CS-235
Computational Geometry

Subhash Suri

Computer Science Department
UC Santa Barbara

Fall Quarter 2002.
1. **Convex hulls** are to CG what sorting is to discrete algorithms.

2. First order **shape approximation**. Invariant under rotation and translation.

3. **Rubber-band analogy.**

4. Many applications in robotics, shape analysis, line fitting etc.

5. **Example:** if $CH(P_1) \cap CH(P_2) = \emptyset$, then objects $P_1$ and $P_2$ do not intersect.

6. **Convex Hull Problem:**
   Given a finite set of points $S$, compute its convex hull $CH(S)$. (Ordered vertex list.)
Classical Convexity

1. Given points $p_1, p_2, \ldots, p_k$, the point $\alpha_1 p_1 + \alpha_2 p_2 + \cdots + \alpha_k p_k$ is their convex combination if $\alpha_i \geq 0$ and $\sum_{i=1}^{k} \alpha_i = 1$.

2. $CH(S)$ is union of all convex combinations of $S$.

3. $S$ convex iff for all $x, y \in S$, $xy \in S$.

4. $CH(S)$ is intersection of all convex sets containing $S$.

5. $CH(S)$ is intersection of all halfspaces containing $S$.

6. $CH(S)$ is smallest convex set containing $S$.

7. In $R^2$, $CH(S)$ is smallest area (perimeter) convex polygon containing $S$.

8. In $R^2$, $CH(S)$ is union of all triangles formed by triples of $S$.

9. These descriptions do not yield efficient algorithms. At best $O(N^3)$. 
1. Start with bottom point $p$.

2. Angularly sort all points around $p$.

3. Point $a$ with smallest angle is on $CH$.

4. Repeat algorithm at $a$.

5. Complexity $O(Nh)$; $3 \leq h = |CH| \leq N$. Worst case $O(N^2)$. 
Quick Hull Algorithm

1. **Form initial quadrilateral** $Q$, with left, right, top, bottom. **Discard points inside** $Q$.

2. **Recursively**, a convex polygon, with some points “outside” each edge.

3. **For an edge** $ab$, find the farthest outside point $c$. **Discard points inside triangle** $abc$.

4. **Split remaining points into “outside” points for** $ac$ and $bc$.

5. **Edge** $ab$ on CH when no point outside.
1. Initial quadrilateral phase takes $O(n)$ time.

2. $T(n)$: time to solve the problem for an edge with $n$ points outside.

3. Let $n_1, n_2$ be sizes of subproblems. Then,

$$T(n) = \begin{cases} 
1 & \text{if } n = 1 \\
1 + T(n_1) + T(n_2) & \text{where } n_1 + n_2 \leq n
\end{cases}$$

4. Like QuickSort, this has expected running time $O(n \log n)$, but worst-case time $O(n^2)$. 
Graham Scan

1. Sort by Y-order; \( p_1, p_2, \ldots, p_n \).
2. Initialize. \textbf{push} \((p_i, stack), i = 1, 2\).
3. \textbf{for} \( i = 3 \) \textbf{to} \( n \) \textbf{do}
   \begin{align*}
   & \text{while} \angle \text{next, top, } p_i \neq \text{Left-Turn} \\
   & \text{pop} \ (stack) \\
   & \text{push} \ (p_i, stack).
   \end{align*}
4. \textbf{return} \( stack \).
5. Invented by R. Graham ’73. (Left and Right convex hull chains separately.)
Analysis of Graham Scan

1. **Invariant:** $\langle p_1, \ldots, \text{top}(\text{stack}) \rangle$ is convex. On termination, points in stack are on $CH$.

2. **Orientation Test:** $D = \begin{vmatrix} 1 & p_x & p_y \\ 1 & q_x & q_y \\ 1 & r_x & r_y \end{vmatrix}$

   $\angle p, q, r$ is LEFT if $D > 0$, RIGHT if $D < 0$, and straight if $D = 0$.

3. **After sorting, the scan takes $O(n)$ time:** in each step, either a point is deleted, or added to stack.
• **Sort points by** \( X \)-coordinates.

• **Let** \( A \) **be the set of** \( n/2 \) **leftmost points, and** \( B \) **the set of** \( n/2 \) **rightmost points.**

• **Recursively** compute \( CH(A) \) **and** \( CH(B) \).

• **Merge** \( CH(A) \) **and** \( CH(B) \) **to obtain** \( CH(S) \).
Merging Convex Hulls

• \( a = \) rightmost point of \( CH(A) \).

• \( b = \) leftmost point of \( CH(B) \).

