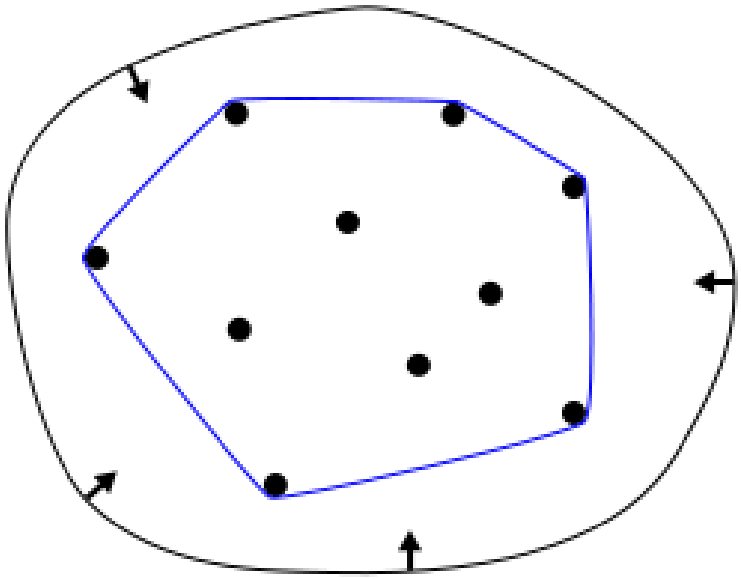
# Convex Hull

Given a set of points $Q$, we may want to find the convex hull, which is a subset of points that form the smallest convex polygon where every point in $Q$ is either on the boundary of the polygon or in the interior of the polygon. We can imagine fitting an elastic band around all of the points. When the band tightens, the points that it rests on form the convex hull.



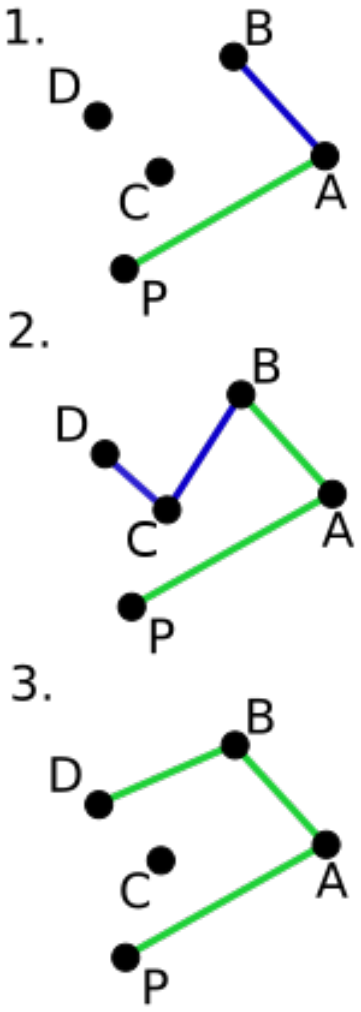
There are several algorithms to solve the convex hull problem with varying runtimes.

## Graham's Scan

The Graham's scan algorithm begins by choosing a point that is definitely on the convex hull and then iteratively adding points to the convex hull.

1. Let $H$ be the list of points on the convex hull, initialized to be empty
2. Choose $p_0$ to be the point with the lowest y-coordinate. Add $p_0$ to $H$ since $p_0$ is definitely in the convex hull.
3. Let $(p_1, p_2, ..., p_n)$ be the remaining points sorted by their polar angles relative to $p_0$ from smallest to largest. Iterating through the points in sorted order should sweep around $p_0$ in counterclockwise order
4. For each point $p_i$:
   (a) If adding $p_i$ to our convex hull results in making a "left turn", add $p_i$ to $H$
   (b) If adding $p_i$ to our convex hull results in making a "right turn", remove elements from $H$ until adding $p_i$ makes a left turn, then add $p_i$ to $H$.

Below is an example of left and right turns. At step 1, our convex hull $H$ is $(P, A, B)$. Adding $C$ resulted in making a left turn at $B$ if we're going from $A$ to $C$, so we add $C$ to $H$, which is now $(P, A, B, C)$. Adding $D$ in step 2 resulted in making a right turn at $C$ if we're going from $B$ to $D$. If we just added $D$ now, we would not have a convex shape. To fix this, we remove elements from $H$ until adding $D$ results in making a left turn. Simply removing $C$ does the trick and we add $D$ after $C$ is removed from $H$. This results in the convex hull solution of $H = (P, A, B, D)$.




Note that there may be cases where we have to remove multiple points from $H$ in order to fix the convexity. Once we iterate through every point, $H$ will form the convex hull.

The bottleneck of the algorithm is sorting the points by polar angles. This operation requires $O(n \log n)$ time. Since every point is added to $H$ exactly once and every point is removed from $H$ at most once, iterating through the points and forming $H$ after the sorting takes $O(n)$ time. Thus, the whole algorithm takes $O(n \log n)$ time.

## Jarvis' march

The Jarvis' march algorithm conceptually is very similar to Graham's scan. Again, we sort the points by their y-coordinates and choose $p_0$ in the same fashion as before. This time, we find the next point in the convex hull by iterating through all $n$ other points and discovering the next point with the smallest relative polar angle to the last point added to the convex hull. The difference here is that each point we add is definitely on the convex hull, as opposed to Graham's scan where each point we add may need to be removed later. The tradeoff is that after each addition to the convex hull, we need to iterate through every other point to find the next point on the convex hull, which takes $O(n)$ time. The algorithm terminates once we try to add $p_0$ to $H$ again.

The runtime of this algorithm is output-sensitive, meaning that the runtime depends on how many points are on the convex hull solution. For each point on the convex hull, we need to spend $O(n)$ time to iterate through all of the other points. Thus, if there are $O(h)$ points on the convex hull, the runtime will be $O(nh)$. If $h = o(\lg n)$, i.e. the convex hull is formed by a small portion of the total points, then Jarvis' march outperforms Graham's scan.