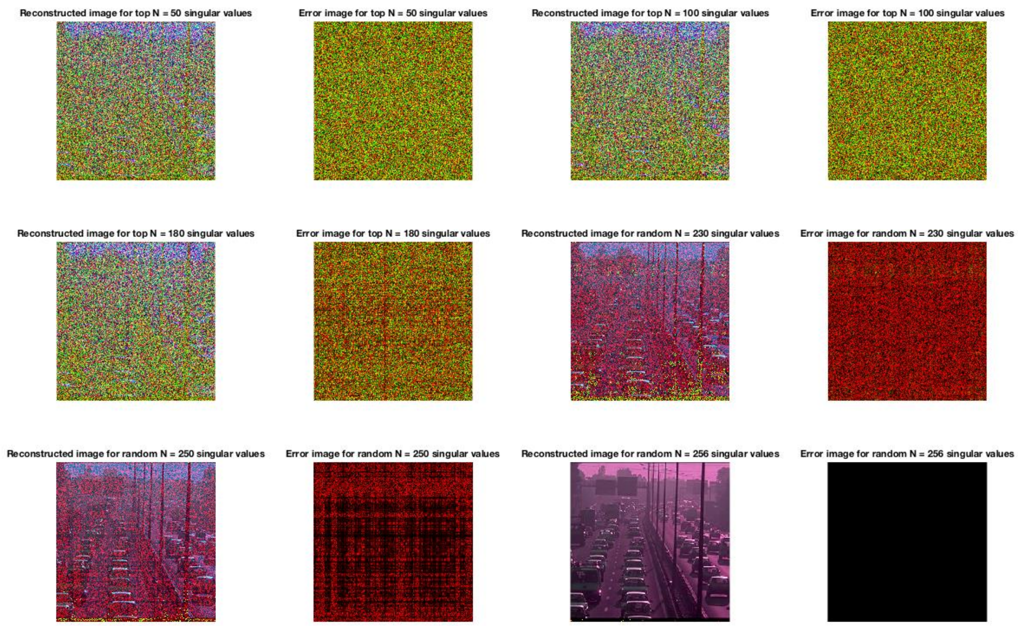

Figure 1.10: SVD on Square image concatenated at GBR

1.8 Comparitive Graph of reconstruction error vs N for 3 different methods of SVD on Square image

Below shows a comparison graph of reconstruction error vs N all three different methods namely

- Gray Scale
- Three color bands - R,G,B
- Three different types of concatenation of 8 bit R,G,B to 24 bit number

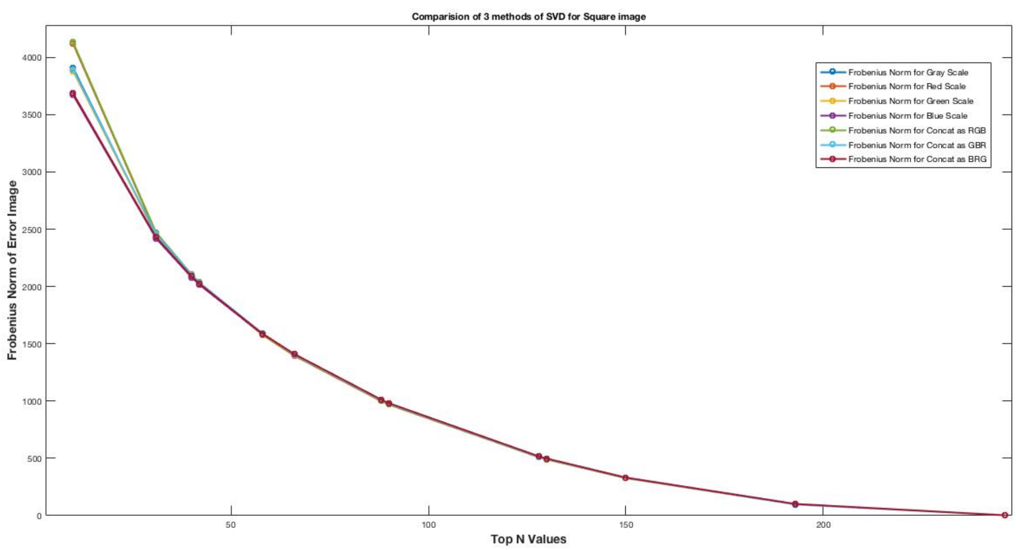

Figure 1.11: Comparitive Graph of 3 methods of SVD

Observation: We observe that as we increase selecting top N singular values the Forbenius Norm Error of error image decreases and it follows almost similar curve for other methods also after N = 30 singular value. Also we observed from the images with permuted R,G,B bands that the color band in the last dominates the whole image. As you can see in RGB blue color is dominating, in GBR red is dominating and in BRG green color is dominating the whole image.

1.9 SVD on Rectangle Image by converting to Gray scale

Similarly, we will perform same experiments for Rectangle image.

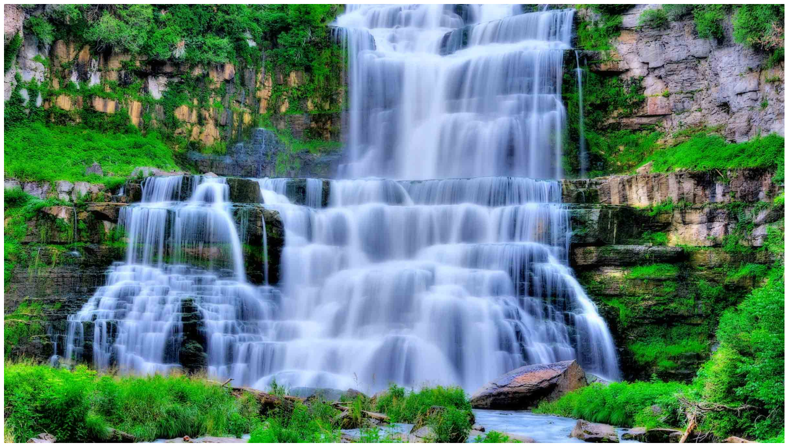

Figure 1.12: Original Rectangle image

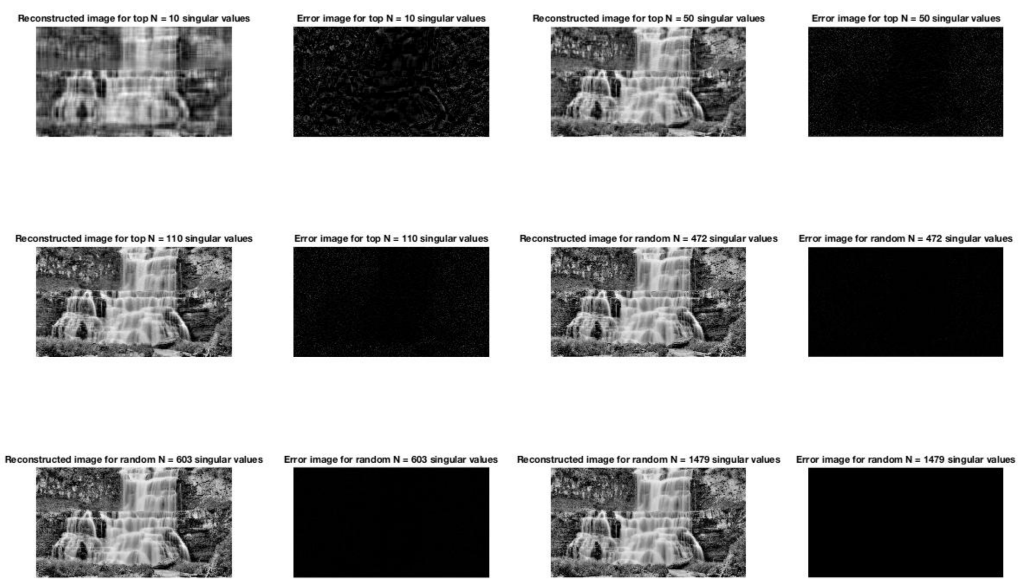

Figure 1.13: SVD on Gray Scale Rectangle image

Below is the comparison graph of Frobenius Norm Vs Top N singular values of Gray scale Rectangle image.

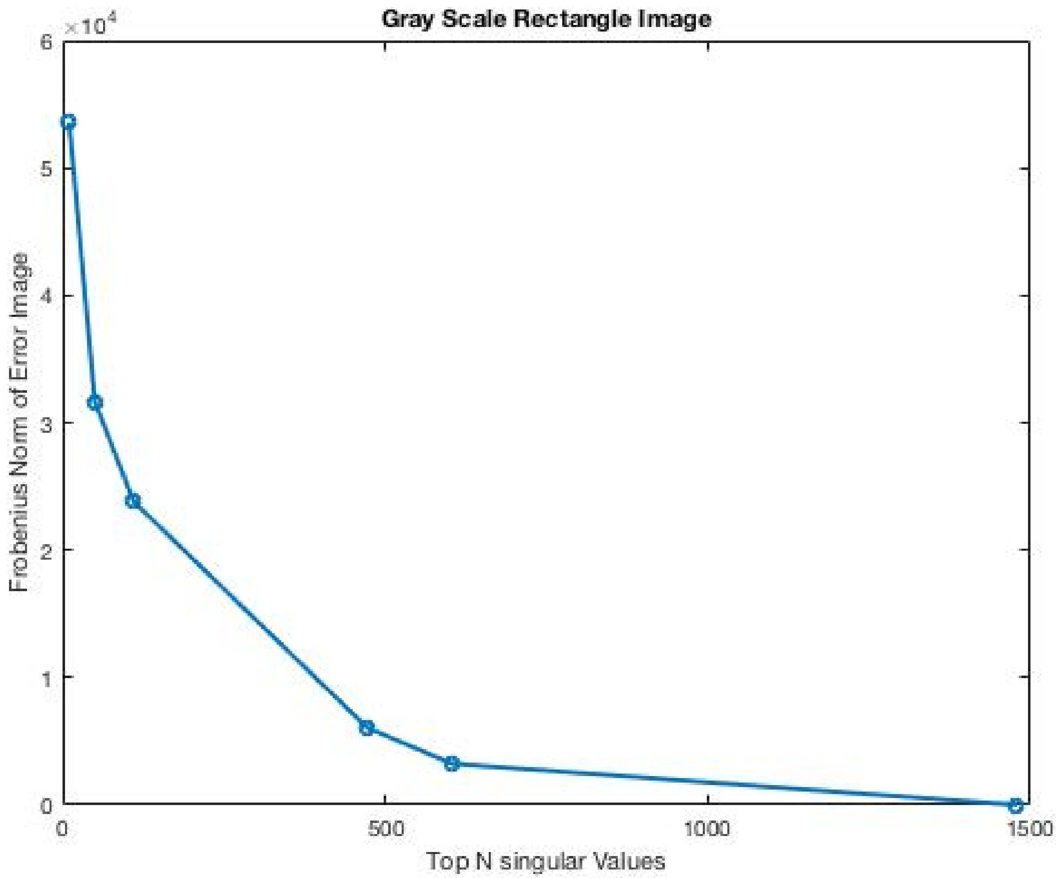

Figure 1.14: Graph of Frobenius Norm Vs Top N singular values of Gray scale Rectangle image

1.10 SVD on Rectangle Image by seperating it's Red color band

Similarly, we extract red color band from original image and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

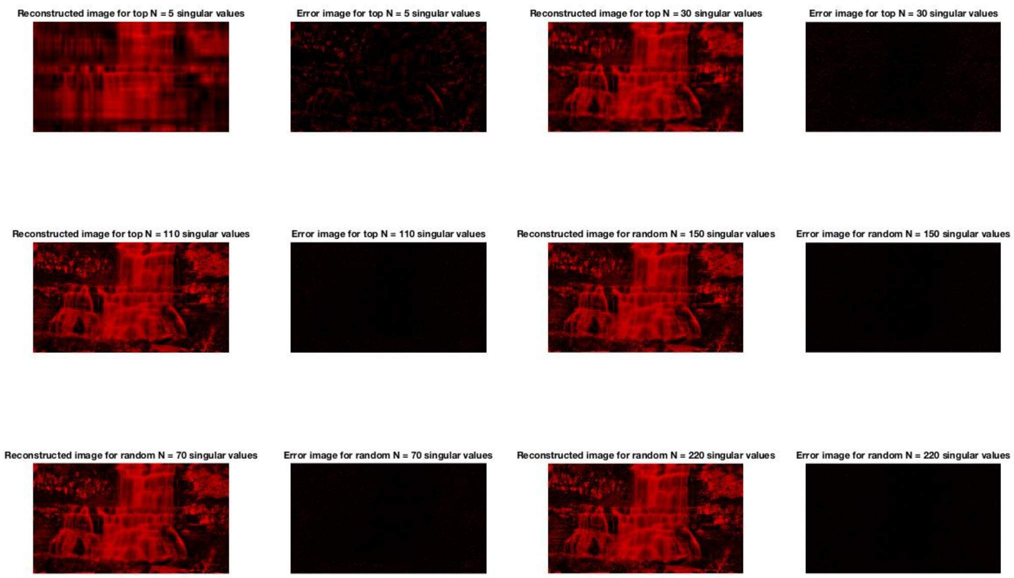

Figure 1.15: SVD on Red color band of Rectangle image

1.11 SVD on Rectangle Image by seperating it's Green color band

Similarly, we extract green color band from original image and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

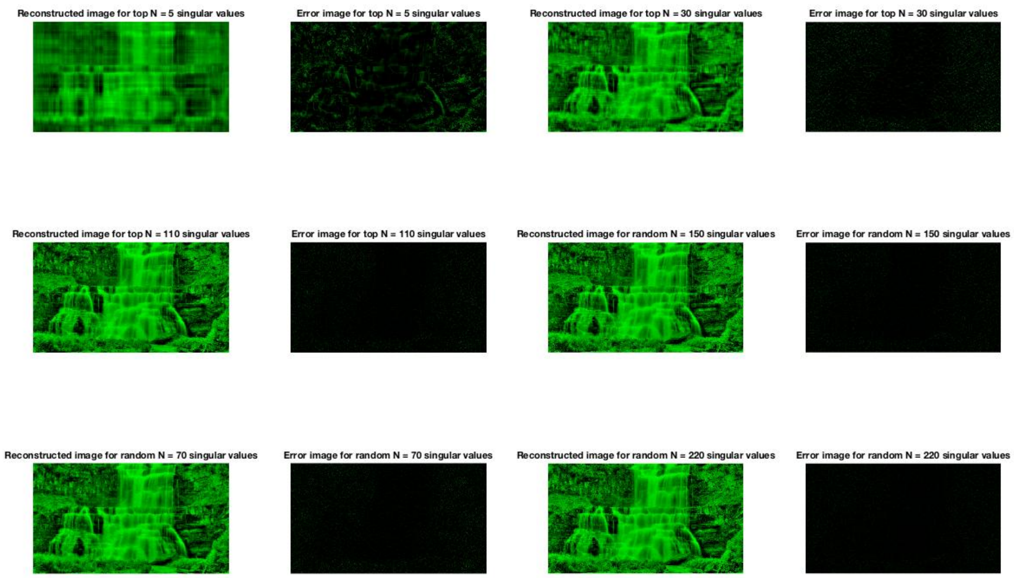

Figure 1.16: SVD on Green color band of Rectangle image

1.12 SVD on Rectangle Image by seperating it's Blue color band

Similarly, we extract blue color band from original image and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

