Project 2 – Convex Hull

**Introduction**

Right out of the gate, I was not able to produce a functional algorithm. I’ve been staring at it for about 18 hours total, and I can’t seem to break through. At smaller values, it seems to be more correct. It will still make mistakes. At higher values, it appears to just pick a side and then connect x values across it. I wrote 3 different algorithms to perform the splitting step, and then 2 different main combine hull methods. I’ve deleted them from the project submission for clarity. The pseudocode for the algorithm is as follows:

**Pseudocode / Runtime Analysis**

Sort the points by increasing x values. – In python, O(nlogn)

Pass the points to a function that recursively divides the points into left and right halves. – Log(n) recursions

Pass the left and right halves into a function that combines the hulls together, starting with a base case of sizes 1,1 1,2 2,1 2,2.

Returns are assumed to be organized in a clockwise fashion, starting with the lower left hand point.

For size 1, 1, all we must do is return [leftpoint, rightpoint]. – linear (n)

For size 1, 2, we return [leftpoint, highestrightpoint, lowestrightpoint] – linear (n)

For size 2,1, we return [lowestleftpoint, highestleftpoint, rightpoint] – linear (n)

For size 2,2, we return [lowestleftpoint, highestleftpoint, highestrightpoint, lowestrightpoint] – linear (n)

For other sizes, we need to locate the rightmost point of the left hull, and the leftmost point of the right hull. – linear (n)

From there, we have to move the point on the left hull counterclockwise until we find the point where the slope of the line between the search point on the left hull and the search point on the right hull is no longer decreasing. From there, we back up a point and we choose that point as our top left tangent point. – linear (n)

Then, we shift the leftmost point of the right hull clockwise until we find the point where the slope between the two points is no longer increasing. Then, we back off a point and choose that as our top right tangent. Linear (n)

Next, we take the same two beginning points as above, except now, we:

Rotate the left hull point clockwise until the slope between the two points is no longer increasing, back up a point, and save. – Linear (n)

Rotate the right hull point counterclockwise until the slope between the two points is no longer decreasing, then back up and save that point. – Linear (n)

From here, we can return the combined hull by starting at the bottom left point, iterating clockwise through the left hull until we reach the top-left point, switching to the top right point, and iterating through the right hull until we reach the bottom right point. – This is linear (n).

We return the list made by the above step. – (1)

Overall:

The merge step takes linear time, O(n).

The divide step takes log time, O(Log(n)).

Sorting at the beginning of everything is O(nlogn).

So, the total runtime should be something like O(2nlogn) or just O(nlogn).

**Empirical Analysis**

Recording n values and the time reported by the UI, it appears to be purely linear in runtime, which is confusing to me. I’m chalking it up to my algorithm not quite being implemented correctly.

Chart, line chart

Description automatically generated

**Screenshots**

As I said earlier, my algorithm seems to get off by more and more as the input size gets larger. For certain values, it also tends to hang. Here are some screenshots of the program. These showcase the shortcomings of my work, but are interesting examples of what NOT to do nonetheless.

Chart

Description automatically generatedChart

Description automatically generatedChart

Description automatically generatedLine chart

Description automatically generatedChart, line chart

Description automatically generated with medium confidenceChart, line chart

Description automatically generated