Recap:
- Given a pt. set $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^2$
  - Compute $\text{conv}(P)$ - smallest convex set containing $P$
- Output: Cyclic sequence of hull vertices
- Algorithms: Graham's Scan $O(n \log n)$
  - Divide & Conquer $O(n \log n)$, Jarvis March $O(nh)$

This Lecture:
- Can we beat $O(n \log n)$ time?
  - We'll give an $\Omega(n \log^2 h)$ lower bound
- Can we achieve $O(n \log h)$?
  - Chan's algorithm

- Good when number of hull vertices $h$ is very small

Theorem: Given $n$ pts unif. distributed in a unit square $E[h] = \log n$
(Chan runs in $O(n \log \log n)$)
Lower bound for convex hulls:

Conv: Given a set P of n pts in \( \mathbb{R}^2 \), compute the vertices of \( \text{conv}(P) \) in cyclic order.

Def: An algorithm is comparison-based if its decisions are based on the sign of a fixed-degree polynomial function of inputs. (Algebraic decision tree model)

Almost all geometric primitives satisfy:

Eg. \( \langle p, q, r \rangle \) form a left-hand turn:

\[
\text{if}( \det \begin{pmatrix} 1 & p_x & p_y \\ 1 & q_x & q_y \\ 1 & r_x & r_y \end{pmatrix} > 0 )
\]

\[
= \text{if}( f(p_x, p_y, q_x, q_y, r_x, r_y) > 0 )
\]

where:

\[
f(\ldots) = (q_x r_y - q_y r_x) - (p_x r_y - p_y r_x) + (p_x q_y - p_y q_x)
\]

A polynomial of degree 2
Theorem: Assuming a comparison-based algorithm, convex has a worst-case lower bound of $\Omega(n \log n)$.

Proof: We will use the well-known fact that any comparison-based algorithm for sorting requires $\Omega(n \log n)$ time in worst case.

We'll reduce sorting to convex. Given set $X = \{x_1, \ldots, x_n\}$ to be sorted in $O(n)$ time, we generate $P = \{p_1, \ldots, p_n\}$ where $p_i = (x_i, x_i^2)$.

If we compute $\text{conv}(P)$, the vertices appear in sorted order of $X$, up to reversal and adjusting starting point in $O(n)$ time.
Letting $T(n)$ denote the time to compute $\text{conv}(P)$, up to constant factors, we can sort $X$ in time $n + T(n) + n$, which must be $\geq c \cdot n \log n$

$\Rightarrow T(n) \geq c \cdot n \log n - 2n \Rightarrow T(n) = \Omega(n \log n)$

\[ \square \]

\textbf{Obs:} This exploits the fact that output is sorted cyclically. What if not?

\textbf{Theorem:} Assuming a comparison-based algorithm determining whether $\text{conv}(P)$ has $h$ distinct vertices requires $\Omega(n \log h)$ time.

$\Rightarrow$ Just counting vertices reqs. log factor.

(See latex lecture notes for proof)

\textbf{Output Sensitivity:} Algorithm's running time depends on output size

$\rightarrow$ Is $O(n \log h)$ possible?

Yes!

- Chan's Algorithm
- combines - Graham scan $O(n \log n)$
- + Jarvis March $O(nh)$
Chan's Algorithm: An $O(n \log h)$ algorithm
- Optimal w.r.t. input size $n$ and output size $h$
- Combines two slow algorithms (Graham + Jarvis) to make faster algorithm
- Chicken + Egg: Algorithm needs to know value of $h$ - How is this possible?

Tangent Lemma:
Given a convex polygon $Q$ given as a cyclic sequence of $m$ vertices $\langle q_1, \ldots, q_m \rangle$ and $P \not\subset Q$, can compute tangent vertices $q^- + q^+$ w.r.t. $P$ in time $O(\log m)$

How? Exercise
Hint: Variant of binary search
How to achieve $O(n \log h)$?

