Name: PRATHMESH R. KUMBHARE

CS6001 Assignment 5: Regression

REPORT

A. OUTLINE FOR PROGRAM ALGORITHM:

The algorithm for implementing the program is given below in steps:

1. Matlab script files are created for each model separately. The file executes the desired model & saves all the workspace variables for corresponding model.

2. Factor which can be varied to get different results for a model is: lambda (only for Bayesian linear, non-linear & dual linear models.)

3. The program loops through all the training images and creates a matrix of column vectors named 'train_image_matrix' with first number in each vector being 1. Mean and covariance for the images is calculated. The program also creates a similar matrix for all the testing images by the name 'test_image_matrix'.

4. If feature selection model is being computed, then the program checks for all the zero rows in the covariance matrix 'covar_train'. This implies that the program selects all the data points in the image which have zero covariance i.e. they do not vary with respect to each other. These rows are then removed from the image matrix which results in reduction of computation for next steps.

5. If Bayesian regression (regularization) is being computed, then the program computes and adds new term (regularizing term) for the calculation for 'phi_hat'. Initially the term 'phi_hat' is obtained (by method explained in point 6). Then, variance ('sig') is computed from it. By selecting the value of 'lambda', the regularizing term is added and finally the new regularized phi ('phi_hat_reg') is obtained.

6. **Computation of Inverse:** The computation of 'phi_hat' requires determination of the term: inverse of 'X*transpose(X)'. For the calculation of inverse, the matrix must be positive definite or it must be full rank. MATLAB does not find this term accurately. To solve this problem, a function file 'near_positive_definite' is created which takes a square non-positive definite matrix as input argument and returns a square positive definite matrix as output.

The image matrix created as mentioned above is passed through this function. Inverse of this near PD matrix is computed and then finally ‘phi_hat’ term is obtained.

7. **Inference Algorithm:** An estimate of world states of testing images is computed (‘w_test_hat’) using the phi obtained as explained in the earlier points. The ground truth for testing images is computed (‘w_test_gt’) by dividing values the given world states (‘w_test’) by ‘3.14*pi’. Estimated rotation values for testing values (‘w_test_hat_rot’) are computed similarly by dividing each value in the ‘w_test_hat’ by ‘3,14*pi’. The variable ‘mod_error’ computes the absolute values of difference between estimated and ground truth values. Finally, the average value of this error is calculated. (‘deviation’)

8. For visualization, a graph showing the ground truth and estimated rotation values is plotted. All the variables in the workspace are saved in a ‘. mat’ file by the appropriated model name.

9. **Info on file ‘nearestPSD’:** For the computation of inverse of matrix, it must be positive definite. Calculating inverse of matrix directly by ‘inv’ or ‘\’ operator results in error: The matrix is not positive definite. I found this function file created by some other person on the Mathworks website. In order to avoid this problem, I am using this function file as I was unable to solve this issue any other way.

B. RESULTS:

This is the table for all the models implemented with all the important values:

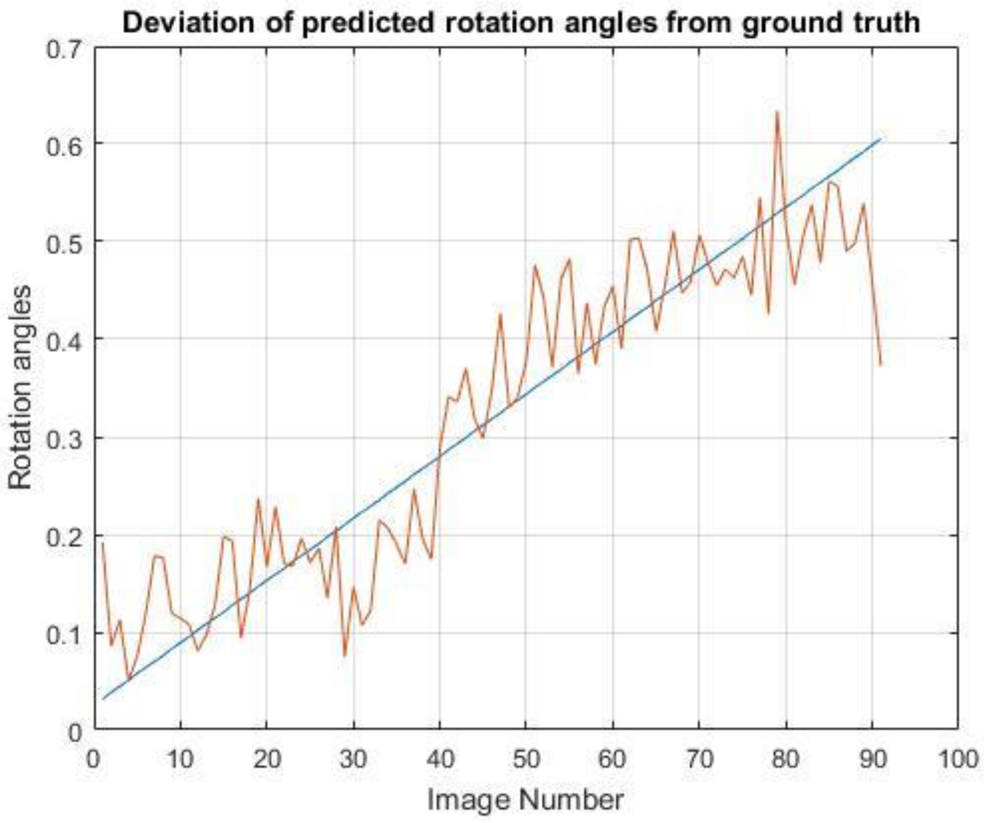
	MODEL NAME	LAMBDA	FILE RUN TIME (SEC)	DEVIATION
1	Linear regression	None	1461.96	0.0501
2	Linear Regression (Feature Selection 1)	None	1213.28	0.0502
	Linear Regression (Feature Selection 2)	None	30.1563	0.0608
3	Bayesian Linear Regression	200	1519.875	0.0480
	Bayesian Linear Regression	0.001	1519.875	0.0501
	Bayesian Linear Regression	5000	1519.875	0.0548
4	Bayesian Linear Regression (Feature Selection 1)	200	1227.82	0.0480
	Bayesian Linear Regression (Feature Selection 1)	0.0001	1227.82	0.0501
	Bayesian Linear Regression (Feature Selection 1)	10000	1227.82	0.0570
	Bayesian Linear Regression (Feature Selection 2)	200	29.8594	0.0552
5	Non-linear regression (X^2)	200	211.953	0.0505
6	Non-linear regression (X^2 & X^3)	200	577.156	3.2973
7	Non-linear regression (Radial Basis)	200	172.0625	0.0507
8	Non-linear regression (Arc Tan)	200	209.968	0.0508
9	Dual linear regression	200	15.4375	0.0480
10	Dual linear regression (Gradient)	200	126.1406	0.0085

