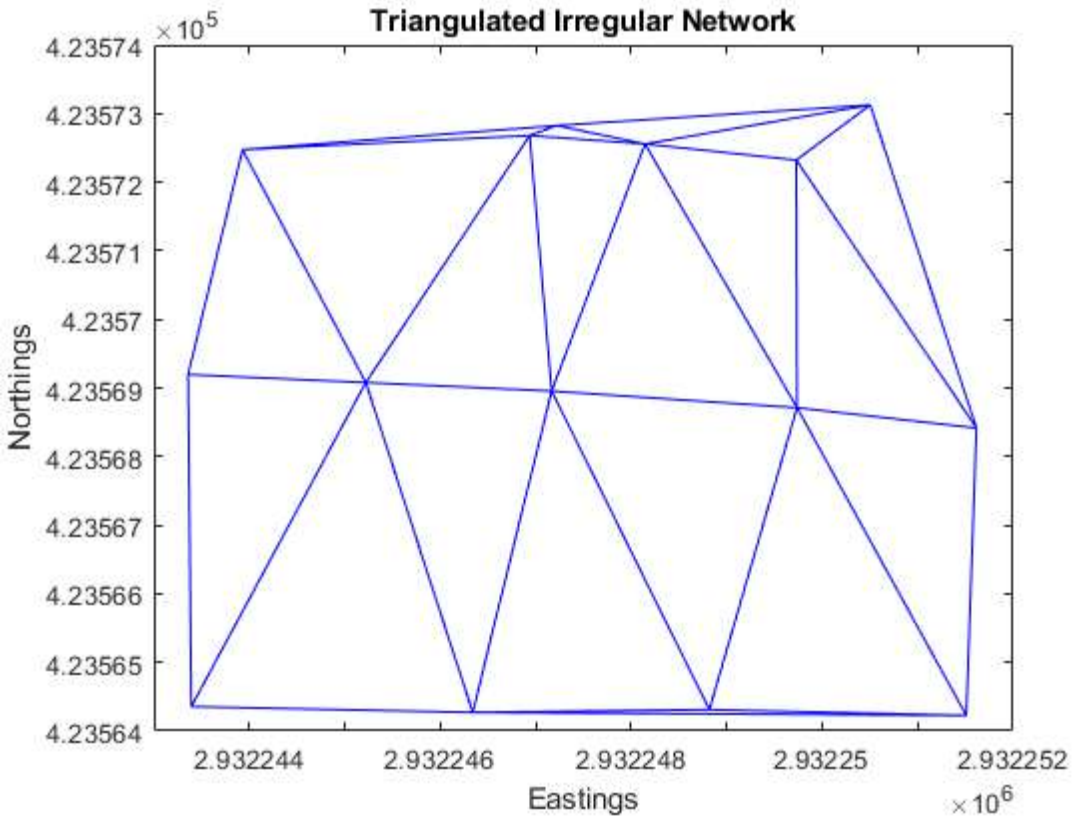
Contents

Plotting of TIN

Load data from CSV, excluding the third column

```
data = readmatrix('gcp.csv');
data(:, 3) = []; % Remove the third column

% Extract northing and easting values
northings = data(:, 1);
eastings = data(:, 2);

% Create Delaunay triangulation
tri = delaunayTriangulation(eastings, northings);

% Plot the triangulation
triplot(tri);
xlabel('Eastings');
ylabel('Northings');
title('Triangulated Irregular Network');
```



