Q2. 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. Using pc1.csv:

a. Compute the covariance matrix.

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

Solution 2.1:

Approach:

We are trying to perform principal component analysis (PCA) on two sets of 3D points represented by csv files **pc1.csv** and **pc2.csv**.

1. We first read the csv files into 'pandas' data-frames.
2. We then extract the x, y, and z coordinates into separate 'numpy' arrays for each point set.
3. We then calculate the mean of each coordinate and uses them to create a covariance matrix.

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

4. We then use the numpy eig function to find the eigenvalues and eigenvectors of the covariance matrix which can be used to calculate the magnitude and direction of the surface normal.
5. The code then prints the covariance matrix, the eigenvalues, and the magnitude (sq root of the min eigen value) and the direction of the surface normal (eigenvector corresponding to the smallest eigenvalue).

Results:

```
The Covariance Matrix is:
[[ 33.6375584   -0.82238647 -11.3563684 ]
 [ -0.82238647  35.07487427 -23.15827057]
 [-11.3563684  -23.15827057  20.5588948 ]]

Eigenvalues of the covariance matrix are: [ 0.66727808 34.54205318 54.06199622]

The surface normal is the Eigenvector corresponding to the smallest Eigenvalue
Therefore,
Direction of the surface normal: [0.28616428 0.53971234 0.79172003]
Magnitude of the surface normal: 0.8168709081065844
```

Challenges: **None**

The normal is shown in the following figure (magnitude is scaled for visualisation)

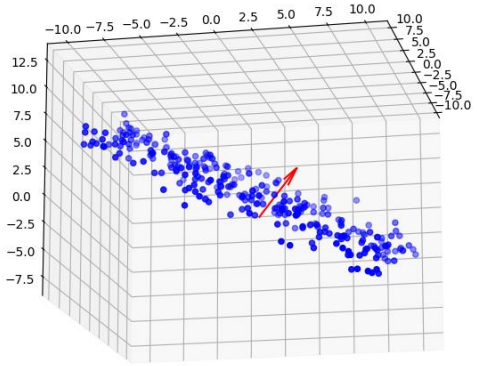


Fig 4. Normal to pc1.csv

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

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.

Standard Least Square Fitting:

1. We write a code that performs a standard least square fitting of a plane to the point cloud represented by the **pc1.csv** and **pc2.csv** files.
2. We first construct a matrix **A** containing the x, y, and constant 1 coordinates of each point in **the csv file**.
3. We then construct a vector **b** containing the z coordinates of each point.
4. We then solve the normal equations

$$\mathbf{ATAx = ATb}$$

using the pseudo inverse method because the matrix is not square.
x contains the optimized coefficients for the plane equation

$$\mathbf{ax + by + c = z}.$$

5. The code then creates a 3D plot of the plane using matplotlib's **plot_surface** function, with **x** and **y** values generated using numpy's **linspace** function.

Total Least Square Fitting:

The code then computes the coefficients of the plane that best fits a set of point cloud using the method of total least squares, and then plots the plane.

1. First, the code computes the mean of the input data points and centers the data by subtracting the mean from each point.
2. Then, it computes the covariance matrix of the centered data and finds the eigenvector corresponding to the smallest eigenvalue, which is the normal vector of the plane.
3. Next, the code extracts the coefficients of the plane equation from the normal vector and the mean of the data points.
4. It then defines a grid of points in the x-y plane and computes the corresponding z-values using the plane equation.
5. Finally, it plots the plane using Matplotlib's **plot_surface** function.

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 of outlier rejection.

