Problem 1

In the given video, a red ball is thrown against a wall. Assuming that the trajectory of the ball follows the equation of a parabola:

Question 1.1 Detect and plot the pixel coordinates of the centre point of the ball in the video. [10]

Approach/Pipeline:

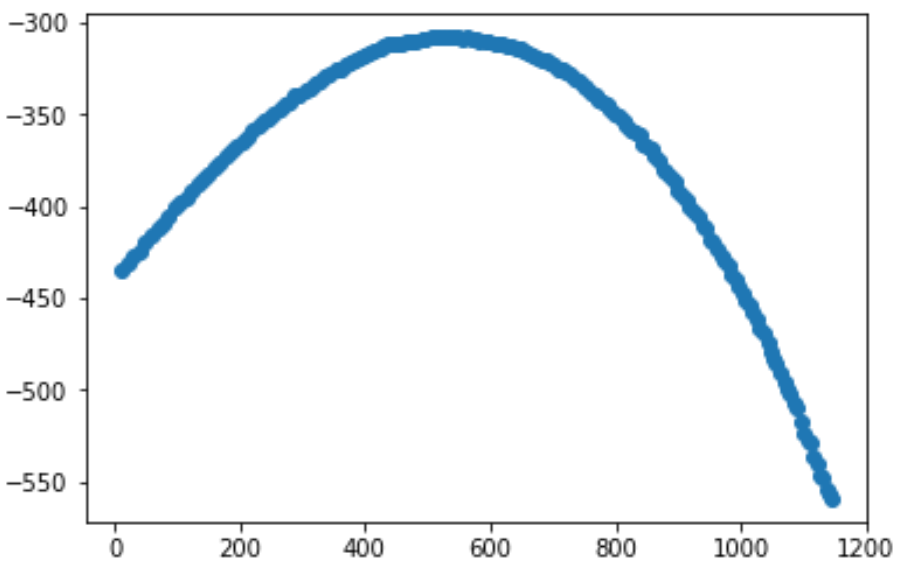
1] Iimage frame RGB values below the threshold are zeroed and others are retained.
2] Extracted ballpoints are saved and a mean is calculated for Ball's centre.
3] For each image frame calculated centres are stored in a list.
4] Generated points are plotted on an intermediate image frame.

Problems encountered:

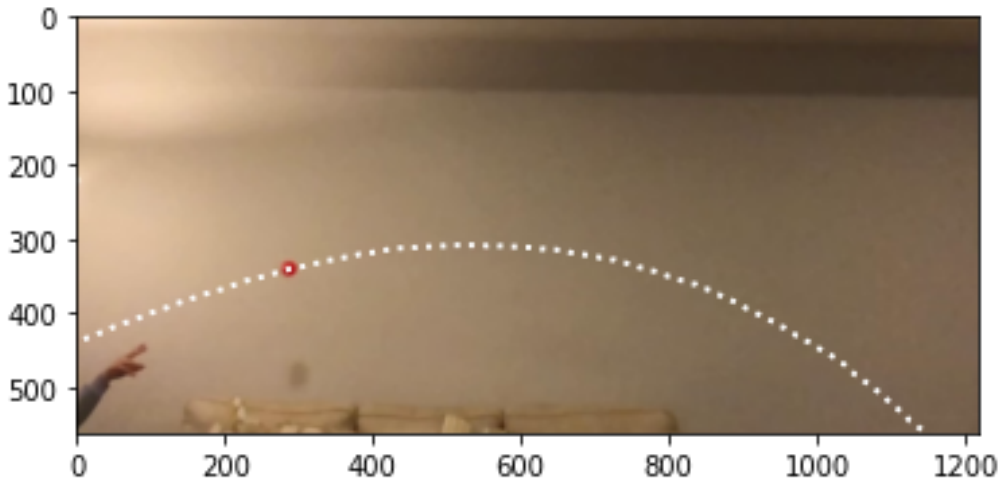
a] Deciding on Threshold for RGB values: Took multiple trial errors for a satisfactory solution.
B] Noise in some image frames: This issue was again tackled by fine tuning the threshold values.

Data plot of all detected centre coordinates of the available image frame (x coordinate plot has been intentionally inverted for better understanding of plot)

Results:



An instance previewing the detection of the ball's centre:



Here, all the white spots signify the ball's centre detected throughout all derived image frames.

Interpretation: The ball trajectory appears to follow a parabolic path.

Question 1.2 **Use Standard Least Squares to fit a curve to the extracted coordinates. For the estimated parabola you must.**

Approach/Pipeline:

1] First we form a matrix with an overdefined system based on x and y coordinates
2] Least Square method is employed for solving this problem. Steps involved are:

```
# Solve the least-squares problem
XT_X = np.dot(X.T, X)
XT_Y = np.dot(X.T, y_data)
coeffs = np.linalg.solve(XT_X, XT_Y)
```

3] These coeffs are used for fitting the curve along the obsered ball centre points.
4] The Fitted curve is plotted alongside Observed data points.

Problem Encountered:

1] Struggled a lot with line fitting - Eventually fixed by correcting input data with transpose and some mathematical operation.