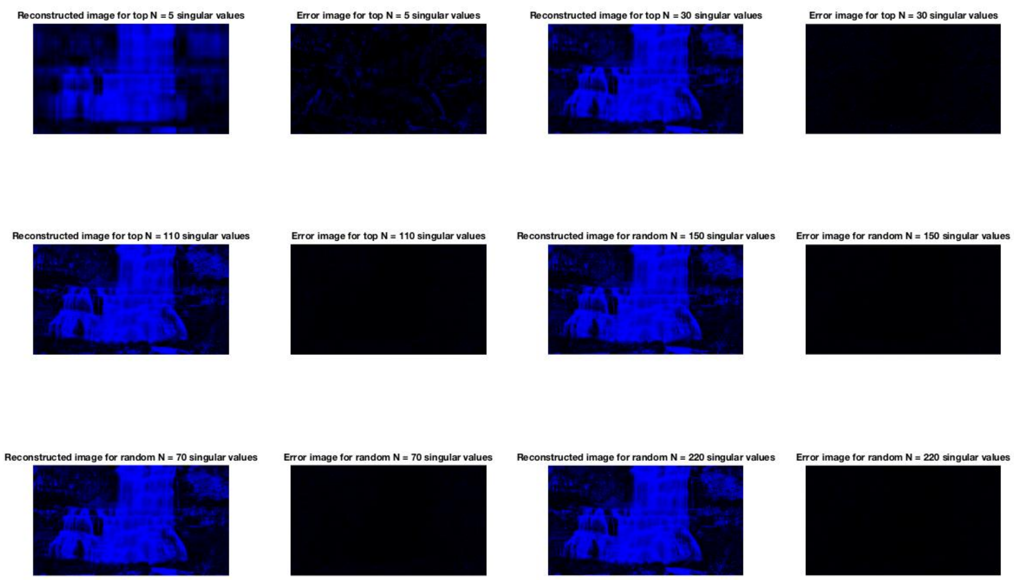

Figure 1.17: SVD on Blue color band of Rectangle image

1.13 SVD on Rectangle Image after concatenating the 8bit R,G,B channel as RGB to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as R then G then B and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

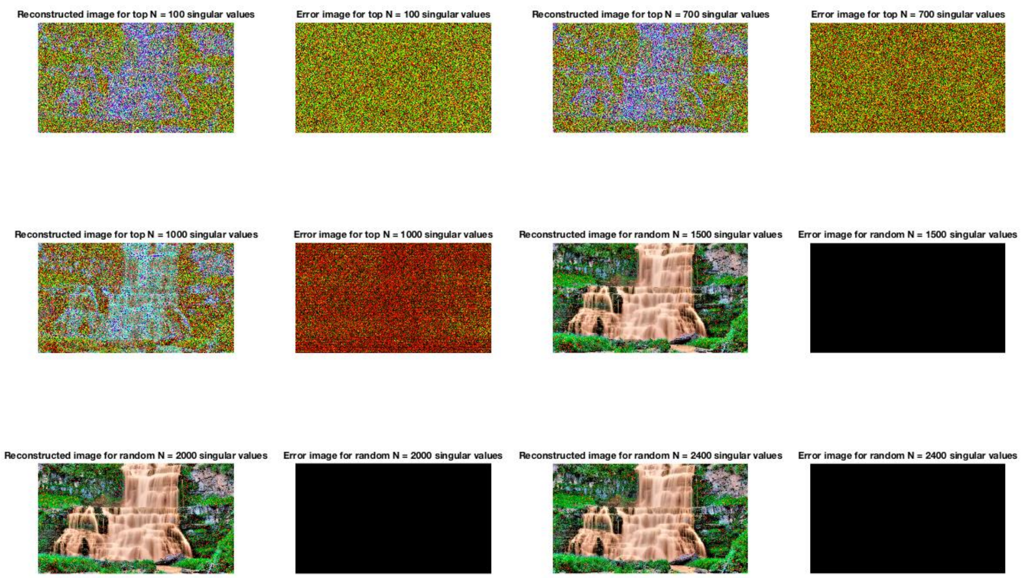

Figure 1.18: SVD on Rectangle image concatenated at RGB

1.14 SVD on Rectangle Image after concatenating the 8bit R,G,B channel as BRG to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as B then R then G and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

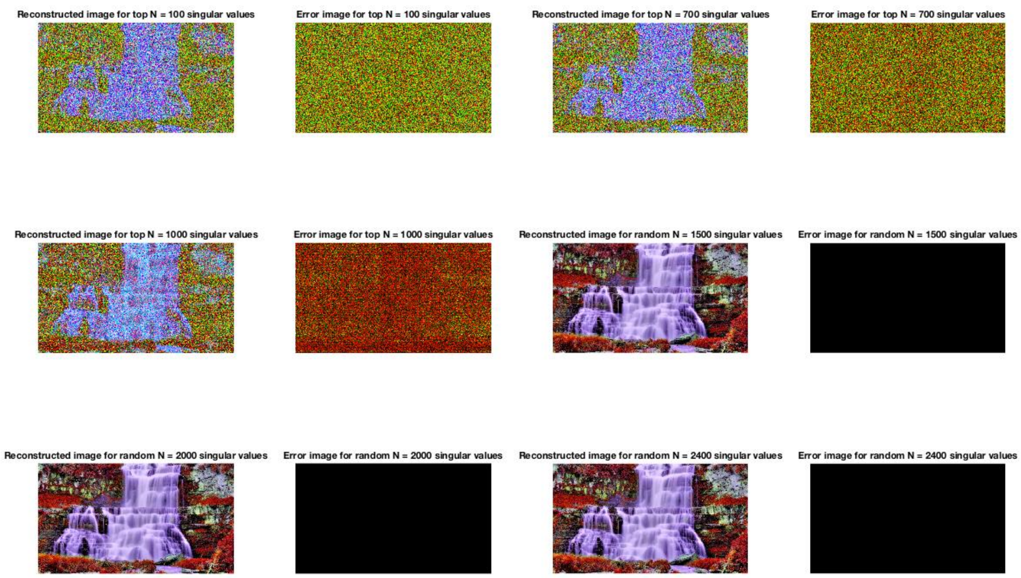

Figure 1.19: SVD on Rectangle image concatenated at BRG

1.15 SVD on Rectangle Image after concatenating the 8bit R,G,B channel as GBR to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as G then B then R and perform SVD and select top N singular values and reconstruct original image along with it's corresponding error image.

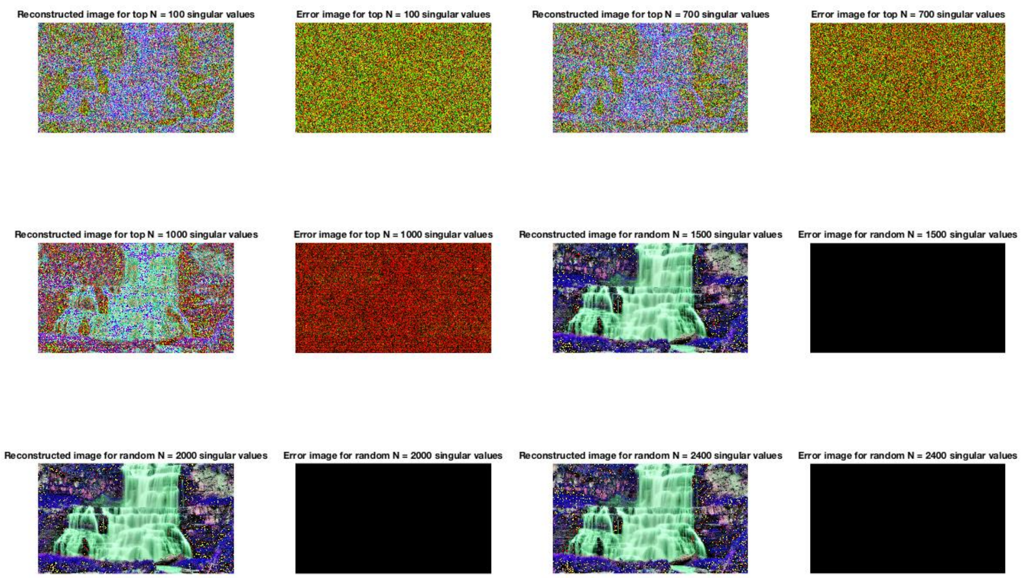

Figure 1.20: SVD on Rectangle image concatenated at GBR

1.16 Comparitive Graph of reconstruction error vs N for 3 different methods of SVD on Rectangle image

Below shows a comparison graph of reconstruction error vs N all three different methods namely

- Gray Scale
- Three color bands - R,G,B
- Three different types of concatenation of 8 bit R,G,B to 24 bit number

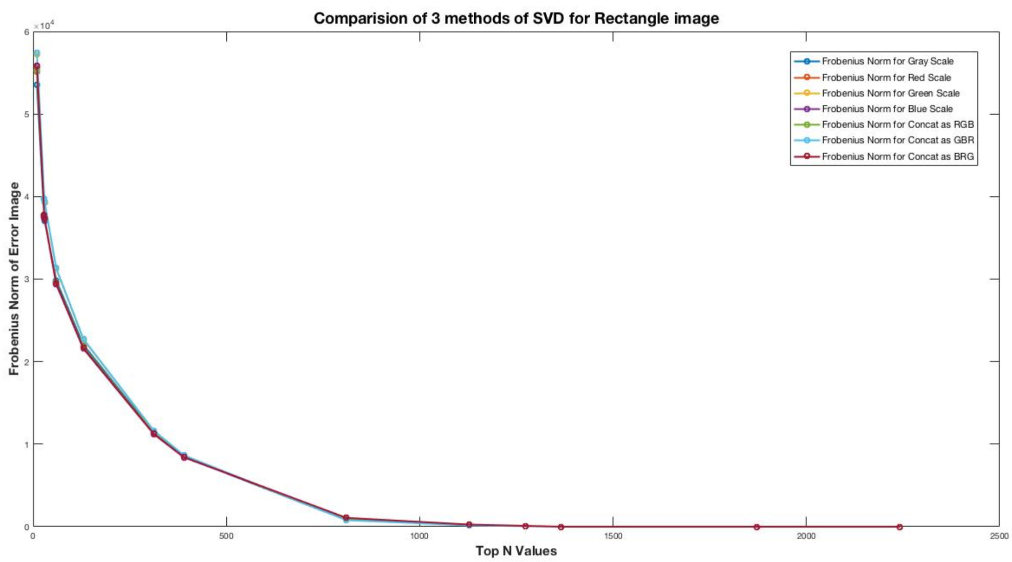

Figure 1.21: Comparitive Graph of 3 methods of SVD

Observation: We observe that as we increase selecting top N singular values the Forbenius Norm Error of error image decreases and it follows almost similar curve for other methods also. Also we observed from the images with permuted R,G,B bands that the color band in the center dominates the whole image. As you can see in RGB green color is dominating, in GBR blue is dominating and in BRG red color is dominating the whole image.

Chapter 2

Eigen Value Decomposition

We will perform Eigen Value Decomposition on both square and rectangular images.
(a) by converting the image to grayscale.
(b) separately on each color bands.
(c) after concatenating the 8bit R,G,B channel to form a 24bit number.

2.1 EVD on Square Image by converting to Gray scale

Eigen Value Decomposition is the factorization of a matrix in terms of its eigenvalues and eigenvectors. A Matric A can be factorized as $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}$ where Q is the square $\mathbf{N} \times \mathbf{N}$ matrix whose ith column is the eigenvector q_i of A and Λ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues.

First we will convert original image to gray scale and then perform Eigen Value Decomposition and reconstruct original image using $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}$

Figure 2.1: Original Square image

Figure 2.2: Original Gray Scale & Reconstructed Square image

Now, we will select Top N eigen values of $\mathbf{\Lambda}$ and reconstruct image along with it's corresponding error image with respect to original image.

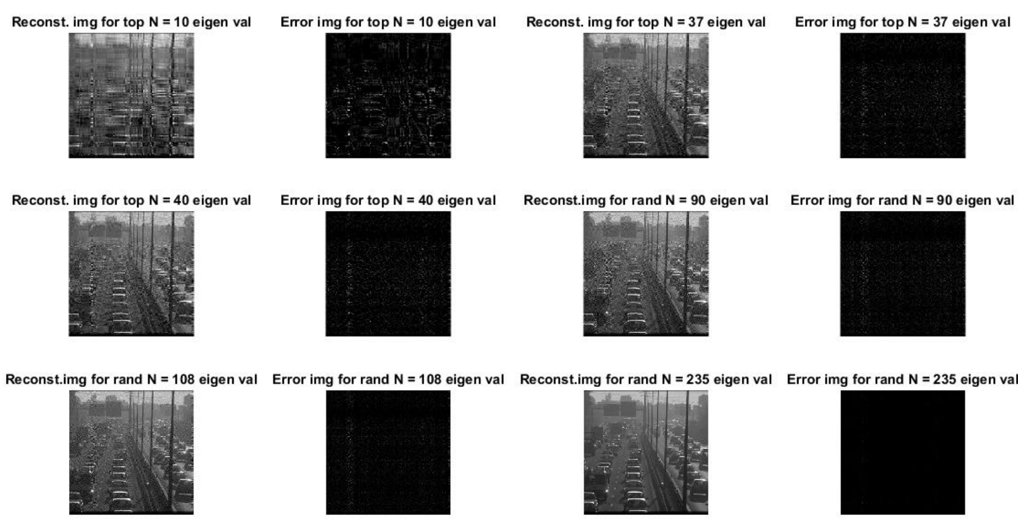

Figure 2.3: EVD on Gray Scale Square image

Below graph shows how Frobenius Norm of error image decreases as we select more top N eigen values.

