# CMSC 426: Computer Vision

# Homework-1: Linear Least Squares

## 1. Plot Eigen

First of all, the data.mat files are loaded in MATLAB and converted into 2 row vectors (x, y) for convenience. Based on the matrix size of x and y, their dimensions are 200 x 1. These points are then plotted for better visualization. Thereafter, the Covariance Matrix, Eigenvector and Eigenvalue are calculated.

Using the formula mentioned in the theory (links provided in the question), the Covariance Matrix is calculated. After that, using the built-in ‘eig’ function, the Eigenvalue (D) and Eigenvector (V) are computed.

For calculating the maximum eigenvalue (to plot the major axis of the ellipse), the maximum value of the eigenvalue is calculated using ‘max’ function twice. The inner 'max' gives the maximum column of the matrix and outer 'max' gives the maximum from that column.

Thereafter, the 'find' returns the row and column subscripts of each nonzero element in the given array using any of the input arguments in previous syntaxes.

Taking the maximum column value from the eigenvector using the value of UV, calculating the direction of the vector to be plotted. Scaling the Magnitude of Vector for better visualization of the axes and then the values of xrangemax and yrangemax are plotted. This gives the major axis of the ellipse.

The same process is carried out for finding the minimum eigenvalue for plotting the minor axis of the ellipse.

## 2. Least Square

### a. Linear Least Square

First of all, the data.mat files are loaded in MATLAB and converted into 2 row vectors (x, y) for convenience.

Based on the matrix size of x and y, their dimensions are 200 x 1. These points are then plotted for better visualization. The transpose of x and y is found and converting them into column vectors. For ease of calculation and to match the matrix dimensions, the column matrix is converted into 200x2 with first column of ones only

Finally, using the formula for calculating the 'theta' parameter as mentioned in the theory (links provided in the question), the line is finally plotted over the scatter plot.

### b. Outlier Rejection using Regularization

This methodology is similar to Linear Least Square with the exception of an additional Lambda factor. Thus, upon varying the value of Lambda, varies the line on the plot. This method is fast compared to RANSAC.

To get a sense of the effect of Lambda, the value of Lambda has been changed from 10 – 100 – 1000. However, the disadvantage is that the output depends upon the Lambda value and it gets influenced by the outlier as compared to RANSAC.

### c. Outlier Rejection using RANSAC

This methodology is different than Regularization in a sense that it randomly selects the points in every iteration and tries to accumulate as many inliers as possible and avoid the effect of outliers. The inliers are decided based on a threshold value. This value is decided using the trial and error method depending upon the distance of the points from the iterative line. However, the limitation is that it is completely random. So, there is no assurance of the number of iterations or the time taken to fetch the result.

First of all, using the 'randi' function, any random point out of the 200 points are selected to plot a line and to determine the inliers and outliers. The equations used are from the theory (links provided in the question) which includes calculating the variables of the line equation and perpendicular distance of the points from the line.

As and when points having values less than the threshold are found, they are used further for plotting of the RANSAC line.

References

1. https://cmsc426.github.io/math-tutorial/

2. http://www.visiondummy.com/2014/04/geometric-interpretation-covariance-matrix/

3. https://en.wikipedia.org/wiki/Random_sample_consensus

Dipam Patel
UID: 115809833