Results:

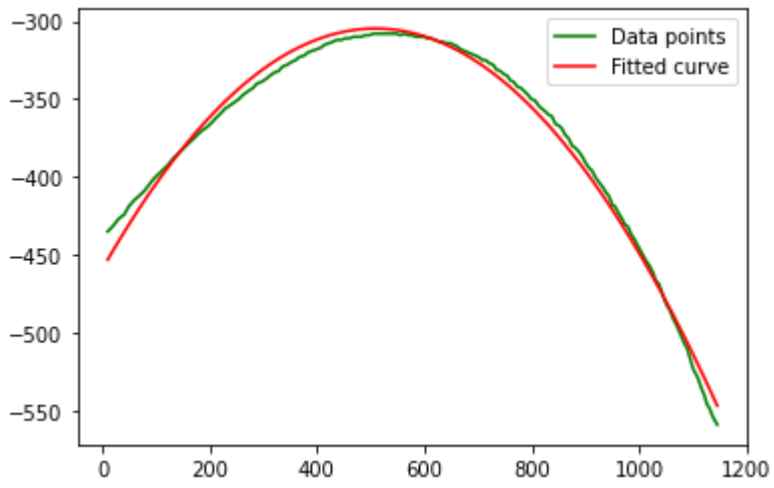
a. Print the equation of the curve. [5]

Fit curve coefficients i.e a,b,c are generated from the fit curve function.

Generalised Quadratic Equation: ax^2 + bx + c = 0

Equation of the fitted curve is: y = 458.620351 + (-0.606833)x + (0.000595)x^2

b. Plot the data with your best-fit curve. [5]



Here, the above plot shows observed data points (green coloured) and the red coloured one indicates the best-fit curve for the generated centre coordinates.

Code Explanation:

- Fits a curve to a given set of data points using the least square method.
- It takes in the x and y coordinates of the data points and the degree of the polynomial to fit and returns polynomial coefficients of fitted curve.
- Creates design matrix based on input data, then transposes it along with y-coordinates to solve normal equation and obtain polynomial coefficients.
- Plots the fitted curve and the data points using Matplotlib.

Question 1.3 **Assuming that the origin of the video is at the top-left of the frame as shown below, compute the x-coordinate of the ball's landing spot in pixels if the y-coordinate of the landing spot is defined as 300 pixels greater than its first detected location. [10]**

Approach/Pipeline:

- Calculates the roots of a quadratic equation given the coefficients a, b, and c.
- Discriminant is calculated using the coefficients and then the two roots are found using the quadratic formula.

Problem Encountered: - NA -

Results:

The roots are = (-364.26694948821057+0j) (1384.1543444461936+0j)

X co-ordinate of the landing spot = (1384.1543444461936+0j)

Problem 2

Question 2 Given are two CSV files, pc1.csv and pc2.csv, which contain noisy LIDAR point cloud data in the form of (x, y, z) coordinates of the ground plane.

1. ***a) Compute the covariance matrix using PC1. [15]***

The covariance matrix is one of the most important matrices in data science and machine learning. The covariance matrix gives the correlation coefficients between features in a dataset and hence it is very useful for feature selection and dimensionality reduction. Plotting the covariance matrix produces a visual plot that displays the correlation coefficients.

$$q_{jk} = \frac{1}{N-1} \sum_{i=1}^{N} \left(X_{ij} - \bar{X}_j\right) \left(X_{ik} - \bar{X}_k\right)$$

Approach/ Pipeline:

1] Based on the Covariance formula, along with corresponding variance and covariances.

$$Covariance\ matrix = \begin{bmatrix} var(x) & cov(x,y) & cov(x,z) \\ cov(y,x) & var(y) & cov(y,z) \\ cov(z,x) & cov(z,y) & var(z) \end{bmatrix}$$

Code Explanation:

Calculates the discriminant of the quadratic equation and uses the math module to calculate the roots of the quadratic equation.

Problems encountered: *- NA -*

Interpretation:

1] The covariances of each pair of variables are displayed in this symmetric matrix.
2] The amplitude and direction of the distribution of multivariate data in multidimensional space are shown by these values in the covariance matrix.
3] By adjusting these variables, we may learn how data is distributed across two dimensions.

Results :

Covariance:

```
Covariance for PC1:
[[ 33.6375584   -0.82238647 -11.3563684 ]
 [ -0.82238647  35.07487427 -23.15827057]
 [-11.3563684  -23.15827057  20.5588948 ]]
```

1.b) Assuming that the ground plane is flat, use the covariance matrix to compute the magnitude and direction of the surface normal. [15]

Approach/ Pipeline:

1] The covariance matrix is a square matrix that summarizes the variance and covariance of a set of variables.

Code Explanation:

- Performs principal component analysis (PCA) on the points to find the surface normal, magnitude, and direction.
- Calculates the surface magnitude and surface direction and prints these values along with the eigenvalues.