Results & description for each model are given below separately. It should be noted that for majority of file execution time is taken for calculating covariance matrix and near PD matrix.

1. Linear Regression Model:

This model uses gray scale image (101*101) as input for estimating ‘phi_hat’. Average deviation of computed values from ground truth ones is calculated in the inference part. Below is the graph showing deviation for each testing image.

Deviation plot:



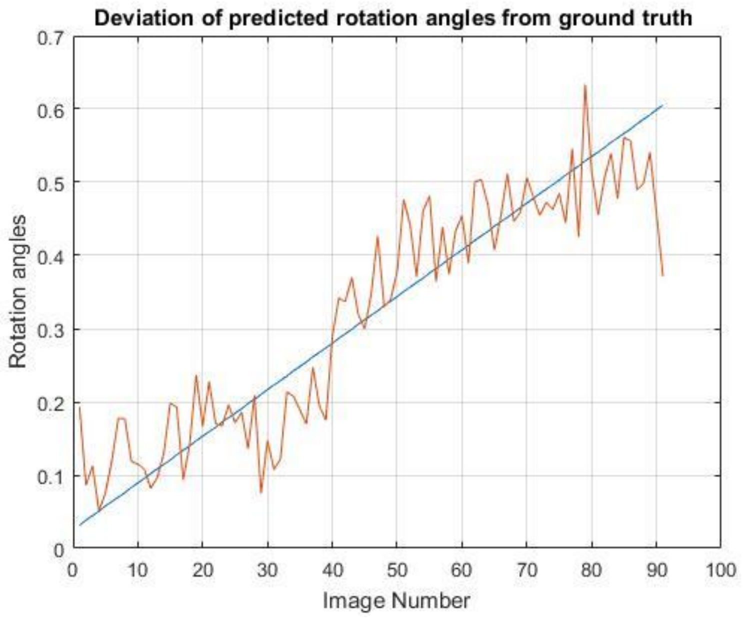
2. Linear regression with feature selection:
    - Model 1:

This model uses gray scale image (101*101) to form image matrix first. Then, by inspecting covariance matrix, it deletes the 'zero variation' data points from the image and creates new image matrix. This matrix is used for computation of 'phi_hat' and finally average deviation. By looking at the results for the model, we get 602 such features. These are not used in the computation. Hence for the computation of the inverse of matrix 'X*transpose(X)', the dimensions are 9600*9600 instead of 10202*10202. Below is image of one actual image & its corresponding 'feature deleted' image (zero values at its deleted feature points). Also, average deviation plot is shown.

Testing image & its 'feature deleted' image:

(Actual rotation for this image is 0.038 while its computed estimate is 0.086)

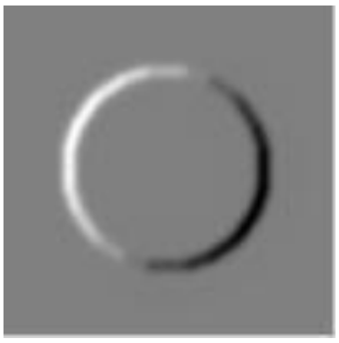
Deviation plot:

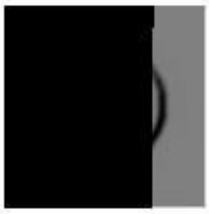


- Model 2:

The working flow of this algorithm is very similar to the above mentioned Model 1 with only one difference in the computation of feature selection points. In the earlier model we remove the ‘zero varying’ data points from the image while in this model we remove data points which vary between: 0-1. By observing the variance between the pixels, we can clearly see that majority of the variance values are near 0. (Like 0.16) These pixels do not contribute significantly in ‘providing information’. By making this slight modification, we obtain number of removable feature points equal to 7486. Hence for the computation of the inverse of matrix ‘X*transpose(X)’, the dimensions are 2716*2716. Below is image of one actual image & its corresponding ‘feature deleted’ image. Also, average deviation plot is shown.

