Simplified Bellman-Ford

1 Initialize: \( d := \infty; \ d[s] \leftarrow 0; \ \pi := \text{nil}; \)
2 for \( m \leftarrow 1 \) to \(|V| - 1\)
3 for all \((u,v)\) in \( E \) do
4 \hspace{1em} if \( d[v] > d[u] + w(u,v) \) then
5 \hspace{2em} \( d[v] \leftarrow d[u] + w(u,v); \)
6 \hspace{2em} \( \pi[v] \leftarrow u \)
Negative Cycles

Claim: A negative cycle exists iff the $|V|=n^{th}$ loop of

3. for all $(u,v) \in E$ do
4. if $d[v] > d[u] + w(u,v)$ then
5. $d[v] \leftarrow d[u] + w(u,v)$;
6. $\pi[v] \leftarrow u$

produces a change.
No Change on Cycle

Use $d$ (resp. $d'$) to denote the array before (resp. after) the $n$th loop. Then

$$d'[p_{i+1}] \leq d[p_i] + w(p_i, p_{i+1}) \quad \text{for } i=1, \ldots, k$$

$$\sum d'[p_{i+1}] \leq \sum d[p_i] + \sum w(p_i, p_{i+1})$$
Negative loops

\[ \sum w(p_i, p_{i+1}) < 0 \]

For all \( i = 1, \ldots, k \)

\[ \sum d'[p_{i+1}] \leq d[p_i] + w(p_i, p_{i+1}) \]

\[ \sum d'[p_{i+1}] \leq \sum d[p_i] + \sum w(p_i, p_{i+1}) < \sum d[p_{i+1}] \]
Computational Geometry and Convex Hulls
The Convex Hull Problem

Given $n$ points $p_0, p_1, \ldots, p_{n-1}$ in the plane, find the smallest convex polygon that contains all points $p_0, p_1, \ldots, p_{n-1}$. 

![Diagram of convex hull with points $p_0, p_1, \ldots, p_{n-1}$]
The Convex Hull Problem

Given $n$ points $p_0, p_1, \ldots, p_{n-1}$ in the plane, find the smallest convex polygon that contains all points $p_0, p_1, \ldots, p_{n-1}$. 
Applications of Convex Hull

• Packing: Smallest box or wrapping
• Robotics: Avoiding obstacles
• Graphics and Vision: Image and shape analysis
• Computational geometry: Many applications, e.g.
  – Finding farthest pair of points in a set
Definitions

• A set $Q$ is **convex** if $x$ in $Q$ and $y$ in $Q$ implies that $xy$ in $Q$.

• The **segment** $xy$ is the set of all points $ax + by$ with $a \geq 0, b \geq 0,$ and $a+b=1$.

• A **convex combination** of points $x_1, \ldots, x_k$ is $a_1x_1 + \ldots + a_kx_k$ with $a_i \geq 0$ and $a_1 + \ldots + a_k = 1$. 
Definitions of convex hull

• The set of all convex combinations (of d+1 points)
• Intersection of all convex sets that contain Q
• Intersection of all half-spaces that contain Q

In the plane:
• Smallest convex polygon P that encloses Q
• Enclosing convex polygon P with smallest area
• Enclosing convex polygon P with smallest perimeter
• Union of all triangles determined by points in Q
Convex Hull by Jarvis March - Example
Jarvis March - Example
Jarvis March - Example
Jarvis March - Example
Jarvis March - Example
Jarvis March - Example
Jarvis March - Example
The Jarvis March Algorithm

$JavisMarch(Q)$
1. Find the index $i_0$ of the lowest point in $Q$
2. $i \leftarrow i_0$
3. repeat
   for each $j \neq i$ do
     compute angle $q_j$ from previous hull edge;
     $k \leftarrow$ index of point with the smallest $q_k$;
     Output $(p_i, p_k)$ as a hull edge;
     $i \leftarrow k$
   until $i = i_0$

Points are output in their order of appearance on the convex hull polygon.
Jarvis March - Complexity

\[ T(n) = \Theta(n \cdot h), \]

where \( h \) = number of vertices on the convex hull
Computing Relative Orientation

1. Given directed line segments $\overrightarrow{p_0p_1}$ and $\overrightarrow{p_0p_2}$, determine whether $\overrightarrow{p_0p_1}$ is clockwise from $\overrightarrow{p_0p_2}$ with respect to point $p_0$?