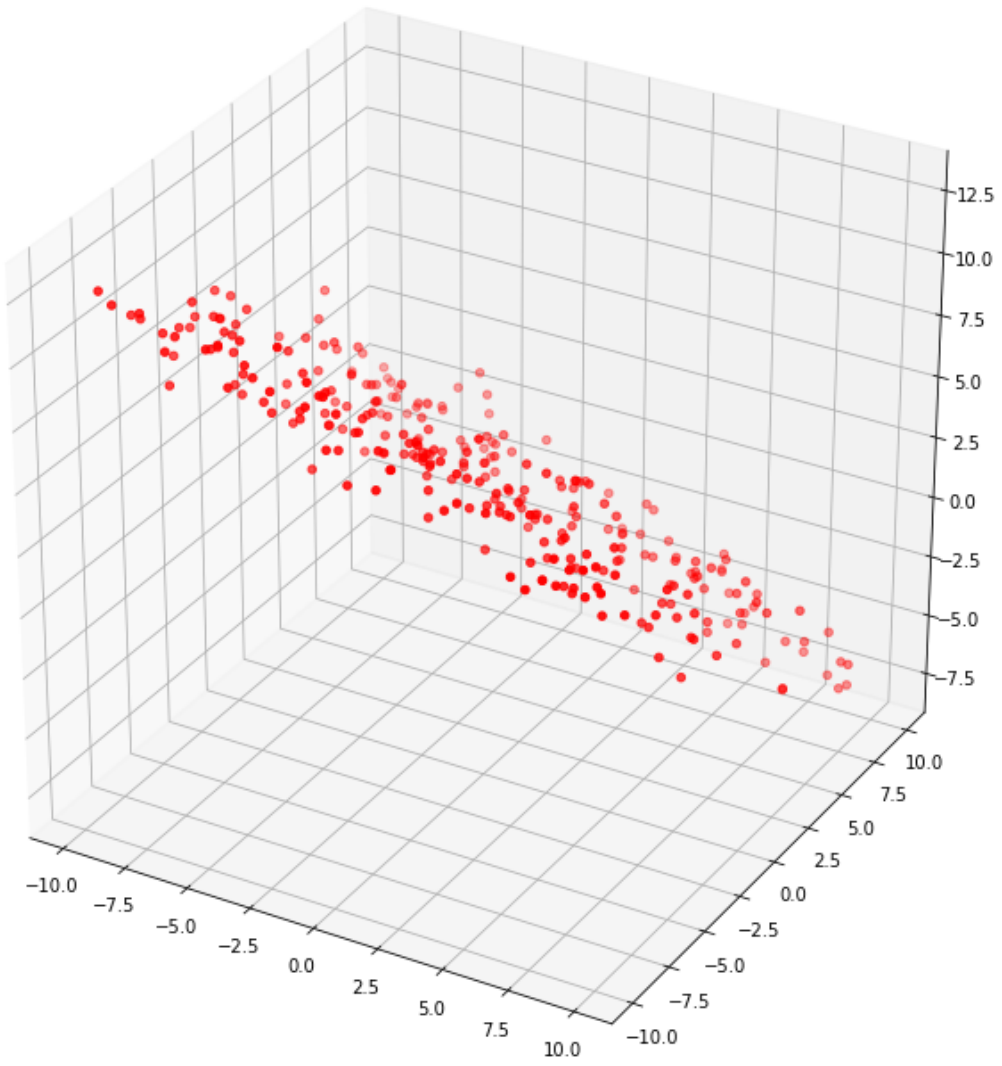
Problems encountered: *- NA -*

Results :

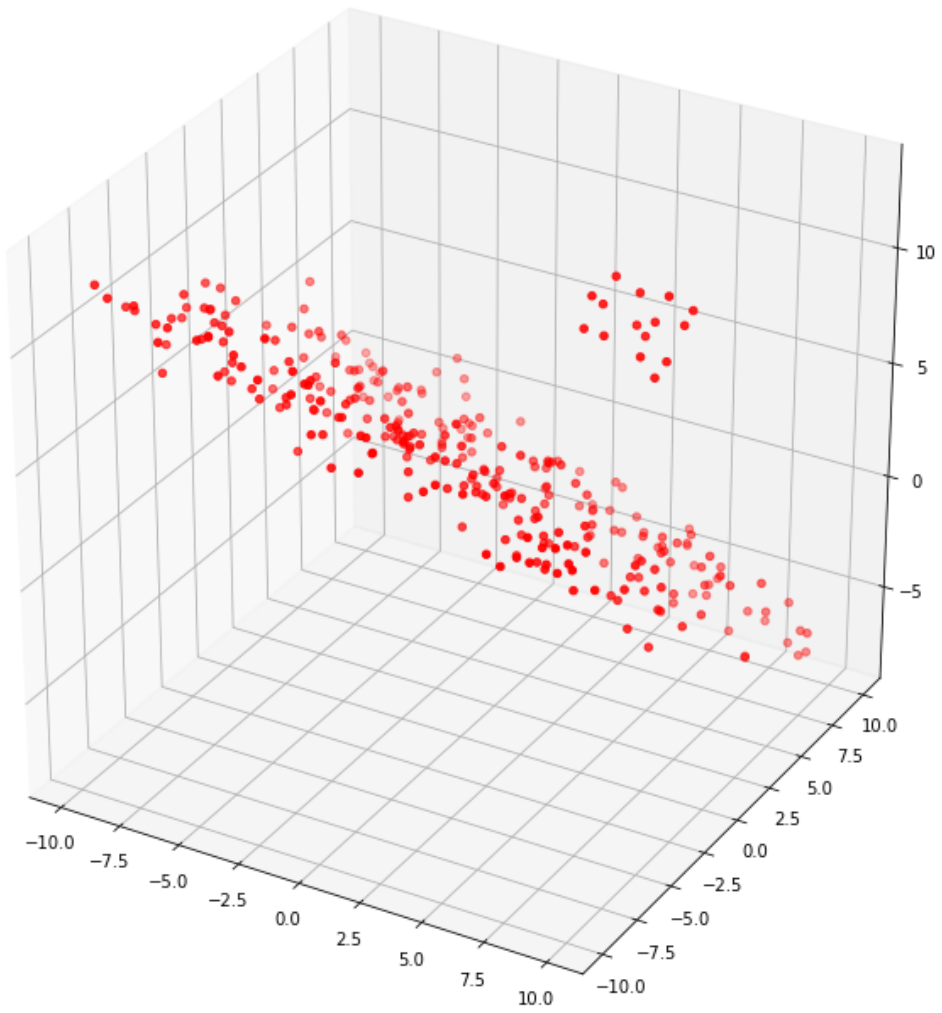
```
Eigenvalues: [ 0.66727808 34.54205318 54.06199622]
Surface magnitude: 0.81687090810658
Surface direction: [0.35031762 0.66070701 0.9692107 ]
```

2] In this question, you will be required to implement various estimation algorithms such as Standard Least Squares, Total Least Squares and RANSAC.