- Can't sort any set of size $>> h$
- Guess the hull size - $h^*$
- Partition $P$ into $\left\lceil \frac{n}{h^*} \right\rceil$ groups, each of size $\leq h^*$
  \[ P_1, \ldots, P_k, \quad k = O\left(\frac{n}{h^*}\right) \rightarrow O(n) \]
- Run Graham on each group forming $k$ mini-hulls $H_1, \ldots, H_k$
  \[ \rightarrow O(k \cdot h^* \log h^*) = O(n \log h^*) \]
- If we guess right ($h^* = h$) $\rightarrow O(n \log h)$

- Run Jarvis, but treat each mini-hull as a "fat point"
- Use the utility function to compute turning angles

Example: Suppose $k = 5$
Merging Mini-hulls:

- By the Tangent Lemma, compute tangents \( q_j^+ + q_j^- \) for each \( H_j \) in time \( O(\log h^*) \)
- Compute all tangents in time \( O(k \cdot \log h^*) \)
- \( v_i \leftarrow \) tangent with smallest turning angle
- Terminates after \( h \) iterations

\[ \Rightarrow \text{Total merge time}: \ O(h \cdot k \cdot \log h^*) \]

- If we guess right \( (h^* = h) \) then
  \[ O(h^* \left( \frac{h^*}{k} \right) \log h^*) = O(n \log h^*) \]
  \[ = O(n \log h) \]

Summary: If we guess correctly \( (h^* = h) \) this computes \( \text{conv}(P) \) in time \( O(n \log h) \).
Conditional Hull \((P, h^*)\):

**Mini-hull Phase:** \(O(n \log h^*)\)

**Merge Phase:** \(O(n \frac{h}{h^*} \log h^*)\)

**If** \(h^* > h\) \(\Rightarrow\) **Mini-hull phase is too slow**

- Note: Can tolerate a polynomial error. E.g., if \(h \leq h^* \leq h^2\)
  \(\Rightarrow O(n \log h^*) = O(n \log (h^2)) = O(2 \cdot n \log h) = O(n \log h)\) ok.

**If** \(h^* < h\) \(\Rightarrow\) **Merge phase too slow**

- If Jarvis finds more than \(h^*\) hull pts - stop + return fail status
  \(\Rightarrow O(n \log h^*)\) time

**Strategy:**

**Start** small and increase until success

- **Arithmetic:** \(h^* = 3, 4, 5, \ldots\) way too slow \(\Rightarrow O(n \cdot h \cdot \log h)\)
- **Exponential:** \(h^* = 4, 8, 16, \ldots, 2^i\) better \(\Rightarrow O(n \log h)\)
- **Double Exponential:** \(h^* = 4, 16, 256, \ldots, 2^{2^i}\) best!

**Note:** \(h_i^* = 2^{2^i} \quad h_i^* \leftarrow (h_i^*)^2\)
Final Algorithm: Chan Hull ($P$):

\[
h^* = 2 \\
\text{repeat} \\
\quad h^* \leftarrow (h^*)^2 \quad \Rightarrow \quad h^*_i = 2^{2^i} \\
\quad (\text{status}, V) \leftarrow \text{conditional Hull}(P, h^*) \\
\text{until} (\text{status} == \text{success}) \\
\text{return } V
\]

Correctness: Already explained

Time:
- Running time per iteration $O(n \log h^*)$
- $h^*_i = 2^{2^i}$
- Stops when $h^* \geq h$
  \[2^{2^i} \geq h \Rightarrow i = \lceil \log \log h \rceil \text{ iterations}\]
- Total time: [up to constants]
  \[\sum_{i=1}^{\lceil \log \log h \rceil} n \cdot \log (2^{2^i}) = n \sum_{i=1}^{\lceil \log \log h \rceil} 2^i \leq 2n \cdot 2^{\log \log h} \leq 2n \log h \leq O(n \log h) \]

\(\smile\)
**Lower Bound:** (Optional)

**Convex Hull Size Verification (CHSV):**

Given a planar point set $P$ of size $n + \text{int } k$, does $\text{conv}(P)$ have $h$ vertices?

**Thm:** CHSV requires $\Omega(n \log h)$ time to solve (worst case in the algebraic-tree decision model)

**Take away:** Just counting num. of hull vertices takes $\Omega(n \log h)$ time.

**Proof (sketch):**

**Multiset Verification Problem (MSV):**

Given a set $S$ of $n$ real numbers and integer $k$, does $|S| = k$?

**Known:** MSV has lower bound of $\Omega(n \log k)$

Can reduce MSV to CHSV in linear time

Map $z_i \rightarrow (z_i, z_i^3)$