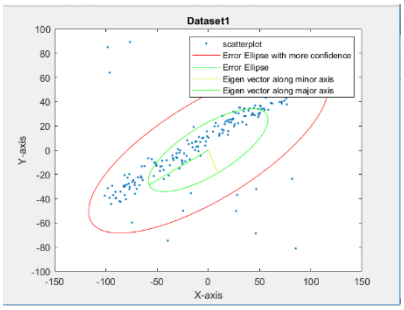
CMSC426 – Homework1

Co-variance matrix is given by $\Sigma = \begin{bmatrix} \sigma(x,x) & \sigma(x,y) \\ \sigma(y,x) & \sigma(y,y) \end{bmatrix}$. Here the first and fourth element gives the variance of x and y respectively. Variance(sigma(x,x)) is the square of standard deviation(sigma(x)). Standard deviation means as the name suggests, deviation of data in a standard way, which we take as mean. So, the standard deviation is a measure of how deviated the data is with respect to its mean. So, variance of a variable x or y is a measure of how data is spread across with respect to its mean. Best example is the Gaussian density function/Bell curve. Higher the standard deviation, higher the variance and thus more spread the data is from its mean => Stretched curve. Also, lower the standard deviation, lower the variance and more data would be centered around the mean value.

But, in the matrix above, we have off-diagonal elements given by sigma(x,y) and sigma(y,x). These elements are called the co-variances. The reason is because we are calculating variance between x and y and not between (x,x) or (y,y). This co-variance tells how the x and y data vary with respect to each other. In a 2-D plane, x is the x-coordinates and y y-coordinates. So, the co-variance tells us how the data is spread diagonally, while the variances of x and y indicate how the data is spread parallel to the x-axis an y-axis respectively. The two terms sigma(x,y) and sigma(y,x) are the same because sigma(x,y)= $E[(x - E(x))(y - E(y))]$, changing x and y would give the same result. If the co-variance(sigma(x,y) or sigma(y,x)) value is positive, it means that if x↑ => y↑ or if x↓ => y↓. If co-variance is negative, the increase/decrease in values of x and y are anti to each other. If x↑ => y↓ and vice versa. Thus, co-variance gives us the spread of data diagonally in 2-D and co-variance matrix gives us the overall spread of data in the 2-D plane.

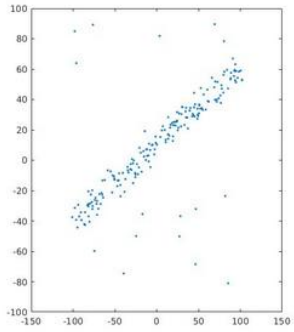
Eigenvectors of any matrix is the set of unit vectors that are orthogonal to each other or called basis vectors. The eigenvectors represent the directions along which the data is spread, and the eigenvalues gives the magnitude of the spreads along these directions. Getting the eigenvectors of the co-variance matrix gives the unit vectors along which the data is spread in other words co-variance matrix represents a linear transformation of any data. If data D is multiplied by a linear transformation matrix T, D'=TD, we have co-variance matrix of D' is TT^T. Taking these eigen vectors as the major and minor axis (eigenvector corresponding to largest eigenvector is considered as major axis and the eigenvector corresponding to next largest eigenvalue as minor axis) of an ellipse, we draw a co-variance ellipse which encompasses the data points. To draw an ellipse, length of the major and minor axis is proportional to the sqrt of respective eigenvalue and is equal to 2*sqrt(eigenvalue*confidence parameter). If we increase the confidence level (try to fit in more data) we will increase the length of these axes, however, we need to keep in mind that the ellipses' area will increase leading to more data points falling within the ellipse but the amount of useful information inside would decrease.



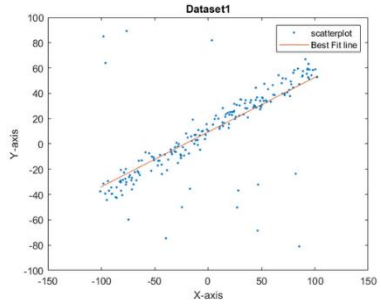
This Figure shows an example of different confidence levels. The Green ellipse is the one with length of major and minor axes equal to sqrt(eigenvalues). Whereas the red ellipse is the one with greater lengths of these axes (2*sqrt(eigenvalues)) and thus occupy more area, confidence parameter=1 => <85% confidence level.

NOTE: When it comes to fitting a line to a sparse data, there is always a tradeoff between quantity(fit max no of points) and quality (how many points are on the line).

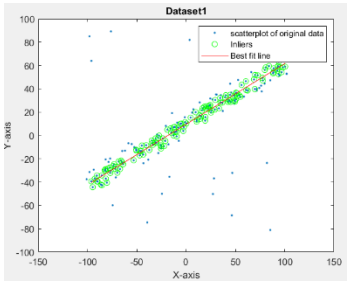
Dataset 1



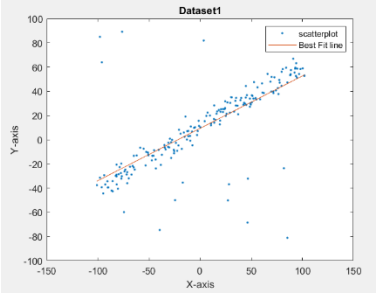
The adjacent figure shows the scatter plot of dataset1.