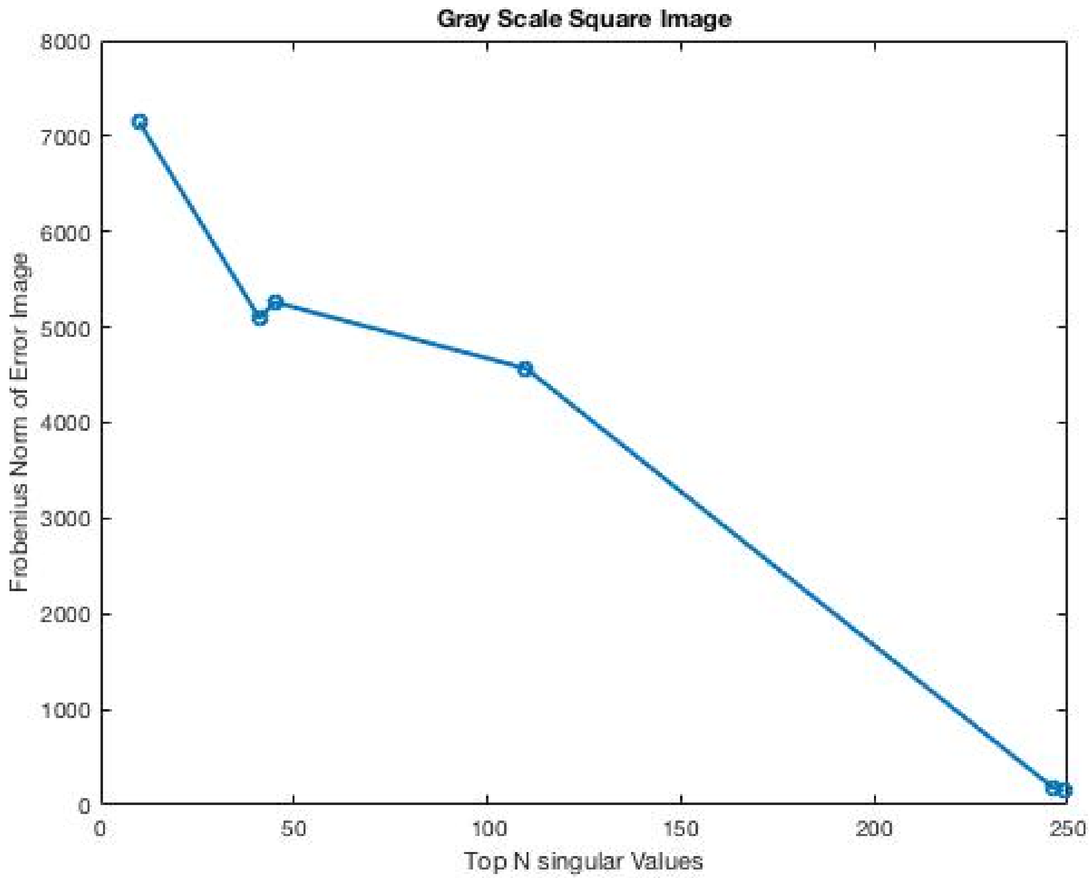

Figure 2.4: Graph of Frobenius Norm Vs Top N eigen values of Gray scale square image

2.2 EVD on Square Image by seperating it's Red color band

Similarly, we extract red color band from original image and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

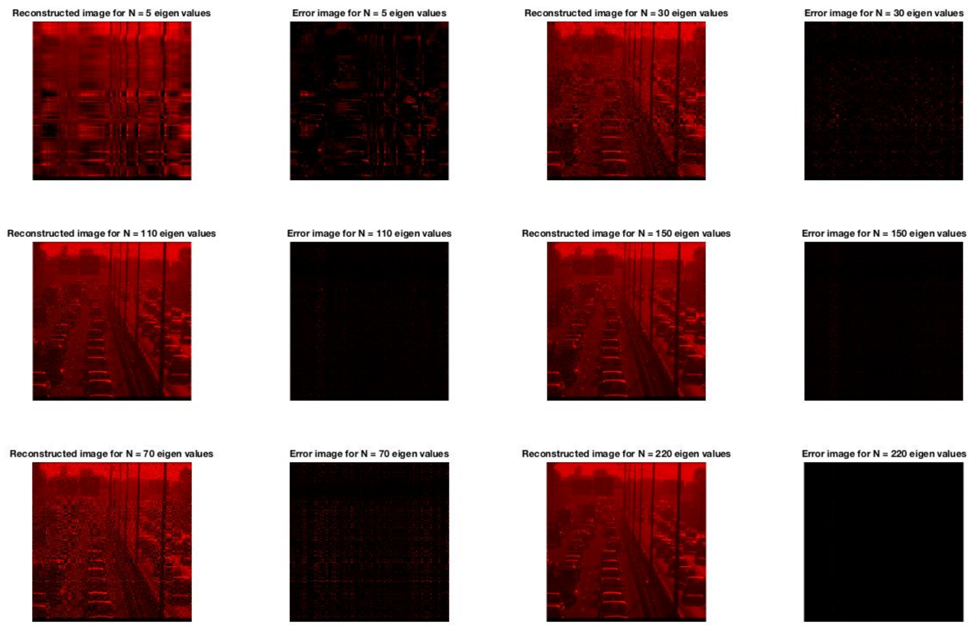

Figure 2.5: EVD on Red color band of Square image

2.3 EVD on Square Image by seperating it's Green color band

Similarly, we extract green color band from original image and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

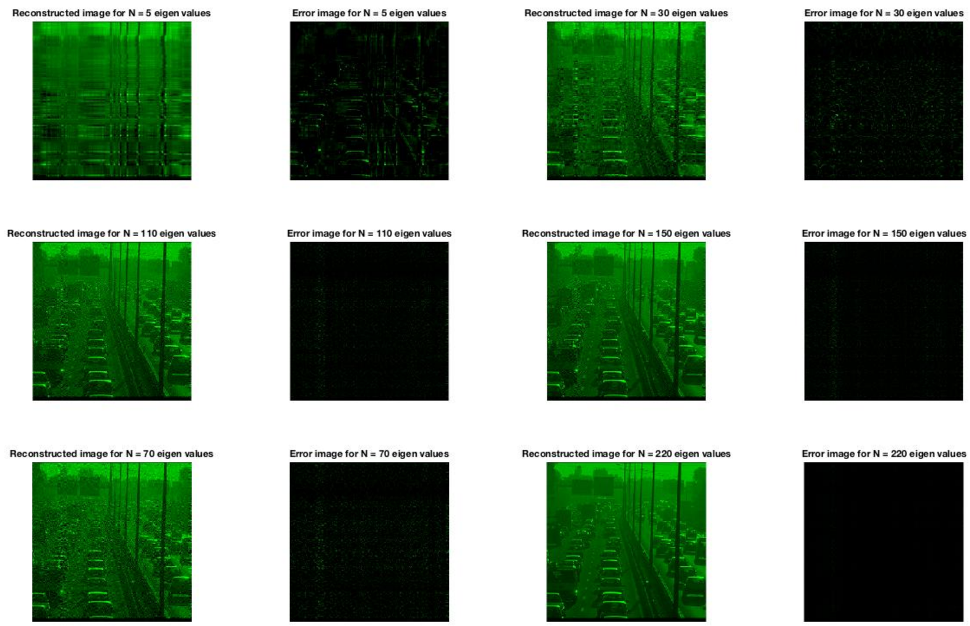

Figure 2.6: EVD on Green color band of Square image

2.4 EVD on Square Image by seperating it's Blue color band

Similarly, we extract blue color band from original image and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

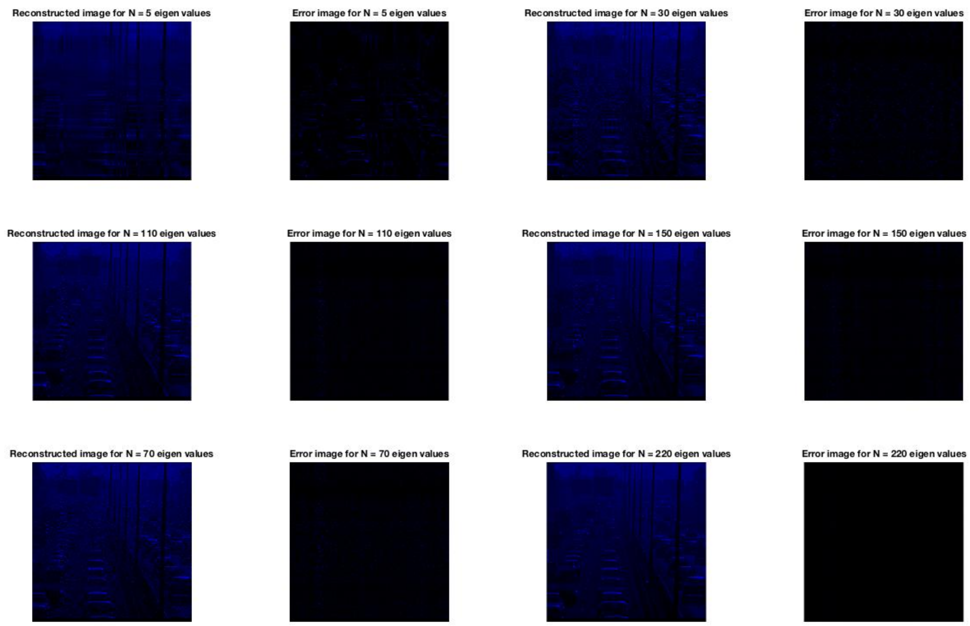

Figure 2.7: EVD on Blue color band of Square image

2.5 EVD on Square Image after concatenating the 8bit R,G,B channel as RGB to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as R then G then B and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

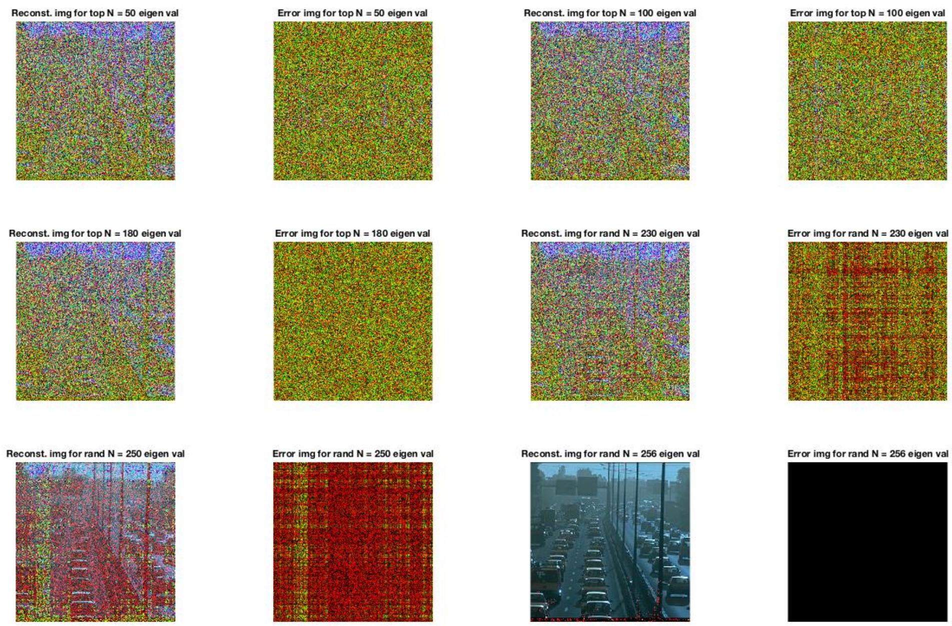

Figure 2.8: EVD on Square image concatenated at RGB

2.6 EVD on Square Image after concatenating the 8bit R,G,B channel as BRG to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as B then R then G and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

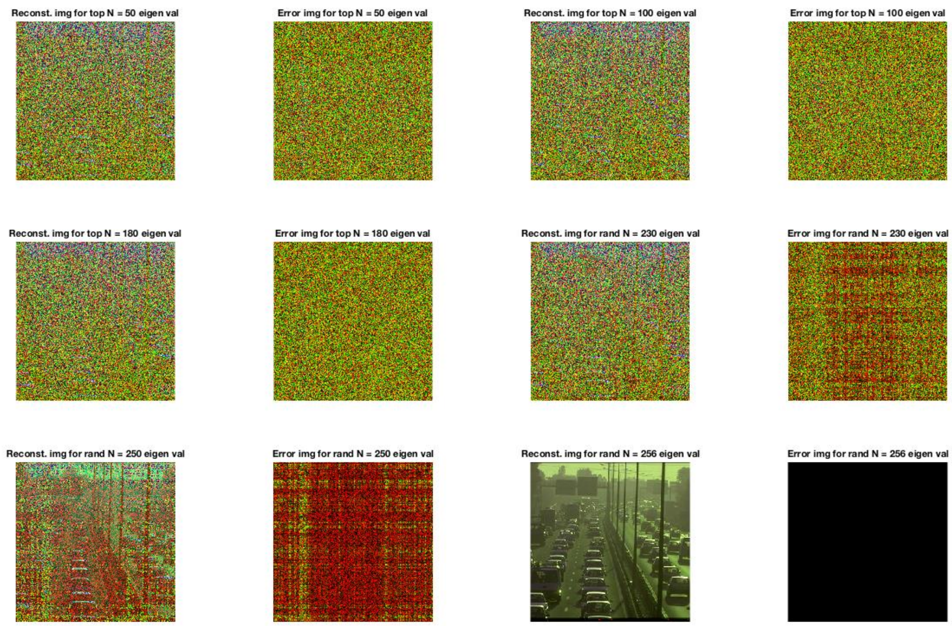

Figure 2.9: EVD on Square image concatenated at BRG

2.7 EVD on Square Image after concatenating the 8bit R,G,B channel as GBR to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as G then B then R and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

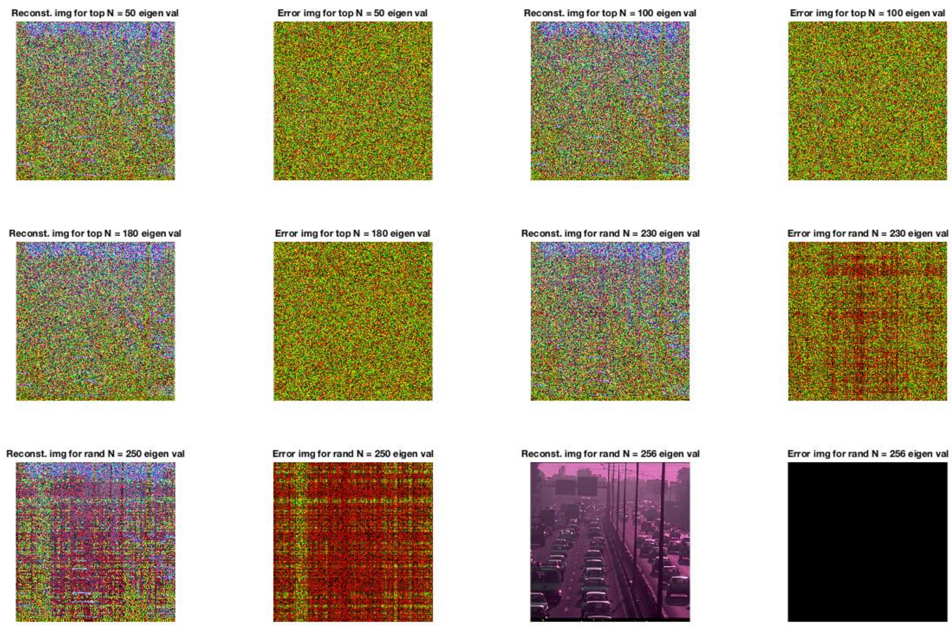

Figure 2.10: EVD on Square image concatenated at GBR

2.8 Comparitive Graph of reconstruction error vs N for 3 different methods of EVD on Square image

Below shows a comparison graph of reconstruction error vs N all three different methods namely

- Gray Scale
- Three color bands - R,G,B
- Three different types of concatenation of 8 bit R,G,B to 24 bit number

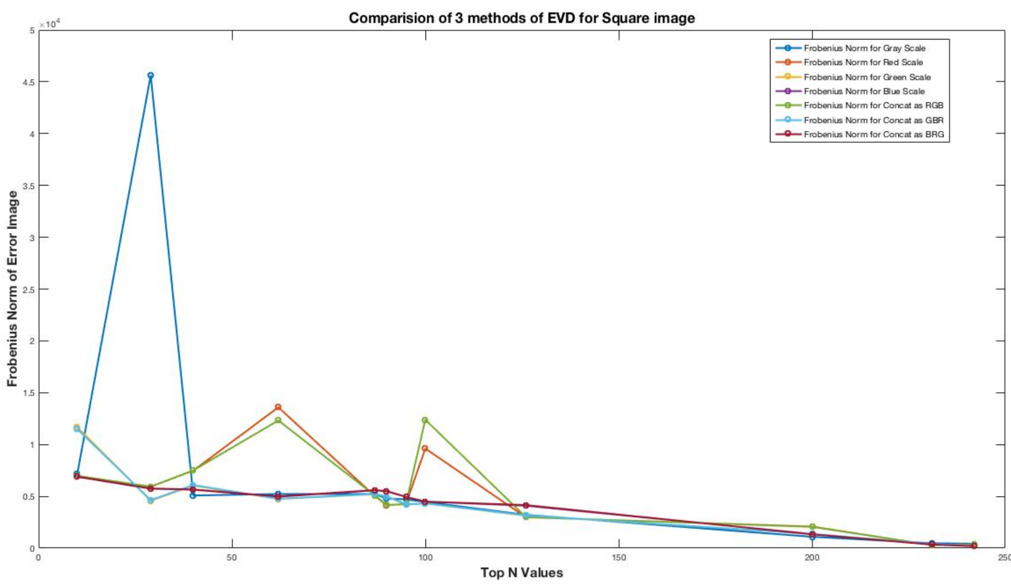

Figure 2.11: Comparitive Graph of 3 methods of EVD

Observation: We observe that as we increase selecting top N eigen values the Forbenius Norm Error of error image decreases. And error is more when red component is more in proportion to other in image for eigen values upto 150 while for other method they stabilises and decreases gradually after eigen value 30.