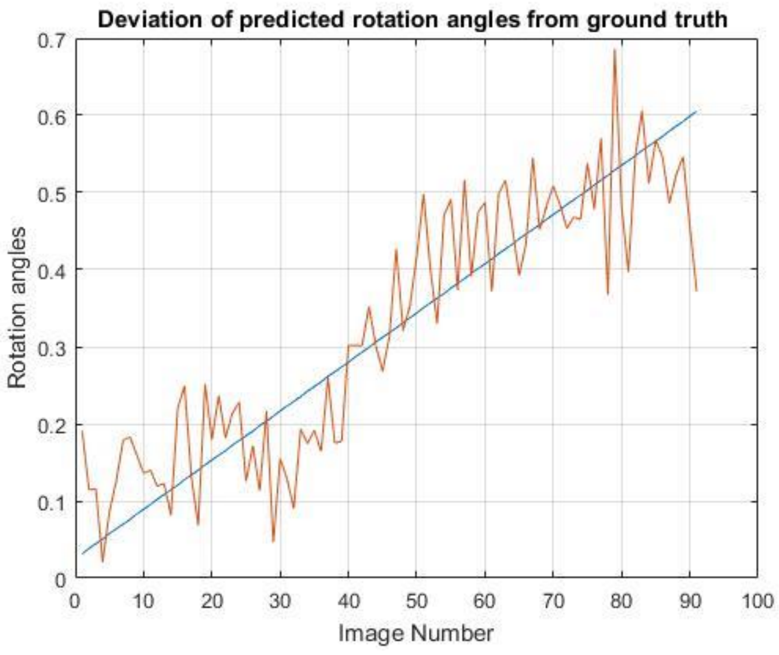
Testing image & its ‘feature deleted’ image:





(Actual rotation for this image is 0.0385 while its computed estimate is 0.1146)

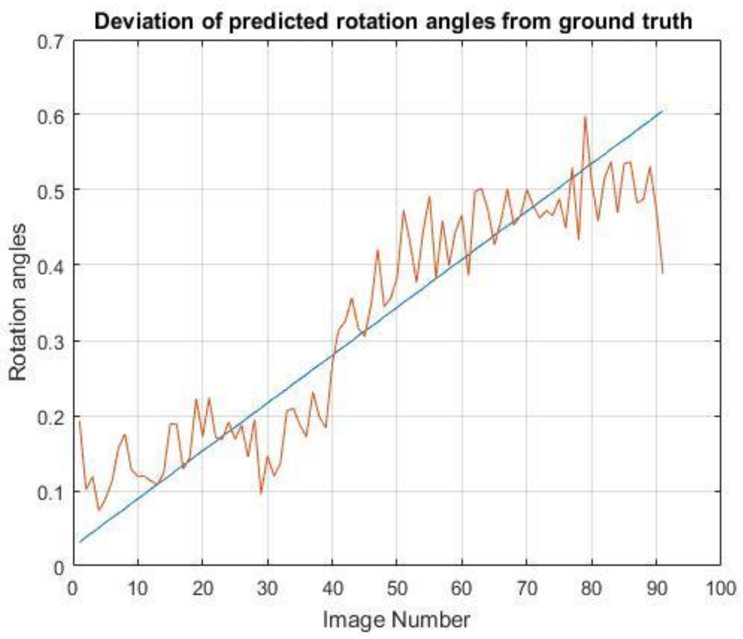
Deviation plot:



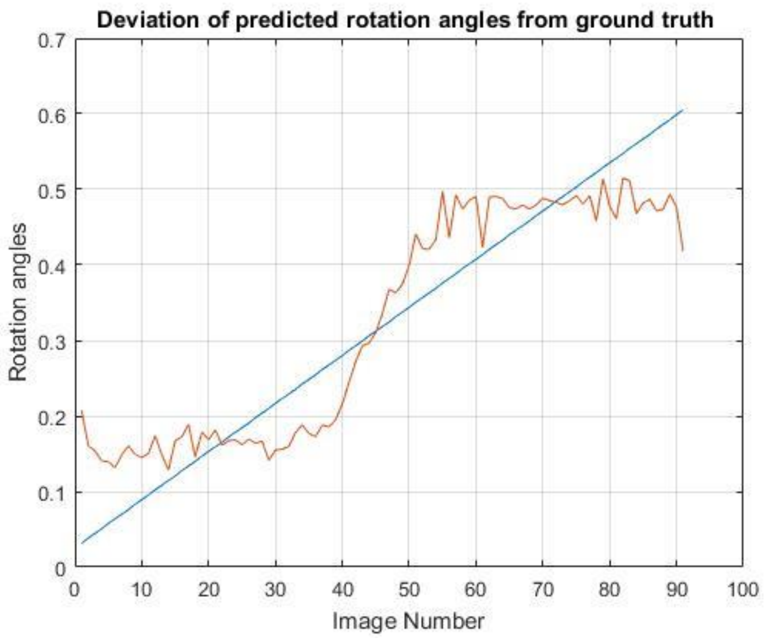
3. Bayesian Regression model:

Like linear regression, this model uses gray scale image (101*101) as input for estimating 'phi_hat'. Then an additional regularizing term 'phi_hat_reg' is calculated using variance of the data. This term is used to predict world state of testing images. Finally, average deviation is computed. The value of lambda can be varied. By trial and error approach, best value for lambda obtained is 200. Plots for deviation at lambda values of 200 & 5000 are shown below. Also, a plot for difference between 'phi_hat' (non-regularized) & 'phi_hat_reg' (regularized) for lambda value of 200 is shown.

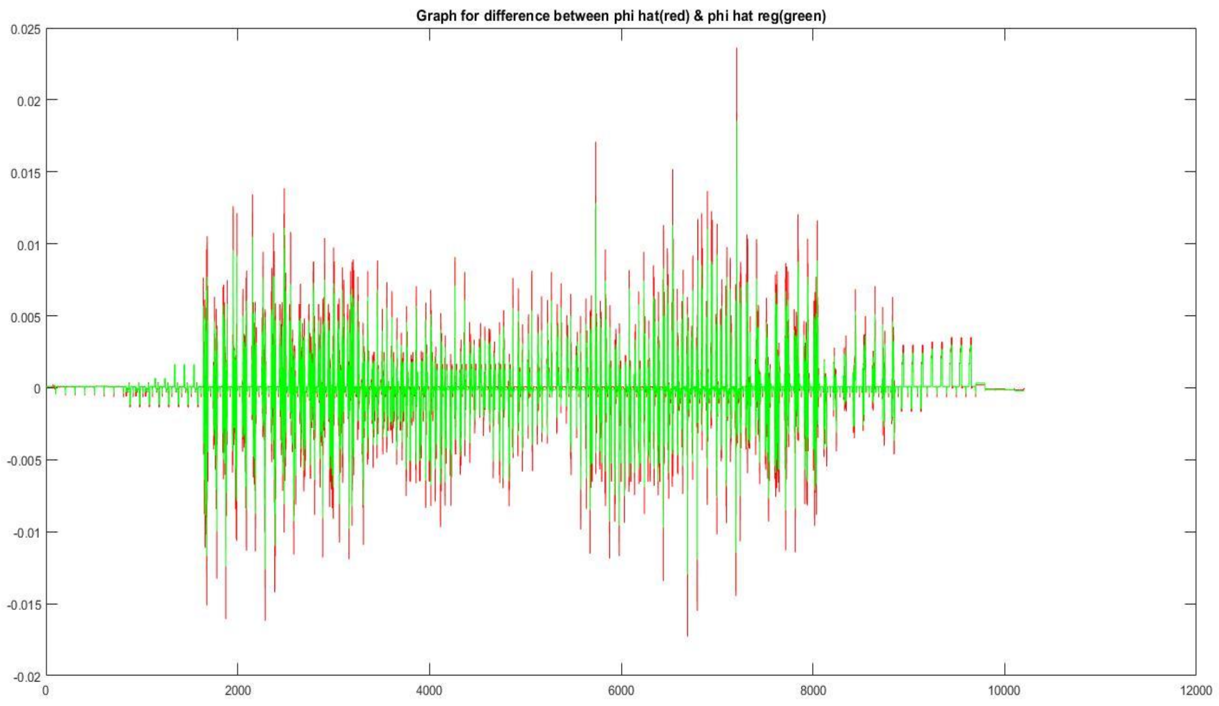
Deviation plot for lambda = 200:



Deviation plot for lambda = 5000:



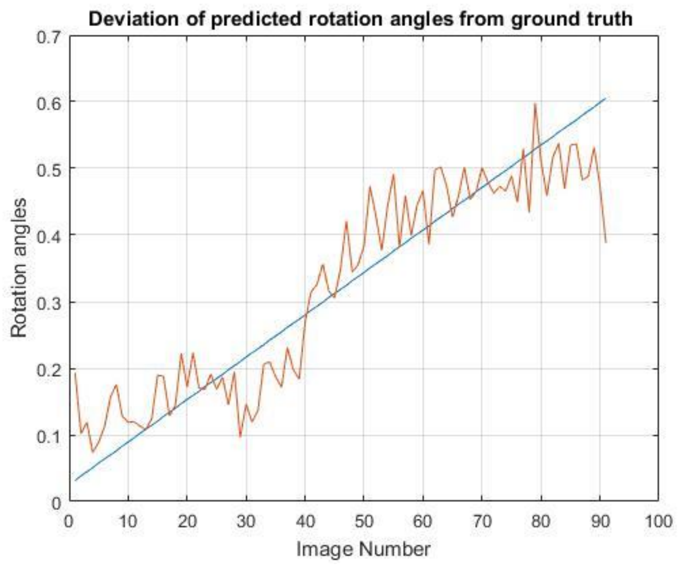
Estimated phi (Non-regularized & regularized) difference:



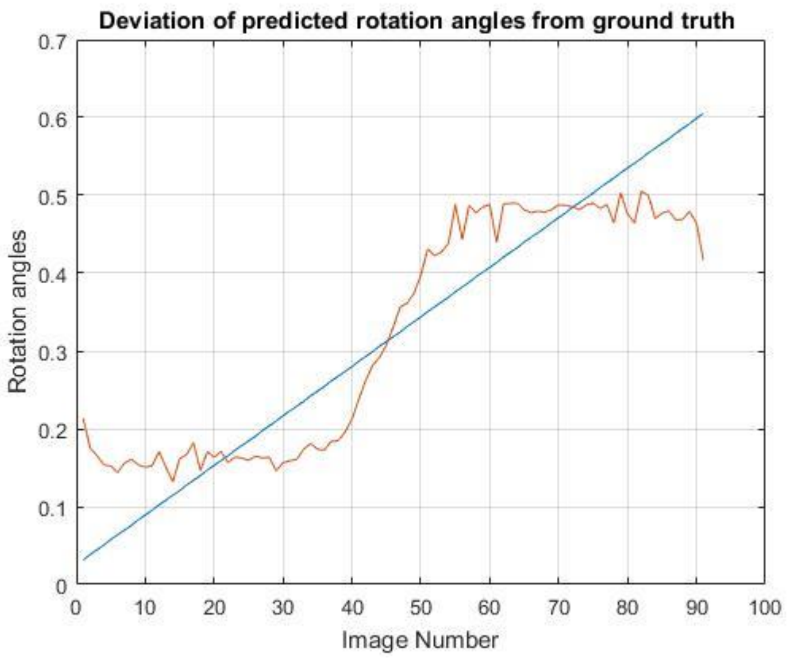
4. Bayesian regression with feature selection:
    - Model 1:

Like the linear feature selection, this model uses gray scale image (101*101) to form image matrix first. Then, by inspecting covariance matrix, it deletes the 'zero variation' data points from the image and creates new image matrix. This matrix is used for computation of 'phi_hat'. Then an additional regularizing term 'phi_hat_reg' is calculated using variance of the data. This term is used to predict rotation of the test images. Finally, average deviation is obtained. Value of lambda can be varied to get different results. Best result is seen at value of lambda = 200. Note that these results are very close to earlier model (Bayesian regression model), but less computationally intensive.

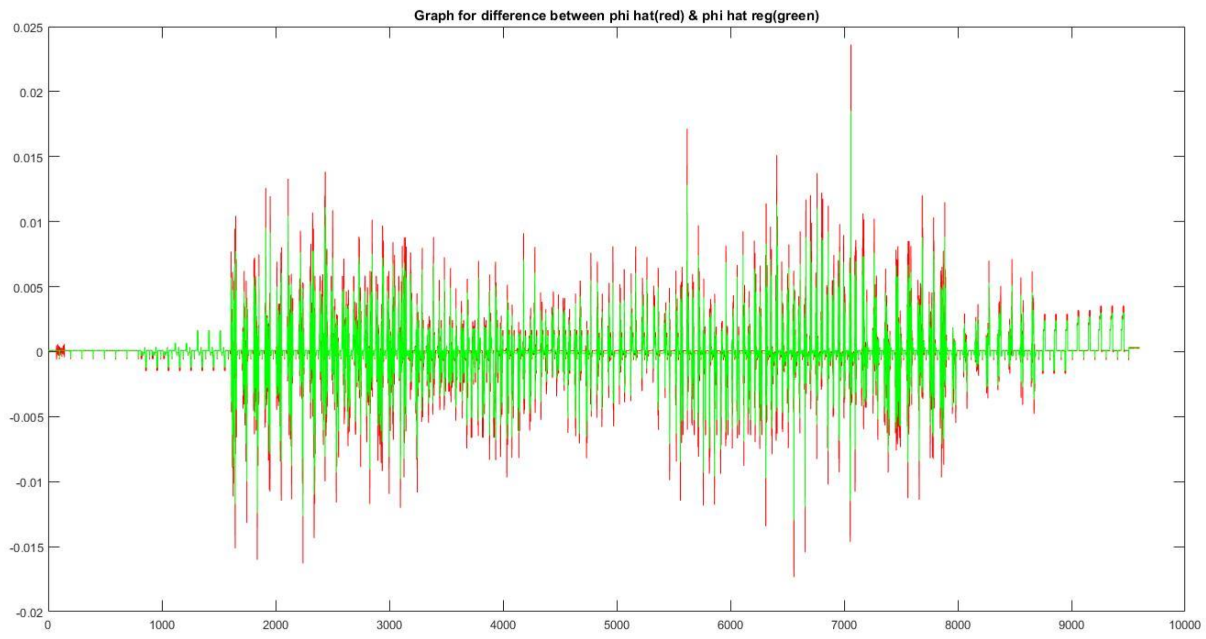
Deviation plot for lambda = 200:



Deviation plot for lambda = 10000:



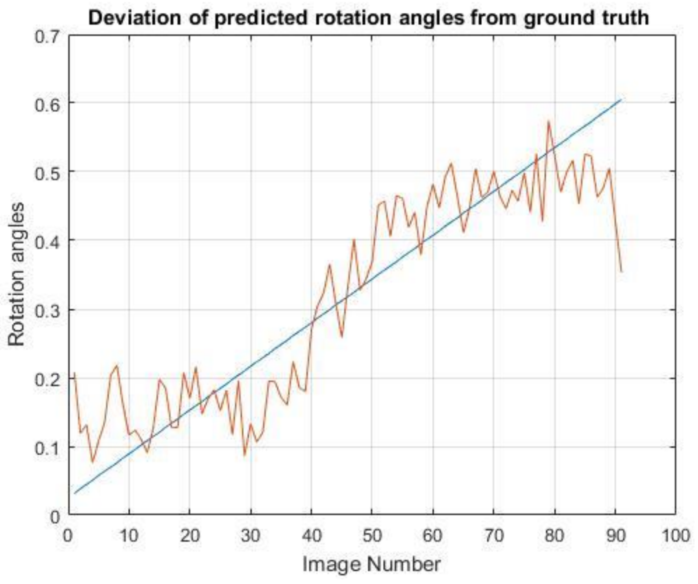
Estimated phi (Non-regularized & regularized) difference:



- Model 2:

The algorithm is very similar to the above mentioned Model 1 with only one difference in the computation of feature selection points. The feature selection is done by deleting pixels of the image having variance between: 0-1. (Feature selection process is identical to the one used in linear regression with feature selection model 2) Number of such feature points obtained are 7486. Therefore, for the computation of the inverse of matrix ‘X*transpose(X)’, the dimensions are 2716*2716.

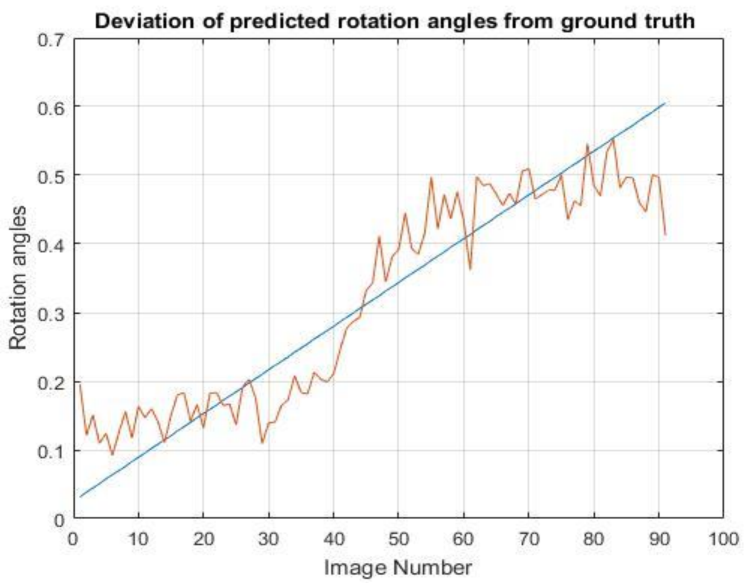
Deviation plot for lambda = 200:



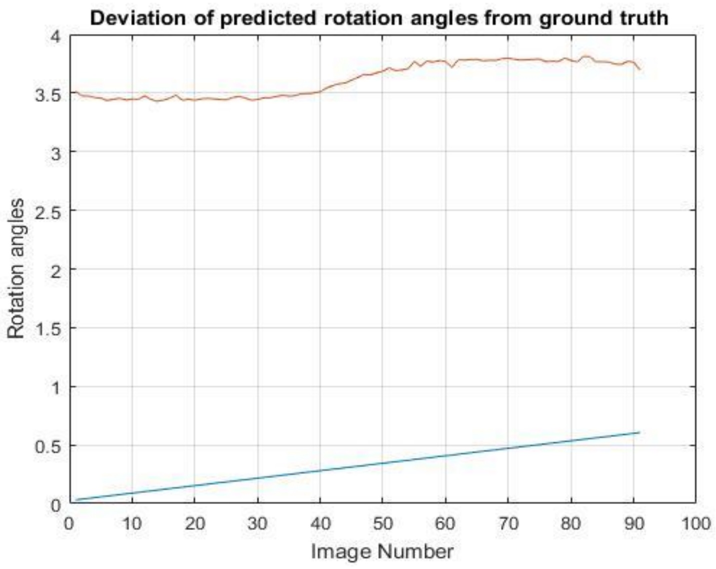
5. Non-linear regression model:

For the image matrix formation, each image is scaled down by factor of 0.5 and then converted into vector. As we are going to append non-linearized values of the image to original image vector, the size of the matrices increases drastically. To keep computation time in limits, we scale down image. Number of 'zero varying' data points after scaling are 106. So, the size of matrix 'X*transpose(X)' is 4992*4992 for only one non-linear function appended. (example: [x; x^2]) Different non-linear functions are appended to the image vector. Deviation plots for each of these models at lambda=200 are given below.

