# Isolines From Scratch - An Exercise

Marcelo Fernandes
Software & Electrical Engineer
https://marcelofern.com

## I. Introduction

Isolines are contours drawn on graphical figures, connecting data points of the same value[1].

This report describes an implementation of the Marching Squares[2] algorithm to calculate isolines using the isovalue of 11 for a graph of 2 dimensions. This algorithm is introduced via interactive steps using an example matrix of dimensions 19x21 (rows x columns).

April 15, 2021

## II. Dataset

The dataset is displayed below:

```
[[162,145,130,117,106,97,90,85,82,81,82,85,90,97,106,117,130,145,162,181,200],
[145,128,113,100,89,80,73,68,65,64,65,68,73,80,89,100,113,128,145,164,185],
[130,113,98,85,74,65,58,53,50,49,50,53,58,65,74,85,98,113,130,149,170],
[117,100,85,72,61,52,45,40,37,36,37,40,45,52,61,72,85,100,117,136,157],
[106,89,74,61,50,41,34,29,26,25,26,29,34,41,50,61,74,89,106,125,146],
[97,80,65,52,41,32,25,20,17,16,17,20,25,32,41,52,65,80,97,116,137],
[90,73,58,45,34,25,18,13,10,9,10,13,18,25,34,45,58,73,90,109,130],
[85,68,53,40,29,20,13,8,5,4,5,8,13,20,29,40,53,68,85,104,125],
[82,65,50,37,26,17,10,5,2,1,2,5,10,17,26,37,50,65,82,101,122],
[81,64,49,36,25,16,9,4,1,0,1,4,9,16,25,36,49,64,81,100,121],
[82,65,50,37,26,17,10,5,2,1,2,5,10,17,26,37,50,65,82,101,122],
[85,68,53,40,29,20,13,8,5,4,5,8,13,20,29,40,53,68,85,104,125],
[90,73,58,45,34,25,18,13,10,9,10,13,18,25,34,45,58,73,90,109,130],
[97,80,65,52,41,32,25,20,17,16,17,20,25,32,41,52,65,80,97,116,137],
[106,89,74,61,50,41,34,29,26,25,26,29,34,41,50,61,74,89,106,125,146],
[117,100,85,72,61,52,45,40,37,36,37,40,45,52,61,72,85,100,117,136,157],
[130,113,98,85,74,65,58,53,50,49,50,53,58,65,74,85,98,113,130,149,170],
[145,128,113,100,89,80,73,68,65,64,65,68,73,80,89,100,113,128,145,164,185],
[162,145,130,117,106,97,90,85,82,81,82,85,90,97,106,117,130,145,162,181,200]]
```

## III. Algorithm

The algorithm is introduced through the sequential steps below:

1) Create a threshold matrix.
2) Map matrix cells to create corners around the isovalue threshold
3) Draw the mapped patterns on the graph
4) Perform a linear interpolation

### A. Creating the threshold matrix

The threshold matrix is created by going over the original matrix and substituting the original values by:

- 1 when the original value is above the isovalue.
- 0 when the original value is below or equal the isovalue.