Generating elevation from the TIN

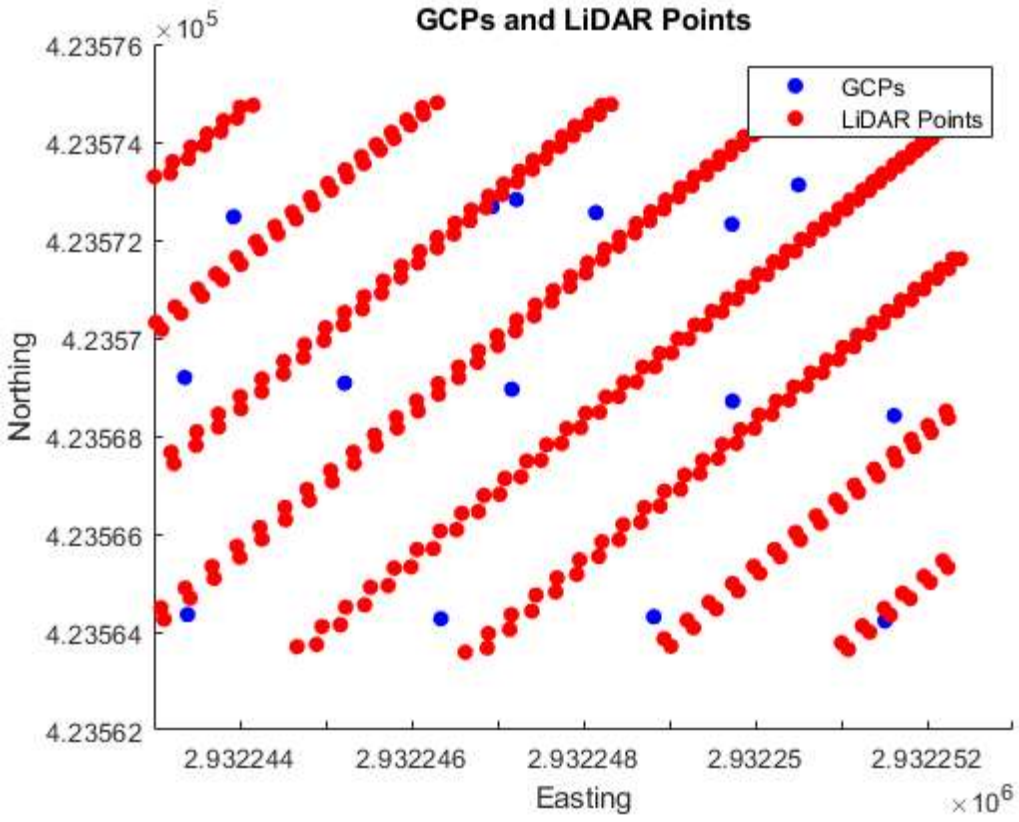
```
% Load LiDAR data (E, N, H)
lidar_data = readmatrix('lidar.csv');  % Skip header row if present

% Load GCPs (E, N, H)
gcps = readmatrix('gcp.csv');  % Skip header row if present

% Generate TIN with GCPs
tri = delaunay(gcps(:, 1), gcps(:, 2));
tin_interp = scatteredInterpolant(gcps(:, 1), gcps(:, 2), gcps(:, 3), 'linear', 'none');

% Interpolate heights for LiDAR points
lidar_easting_northing =  lidar_data(:, 1:2);
lidar_heights = tin_interp(lidar_easting_northing);

% Find valid indices (non-NaN values)
valid_indices = ~isnan(lidar_heights);

% Filter out NaN values and corresponding lidar data
lidar_easting_northing_valid = lidar_easting_northing(valid_indices, :);
lidar_heights_valid = lidar_heights(valid_indices);
lidar_data_valid = lidar_data(valid_indices, :);

% Calculate height differences
height_diffs = lidar_data_valid(:, 3) - lidar_heights_valid;

% Compute RMSE
rmse = sqrt(mean(height_diffs .^ 2));

disp(['Root Mean Square Error (RMSE): ', num2str(rmse)]);
```

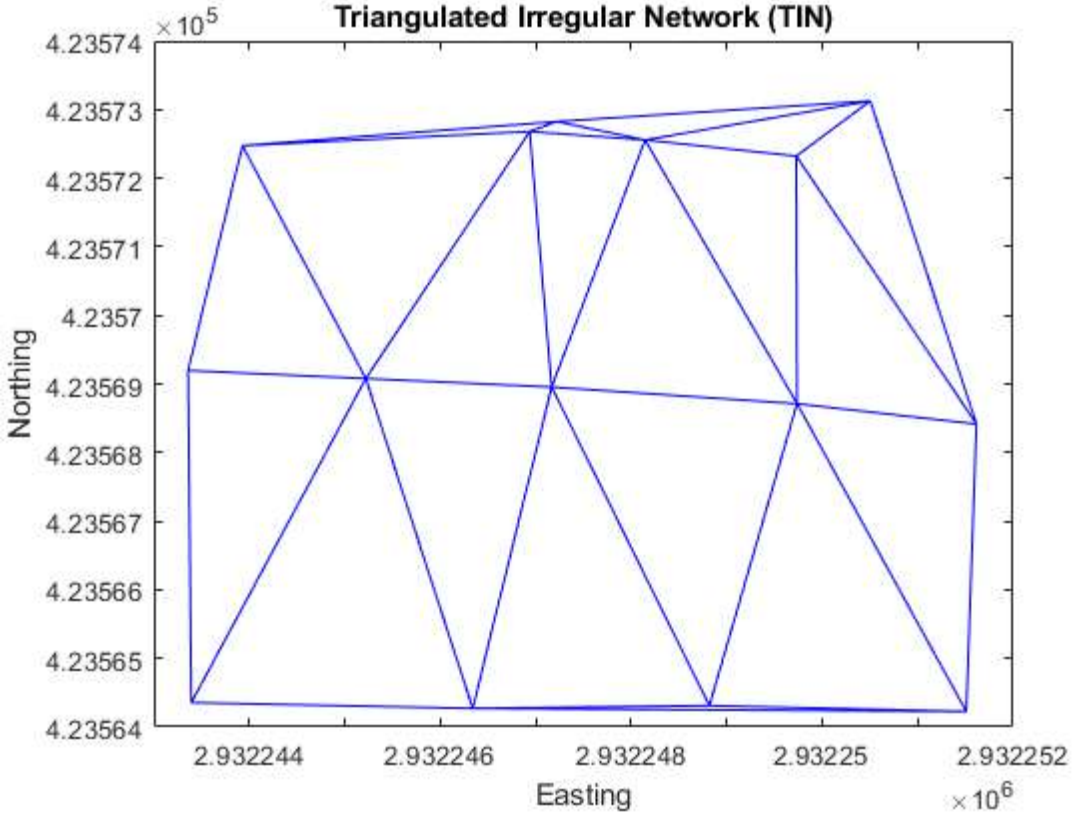
```
Root Mean Square Error (RMSE): 0.40745
```