• while \( ab \) not lower tangent of \( CH(A) \) and \( CH(B) \) do

  1. while \( ab \) not lower tangent to \( CH(A) \) set \( a = a - 1 \) (move \( a \) CW);
  2. while \( ab \) not lower tangent to \( CH(B) \) set \( b = b + 1 \) (move \( b \) CCW);

• Return \( ab \)
Analysis of D&C

- Initial sorting takes $O(N \log N)$ time.
- Recurrence for divide and conquer
  $T(N) = 2T(N/2) + O(N)$
- $O(N)$ for merging (computing tangents).
- Recurrence solves to $T(N) = O(N \log N)$. 
Applications of CH

A problem in statistics

• Given a set of $N$ data points in $R^2$, fit a line that minimizes the maximum error.

• A data point’s error is its $L_2$ norm distance to the line.
• Minimizing max error = parallel lines of support with Min separation.

• Max error \( D \) implies parallel lines of support with separation \( 2D \), and vice versa.

• Min separation between parallel support lines is also called width of \( S \).
Algorithm for Width

- Call \( a, b \) antipodal pair if they admit parallel lines of support.

- In \( \mathbb{R}^2 \), only \( O(N) \) antipodal pairs.

- If \( L_1, L_2 \) are parallel support lines, with minimum separation, then at least one of the lines contains an edge of \( CH(S) \).

- We can enumerate all antipodal pairs by a linear march around \( CH \).
Noncrossing Matching

• Given $N$ red and $N$ blue points in the plane (no three collinear), compute a red-blue non-crossing matching.

• Does such a matching always exist?

• Find if one exists.
Noncrossing Matching

- A non-crossing matching always exists.
- (Non-constructive:) Matching of minimum total length must be non-crossing.

- But how about an algorithm?
Algorithm

• Compute $CH(R)$ and $CH(B)$.

• Compute a common tangent, say, $rb$.

• Output $rb$ as a matching edge; remove $r$, $b$, update convex hulls and iterate.
When CH Nest?

- Algorithm fails if $CH(R)$ and $CH(B)$ nest.

- Split by a vertical line, creating two smaller, hull-intersecting problems.

- [Hershberger-Suri ’92] gives optimal $O(N \log N)$ solution.
• **Reduce sorting** to convex hull.
• **List of numbers to sort** \( \{x_1, x_2, \ldots, x_n\} \).
• **Create point** \( p_i = (x_i, x_i^2) \), for each \( i \).
• **Convex hull of** \( \{p_1, p_2, \ldots, p_n\} \) **has points in sorted** \( x \)-**order.** \( \Rightarrow \) **CH of** \( n \) **points must take** \( \Omega(n \log n) \) **in worst-case time.**

• **More refined lower bound is** \( \Omega(n \log h) \). **LB holds even for identifying the CH vertices.**
Output-Sensitive CH

1. Kirkpatrick-Seidel (1986) describe an $O(n \log h)$ worst-case algorithm. Always optimal—linear when $h = O(1)$ and $O(n \log n)$ when $h = \Omega(n)$.

2. T. Chan (1996) achieved the same result with a much simpler algorithm.

3. Remarkably, Chan’s algorithm combines two slower algorithms (Jarvis and Graham) to get the faster algorithm.

4. Key idea of Chan is as follows.
   
   (a) Partition the $n$ points into groups of size $m$; number of groups is $r = \lceil n/m \rceil$.
   (b) Compute hull of each group with Graham’s scan.
   (c) Next, run Jarvis on the groups.
Chan’s Algorithm

1. The algorithm requires knowledge of CH size $h$.

2. Use $m$ as proxy for $h$. For the moment, assume we know $m = h$.

3. Partition $P$ into $r$ groups of $m$ each.

4. Compute Hull($P_i$) using Graham scan, $i = 1, 2, \ldots, r$.

5. $p_0 = (-\infty, 0)$; $p_1$ bottom point of $P$.

6. For $k = 1$ to $m$ do
   - Find $q_i \in P_i$ that maximizes the angle $\angle p_{k-1}p_kq_i$.
   - Let $p_{k+1}$ be the point among $q_i$ that maximizes the angle $\angle p_{k-1}p_kq$.
   - If $p_{k+1} = p_1$ then return $\langle p_1, \ldots, p_k \rangle$.

7. Return “$m$ was too small, try again.”
Time Complexity

- **Graham Scan**: $O(rm \log m) = O(n \log m)$.

- Finding tangent from a point to a convex hull in $O(\log n)$ time.

- **Cost of Jarvis on r convex hulls**: Each step takes $O(r \log m)$ time; total $O(hr \log m) = ((hn/m) \log m)$ time.

- Thus, total complexity

  $$O \left( \left( n + \frac{hn}{m} \right) \log m \right)$$

- If $m = h$, this gives $O(n \log h)$ bound.