For the isovalue of 11, we then have the following threshold matrix:

```
[[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

### B. Mapping the threshold matrix into squares

The threshold matrix can be mapped into squares, where each data value in the matrix is the vertex of a square. Each of the four corners of a square can either be true or false. When the threshold matrix value is 0, we map it to ”false”, and where the value is 1, we map it to ”true”. This gives us a total of $2^4 = 16$ different combinations of squares. These squares are further mapped against a look-up table of lines that will be drawn onto the original table.

The 16 distinct cases are:

0) All corners are True
1) All corners except the bottom left are True
2) All corners except the bottom right are True
3) Only the top corners are True
4) All corners except the top right are True
5) Left upper corner and low right corner are True
6) Left top/bottom corners are True
7) Only the top left corner is True
8) All corners except the top left are True
9) Right top/bottom corners are True
10) Top right corner and bottom left corner are True
11) Only the top right corner is True
12) Only the bottom corners are True
13) Only the bottom right corner is True
14) Only the bottom left corner is True
15) No corners are True

Each one of these cases is mapped to a line as depicted in Fig. 1.

### C. Drawing the patterns on the graph

Mapping each square from the threshold matrix using the lookup lines on Fig. 1. will then give us the graphical representation showed by Fig. 2.

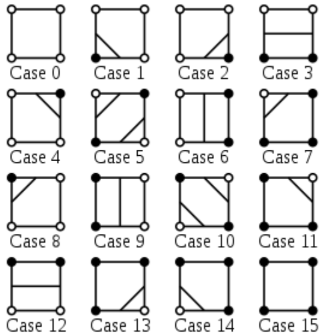


Fig. 1. Look-up lines

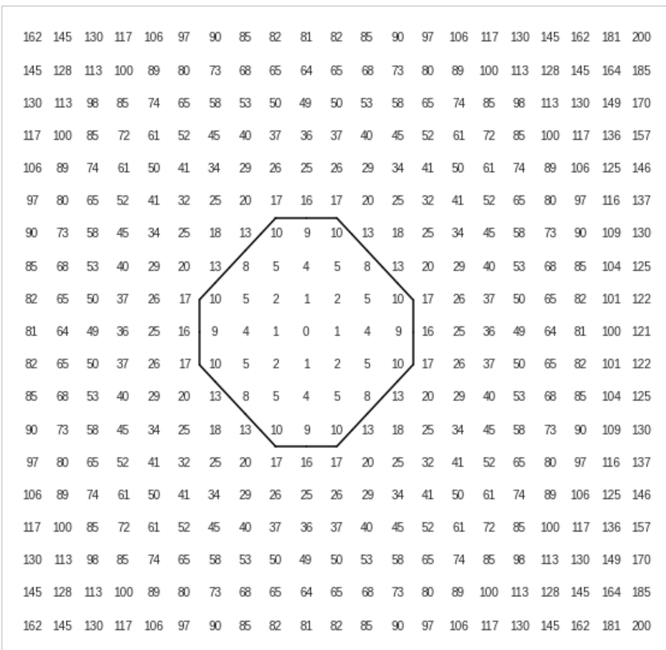


Fig. 2. Isolines before linear interpolation

### D. Interpolation

The last step is to use the linear interpolation technique to find out a more appropriate position for all the lines drawn from the lookup table. The manual result of this is represented on Fig. 3. and Fig. 4.

## IV. Isolines Using Python and Matplotlib

The Python package Matplotlib is ideal for quickly and efficiently generating isolines from a dataset. An example with various isovalues (10, 20, 30, 50, 80, 130, 180) is showed on Fig. 5. The code is provided on Appendix A.

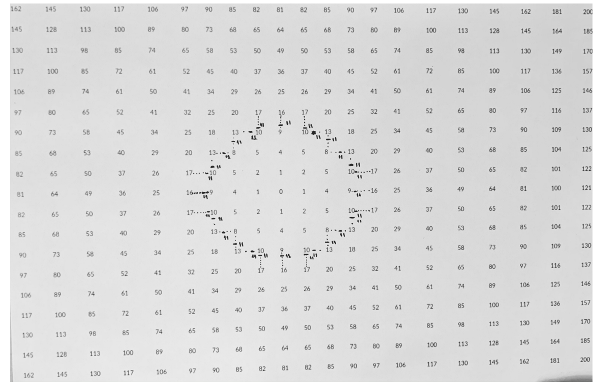

Fig. 3. Estimation points for the isovalue of 11

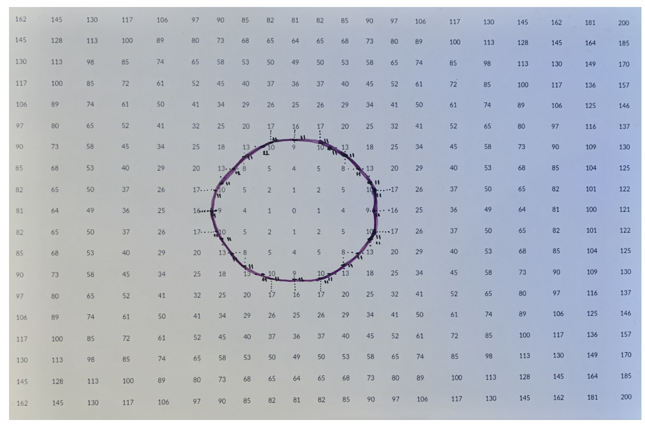

Fig. 4. Isolines after linear interpolation

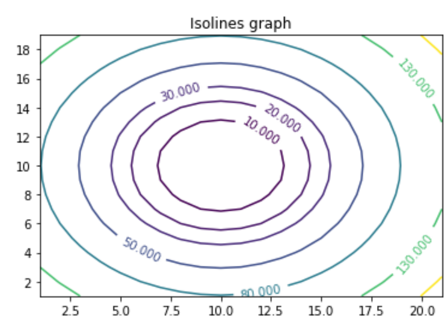


Fig. 5. Isolines using Matplotlib

## V. Appendix A

```
import matplotlib
import numpy as np
import matplotlib.cm as cm
import matplotlib.pyplot as plt

x = np.arange(1, 22, 1)
y = np.arange(1, 20, 1)
X, Y = np.meshgrid(x, y)

Z = [#...omitted to avoid breaking the template...]

fig, ax = plt.subplots()
CS = ax.contour(
    X, Y, Z,
```

```
    levels=[10, 20, 30, 50, 80, 130, 180]
)
ax.clabel(CS, inline=True, fontsize=10)
ax.set_title('Isolines graph')
```

## REFERENCES


[1] Courant, Robbins. and Stewart, What is mathematics?. Oxford: University press, 1996.

[2] C. Maple, "Geometric design and space planning using the marching squares and marching cube algorithms", 2003 International Conference on Geometric Modeling and Graphics, 2003. Proceedings. Available: 10.1109/gmag.2003.1219671 [Accessed 15 April 2021].

[3] Wikipedia marching squares picture. 2021. Available: https://en.wikipedia.org/wiki/File:Marching_squares_algorithm.svg [Accessed 15 April 2021].