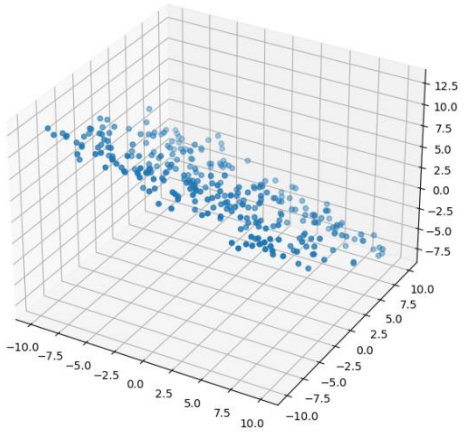
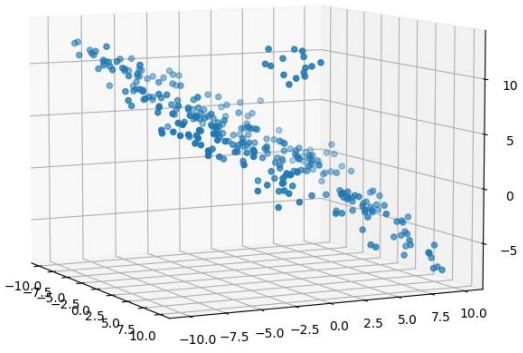
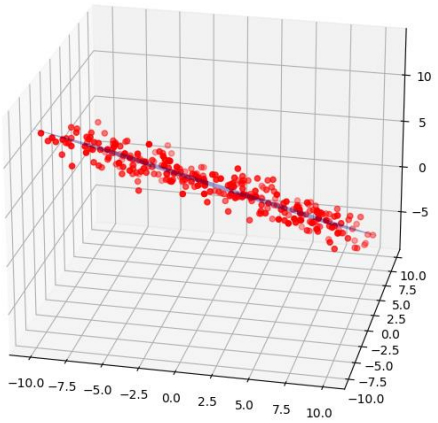
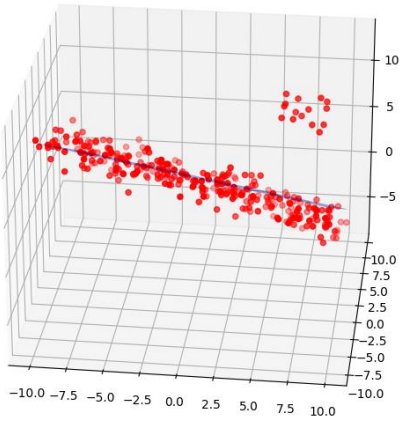
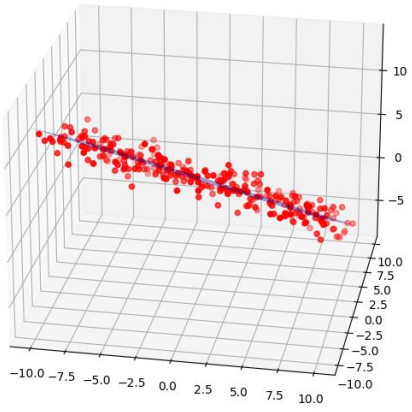
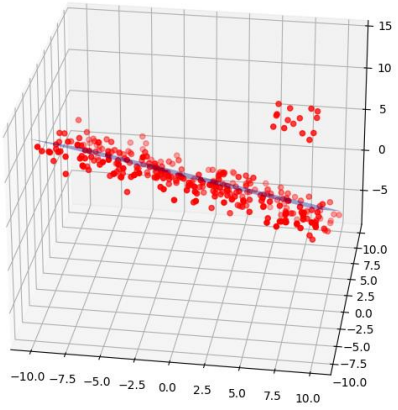
The RANSAC algorithm is a random sampling method used to estimate the parameters of a mathematical model from a set of observed data that contains outliers. It consists of several steps:

1. Randomly select a subset of the observed data that is the minimum number required to fit the model.
2. Compute the parameters of the model using the selected data subset.
3. Find the inliers, which are the observed data points that are consistent with the model, by comparing all observed data points with the model.
4. If the number of inliers is greater than a predefined threshold, re-estimate the model parameters using all the inliers.
5. Repeat steps 1 to 4 for a fixed number of iterations or until a certain convergence criterion is met.
6. Choose the model with the most inliers as the final solution.