2.9 Comparitive Graph of reconstruction error vs N for 3 different methods of EVD & SVD on Square image

Below shows a comparison graph between SVD & EVD of reconstruction error vs N for all three different methods namely

- Gray Scale
- Three color bands - R,G,B
- Three different types of concatenation of 8 bit R,G,B to 24 bit number

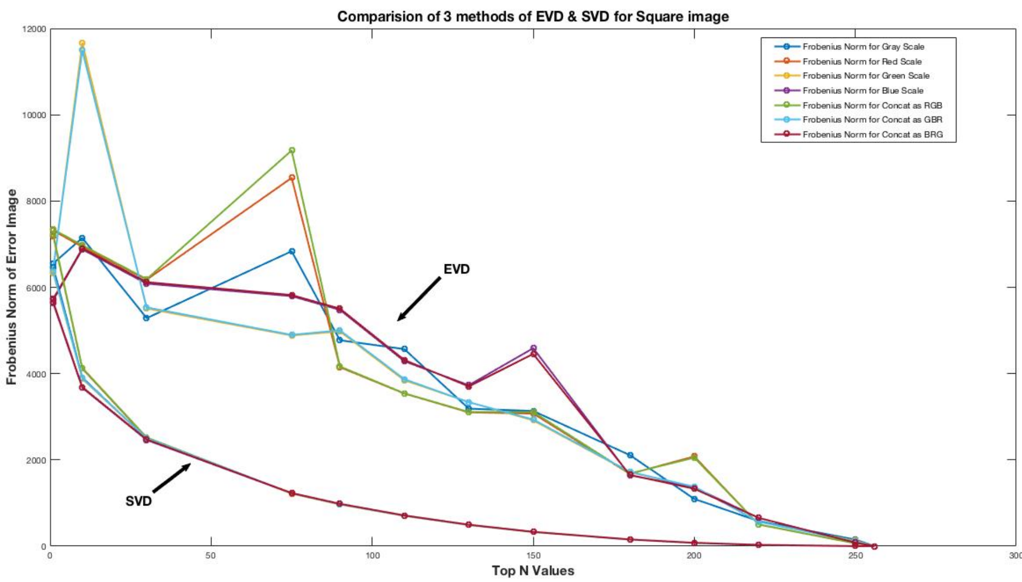

Figure 2.12: Comparitive Graph of 3 methods of EVD & SVD

Observation: We observe that as we increase selecting top N eigen/singular values the Forbenius Norm Error of error image decreases. For SVD the graph is smooth and it keeps on gradually decreasing with increase in N, While graph of EVD is not smooth it spikes up when red component is more in image. SVD error decreases rapildy to 2000 at N = 35 while EVD takes N = 180 to reach at 2000 error.

2.10 EVD on Rectangle Image by converting to Gray scale

To perform Eigen Value Decomposition for rectangle image $\mathbf{A}$, we first find eigen values $\mathbf{\Lambda}$ and eigen vector $\mathbf{Q}$ for matrix $\mathbf{A} * \mathbf{A'}$, then we use formula $\mathbf{M} = \mathbf{Q\Lambda Q}^{-1}$ to reconstrust $\mathbf{A} * \mathbf{A'}$. We then multiply reconstructed matrix $\mathbf{M}$ with pseudo inverse of $\mathbf{A'}$ which is $(\mathbf{A} * \mathbf{A'})^{\mathbf{-1}} * \mathbf{A}$ to reconstrust original image $\mathbf{A}$.

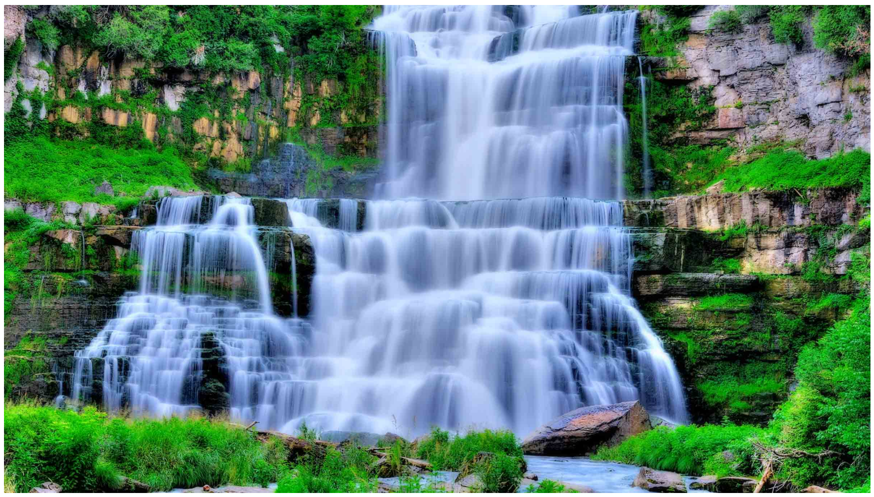

Figure 2.13: Original Rectangle image

Figure 2.14: Original Gray Scale & Reconstructed Rectangle image using EVD

Now, we will select Top N eigen values of $\mathbf{\Lambda}$ and reconstruct image along with it's corresponding error image with respect to original image.

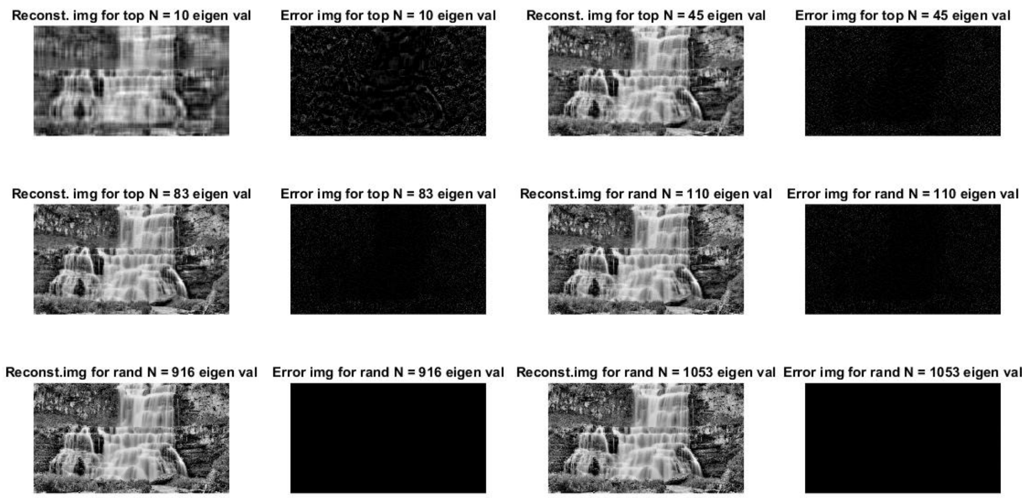

Figure 2.15: EVD on Gray Scale Rectangle image

2.11 EVD on Rectangle Image by seperating it's Red color band

Similarly, we extract red color band from original image and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

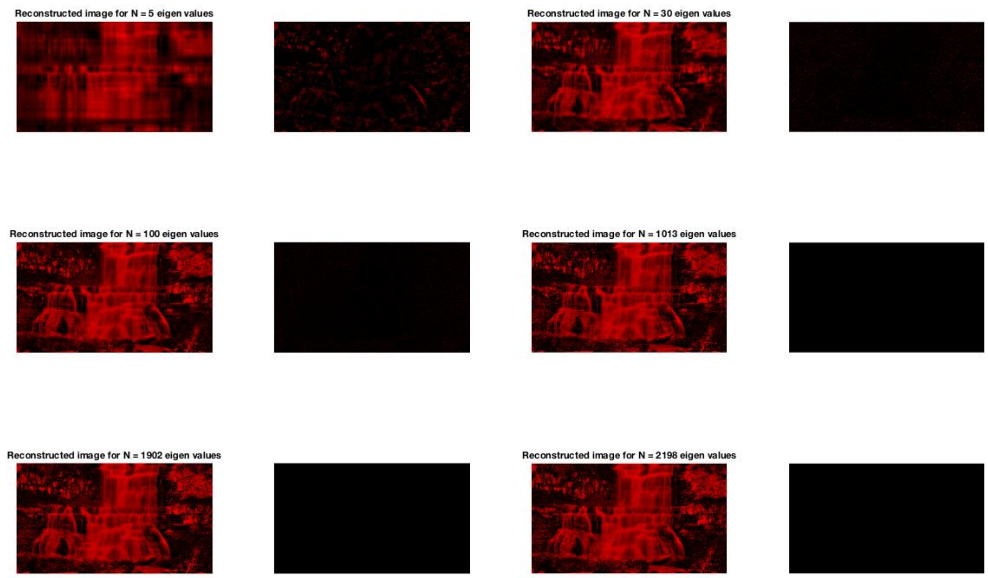

Figure 2.16: EVD on Red color band of Rectangle image

2.12 EVD on Rectangle Image by seperating it's Green color band

Similarly, we extract green color band from original image and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

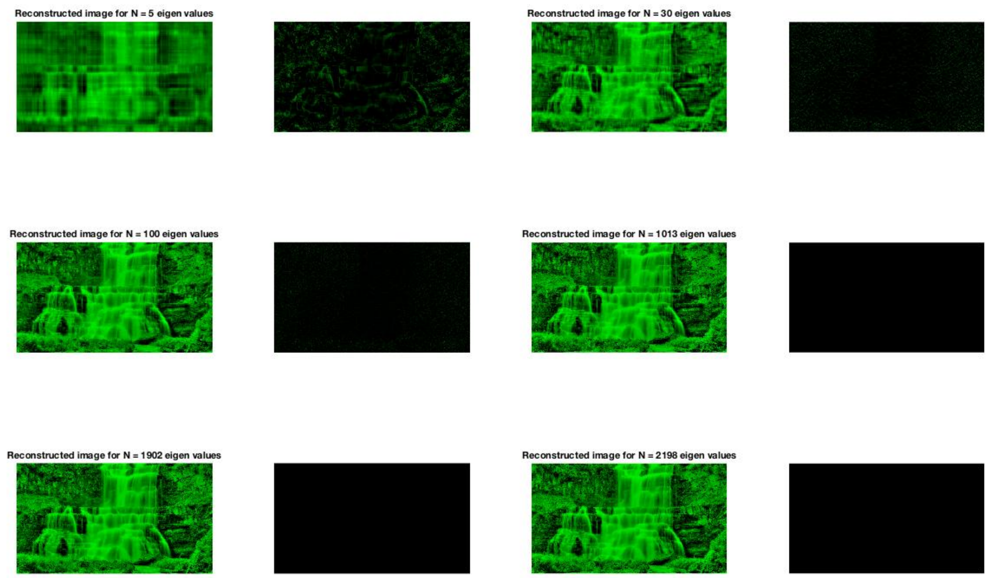

Figure 2.17: EVD on Green color band of Rectangle image

2.13 EVD on Rectangle Image by seperating it's Blue color band

Similarly, we extract blue color band from original image and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

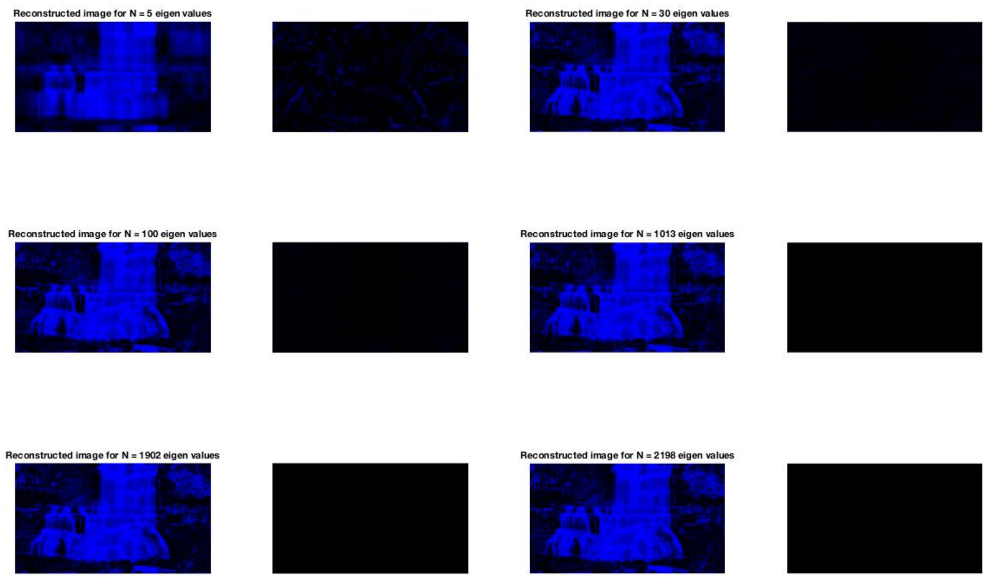

Figure 2.18: EVD on Blue color band of Rectangle image

2.14 EVD on Rectangle Image after concatenating the 8bit R,G,B channel as RGB to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as R then G then B and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

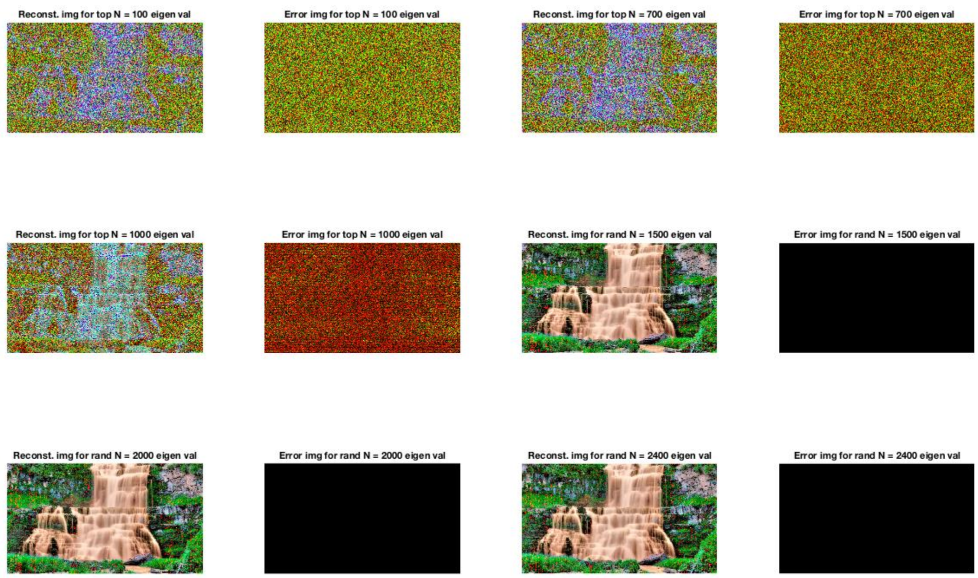

Figure 2.19: EVD on Rectangle image concatenated at RGB

2.15 EVD on Rectangle Image after concatenating the 8bit R,G,B channel as BRG to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as B then R then G and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

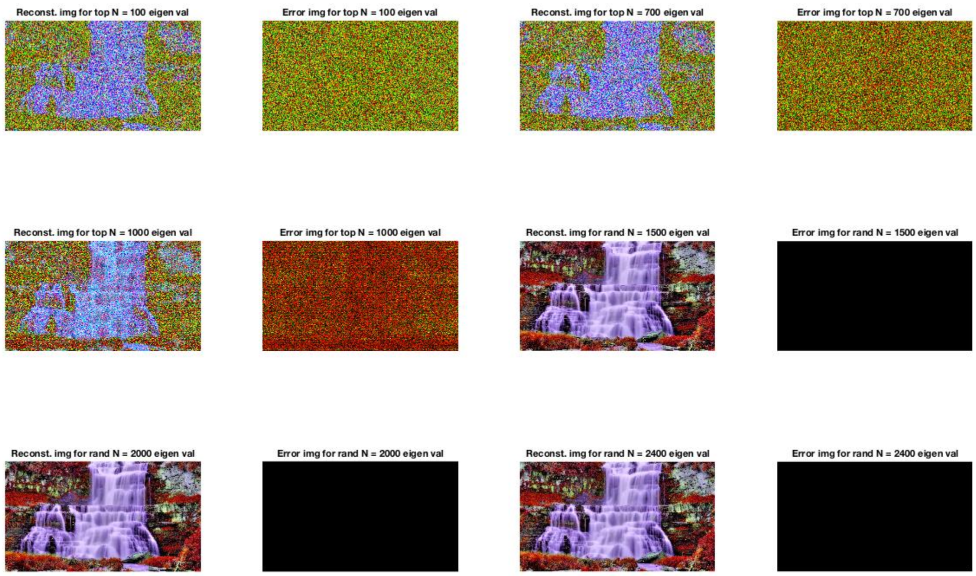

Figure 2.20: EVD on Rectangle image concatenated at BRG

2.16 EVD on Rectangle Image after concatenating the 8bit R,G,B channel as GBR to form a 24bit number

We extract 8bit R,G,B color bands from original image, then concatenate into a 24bit number as G then B then R and perform EVD and select top N eigen values and reconstruct original image along with it's corresponding error image.