- **Problem**: We don’t know $h$. 
Finishing Chan

Hull($P$)

- for $t = 1, 2, \ldots$ do

  1. Let $m = \min(2^t, n)$.
  2. Run Chan with $m$, output to $L$.
  3. If $L \neq \text{“try again”}$ then return $L$.

1. Iteration $t$ takes time $O(n \log 2^t) = O(n2^t)$.

2. Max value of $t = \log \log h$, since we succeed as soon as $2^t > h$.

3. Running time (ignoring constant factors)

\[
\sum_{t=1}^{\lg \lg h} n2^t = n \sum_{t=1}^{\lg \lg h} 2^t \leq n2^{1+\lg \lg h} = 2n \lg h
\]

4. 2D convex hull computed in $O(n \log h)$ time.
Convex Hulls in $d$-Space

- New and unexpected phenomena occur in higher dimensions.

- Number of vertices, faces, and edges not the same.

- How to represent the convex hull? Vertices alone may not contain sufficient information.
Faces

- **In** $d$-dimensions, a face can have any dimension $k$, where $k = 0, 1, \ldots, d - 1$.

- **Special names**: Vertices (dim 0), Edges (dim 1), and Facets (dim $d - 1$).

- **In general**, a $k$-dim face.

- **In 4-dimension**, faces are 3d subspace, 2d faces, edges and vertices.
Facial Lattice

- Complete description of how faces of various dimension are incident to each other.

Face lattice of $f$
Complexity

Cubes of dim 1, 2, 3....

- How many vertices does \( d \)-dim cube have?
- How many facets does \( d \)-dim cube have?
- So, already as a function of \( d \), there is exponential difference between \( V \) and \( F \).
- But, for a fixed dimension \( d \), how large can the face lattice be as a function of \( n \), the number of vertices?
3 Dimensions

- **Steinitz**: The facial lattice of a 3-d convex polytope is isomorphic to a 3-connected planar graph and vice versa.

- **By Euler’s formula**, \( V - E + F = 2 \).

- **Verify this for cube**: \( V = 8, E = 12, F = 6 \).

- **In 3D**, \( E \) and \( F \) are linear in \( n \).
  \( E \leq 3n - 6 \), and \( F \leq 2n - 4 \).
Higher Dimensions

• Convex polytopes in higher dimensions can exhibit strange and unexpected behavior.

• In 4D, there are \( n \) points in general position so that the edge joining every pair of points is on the convex hull!

• That is, a 4D convex hull of \( n \) points can have \( \Theta(n^2) \) edges!

• In \( d \) dimensions, the number of facets can be \( n^{\lfloor d/2 \rfloor} \).

• Thus, explicit representation of convex hulls is not very practical in higher dimensions.

• But this does not mean they are useless: after all linear programming is optimization over convex polytopes.
Cyclic Polytopes

- Discovered in 1900’s, their importance comes from the Upper Bound Theorem by McMullen and Shephard 1971).

- Moment curve: $\gamma = \{(t, t^2, \ldots, t^d) \mid t \in \mathbb{R}\}$.

- A point $p = (u, u^2, \ldots, u^d)$ is given by the single parameter $u$.

- Consider $n$ values $u_1 < u_2 < \cdots < u_n$. Let $p_1, p_2, \ldots, p_n$ be the corresponding points on the moment curve.

- Then, any $k$-tuple of points, where $k \leq d/2$, is a face of their convex hull.
4D Example

- **Moment curve** is $\gamma = \{(t, t^2, t^3, t^4)\}$.

- **Fix any two** $i, j$. **Consider the polynomial**

$$P(t) = (t - u_i)^2(t - u_j)^2$$

- **This polynomial can be written as**:

$$P(t) = t^4 + a_3t^3 + a_2t^2 + a_1t + a_0$$

- **Clearly,** $P(t) \geq 0$, for all $t$. Furthermore, the only zeros of the polynomial occur at $t = u_i$ and $t = u_j$. 
4D Example

- But \( x_4 + a_3x_3 + a_2x_2 + a_1x_1 + a_0 = 0 \) is the equation of a hyperplane. This evaluates to zero when \( x = p_i \) or \( p_j \).

- Since for all other points, the polynomial evaluates to \( \geq 0 \), it means that the moment curves lies on the same side of this plane.

- Thus, this plane is the witness that \( p_ip_j \) is on the convex hull.

- We chose \( i, j \) arbitrarily, so all pairs are on the convex hull.
Upper Bound Theorem

- Among all $d$-dim convex polytopes with $n$ vertices, the cyclic polytope has the maximum number of faces of each dimension.

- A $d$-dim convex polytope with $n$ vertices has at most $2\binom{n}{d/2}$ facets and at most $2^{d+1}\binom{n}{d/2}$ faces in total.

- Thus, asymptotically, a $d$-dim convex polytope has $\Theta(n^{\lfloor d/2 \rfloor})$ faces.

- A worst-case optimal algorithomn of this complexity is by Chazelle [1993].