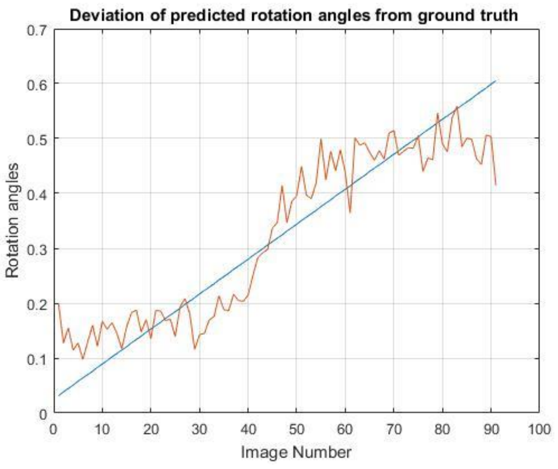
Deviation plot for [x; x^2] model:



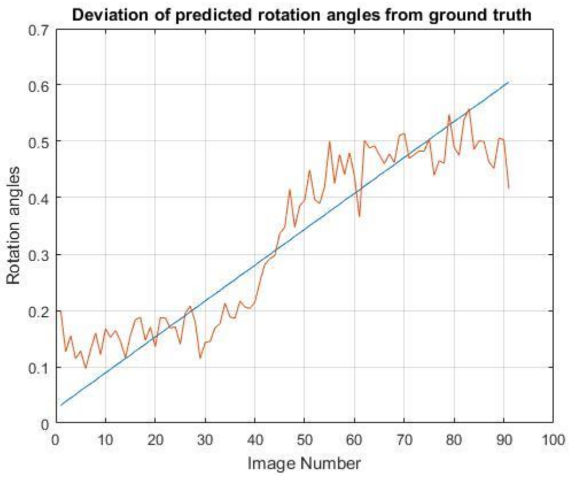
Deviation plot for [x; x^2; x^3] model:



Deviation plot for [x; radbas(x)] model:



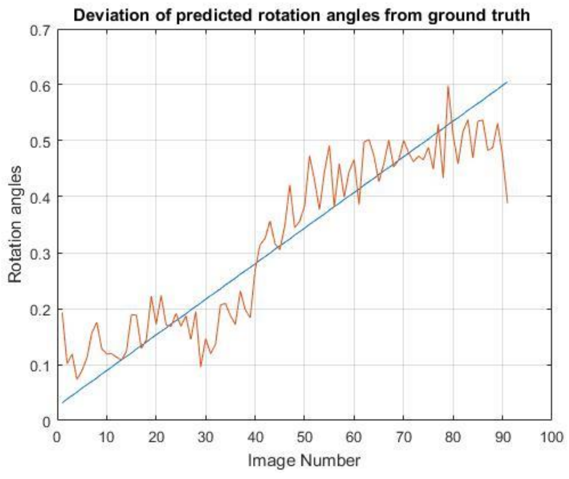
Deviation plot for [x; arctan(x)] model:



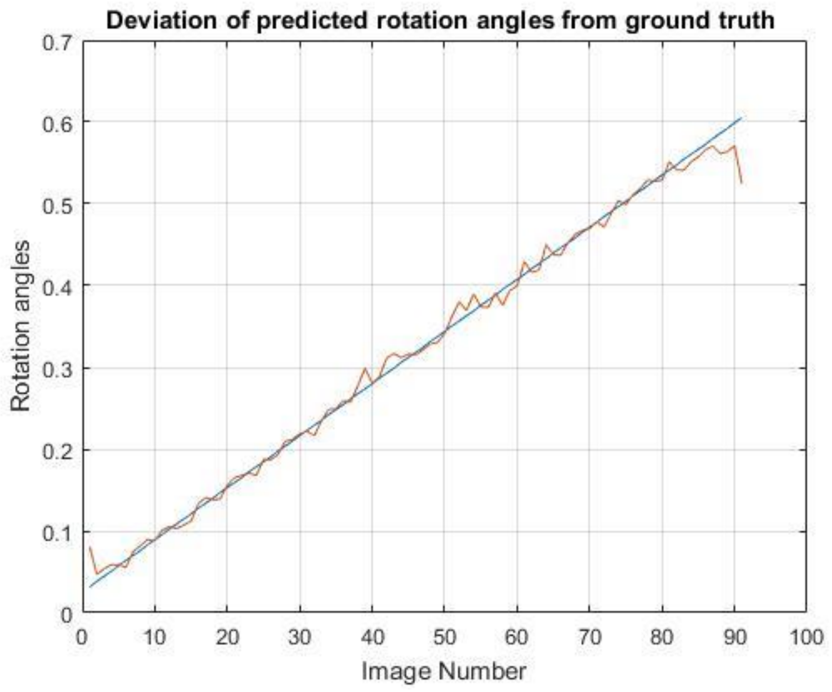
6. Dual linear regression model:

This model uses gray scale image (101*101) to form image matrix first. For gradient model, each image is passed through the function ‘img_grad’, which produces magnitude and direction of the gradient of image points. Then, a matrix of image vector columns is formed for further calculation. Like Bayesian linear regression model, it computes ‘phi_hat’ & ‘phi_hat_reg’. One very important point to note in this method that it computes ‘transpose(X)*X’ instead of ‘X*transpose(X)’. So, its matrix size is 90*90 instead of 20403*20403 for this gradient vector. Deviation plot at lambda value equal to 200 is shown below. Also, one training image is shown after it is converted in magnitude and direction form. Sobel operator is used to achieve this gradient image.