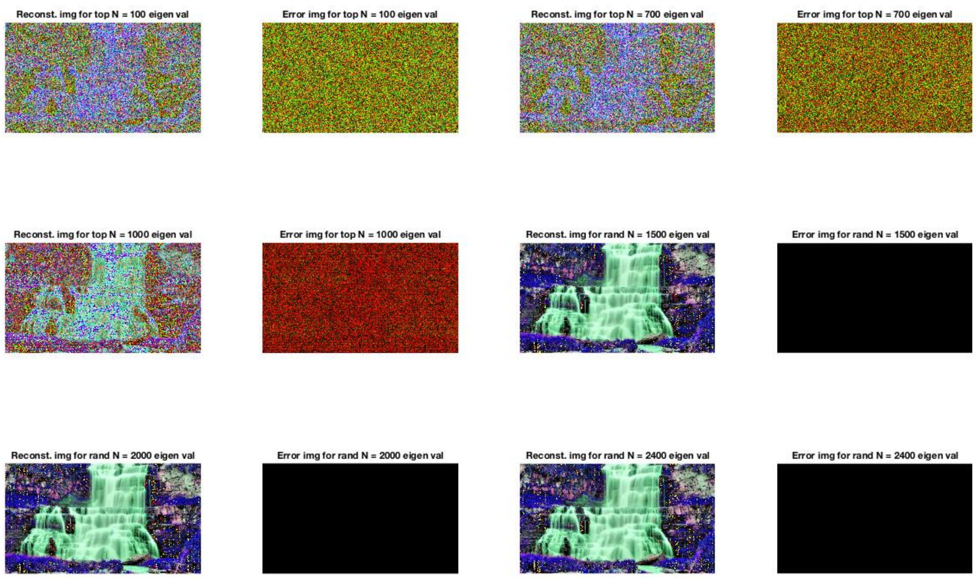

Figure 2.21: EVD on Rectangle image concatenated at GBR

2.17 Comparitive Graph of reconstruction error vs N for 3 different methods of EVD on Rectangle image

Below shows a comparison graph of reconstruction error vs N all three different methods namely

- Gray Scale
- Three color bands - R,G,B
- Three different types of concatenation of 8 bit R,G,B to 24 bit number

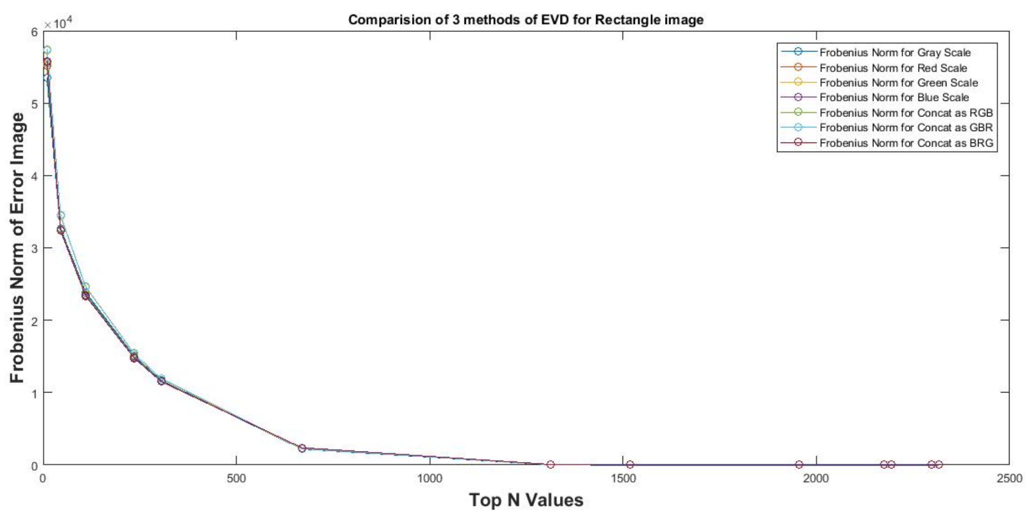

Figure 2.22: Comparitive Graph of 3 methods of EVD

Observation: We observe that as we increase selecting top N eigen values the Forbenius Norm Error of error image decreases and it follows almost similar curve for other methods also.

2.18 Comparitive Graph of reconstruction error vs N for 3 different methods of EVD & SVD on Rectangle image

Below shows a comparison graph between SVD & EVD of reconstruction error vs N for all three different methods namely

- Gray Scale
- Three color bands - R,G,B
- Three different types of concatenation of 8 bit R,G,B to 24 bit number

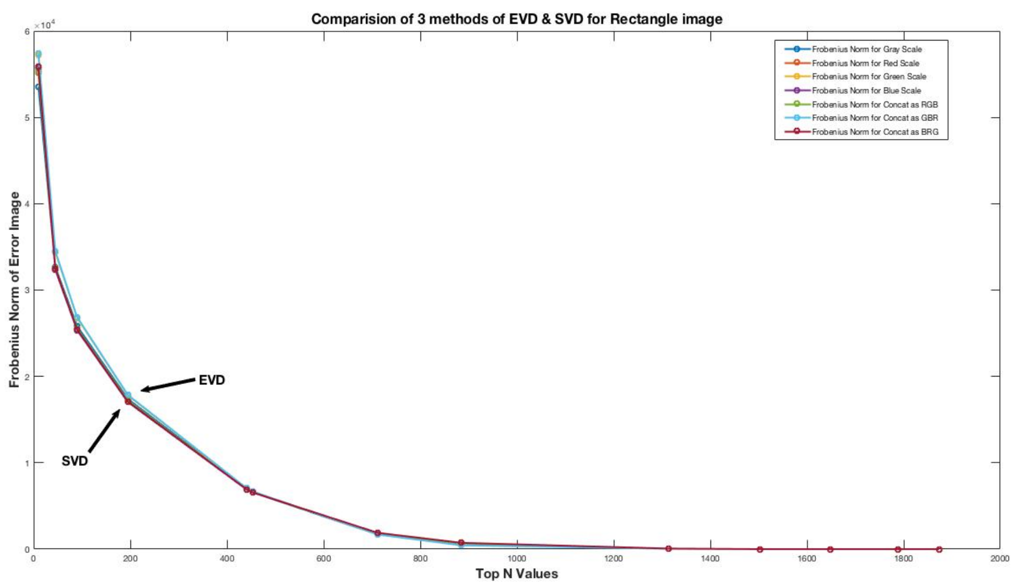

Figure 2.23: Comparitive Graph of 3 methods of EVD & SVD

Observation: We observe that as we increase selecting top N eigen/singular values the Forbenius Norm Error of error image decreases. For both EVD & SVD the graph is almost similar and it keeps on gradually decreasing with increase in N.

Chapter 3

Polynomial Regression

The goal of regression analysis is to model the expected value of a dependent variable y in terms of the value of an independent variable (or vector of independent variables) x. In general, we can model the expected value of y as an nth degree polynomial, yielding the general polynomial regression model.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \varepsilon.$$

To come up with the best model which fits the give data we will try out with different orders of polynomial of input features x. Then for every model with polynomial order M we will find out mininum parameter which will give is it's best line/curve for that model by using normal equation.

$$\theta = (\mathbf{X^T X})^{-1}\mathbf{X^T}y$$

By using this parameter θ we will test our model on validation data and calculate Sum of Squared Error, then whichever models gives minimum value of error will be selected as best model for given data.

We will divide out data set into three parts:

- Training data - 70
- Validation data - 20
- Testing data - 10

We will find parameter θ for every model from training data and then we will use that parameter θ to calculate estimated value for validation data using $\mathbf{E} = X * \theta$ and find Sum of Squared Error between estimated and actual output. Then select one which gives minimum error.

3.1 1-dimensional data

Below figure shows SSE Vs Degree for testing and validation data, we can see as we increase degree the error for training data decreases but error for validation data increases which means we are overfitting our parameters on input data so it will not work correctly for new unseen data, hence error increases for validation data.

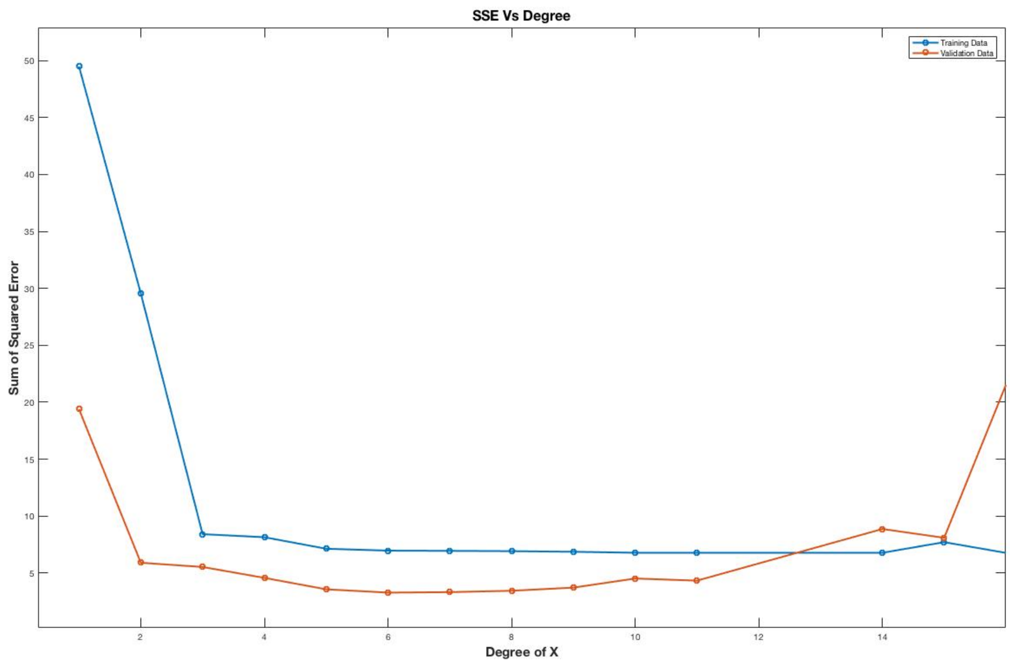

Figure 3.1: SSE Vs Degree for 1D training and validation data

Now from above graph observation, we will select degree M as 5 since it is giving minium error for validation data. Then we will use parameter θ for degree 5 which we calculated from training data on testing data to estimate output.

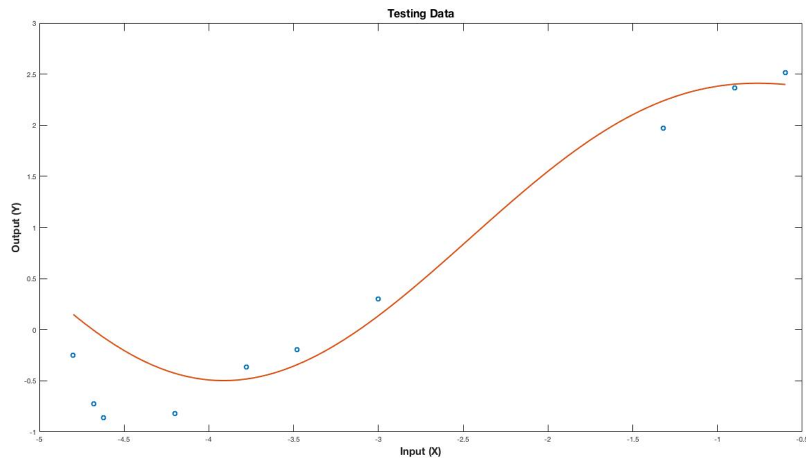

Figure 3.2: Plotting polynomial for M = 5 on testing data

And below is graph of Actual output Vs Estimated output for selected parameter θ and degree M = 5.

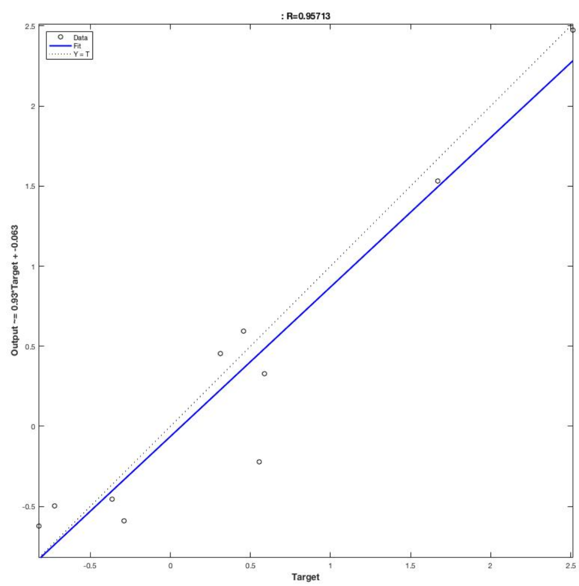

Figure 3.3: Actual output Vs Estimated output

Observation: We observe how error decreases as we increase degree for training data but error for validation data increases which means we are overfitting our parameters on input data so it will not work correctly for new unseen data, hence error increases for validation data.

3.2 Ridge Regression on 1-dimensional data

As we saw in previous experiment our error increases with increase in degree for unseen validation data, means it the regression model becomes tailored to fit the quirks and random noise in your specific sample rather than to approximate the true model.
So, as we increase the degree of M the value of coefficiencts in parameter θ increases to very high value, so to control this we add regularization parameter $\mathbf{\Lambda}$ to our original formula for finding parameter θ.

$$\theta = (\mathbf{X^T X} + \mathbf{\Lambda})^{-1}\mathbf{X^T}y$$

We will try out different models with different order of polynomial degree and different $\mathbf{\Lambda}$ to find out which gives least error on validation data.