2. Given two line segments $\overrightarrow{p_0p_1}$ and $\overrightarrow{p_1p_2}$, if we traverse $\overrightarrow{p_0p_1}$ and then $\overrightarrow{p_1p_2}$, do we make a left turn at point $p_1$?
Cross Products

\[ p_1 \times p_2 = x_1y_2 - y_1x_2 \]
\[ = -p_2 \times p_1 \]

\[ p_1 = (x_1, y_1) \]
\[ p_2 = (x_2, y_2) \]
Relative Orientation

\[(0,0)\]

\[p_1\]
Relative Orientation

1. Given directed line segments $\overrightarrow{p_0p_1}$ and $\overrightarrow{p_0p_2}$, determine whether $\overrightarrow{p_0p_1}$ is clockwise from $\overrightarrow{p_0p_2}$ with respect to point $p_0$?

Compute $\Pi = p_1 \times p_2$

if $\Pi > 0$, then $\overrightarrow{p_0p_1}$ is clockwise from $\overrightarrow{p_0p_2}$

if $\Pi < 0$, then $\overrightarrow{p_0p_1}$ is counterclockwise from $\overrightarrow{p_0p_2}$
Relative Orientation

2. Given two line segments \( p_0p_1 \) and \( p_1p_2 \), if we traverse \( p_0p_1 \) and then \( p_1p_2 \), do we make a left turn at point \( p_1 \)?

Compute \( \Pi = (p_1 - p_0) \times (p_2 - p_0) \)

if \( \Pi > 0 \), then we make a right turn at \( p_1 \)

if \( \Pi < 0 \), then we make a left turn at \( p_1 \)
The Graham Scan Algorithm

GrahamScan(Q)
1. $p_0 \leftarrow$ the point with the minimum $y$-coordinate
2. sort the remaining points $<p_1, \ldots, p_m>$ in $Q$, by the angle in counterclockwise order around $p_0$
3. Top(S) $\leftarrow 0$
4. Push $(p_0, S)$; Push $(p_1, S)$; Push $(p_2, S)$
5. for $i \leftarrow 3$ to $m$ do
6. while angle formed by the points NextToTop(S), Top(S), $p_i$ makes a non-left turn do
7. Pop(S)
8. Push $(p_i, S)$
9. Endfor
10. return S

Show the example!
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
Graham Scan - Example
The Graham Scan Algorithm

GrahamScan(Q)

1. \( p_0 \leftarrow \) the point with the minimum y-coordinate
2. sort the remaining points \(<p_1, \ldots, p_m>\) in \( Q \), by the angle in counterclockwise order around \( p_0 \)
3. \( \text{Top}(S) \leftarrow 0 \)
4. \( \text{Push}(p_0, S); \text{ Push}(p_1, S); \text{ Push}(p_2, S) \)
5. for \( i \leftarrow 3 \) to \( m \) do
6. while angle formed by the points \( \text{NextToTop}(S), \text{Top}(S), p_i \) makes a right turn do
7. \( \text{Pop}(S) \)
8. \( \text{Push}(p_i, S) \)
9. Endfor
10. \( \text{return } S \)
Graham Scan - Complexity

\[ T(n) = \Theta(n \log n) + c \cdot n = \Theta(n \log n) \]

*Can we do better than \( O(n \log n) \)?*

*If we could, we could sort faster than \( O(n \log n) \). So*

**Lower bound:** Computing the Convex Hull requires \( \Omega(n \log n) \) in the worst case.
Improving Hull algorithms

- Compute extreme points in 4 directions
- Eliminate internal points (How?) in $\Theta(n)$
- Run a standard Convex Hull algorithm

- On a random distribution, only $\Theta(n^{0.5})$ points are left