Raw Data Plot for PC1



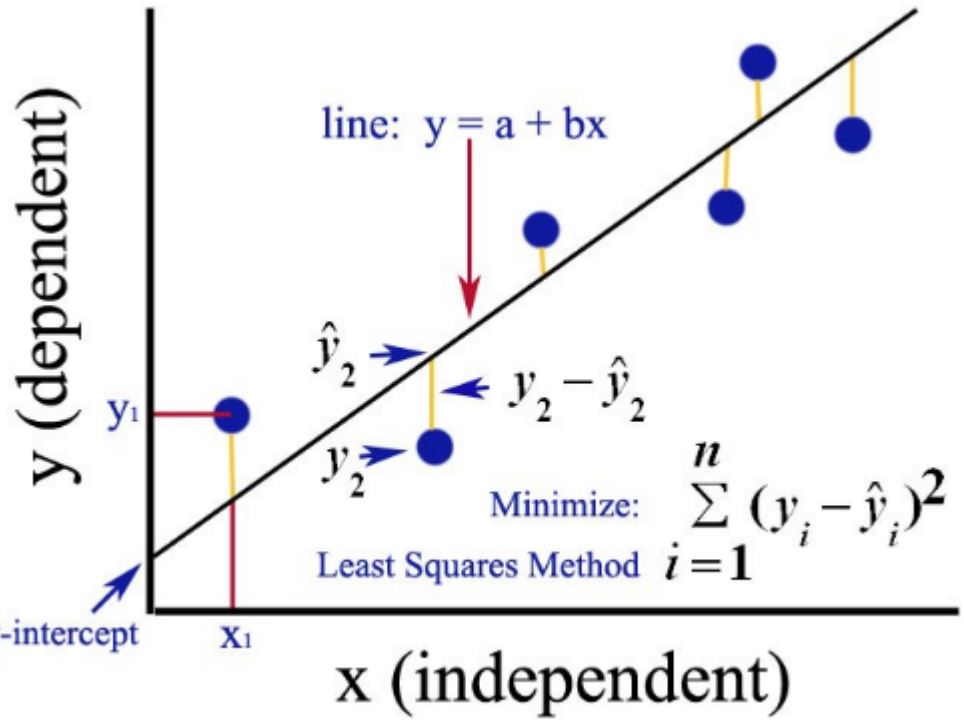
Raw Data Plot for PC2



Question *2 a) Using pc1.csv and pc2, fit a surface to the data using the standard least square method and the total least square method. Plot the results (the surface) for each method and explain your interpretation of the results. [20]*

Standard Least Square

Interpretation



Reference: https://medium.com/analytics-vidhya/ordinary-least-square-ols-method-for-linear-regression-ef8ca10aadfc

Approach/ Pipeline:

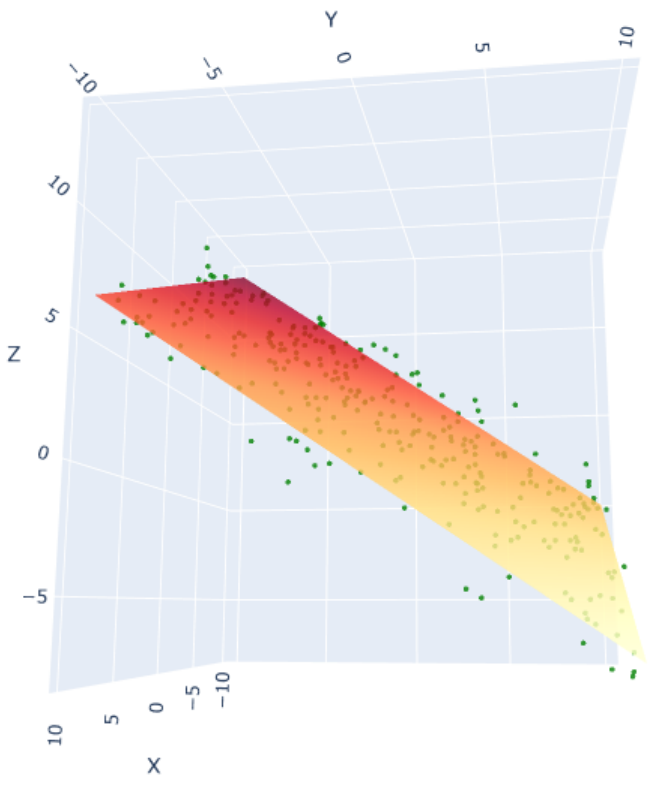
- Data is first translated to its origin by subtracting the centroid coordinates from each data point.
- Later form a design matrix A by appending a column of ones to the (x, y) coordinates of each data point. This matrix represents the independent variables (overdefined System)
- Construct a response vector b by using the z-coordinate of each data point. This vector represents the dependent variable.
- Use the standard least squares method to calculate the coefficients of the plane. This involves solving the equation Ax = b for the coefficients x, where x = (a, b, c) represents the coefficients of the plane equation z = ax + by + c.
- Once the coefficients are computed, a plane can be defined by the equation z = ax + by + c. This can be used to compute the z-coordinate of any point on the plane.
- The fitted plane can be visualized by plotting the data points and the plane in a 3D plot. This can help to validate the accuracy of the fitting and provide insights into the 3D structure of the object.

