Chris Bohlman

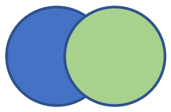
April 24, 2018

CSc 445

Algorithms Homework #6

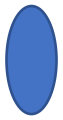
1.

a. Assume A and B are discs.
A U B could create a shape such as:



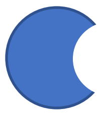
And this shape is **NOT convex**

b. Assume A and B are discs.
A ∩ B could create a shape such as:

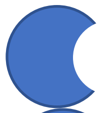


And this shape **IS convex**

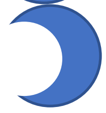
c. Assume A and B are discs
A / B could create a shape such as:
And this shape is **NOT convex**



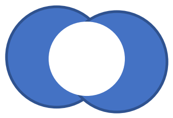
d. A / B =



B / A =



AΔB =



which is **not convex**

2.

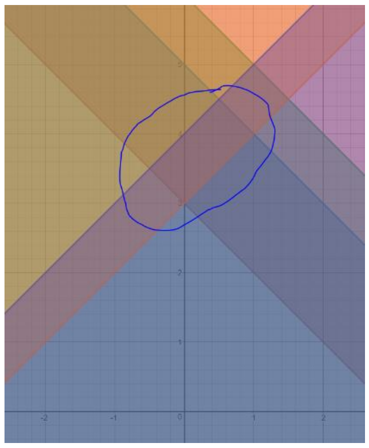
   a. Y >= x
      X >= 0
      Y <= 2

   b. Y >= 3x
      Y >= -3x
      Y >= 10

   c. Y >= 3x
      Y >= -3x
      Y <= 10

3. For this problem, you want to use and abuse the Simplex function.
   Simplex assumes that the feasible region points downwards and finds the local minimum of that region. Therefore, you can use the simplex algorithm to find (on a normal cartesian graph) the northernmost, southernmost, easternmost, and westernmost point. At each of those points, construct a line parallel to an axis: northmost/southernmost = x axis, easternmost/westernmost = y axis. And these 4 lines should give you your bounding box.

   To find southernmost point on cartesian grid, just run simplex and return result
   To find northernmost, reflect graph on x axis, run simplex, and take absolute value of y coordinate found.
   To find westernmost point, rotate graph by 90 degrees counterclockwise and run simplex. Fix signs of coordinates, and this should be the answer.
   To find easternmost point, rotate graph by 90 degrees clockwise and run simplex. Fix signs of coordinates, and this should be the answer.



4. The quadrilateral is the feasible area for this set of half planes. To maximize the cost function 1000x + 2y, it's very easy to see that we should probably pick the point with the largest x coordinate. In this quadrilateral, the point that maximizes the cost function is (1, 4).

5. O(n^2) trivial algorithm: check every point against every other point.

   Have an axis with several points plotted on it. Have parameter ε that is a given error margin, and you want to find if there is some point q where distance to every other point <= ε.

   Write the problem as an LP problem with one variable.

   Want for every point: $|q - p_i| <= ε$

   $|q - p_1| <= ε$

   $|q - p_2| <= ε$

   …

   $|q - p_n| <= ε$

   Preprocess data suck that every point makes 2 half planes, with $p_i - ε$ facing +x direction and $p_i + ε$ facing -x direction. being the two half planes on the x axis, run the LP algorithm with the 2n half planes, hopefully finding a feasible region with a point inside of it, which would be the most feasible point for this algorithm. If no point is in feasible region, or if the feasible region does not exist, terminate the algorithm.

   This algorithm, pursuant to our discussion of LP run time, runs in O(2n* α(2n)) time, which is approximately O(n) time.

6. Linear programming: Assume that a line exists
   Start with inserting point p_1 and q_1 and draw a line between them. The equation for this line should be $y = (y_2-y_1/x_2-x_1)x + b$, where b is the y intercept of the line. To find b, just plug one of the points into the equation for x and y and get b.
   This is allowed since the points can in fact be on line l.

   Next, insert every P and every Q one at a time, ensuring that every point falls where it's supposed to be in relation to the line.
   If it does, continue

   If it doesn't:

```
        If you've added point p:
                The line's slope has to decrease towards zero
                        Choose the p point just added and the previous q point, and make
                        that the new line through process described above
        If you've added point q:
                The line's slope has to increase.
                        Choose the q point just added and the previous p point, and make
                        that the new line through process described above
```

   Line as a cost function:
   For p: want $y >= \alpha x + \beta$
   For q: want $y <= \alpha x + \beta$
   Where we want to find the variables α and β, which is the slope and y intercept of the line.
   X and y are given to you already.
   You must randomize the point order. Doing so will allow the expected (predicted) runtime to be O(n).
   Worst case running time would be picking the least optimal path of points, so a new line is generated every single iteration: O(n^2) running time

7. Checking every single point results in quadratic run time, but run time isn't restricted

Basically, what we want to do is fit a line to these points.
$|y_i-\alpha x_i-\beta| \le \varepsilon$ for all n points.
The variables are α and β while ε and all points are given

```
Algorithm:
Start with 2 points and draw a line between them (same method as in problem 6). Get
    values of $\alpha$ and $\beta$ from that.
Then, continue to add points.
For each point:
    Check if (using previous values) $|y_i-\alpha x_i-\beta| \le \varepsilon$
    If point passes, continue to next point
    Else you must redraw the line.
            If point $y_i-\alpha x_i-\beta > \varepsilon$,
                    You must redraw the line somehow in order to increase the slope but go
                    back and check other points to see if they are all still within the
                    parameters of the best fit line. This takes O(n) time in worst case.
            If point $y_i-\alpha x_i-\beta < \varepsilon$,
                    You must redraw the line somehow in order to decrease the slope but go
                    back and check other points to see if they are all still within the
                    parameters of the best fit line. This takes O(n) time in worst case.
```

Total time this takes: Worst case is O(n^2). Can't be O(n), because you have to go back and re check all of the points previously added to the best fit line.

Don't know if this is the fastest algorithm. Don't know intricacies of re drawing line, but I assume it's close to my problem #6 solution. I just am not sure how to redraw if new line is unsatisfactory and how to deal with outlier points.