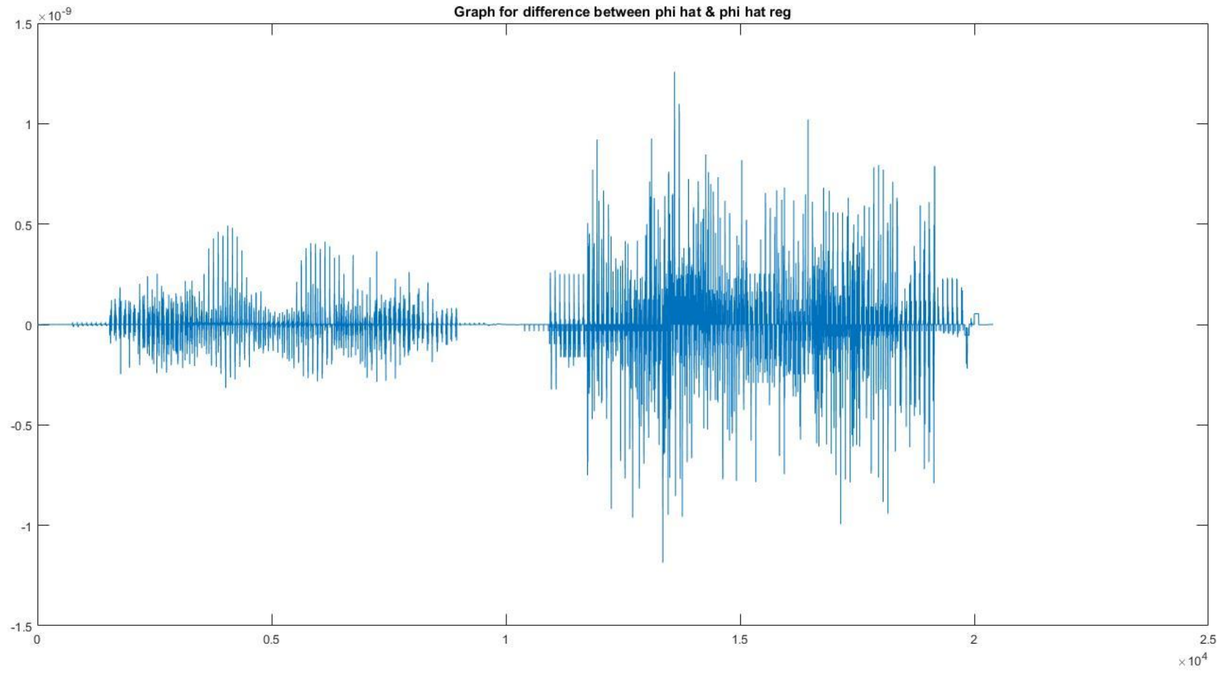
Deviation plot (Gray scale image model):



Deviation plot (Gradient of gray scale image model):

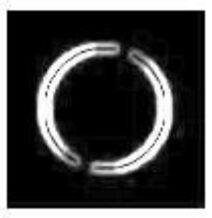


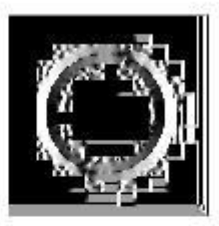
Plot for phi difference:



Images for magnitude & direction of gradient:

- Magnitude (left) and Direction (Right) of gradient.





C. General Observation & Comments:

1. After carefully looking at table of model readings at the beginning, it is clear that determining the suitable value of lambda will result in more accurate result.
2. One factor of major concern is the time taken for execution of the program. Another possible way to reduce this time might be to scale down the image by a factor of 2 or 4 to reduce computation time. (This approach was used in the non-linear regression model. Considerable reduction in execution time is achieved with negligible difference in the results.)
3. Feature selection helps in reduction of file execution time with almost same results as that of original ones. Results for linear and Bayesian linear model with feature selection 2 shows drastic reduction in run time with almost similar values in deviation.
4. Non-linear approach gives almost same results as linear and Bayesian regression models.
5. Dual linear regression approach reduces the computation time significantly while giving accurate results. By using the gradient of image (magnitude and direction) as input to this model, most accurate results (almost better by 7 times than the results given by other models) are achieved. It can be clearly seen from the results that this combination model is the best choice for accuracy and is very much within limits of computation time.

D. Summary and Concluding Comments:

1. The aim of this assignment was to predict the rotation angle of the given image. By choosing the best model as mentioned above (Dual linear regression with image gradient as input) we can conclude that the algorithm can correctly predict the angles with satisfactory accuracy.
2. The dual linear regression model can be applied in HSV or YCbCr color spaces to see the results.
3. As explained in the observation section, appropriate image scaling down factor can be determined in order to reduce the computation time. Given the nature of images, scaling down possibly won't result in significant 'loss' of important data.