Given points in \( d \)-dimensional space, find a good "container" = convex polyhedron

Applications: collision detection, pattern recognition, motion planning.

2D - rubber band around pts — see slides.

More formally:

A set \( S \) is **convex** if \( \forall \) pts \( p, q \in S \), all pts on line segment \( pq \) are also in \( S \).

**Def:** Convex hull \( (P) = CH(P) = \)

intersection of all convex sets containing \( P \)

**Note:** \( CH(P) \) is convex.

The fact that \( CH(P) \) is a convex polyhedron with vertices in \( P \) needs proof — LATER

---

Convex hull algorithms in 2D

- almost any algorithmic paradigm will work, so this problem is great for algorithm courses.

**Incremental Algorithm** — add pts in \( x \)-order

**general situation:**

- We have \( CH(\{p_1, \ldots, p_{i-1}\}) = H_{i-1} \) — convex hull
- as a doubly linked list
- \( p_{i-1} \) is a vertex of \( H_{i-1} \)

Want \( H_i = CH(\{p_1, \ldots, p_i\}) \)

Must find upper & lower bridge — edges incident to \( p_i \) in \( H_i \)

and their other endpoints \( p_u \& p_e \).
Scan forward (clockwise) around \( H_{i-1} \) from \( p_{i-1} \) to find lower bridge to \( p_e \)

Scan backward around \( H_{i-1} \) from \( p_{i-1} \) to find upper bridge to \( p_u \)

Note that segment \( p_i p_{i-1} \) is outside \( H_{i-1} \) because \( p_{i-1} \) is rightmost pt of \( H_{i-1} \).

As we scan forward around \( H_{i-1} \), how do we identify \( p_e \)?

Test: is \( p_s \) above/below line \( p_i p_r \)

Timing Analysis

Adding one point can take \( \Theta(n) \) time, so total is \( O(n^2) \). Is that bad?

Amortized Analysis: each input point is deleted from the \( \mathcal{H} \) at most once at \( O(1) \) cost.
So total time is \( O(n) + \text{time to sort} = O(n \log n) \)
Graham's Algorithm (another sorting-based alg.)
sort pts. radially around some point X inside
the convex hull. (how to compute X ?)

Then start from \( p_1 = \min x \) coordinate
and add the points in sorted order,
repeatedly removing the 2nd last point
if it forms a reflex angle

e.g. here we remove \( p_1 \), then \( q \)

Note: to find \( X \), take avg. of 3 pts not on line.
Note: to sort radially, do not compute angles, use sidedness.

---

Divide and Conquer.
- divide pts in two by vertical line (sort(once!) by x coord)
- recurse on each side
- combine ?

To combine:
Find upper and lower
bridges.
Start with line segment
from max pt on left
to min pt on right.
Walk up/down triangle
by triangle
(similar to incremental)
Combine step takes $O(n)$.

Total time $T(n) = 2T\left(\frac{n}{2}\right) + c \cdot n$ 
+ time for initial sort

Solves to $O(n \log n)$ 
(prove by induction).

---

Lower Bound

$\Omega(n \log n)$ to compute ordered convex hull in 2D 
on a RAM with $+, -, \times$

Proof: Reduce sorting to finding the convex hull

map points to parabola 
$x \rightarrow x^2$

convex hull gives sorted order

input points we want to sort

Note: Even finding the CH vertices (unsorted) 
takes $\Omega(n \log n)$ - different pf.
Output-sensitive algorithm

Idea:

Express runtime as function of $n$, input size

Gift-wrapping (Jarvis's March) $- h$, output size

$p_1 = \min x$ then $\max y$

$l_i = \text{vertical line through } p_i$

"wrap" line (rotate thru $p_i$) until it hits the first point $p_2$

Each "wrap" is like finding a max

Compare $p_k$, $p_e$ by testing:

is $p_e$ above/below $p_i p_k$

Time for wrap $O(n)$

Total time $O(n^2)$ in worst case

But as a function of $n$ and $h$, size of CH:

$O(n \cdot h)$
What is the best alg. in terms of \( n \) and \( h \) ?
\( O(n \log h) \) - first developed in '86, Kirkpatrick & Seidel improved by Timothy Chan '96

Chan's Algorithm
Assume \( h \) is known (will fix this later)
\( m = h \)
Partition the points into \( \lceil \frac{n}{m} \rceil \) subsets of \( \leq m \) points each (arbitrarily)
Find CH of each subset using, e.g. Graham's Alg.

Time so far \( O(\frac{n}{m} \cdot m \log m) = O(n \log m) \)

Now run Gift Wrapping, but for the wrap step, don't check all \( n \) points

We only need to check the most extreme pt. (wrt rotating line thru \( p_i \)) of each of the \( \lceil \frac{n}{m} \rceil \) convex hulls

How to find the most extreme pt of a convex hull:

Use binary search
Have subchain of candidates
Test midpoint to cut chain in half
Use sidedness tests
Time for wrap step: \( O\left( \frac{n}{m} \log m \right) \)
If we do all \( h \) wrap steps we find the CH total time is \( O\left( \frac{h}{m} \frac{n}{m} \log m \right) + O\left( n \log m \right) \)
good if \( h = m \) — get \( O\left( n \log h \right) \)

How do we find the right \( m \)?
we will try out values of \( m \).

Careful: if \( m \) is small \( O\left( \frac{h}{m} \frac{n}{m} \log m \right) \) is too big (more than \( O(hn) \))
So stop gift-wrapping after \( m \) steps.
Then time to try \( m \) is \( O(m \frac{n}{m} \log m) = O( n \log m ) \)
If \( m > h \) we compute the CH

How do we find the right \( m \)?
Related problem
Search in a sorted but unbounded array of distinct natural numbers
(in bounded array \( A[1..k] \) can search in \( \log k \) steps)

use doubling technique:
Try \( i = 1, 2, 4, 8 \ldots \)
will find \( A(2^i) \leq x \leq A(2^{i+1}) \)
\( k \)-value we want
\[ \log x + 1 \text{ steps.} \]
Then use binary search — another \( \log x \)
So the idea is to try an increasing seq. of $m$ values $m = 2, 4, 8, 16, \ldots, 2^i, \ldots$ until we get one bigger than $h$.

First try

Time $\geq \sum_{i=1}^{\log h} n \log 2^i = n \sum_{i=1}^{\log h} i = n \log h \cdot i$. Too big!

Second try

Time $\geq \sum_{i=1}^{\log h} n \log 2^{2^i} = n \sum_{i=1}^{\log h} 2^i = n \log h$.

$2^i \geq h \Rightarrow i \geq \log h$. This works!

Next: Convex hull in higher dimensions.