Problem Encountered:

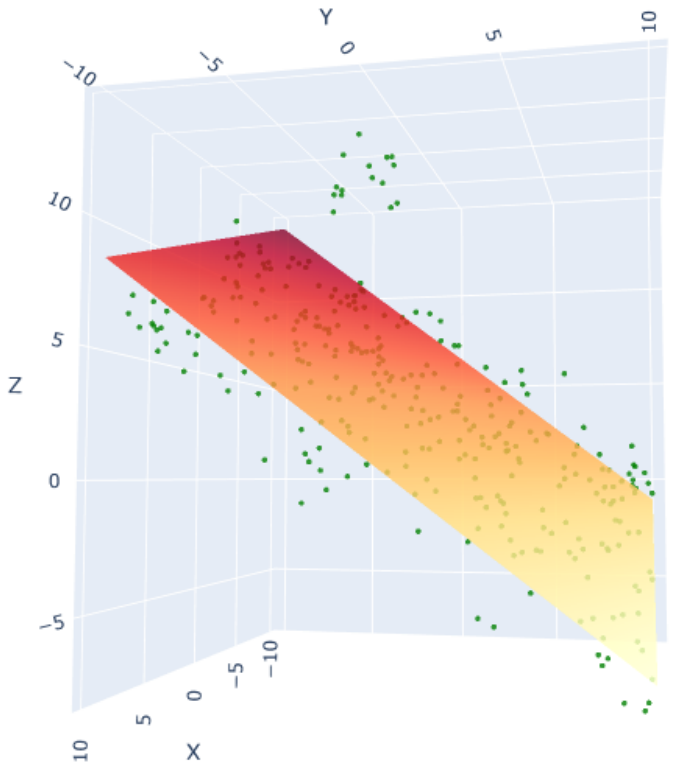
1] Applying the standard least square from scratch made the implementation challenging and we encountered multiple errors during mathematical operation on matrices.

Results:

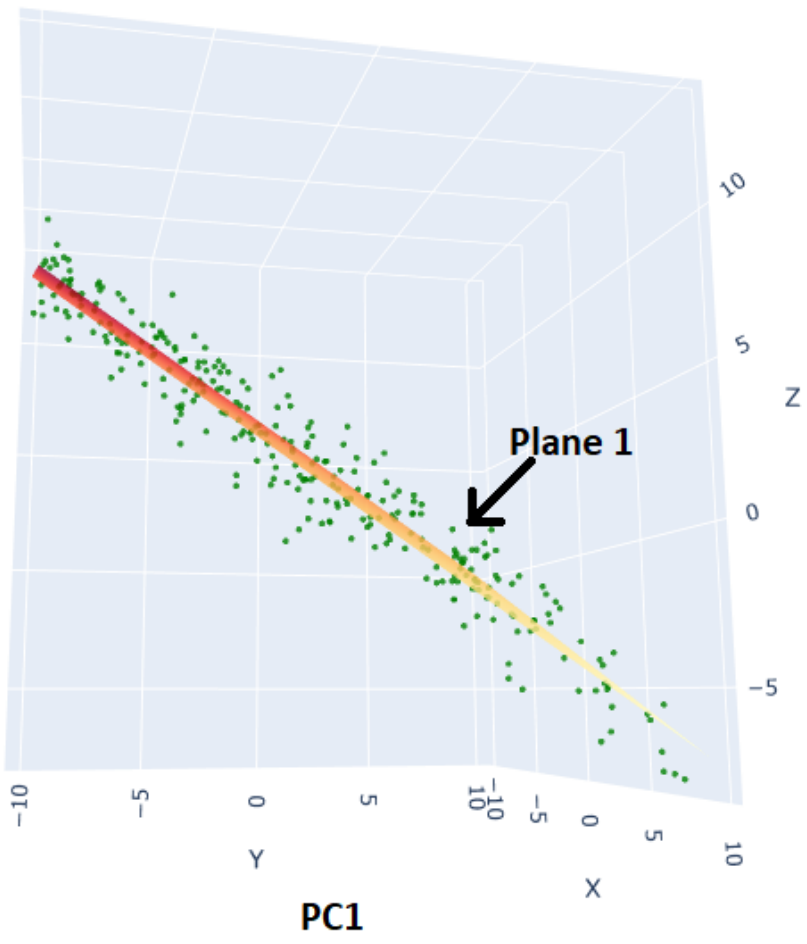
Applying Standard Least Square on PC1



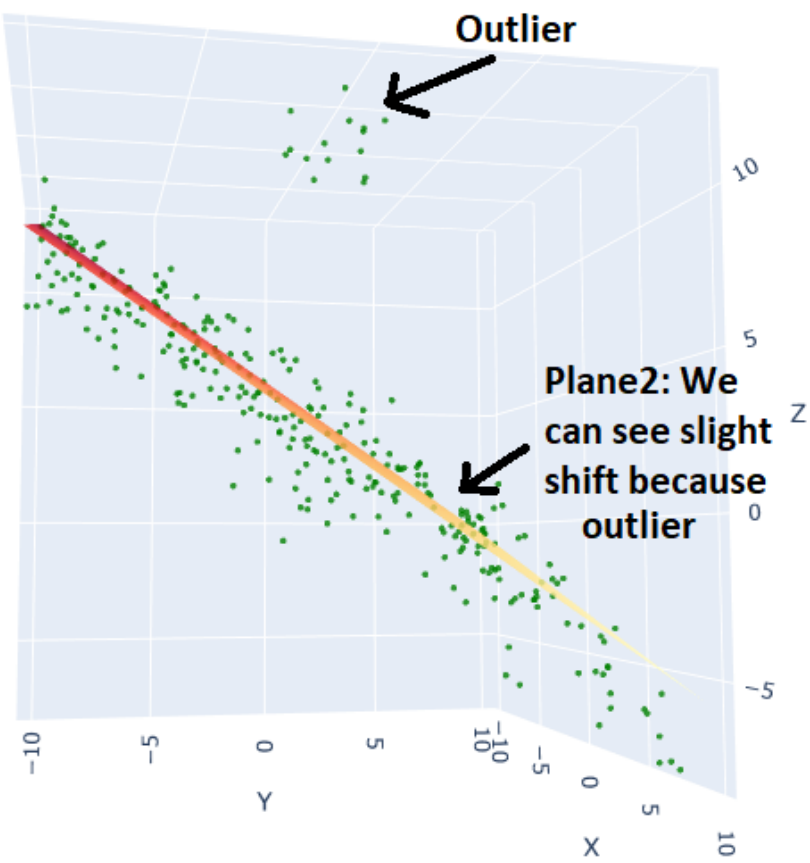
Applying Standard Least Square on PC2



Plot Comparison

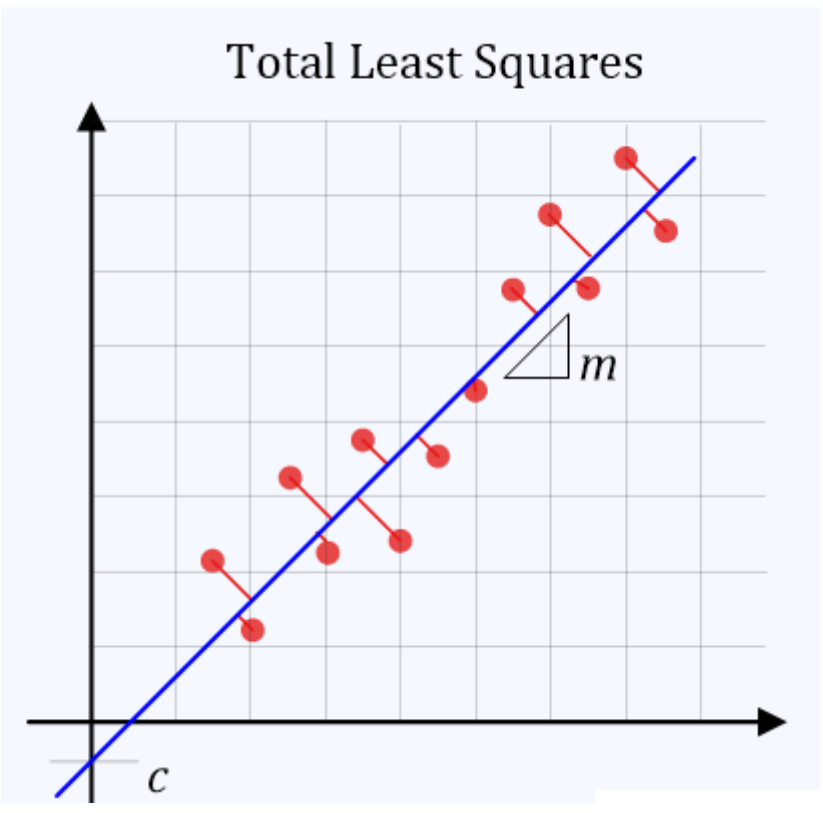


PC1



PC2

Total Least Square



Reference: https://wirelesspi.com/an-intuitive-guide-to-linear-regression/

Approach/ Pipeline:

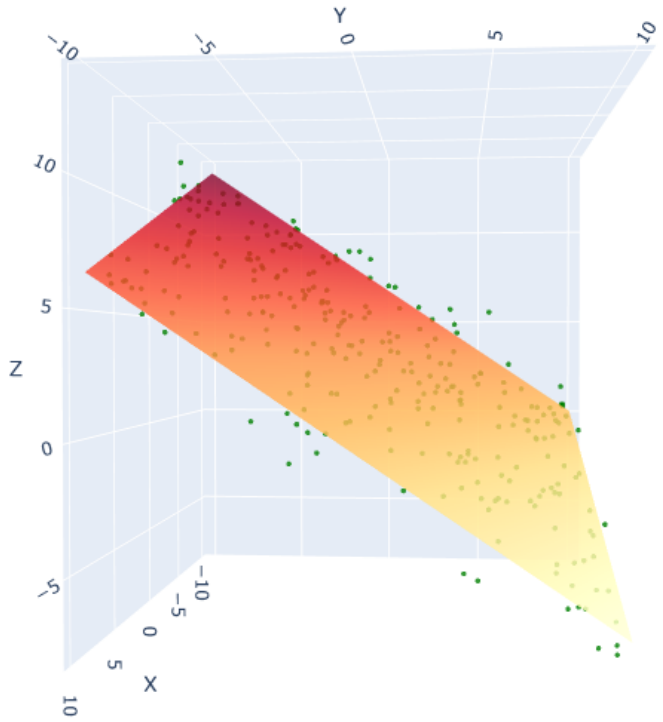
1] First compute the covariance matrix and eigenvalues and eigenvectors of the covariance matrix.
2] Then chooses the eigenvector with the smallest eigenvalue as the surface normal and calculate the surface magnitude and direction.
3] Then total least square regression on the two sets of point cloud data is applied. The resulting total least square plane equation coefficients are printed, and a 3D scatter plot of the first set of point cloud data with the total least square plane superimposed is created using plotly library.

Problem Encountered:

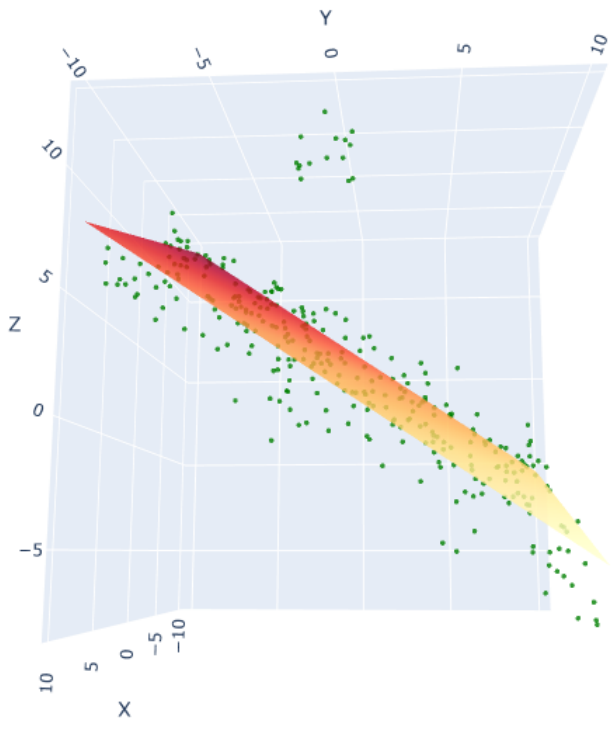
1] Applying the Total least square from scratch made the implementation challenging and we encountered multiple errors during mathematical operation on matrices.
2] Visualizing the change in the application of standard least square and total least square as the change was barely noticeable.