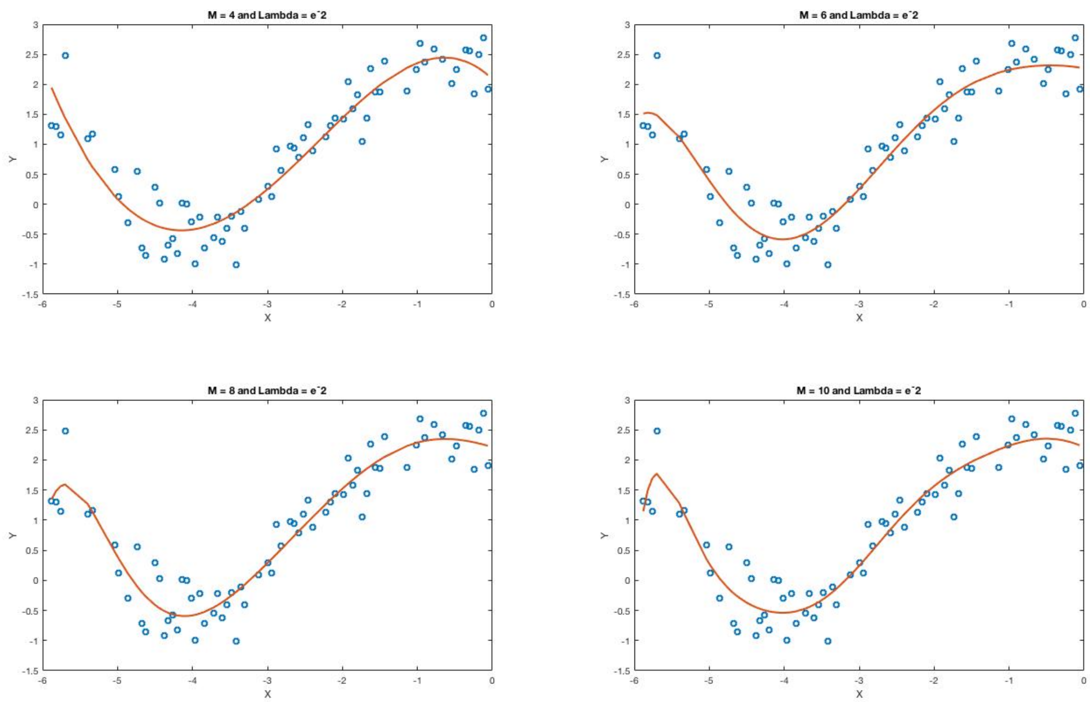

Figure 3.4: 1D Training Data with polynomial of degree 4 and lambda eˆ2

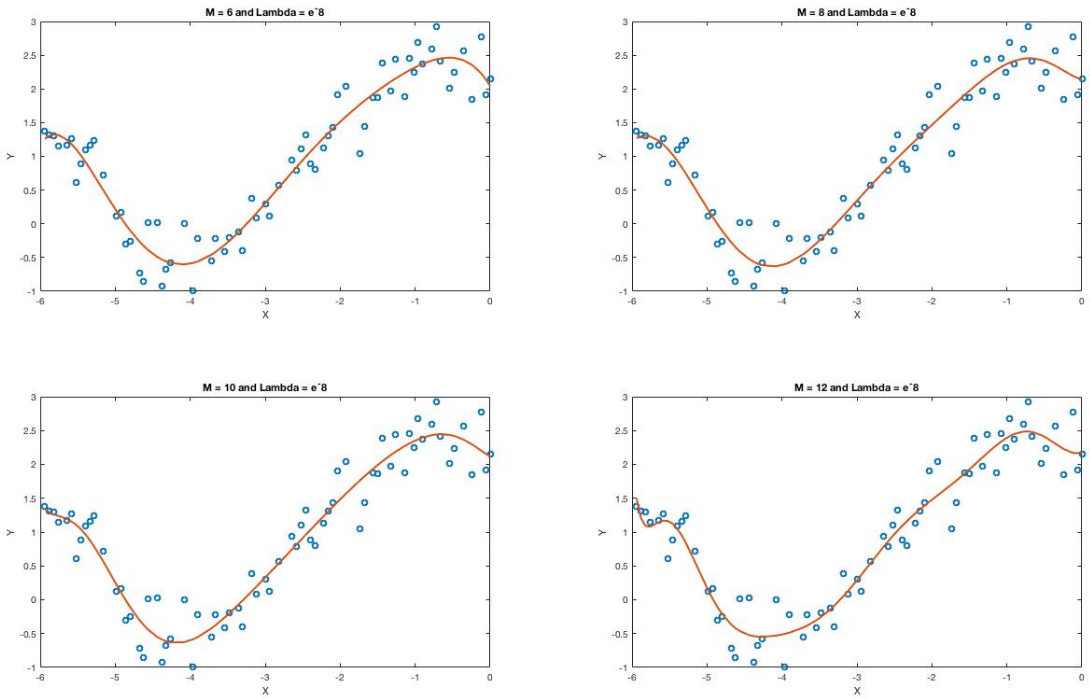

Figure 3.5: 1D Training Data with polynomial of degree 6 and lambda eˆ-8

Now, we will try out with different values of λ on few models to find which lambda is best for our data. Below is graph of SSE Vs Lambda for M = 5 & 15.

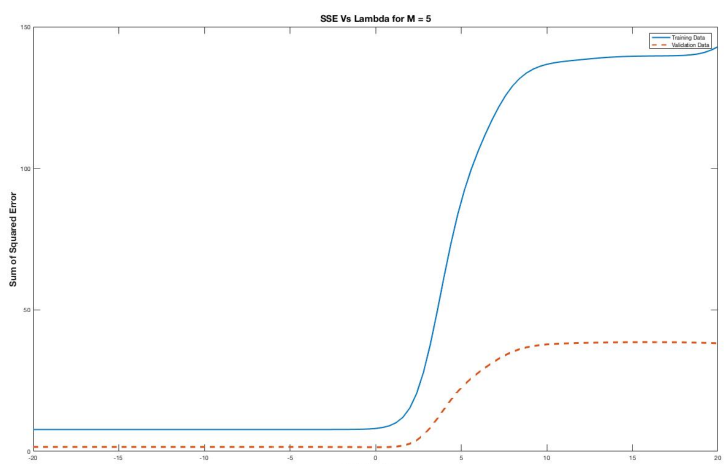

Figure 3.6: SSE Vs Lambda for M = 5 on training and validation data

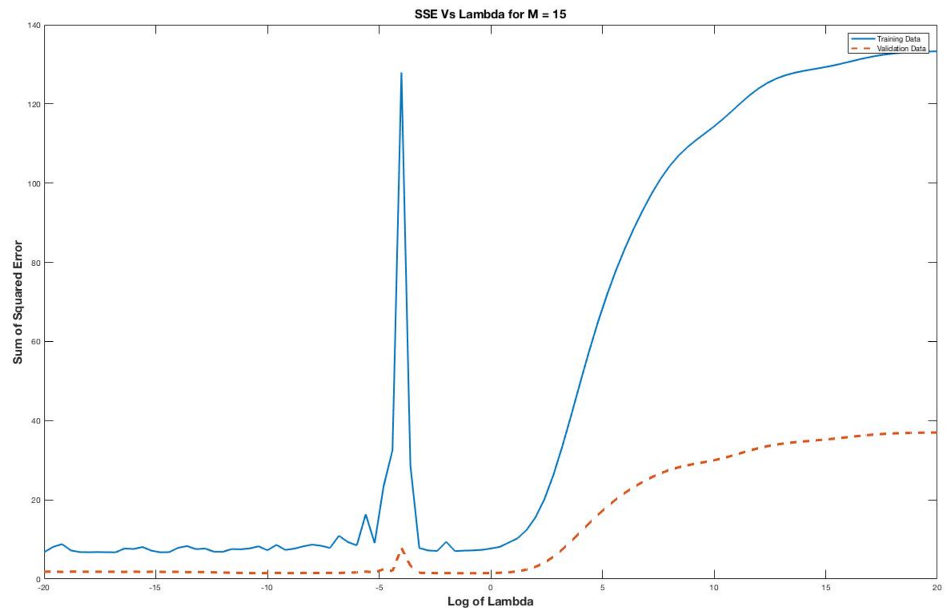

Figure 3.7: SSE Vs Lambda for M = 15 on training and validation data

We observe that as value of λ increases, error increases as we are making coefficeints of parameter θ small and eventually it stabilizes and follows straight line. For higher order polynomial also the error is same and regulariation works with few fluctuations. And we can observe that validation error is less compared to training error as we have not allowed data to overfit on training data. Thus we select lambda = eˆ-8 as it error is low and it stabilies around that.

Below figure shows SSE Vs Degree for testing and validation data, we can observe how error decreases as we increasing the degree and since we have used regulariation the error is high for training data but less for validation data as we have reduced coefficients of higher order through regulariation.

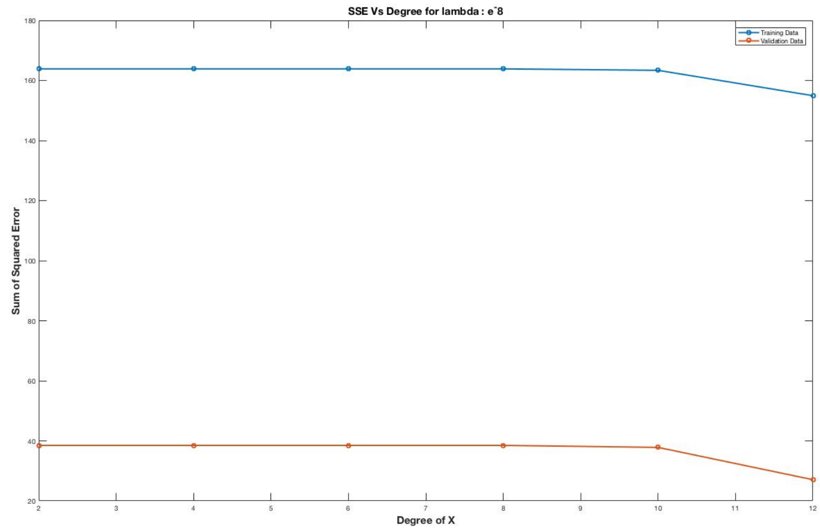

Figure 3.8: SSE Vs Degree for 1D ridge training and validation data for lambda = e$\hat{-}$8 & M = 8

Now from above graph observation, we will select degree lambda = e$\hat{-}$8 and M as 8 since it is stable around that order for validation data. Then we will use parameter θ for degree 5 which we calculated from training data on testing data to estimate output.

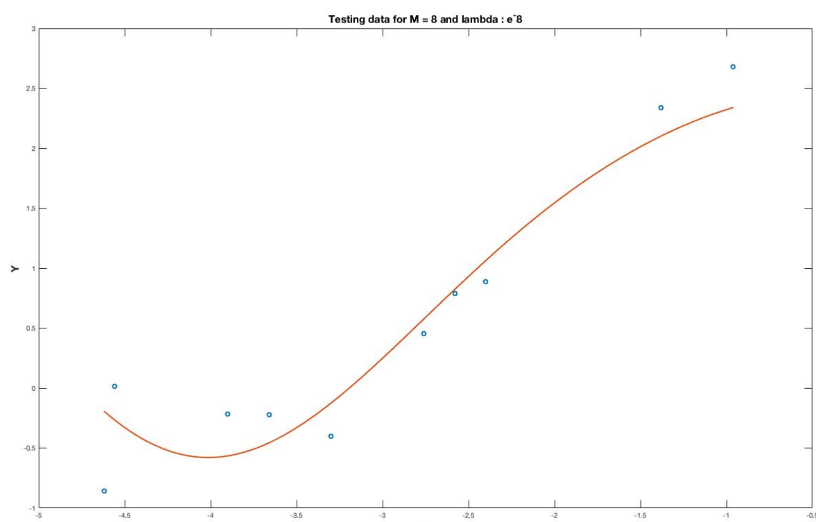

Figure 3.9: Plotting polynomial for M = 8 and lambda = e$\hat{-}$8 on testing data

And below is graph of Actual output Vs Estimated output for selected parameter θ and degree M = 8 and lambda = eˆ-8.

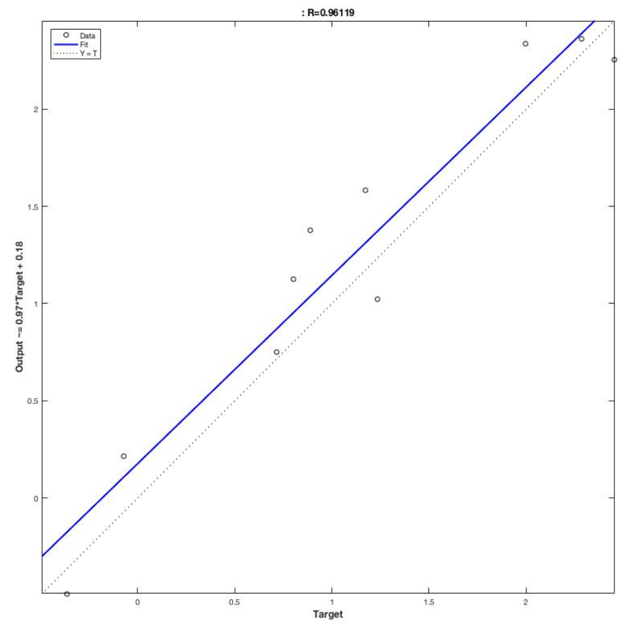

Figure 3.10: Actual output Vs Estimated output

Observation: We observe how error decreases slowly as we increase degree for training data and as we are using lambda = eˆ-8 it's error is more compared to validation dara as we are not allowing data to overfit and error decreases for validation data so it will work correctly for new unseen data.

3.3 2-dimensional data

Now we will perform polymonial regression on 2-dimensional data. We will try out different models with different order of polynomial degree and find out which gives least error on validation data.