This is the Linear Least square implementation to get the best fit line.



This is the RANSAC implementation to get the best fit line. Threshold distance is 6 units and min no. of points are 140.

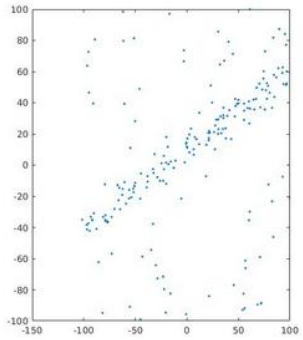


This is the Regularization implementation with penalty matrix given by

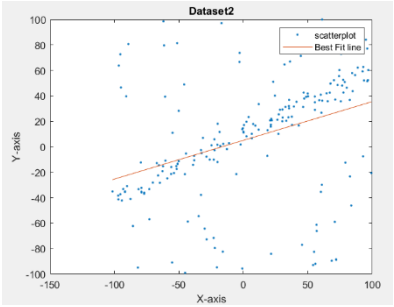
```
lamda = [0.0000000001 0;0 0.000001];
```

From the above figures, LLS method would be best for dataset1 as the dataset is majorly concentrated along the diagonal and only few data points are outliers. Thus, LLS would be the best fit for the dataset because of very less noise in dataset and it would result in the least error.

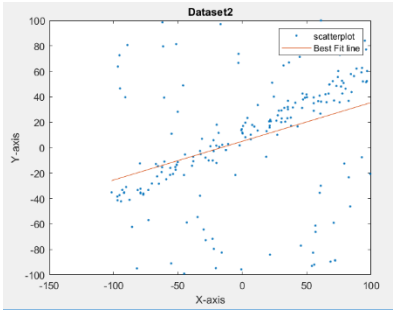
DATASET2



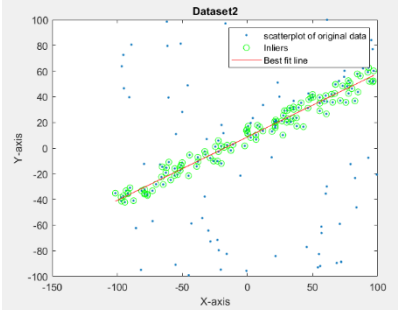
Scatter plot of dataset2. This contains more noise than the previous one.



LLS implementation



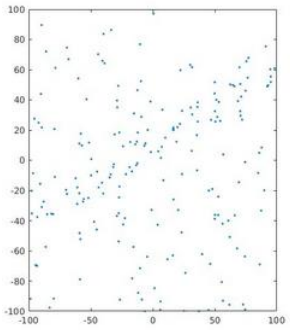
Regularization with `lamda = [0.00001 0;0 1];`



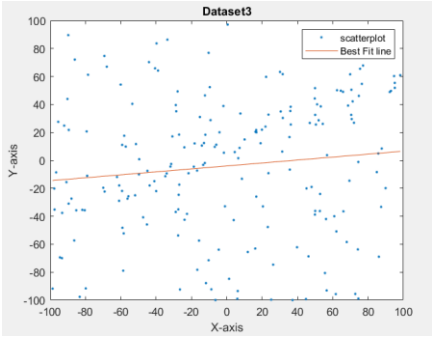
Ransac algorithm. Threshold distance = 10units. Min no. points is 120

Comparing the above figures I would say that the RANSAC algorithm is the best fit for the dataset2 as it would consider only the area with high datapoints density. Outliers are rejected in a good way.

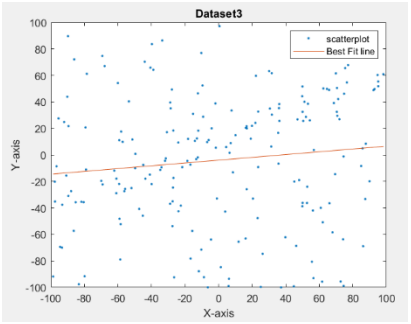
DATASET3



Scatter plot of dataset3.

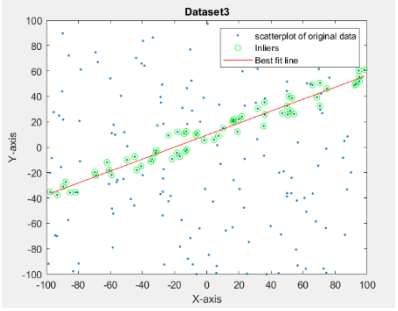


LLS implementation.



Regularization.

```
lamda = [0.00001 0;0 1];
```



RANSAC implementation. With distance threshold =10 units. Min no. of points is 60.

In these kinds of dataset wherein the data is very sparse, RANSAC is the best algorithm to fit in a line as it would enable us to get a line that would pass through max no. of points rejecting many outliers.