Results:

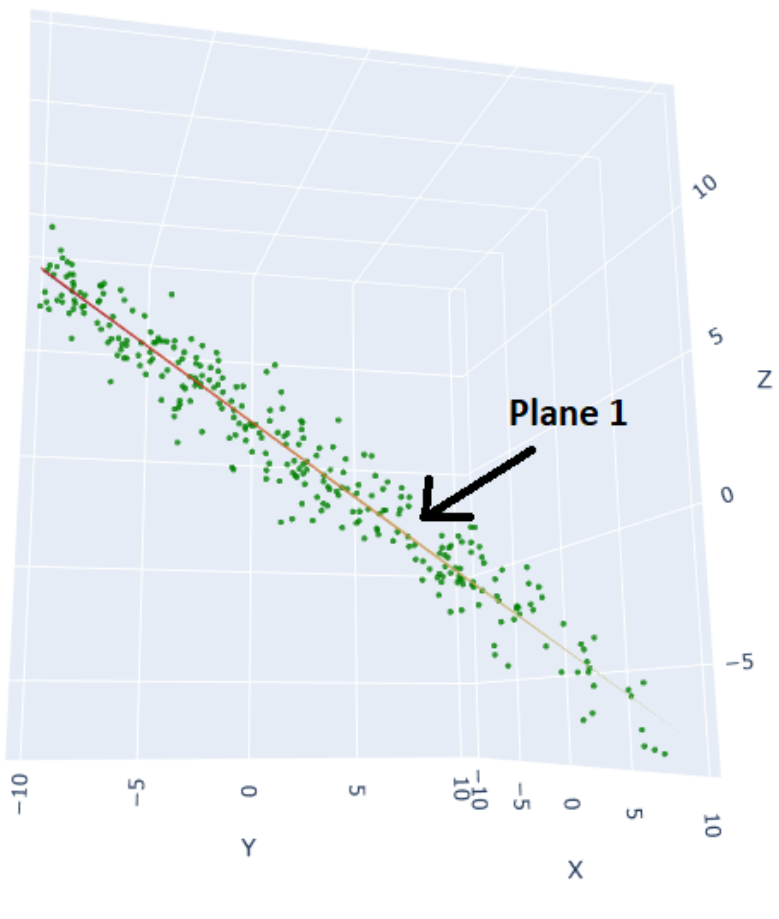
Applying Total Least Square on PC1

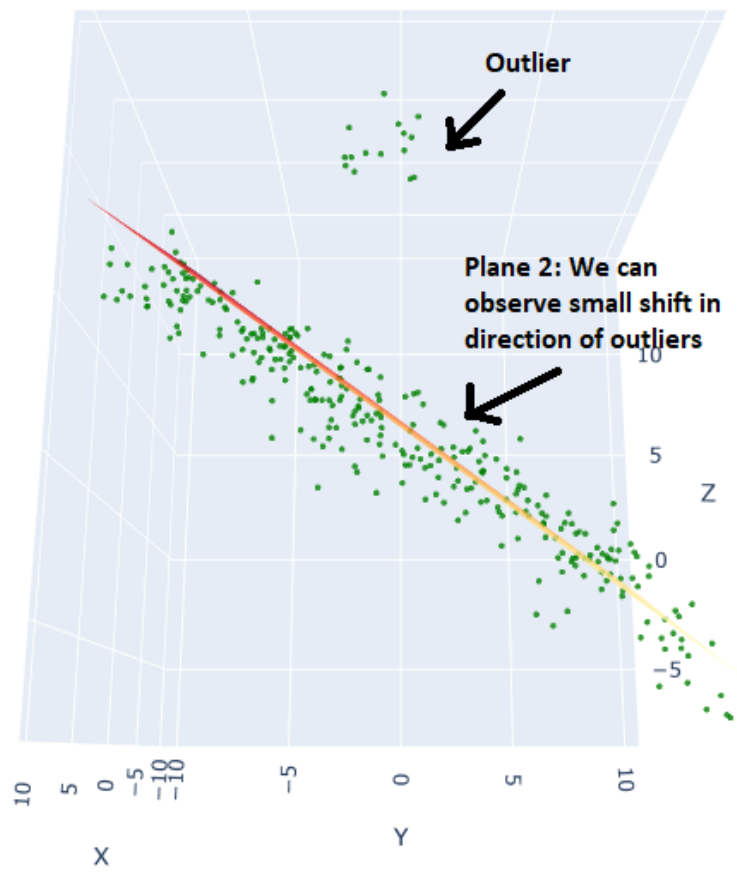


Applying Total Least Square on PC2



Comparison





Discussion/ Interpretation:

Both standard least squares (SLS) and total least squares (TLS) are methods for fitting a line or plane to a set of data points. However, when dealing with outlier-prone data sets, TLS may be more appropriate as it takes into account both the vertical and horizontal distances between the data points and the line or plane, resulting in a better fit to the majority of the data points and less influence from the outliers.

We can observe that not only does the plane is shifted towards the outliers but we can also observe a slight orientation change in the second plot as we are using the total least square where the output is affected by perpendicular distance and not vertical distance which is in the case of standard least square.