Below is the plot of training data with polynomial of degree 1,2 & 3

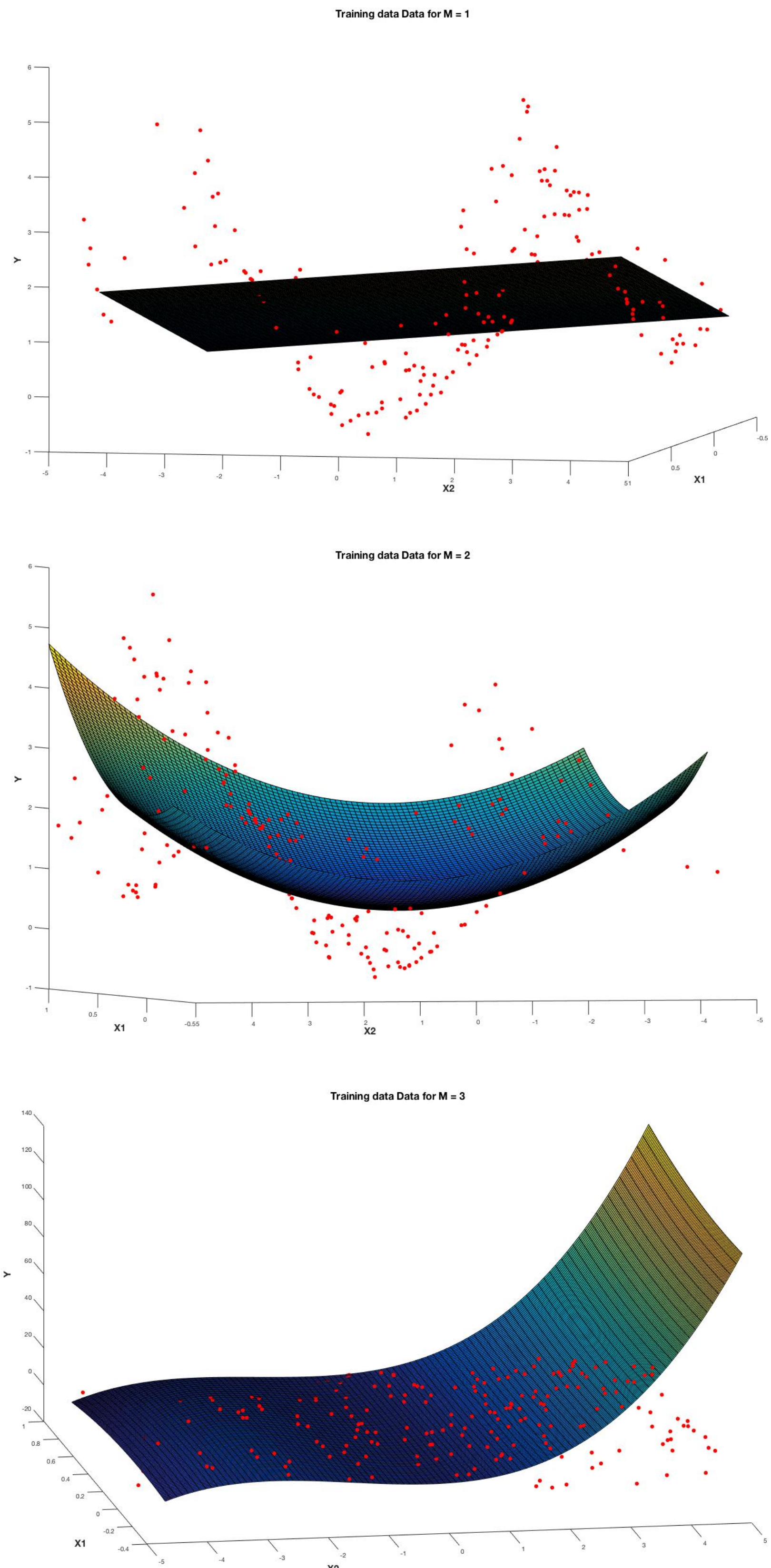

Figure 3.11: 2D Training Data with polynomial of degree 1,2 & 3

Below figure shows SSE Vs Degree for testing and validation data, we can observe how error decreases as we increasing the degree and it stabilies to zero after degree 7.

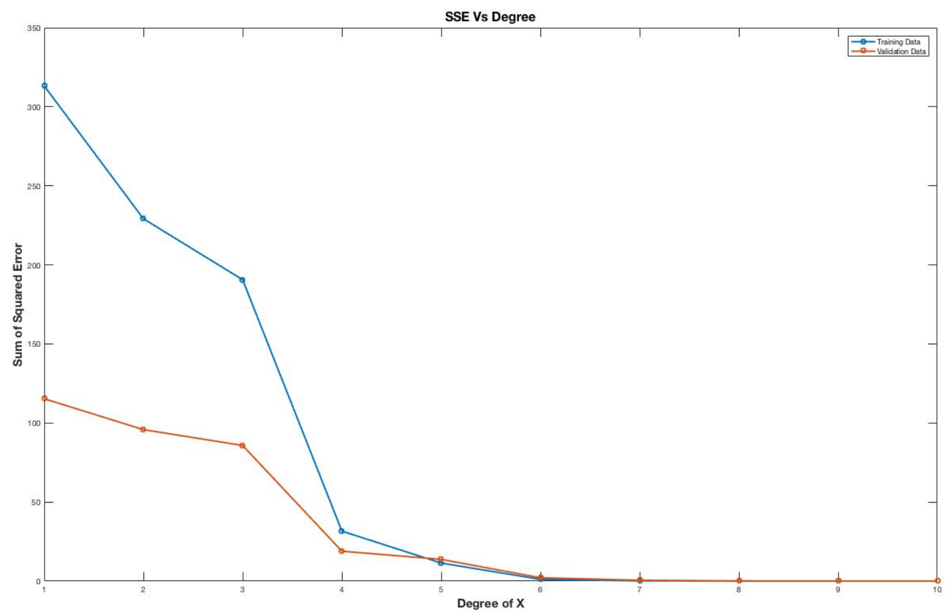

Figure 3.12: SSE Vs Degree for 2D training and validation data

Now from above graph observation, we will select degree M as 7 since it is giving minium error for validation data. Then we will use parameter θ for degree 7 which we calculated from training data on testing data to estimate output.

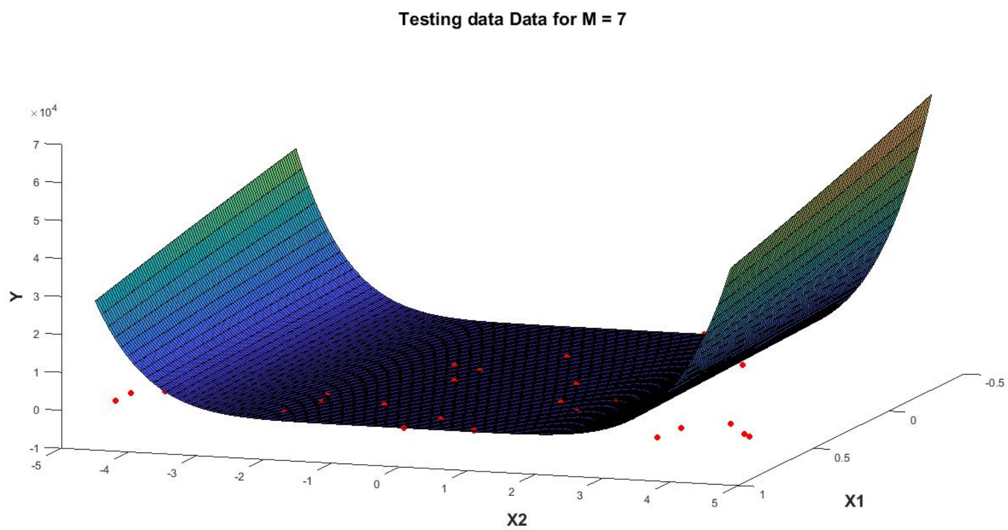

Figure 3.13: Plotting polynomial for M = 7 on testing data

And below is graph of Actual output Vs Estimated output for selected parameter θ and degree M = 5.

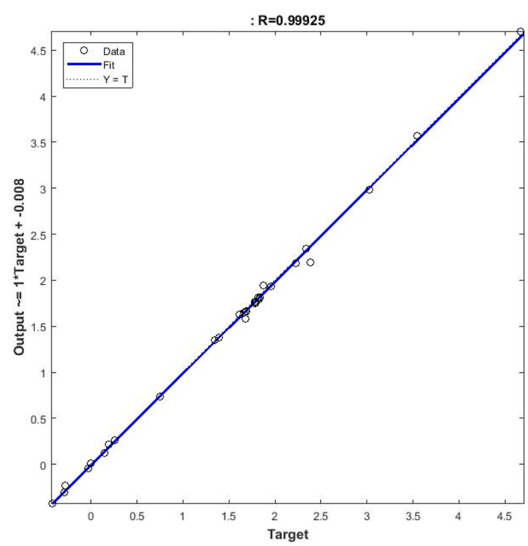

Figure 3.14: Actual output Vs Estimated output

Observation: We observe how error decreases as we increase degree for training data, but since error becomes almost close to zero which means we have accurately predicted best model for our data, hence error becomes zero for unseen validation data also.

3.4 Ridge Regression on 2-dimensional data

Now we will perform ridge regression, We will try out different models with different order of polynomial degree and different $\mathbf{\Lambda}$ to find out which gives least error on validation data.

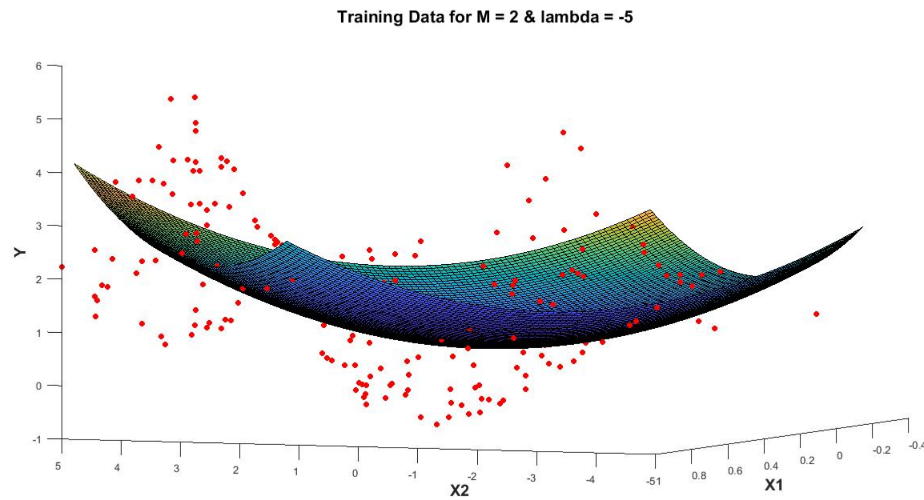

Figure 3.15: Multi Dimension Training Data with polynomial of degree 2 and lambda e-̂5

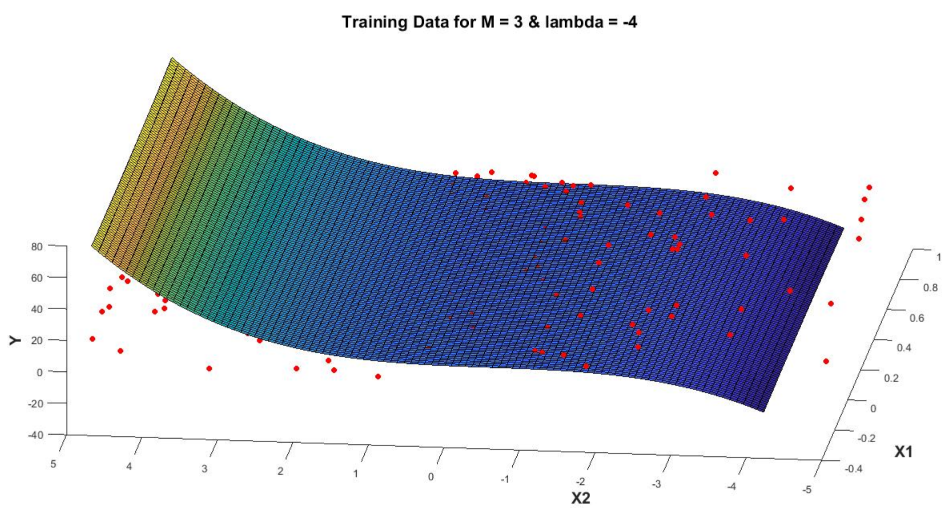

Figure 3.16: Multi Dimension Training Data with polynomial of degree 3 and lambda e-̂4

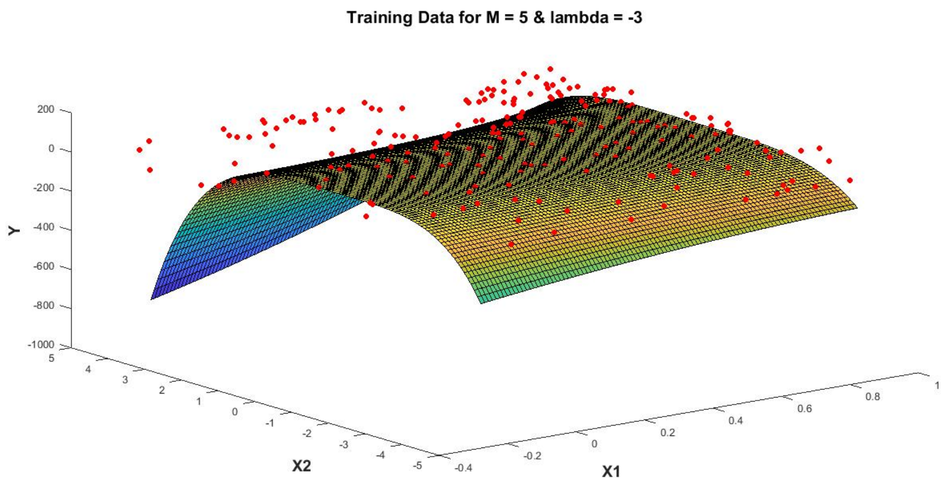

Figure 3.17: Multi Dimension Training Data with polynomial of degree 5 and lambda e$\hat{-}$3

Now, we will try out with different values of λ on few models to find which lambda is best for our data. Below is graph of SSE Vs Lambda for M = 5 & 7.

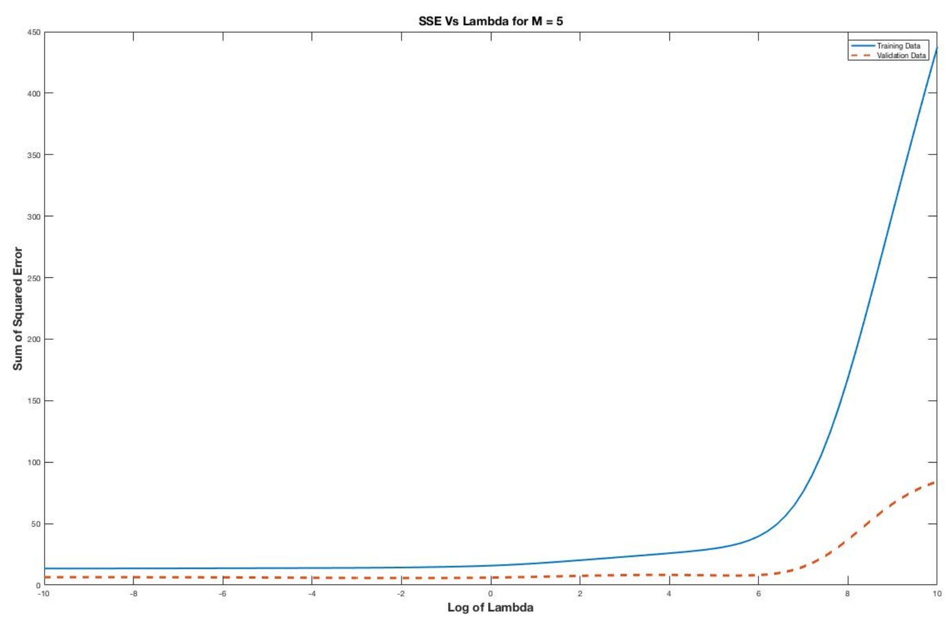

Figure 3.18: SSE Vs Lambda for M = 5 on training and validation data

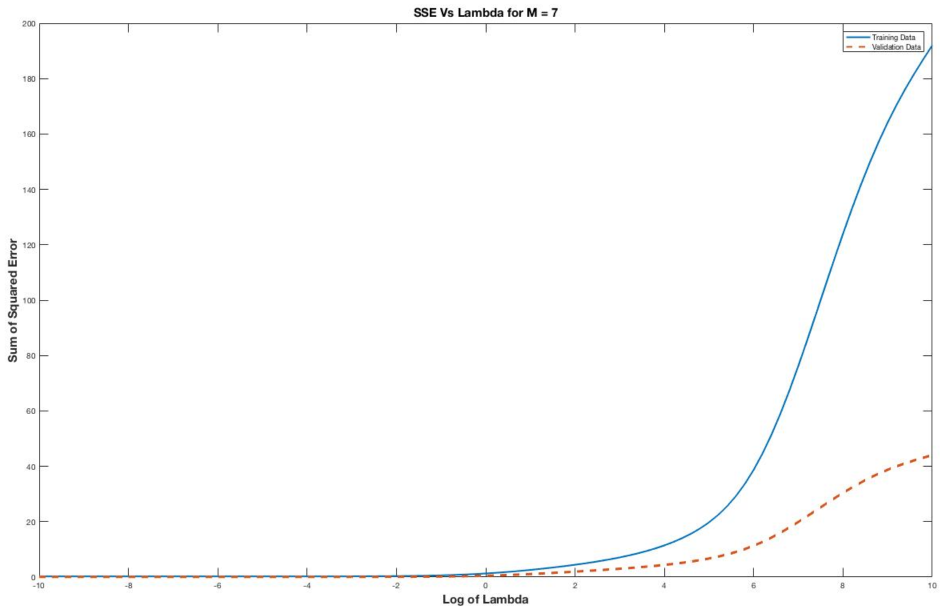

Figure 3.19: SSE Vs Lambda for M = 7 on training and validation data

We observe that as value of λ increases, error increases as we are making coefficeints of parameter θ small and eventually it stabilizes and follows straight line. For higher order polynomial also the error is same and regulariation works with few fluctuations. Thus we select lambda = e$\hat{-}$8 as error is low and it stabilies around that.

