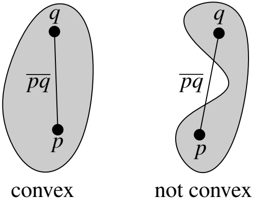
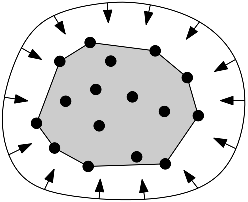
Convex Hull



- A point set S is convex if for all points p and q in S the line segment $\overline{pq}$ is in S.

Polygons



- The convex hull of a polygon is the convex hull of its vertices.
- It is the smallest polygon that contains all the vertices.
- More precisely, it is the intersection of all such polygons.
- Intuition: shrink wrap the polygon.

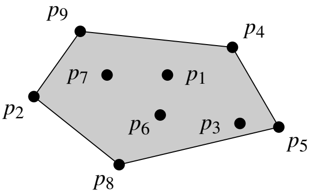
Problem Statement

input = set of points:

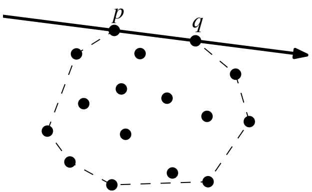
$p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$

output = representation of the convex hull:

p_4, p_5, p_8, p_2, p_9

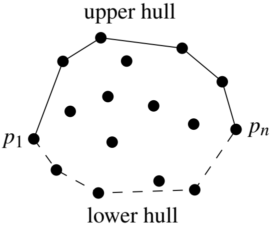


Hull Edge



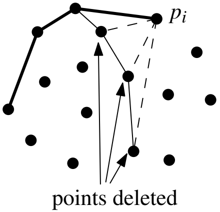
- pq is a hull edge if every other point r lies on the same side of its line.
- $\mathrm{circ}(p, q, r)$ is negative if the hull is listed in clockwise order and is positive otherwise.

Upper Hull



- The upper hull is the edges whose supporting lines are above all the other points.
- It consists of a polygonal curve from the leftmost point to the rightmost point.
- The lower hull is the edges whose supporting lines are below all the other points.
- It consists of a polygonal curve from the rightmost point to the leftmost point.
- Algorithm: construct the two hulls then append them.

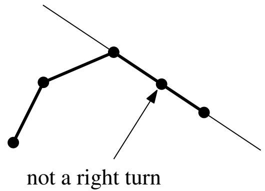
Upper Hull Algorithm



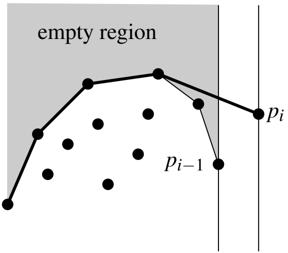
1. Sort the points in increasing x order: $p_1, \ldots, p_n$.
2. Initialize an empty hull $h = ()$.
3. For $i = 1$ to n
   - 3.1 Append p_i to h.
   - 3.2 While h contains $m \geq 3$ points and $\operatorname{circ}(h_{m-2}, h_{m-1}, h_m) > 0$
     - 3.2.1 Set h_{m-1} to h_m.
     - 3.2.2 Remove the last element of h.

Degenerate Cases

- Degeneracy 1: points with equal x coordinates.
- Handling: sort by y coordinate.
- Degeneracy 2: collinear points.
- Handling: treat as left turn.



Correctness



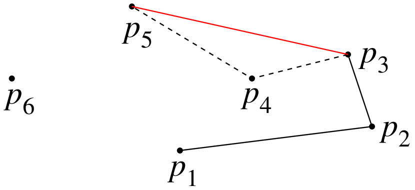
Inductive proof of correctness for i points.

- $i = 1$ is trivial.
- Assume $i - 1$.
- The update creates a curve h from p_1 to p_i with right turns.
- Consider a point p_j with $j < i$ that is not in h.
- p_j is not above the $i - 1$ hull by inductive hypothesis.
- The curve h is above the $i - 1$ hull.
- Hence, p_j is not above h.

Complexity

- Sorting the points takes $\mathrm{O}(n \log n)$ time.
- Each point is removed at most once from h.
- Hence, the time spent on updating the hull is $\mathrm{O}(n)$.
- Thus, the running time is $\mathrm{O}(n \log n)$.
- The space complexity is $\mathrm{O}(n)$.
- These bounds are optimal.

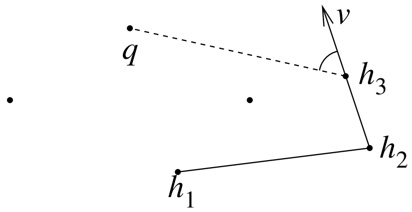
Improved Version



Construct the entire hull with one subroutine.

1. Set p_1 to the lowest point.
2. Sort the other points counterclockwise around p_1.
3. Construct the hull as before, but keeping left turns.

Gift Wrapping Algorithm



1. Initialize the hull to $h = (p)$ with p the lowest point.
2. Initialize v to $(1, 0)$.
3. Repeat
   3.1 Let $h = (h_1, \dots, h_i)$.
   3.2 Find the point q that minimizes the angle $\angle(v, q - h_i)$.
   3.3 Append q to h.
   3.4 Set v to $q - h_i$.
   3.5 If $h_1 = q$ return h.

Analysis

- All n points can be on the hull.
- Adding a point to the hull takes $\mathrm{O}(n)$ time.
- Hence, the running time is $\mathrm{O}(n^2)$.
- The running time is also $\mathrm{O}(nm)$ with m the size of the hull.
- Gift wrapping is faster than the $\mathrm{O}(n \log n)$ algorithms when most of the points are in the interior of the hull.
- An algorithm whose running time depends on the output size is called output sensitive.