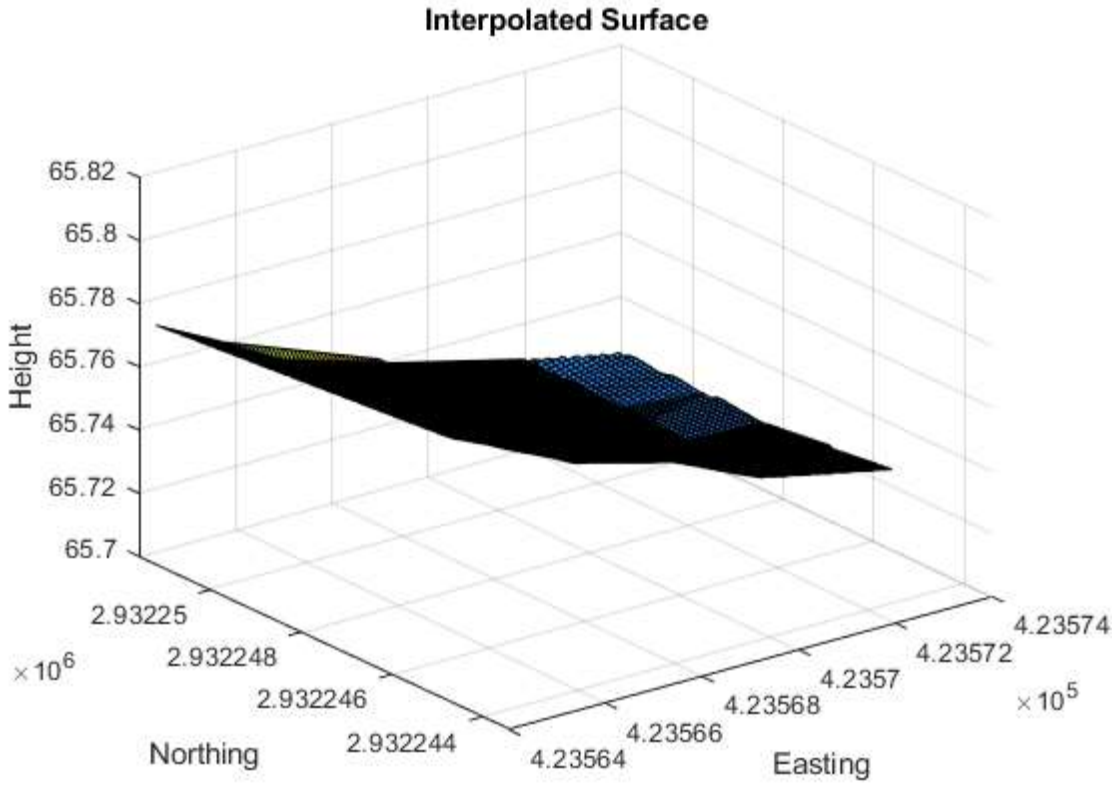
Plot GCPs and LiDAR points

```
figure;
scatter(gcps(:, 2), gcps(:, 1), 'b', 'filled'); % GCPs
hold on;
scatter(lidar_data(:, 2), lidar_data(:, 1), 'r', 'filled'); % LiDAR points
xlabel('Easting');
ylabel('Northing');
legend('GCPs', 'LiDAR Points');
title('GCPs and LiDAR Points');

% Plot Triangulated Irregular Network (TIN)
figure;
triplot(tri, gcps(:, 2), gcps(:, 1));
xlabel('Easting');
ylabel('Northing');
title('Triangulated Irregular Network (TIN)');

% Check points outside convex hull
in_hull = inpolygon(lidar_data(:, 1), lidar_data(:, 2), gcps(:, 1), gcps(:, 2));
outside_hull_indices = find(~in_hull);
%disp('Indices of LiDAR points outside convex hull:');
```

```
%disp(outside_hull_indices);

% Plot interpolated surface (optional)
[X, Y] = meshgrid(linspace(min(lidar_data(:, 1)), max(lidar_data(:, 1)), 100), ...
                  linspace(min(lidar_data(:, 2)), max(lidar_data(:, 2)), 100));
Z = tin_interp(X, Y);
figure;
surf(X, Y, Z);
xlabel('Easting');
ylabel('Northing');
zlabel('Height');
title('Interpolated Surface');

% Handle edge cases as necessary
```


GCPs and LiDAR Points
GCPs
LiDAR Points
×10^5
4.23576
4.23574
4.23572
4.2357
4.23568
4.23566
4.23564
4.23562
Northing
2.932244
2.932246
2.932248
2.93225
2.932252
Easting
×10^6


Triangulated Irregular Network (TIN)
×10^5
4.23574
4.23573
4.23572
4.23571
4.2357
4.23569
4.23568
4.23567
4.23566
4.23565
4.23564
Northing
2.932244
2.932246
2.932248
2.93225
2.932252
Easting
×10^6



Published with MATLAB® R2023b