Below figure shows SSE Vs Degree for testing and validation data, we can observe error is low for lower order of polynomial and error increasing then stabilises for higher degree.

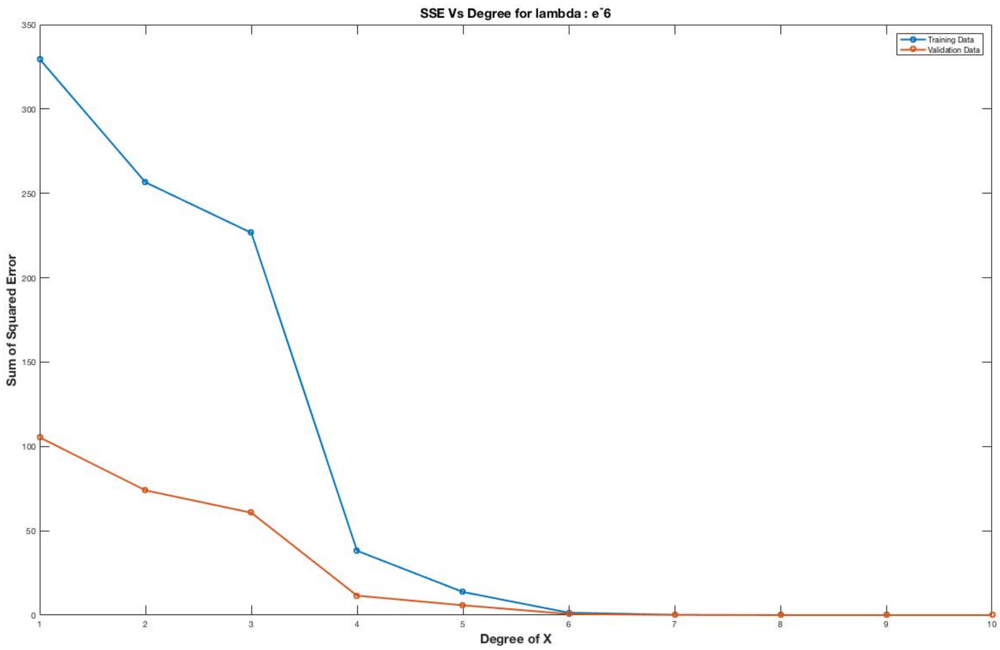

Figure 3.20: SSE Vs Degree for Multi dimension ridge training and validation data for lambda = eˆ-6 & M = 7

Now from above graph observation, we will select degree lambda = eˆ-8 and M as 7 since it is stable around that order for validation data. Then we will use parameter θ for degree 7 which we calculated from training data on testing data to estimate output.

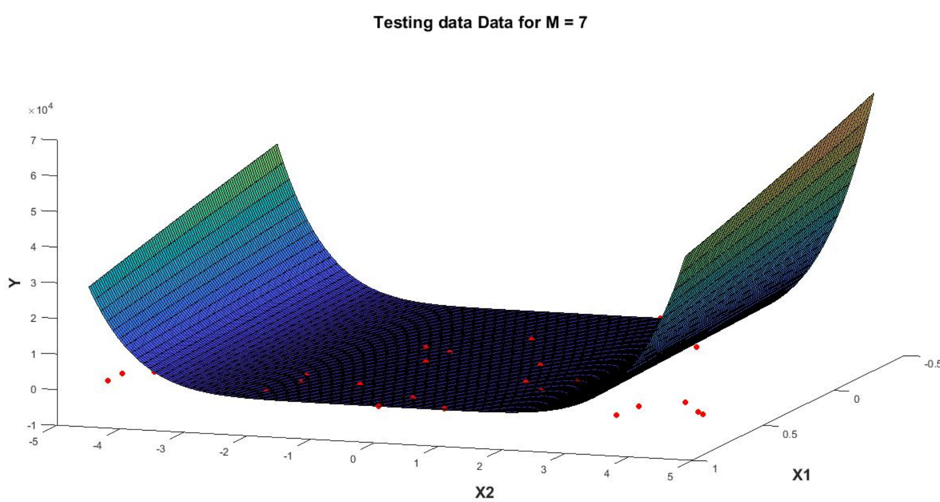

Figure 3.21: Plotting polynomial for M = 7 and lambda = eˆ-6 on testing data

And below is graph of Actual output Vs Estimated output for selected parameter θ and degree M = 7 and lambda = eˆ-8.

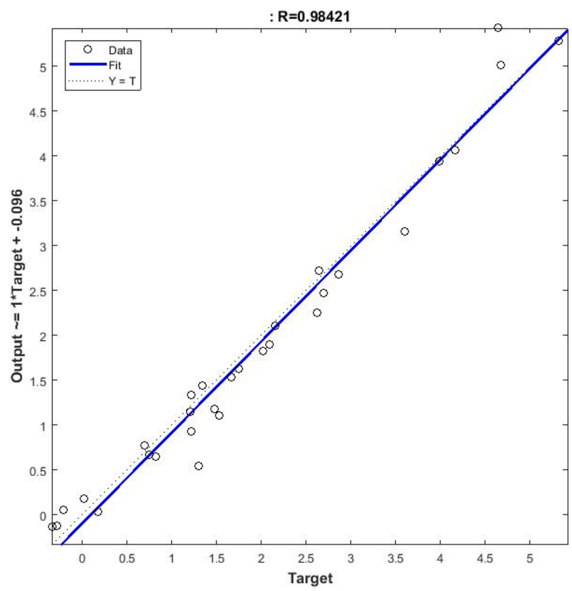

Figure 3.22: Actual output Vs Estimated output

Observation: We observe how error decreases as we increase degree for lambda = eˆ-6 training data, but since error becomes almost close to zero which means we have accurately predicted best model for our data, hence error becomes zero for unseen validation data also.

3.5 Multidimensional data

Similarly, we will perform same experiment for Multi-dimensional data. First we will calculate parameter θ for different order of polynomial from training data and use those parameter on validation data to find out best model i.e order of polynomial which gives least error.

Before performing experiment we observed data and found that two columns of data are same, which will not add any more information to our model hence we didn't consider that column for calculating parameter θ

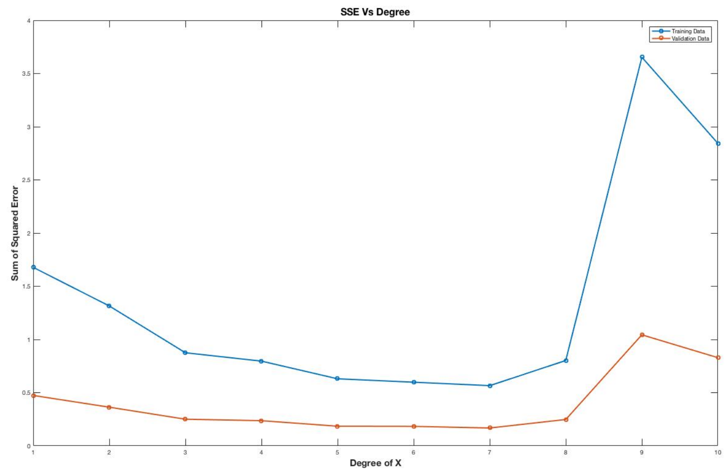

Figure 3.23: SSE Vs Degree for Multi-Dimension on training and validation data

From the above graph we observe that it gives least error for M = 7. Hence we will select M as 7 as use our calculated parameter θ from training data on test data and plot Actual output Vs Estimated output graph which show how good/bad our model works on unseen test data.

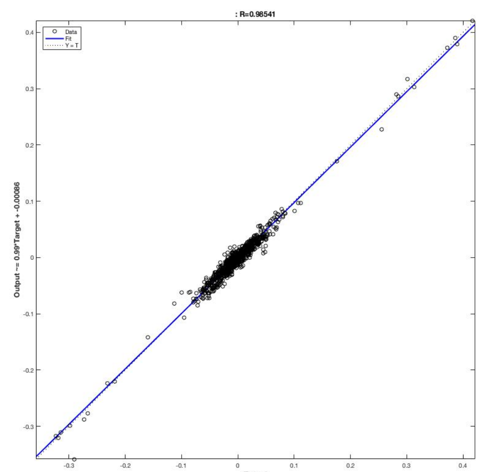

Figure 3.24: Actual output Vs Estimated output

Observation: We observe how error decreases as we increase degree for training data but error increases for few higher order then it again decreases and we observe similar curve for validation data also but with far less error, which means our prediction is close to accurate.

3.6 Ridge Regression on Multidimensional data

Now we will perform ridge regression, We will try out different models with different order of polynomial degree and different $\mathbf{\Lambda}$ to find out which gives least error on validation data.

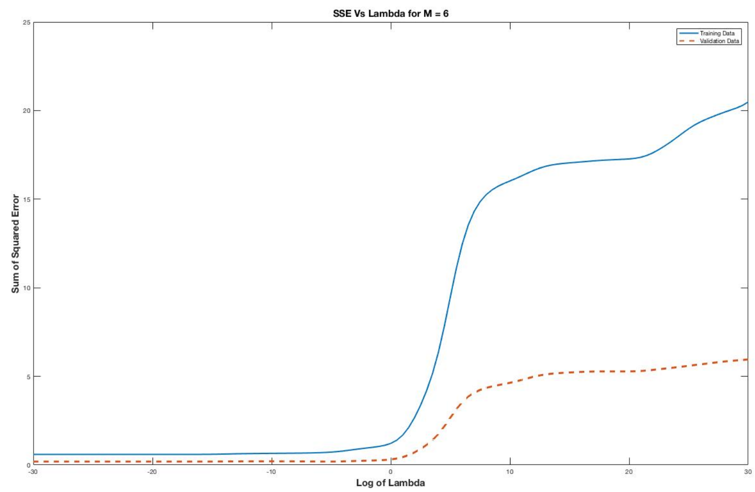

Figure 3.25: SSE Vs Lambda for M = 6 on training and validation data

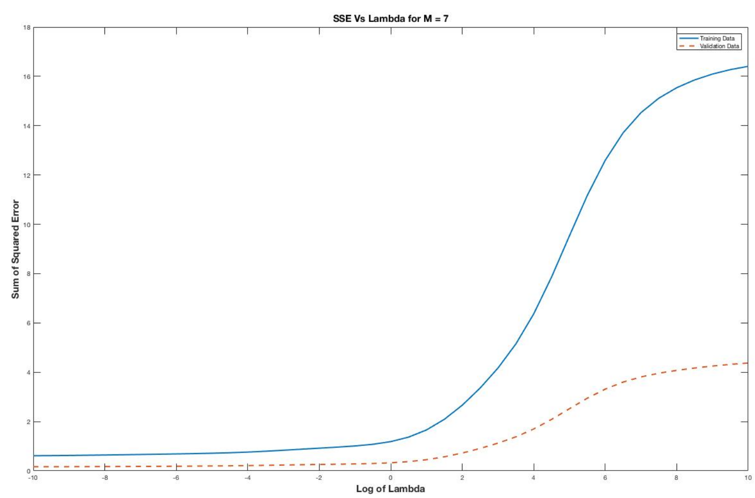

Figure 3.26: SSE Vs Lambda for M = 7 on training and validation data

We observe that as value of λ increases, error increases as we are making coefficeints of parameter θ small and eventually it stabilizes and follows straight line. For higher order polynomial also the error is same and regulariation works with few fluctuations. And we can observe that validation error is less compared to training error as we have not allowed data to overfit on training data. Thus we select lambda = eˆ8 as it error is low and it stabilies around that.

Below figure shows SSE Vs Degree for testing and validation data, we can observe error is low for lower order of polynomial and error increasing then stabilises for higher degree and since we have used regulariation the error is high for training data but less for validation data as we have reduced coefficients of higher order through regulariation.

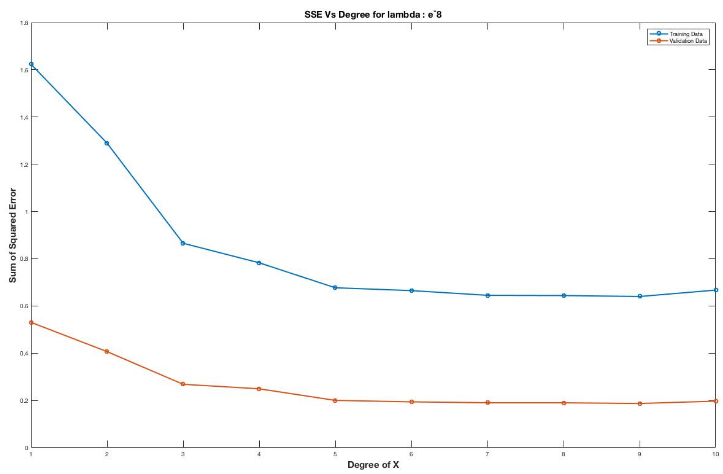

Figure 3.27: SSE Vs Degree for 1D ridge training and validation data for lambda = eˆ8 & M = 7

Now from above graph observation, we will select degree lambda = eˆ10 and M as 7 since it is stable around that order for validation data. Then we will use parameter θ for degree 5 which we calculated from training data on testing data to estimate output and plot graph of Actual output Vs Estimated output.

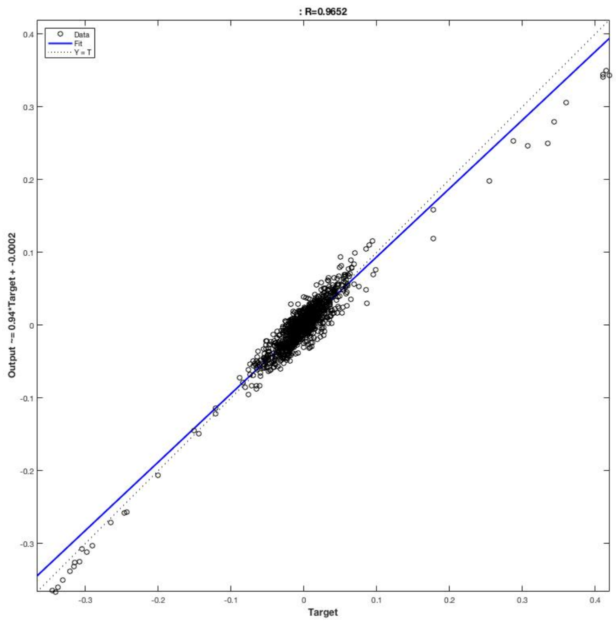

Figure 3.28: Actual output Vs Estimated output

Observation: We observe how error decreases as we increase degree for training data but error does increases for few higher order as we have use regularisation with lambda = eˆ8 which limits coefficients to take large value and we observe similar curve for validation data also but with far less error, which means our prediction is close to accurate.