RAY PATH DENSITY: THE NO RAYS PASSING THROUGH MESH GRID

For 2-D

Suppose n by n grid is chosen and many lines are passing through the boxes with definite starting and end points. The question is to find out the no of lines passing through each box.

Following are the steps undertaken to solve the problem.

The additional python module to install is Numpy

1. The grid is prepared based on the inputs from the user. Latitude range is chosen based on the lowest value of latitude, highest value of latitude and incremental width. Similarly, longitude range is chosen based on the lowest value of longitude, highest value of longitude and incremental width. The incremental width is chosen to be the same in both axes. The latitudes and longitudes values are stored in two lists named, *latitudes and longitudes.* Two lists generated will be used later on.
2. The next goal is to take inputs of the location of source and receiver. The lat, long information of source and receiver is stored in two separate text files; source.txt and receiver.txt. Two lists are generated named source and receiver which stores the lat, long values of sources and receivers.
3. A zero matrix is created to count the number of rays passing through each box. Integral numbers are assigned to latitudes and longitudes and stored in two dictionaries; lat_dict and long_dict.
4. The basic idea of the program is to find the midpoint of red (e.g. point B) and navy-blue dots (e.g. point A) as shown in figure 1. The midpoint (e.g. point C) will lie somewhere is the box and the line has passed through the box. Considering a

special case, if line is passing through corners then the midpoint of the consecutive points will definitely lie inside the box.

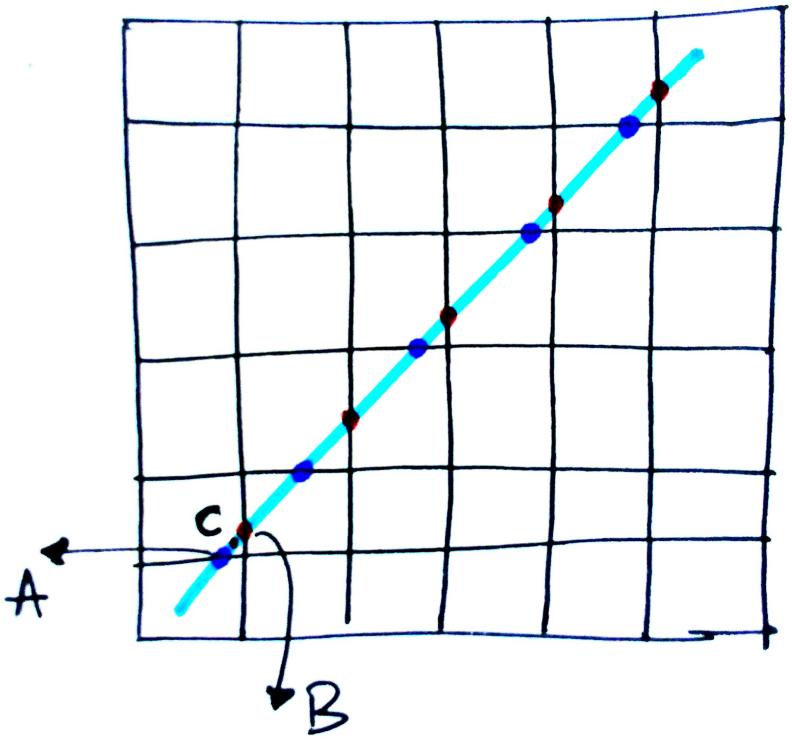

Figure 1 - showing the grids and the line passing through the grids

5. It is now required to find the points of the intersection of ray and the vertical lines (red points) and the intersection points of ray and the horizontal lines (navy-blue).

6. The source is assumed to lie inside a box and not on a grid point. A loop will run varying the latitude values and which are nothing but the vertical lines. To find the red dots, equation of line is written based on the given values of source and receiver coordinates. The value of y(longitude) is obtained for each x(latitude - as latitude values are stored in a list) the list [x,y] is stored in a list named *vertical_dots or red dots*. Similarly, to find the blue dots the equation of line is written by changing the coordinates. The [x,y] pair is stored in a list named horizontal_dots. Both the lists are combined which

is nothing but the coordinates of all the points of the intersection of line (or ray) with the grids.

7. The final list is arranged based on the distance of each point from the source. The sum of coordinates of the adjacent points gives the location of an interior point (of box) from which the line has passed.

8. The zero matrix which was created earlier each element corresponds to a box. T. If the ray has passed through a box, the corresponding elements of the matrix is assigned a number which one more than the existing number.

9. The list mid_point contains the coordinates of the points which lies inside a box. Based on the latitude and longitude values, and long are assigned integral values using the dictionaries.

10. Finally, the value of the corner coordinates of each box is stored in text file along with the weight-age assigned to the particular box.

Please refer to the appendix for the code.

For 3D

Suppose n by n by n grid is chosen and many lines are passing through the cubes with definite starting and end points. The question is to find out the no of lines passing through each cube.

Following steps are undertaken to solve the problem

1. Latitude, Longitude, Depth and incremental width are taken as the inputs for the user to create n by n by n mesh grid in three dimensions. The only valid inputs are accepted and invalid inputs are rejected. The list of latitude, longitude and depth values are stored in a list based on the two end points and the incremental width.

2. Two text files named *source_3D.txt* and *receiver_3D.txt* contains the three dimensional coordinates of the sources and stations. These two files are imported and the coordinates are stored in a list.
3. In order to count the no of rays passing through each differential cube, a matrix is assigned to each depth plane and the collection of matrices is stored in a list. Latitude, Longitude and Depth values in the list are assigned a whole no which makes it easier to access the matrix elements and update them.
4. A loop is run for each source and receiver pair; the value of depth (z) is varied and the corresponding latitude (x) and longitude (y) is obtained using the following equation of line.

$$\frac{x - x_1}{a} = \frac{y - y_1}{b} = \frac{z - z_1}{c}$$

5. If the source and receiver are at the same depth, a procedure similar to the 2D case is applied. Otherwise, the coordinates of the points of intersection of line connecting source and receiver and the depth planes are obtained. The midpoint of the adjacent points are calculated and stored in a list. The point has passed through the cube which contains the midpoint and the cube is assigned one more point to increase its weightage. Similar to the previous case, the coordinates of the corner points and the weightage assigned to the cube is extracted on a text file.