March 2024

Stochastic Optimization

Submitted by:

Nitay Vilner

Tomer Gal

Guy Green

Ron Chechik

Tal Dasht

• Motivation:

GIven N points in the (x,y) plane, saturate weights of given points to maximize their sum of weights and sum of squares of weights, while maintaining that the heaviest route will never exceed the longest.

- A route is a list of points for which each point is "bigger" than all of the previous points.
- A point is considered "bigger" than another point if both its axis are larger than the other's.

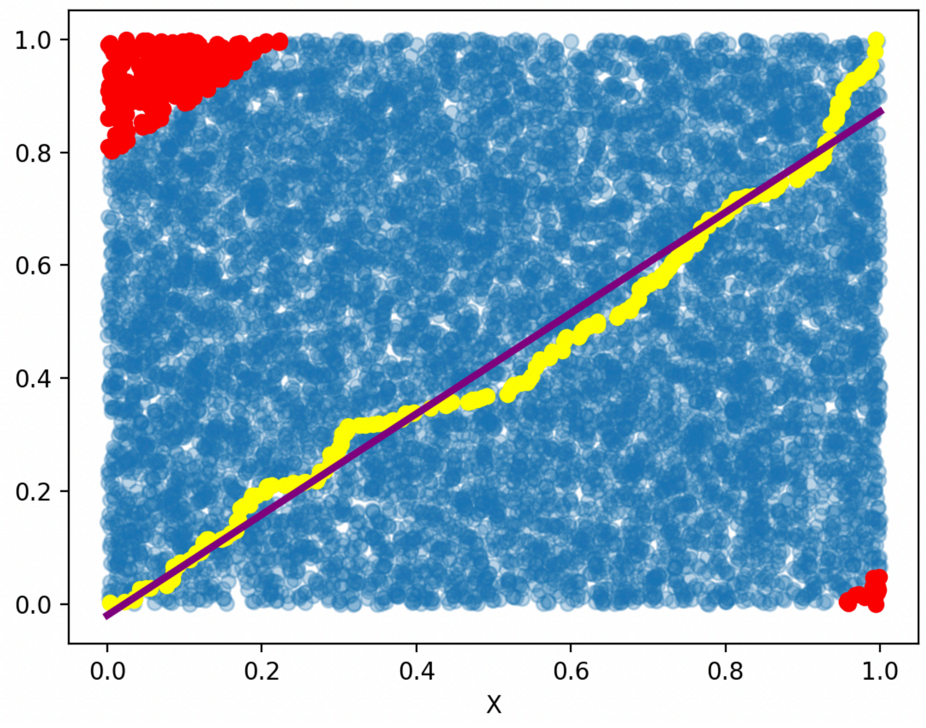
• The Problem:

GIven a big data set, how could we find a close-to-optimal solution which runs in a reasonable time-span?

• Idea \ Heuristics:

- We first find the list of points which make up the longest route in the given data set.
- By using **Least Squares** method we then find the line which fits the best the longest route.
- We sort all the points by their distance from the line, starting from furthest to nearest.
- For each point we find the heaviest route that goes through it, than we saturate the point's weight by the difference from the longest route in the data set to the heaviest route that goes through it.

The **idea** is to saturate all routes that are the furthest away from the center, which probably makes them the routes that "less interfere" with others.
That assumption is based on the fact that given a big data set, the longest path will probably pass somewhere in the middle of all points, as shown in the graph below:



Yellow dots make up the longest route,

and the purple line make up the line resulting from least squares approximation over the yellow dots.

- **Overview of the Algorithm:**

1. Read the set of data from given excel file.
2. Initialize a list of weights, starting with a weight of 1 for each point.
3. Sort the points first by their x in ascending order and then by their y in descending order, later to be used to find longest route and heaviest route for each point.
4. Find the longest route in the data set using **Longest Increasing Subsequence** algorithm, as seen on multiple sources online - O(nlogn).
5. Find the line which approximates best the longest route using **Least Squares** method via numpy library.
6. Sort all the points by their distance from the line, while remembering their original sort by their (x,y) axis. That is necessary for the Heaviest Increasing Subsequence algorithm.

   At that point in the code we give the option to view in a graph the longest route and approximating line (exists in a marked line).
7. For each point in the freshly sorted order, find the heaviest route that goes through it by using the **Heaviest Increasing Subsequence** algorithm, as published by *Guy Jacobson and Kiem-Phong Vo.* Saturate the point's weight by the difference from the longest route in the data set to the heaviest route that goes through it. - $O(n^2 log n)$.

8. Write back the resulting weights to the excel file using a dictionary which matches the weights to their corresponding points in the file.

- **Overall Runtime:** $O(n^2 logn)$.

- **Results:**

We get a fairly fast algorithm that runs around 60s for each given data-set of 10,000 points.

The results are as follows:

name: square_10000_samples_V0, sum: 21335, sum of squares: 840925
name: rhombus_10000_samples_V0, sum: 14060, sum of squares: 27902
name: square_10000_samples_V1, sum: 19204, sum of squares: 544282
name: rhombus_10000_samples_V1, sum: 14361, sum of squares: 29779
name: square_10000_samples_V2, sum: 19030, sum of squares: 520410
name: rhombus_10000_samples_V2, sum: 14185, sum of squares: 32249
name: square_10000_samples_V3, sum: 19652, sum of squares: 630644
name: rhombus_10000_samples_V3, sum: 14430, sum of squares: 29378
name: square_10000_samples_V4, sum: 19629, sum of squares: 658677
name: rhombus_10000_samples_V4, sum: 13950, sum of squares: 26810