Question 2 b. Additionally, fit a surface to the data using RANSAC. You will need to write RANSAC code from scratch. Briefly explain all the steps of your solution, and the parameters used. Plot the output surface on the same graph as the data. Discuss which graph fitting method would be a better choice for outlier rejection. [20]

Approach/ Pipeline:

RANSAC

RANSAC is a robust estimation algorithm used for modelling data points that may contain outliers or noise. It works by randomly selecting a subset of points to form a hypothesized model, then iteratively improving the model by re-estimating parameters based on inliers and rejecting outliers.

Code Steps Explanation:

The main steps of the code are as follows:

- Defined a function which takes a subset of the data and a threshold value as inputs, and returns the number of inliers and the plane coefficients (normal vector and distance from origin).
- We start by randomly selecting a few points (three in this code) to create an initial model. Then, we test all the other points in the dataset to see if they fit the model within a certain threshold.
- If enough points (inliers) fit the model, we re-estimate the model using all of the inliers. We repeat these steps for a set number of iterations (10 million *(Apologies for the inconvenience during the code test - It would take > 25mins)* in this code)
- Choose the model that has the most inliers.
- Create a mesh grid to represent the plane and compute the inlier and outlier masks using the plane coefficients and threshold.
- Create three traces for the inliers, outliers, and the plane, and add them to a Plotly Figure object.
- Show the 3D plot using the "fig.show()" method.

Parameters used:

Threshold:

- Tuned to define in liners and outliers, this parameter helps the RANSAC algorithm to classify point data into inliers and outliers, depending on the error magnitude.
- Similarly, smaller threshold results in more points being considered inliers, but may also include outliers.
- Conversely, a larger threshold reduces the number of inliers and increases the number of outliers.
- The choice of these parameters depends on the specific problem and the desired balance between accuracy and robustness.

Iterations:

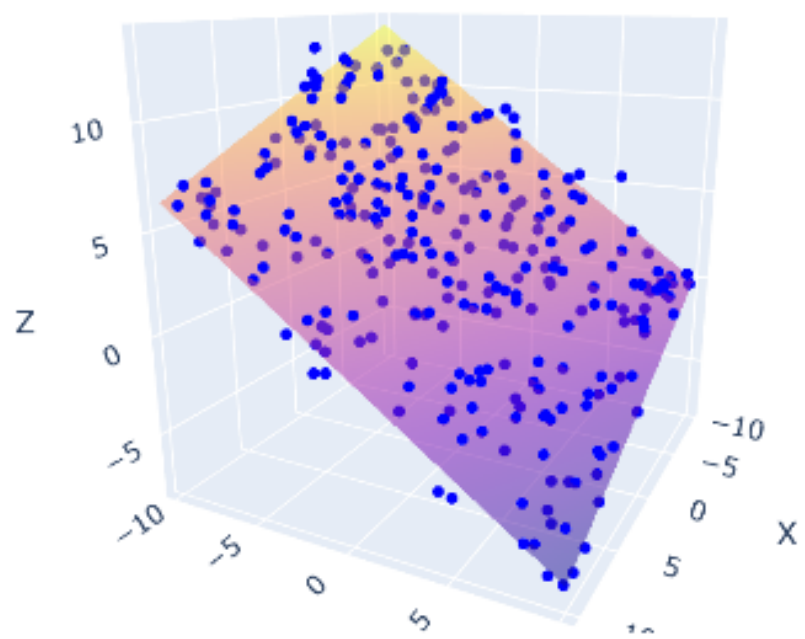
The number of iterations and the threshold is important parameters in RANSAC. The number of iterations affects how many times we will try to fit the model, while the threshold determines how many points will be considered inliers.

Problem Encountered:

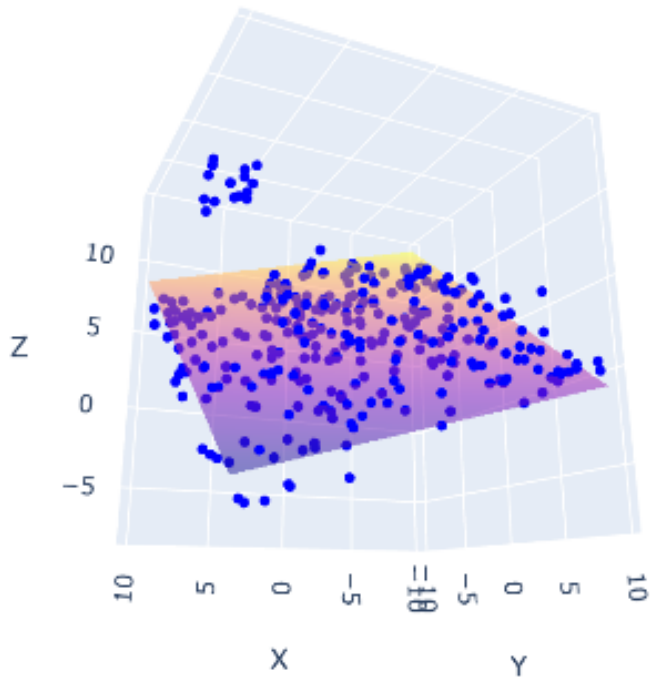
1] Deciding the threshold and the number of iterations of very time-consuming and took more effort compared to previous methods

Results:

Applying RANSAC on PC1:



Applying RANSAC on PC2:



Interpretation/ Observation/ Discussion (Better Choice for Outlier Rejection):

- RANSAC is preferred over standard least square and total least square methods for the given point cloud because can handle outliers in the data.
- Least square methods, such as standard and total least square, are sensitive to outliers and can result in a poor fit if there are outliers present in the data.
- RANSAC can handle outliers by randomly selecting subsets of the data to fit the model and then selecting the subset with the most inliers as the final model.
- Least square methods try to minimize the sum of the squared residuals, which makes them sensitive to outliers because the squared residuals of outliers are typically much larger than those of inliers.

Therefore, when the data contains outliers, RANSAC is a better choice than the least square methods.