In the given code, the RANSAC algorithm is used to fit a plane to a set of 3D points. The **fit_surface()** function implements the RANSAC algorithm, taking as input a set of points, the maximum number of iterations to run, and the distance threshold for identifying inliers. It returns the plane parameters (normal

vector and distance from origin) and the indices of the inliers. After applying the RANSAC algorithm to the set of points in **pc1.csv** and **pc2.csv**, the plane equation is computed using the coefficients a, b, c and d returned by **fit_surface()**. The inliers are plotted in red, and the plane is plotted using the surface plot.

Results, interpretations:

	pc1	pc2
Scatter Plot		
Standard Least square		
Total Least square		

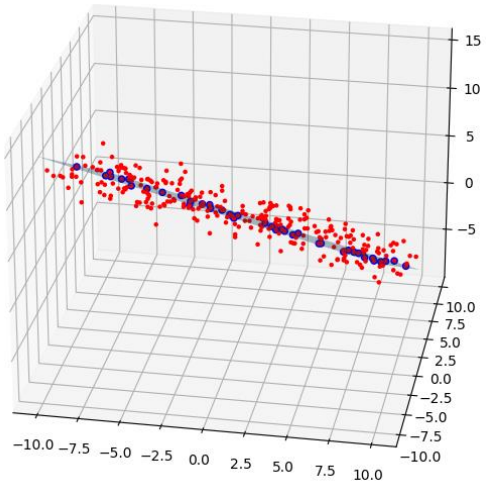
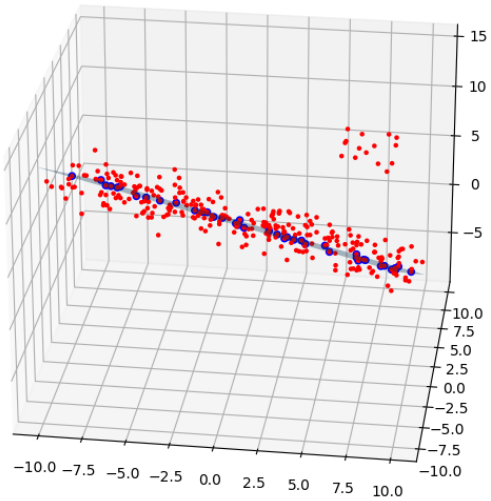
RANSAC		

Table 1. Comparison of planes plotted to fit a data using various estimation methods
(The plots are viewed from a specific angle on purpose so as to easily visualize the shift due to outliers)

- Based on the plots, it can be seen that the while fitting the dataset pc2 with a plane using standard least squares, the plane slightly deviates from the surface points in the direction of the outliers, demonstrating its sensitivity to outliers and consequent susceptibility to noise.

- In case of plane fitting using the method of total least squares, the plane displayed a comparable amount of deviation toward the outliers but less than that with Standard least squares method, demonstrating superior noise resistance.

- In the case of outliers, RANSAC is generally the most robust method compared to least squares and total least squares. This is because RANSAC can identify and reject outliers by fitting the model only on a subset of the data that is likely to contain inliers. Least squares and total least squares, on the other hand, try to fit the model to all data points, including outliers, which can significantly affect the accuracy of the model.

- Least squares and total least squares can still be useful when there are no or only a few outliers in the data. However, in the presence of a large number of outliers, RANSAC is likely to give better results.

Challenges:

- It was difficult to come up with a technique for total least squares (TLS). The longest period was spent on this part of the project. Debugging simple errors in the signs used in the equations of the plane required a lot of time.
- Writing a code for the RANSAC function took a lot of time.

REFERENCES:

Lecture Slides for Standard Least Square, Total Least Square and RANSAC

Lecture Slides for Pseudo Inverse procedure

Python tutorials by TAs for reading the video file and slicing methods

Stackoverflow for inRange() function