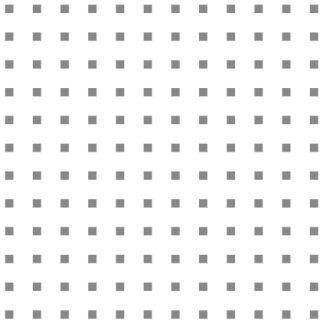
Code Sample – Fitting an Ellipse to points on a grid

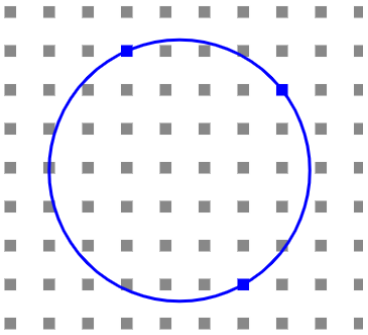
Constantine A. Raftopoulos, PhD

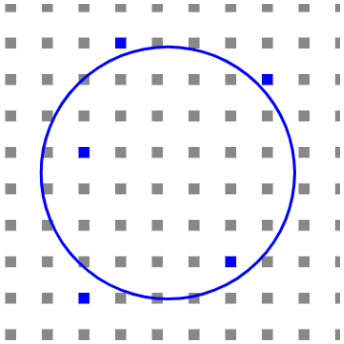
raftop@cs.ucla.edu

This is a program capable of fitting an Ellipse to chosen points on a grid. The system starts with a single window containing a 20x20 grid of square points similar to this:



All points start out gray. The user is allowed to toggle the individual points on the grid on and off. A button is added at the bottom of the window. When the user clicks this button, an ellipse that best fits the highlighted points is generated by means of an iterative, geometric least squares-based algorithm that does not rely on an external library or code to find the best fit. Such a fit is shown below for a circle.





This code fits an ellipse (least square sense) to the chosen by the user points on the grid using a guided random walk. The source code is in Python 3.8. Basic packages are used: numpy for vector matrix operations, random generators, pdist and cdist from distances package for fast Euclidean distance calculation. For Uix, Kivy is used for better integration with Python but also capability for mobile devices. PyInstaller was used to package for windows and linux executables.

An ellipse is uniquely defined by 5 parameters: Center coords, 2 axis lengths and a slope. A random walk is thus initiated in a 5 dimensional space each point in this space representing a different ellipse. The initial parameter estimation is performed by means of mean point coords for the center, max of in-between points distances for the bigger axis and the mean of the same distances for the smaller axis. For estimating the ellipse slope a PCA treatment is used, where the dominant eigenvector of the covariance of the points coords matrix indicates the direction of maximum variance of point projections therefore a good estimate of the initial ellipse slope.

The mean square error is calculated in each iteration by adding each point's distances to the foci and taking the squared residual from the large axis. This is because for a point on any ellipse the sum of its distances to the foci is equal to the large axis. The process iterates 100 times at max. Every time the button is pressed it further improves by trying 100 more times (or at least trying).