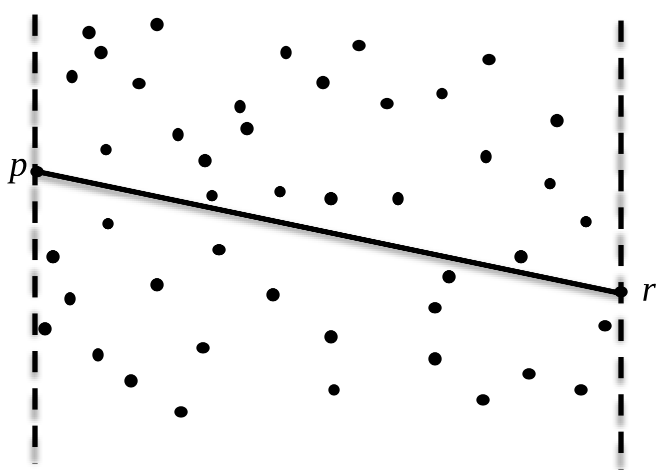
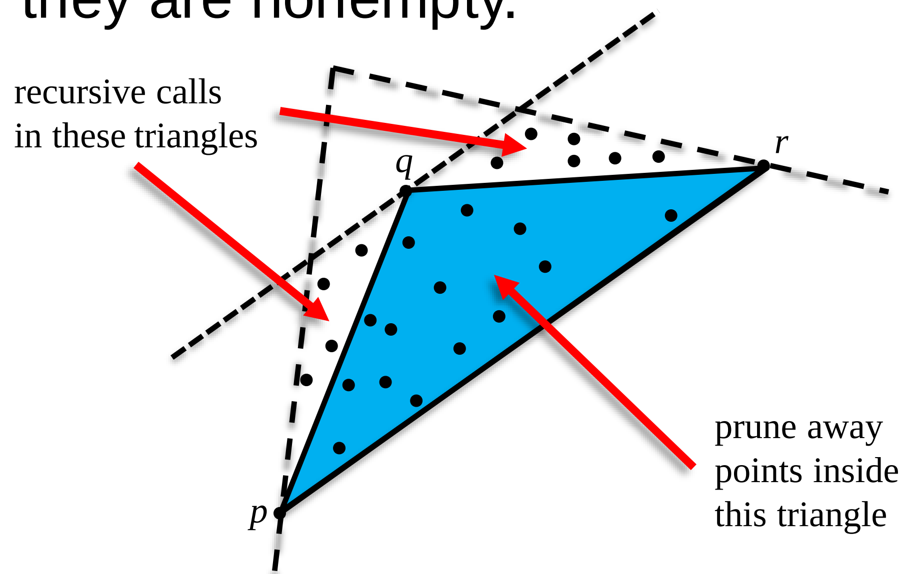
QuickHull Algorithms

Michael T. Goodrich

QuickHull

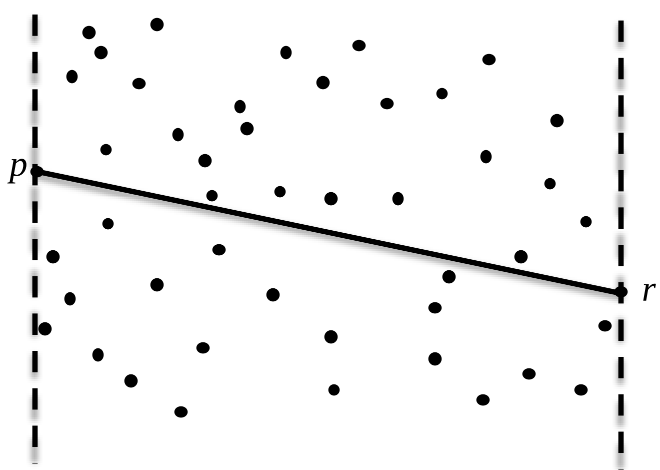
- Find a point, p ∈ S, with minimum x-coordinate, and a point, r ∈ S, with maximum x-coordinate. Clearly, p and r are on the convex hull of S.
- For each recursive call, we have a set of points, S′ ⊆ S, inside a triangle with base pr, for which Quickhull determines the point, q in S′, that is farthest from the segment pr. Then we prune away points inside the triangle (p, q, r)
- Partition the remaining points of S′ into those above pq and qr and in bounding triangles defined by the tangents,respectively, and we recursively solve the problem for each of these subsets if they are nonempty.

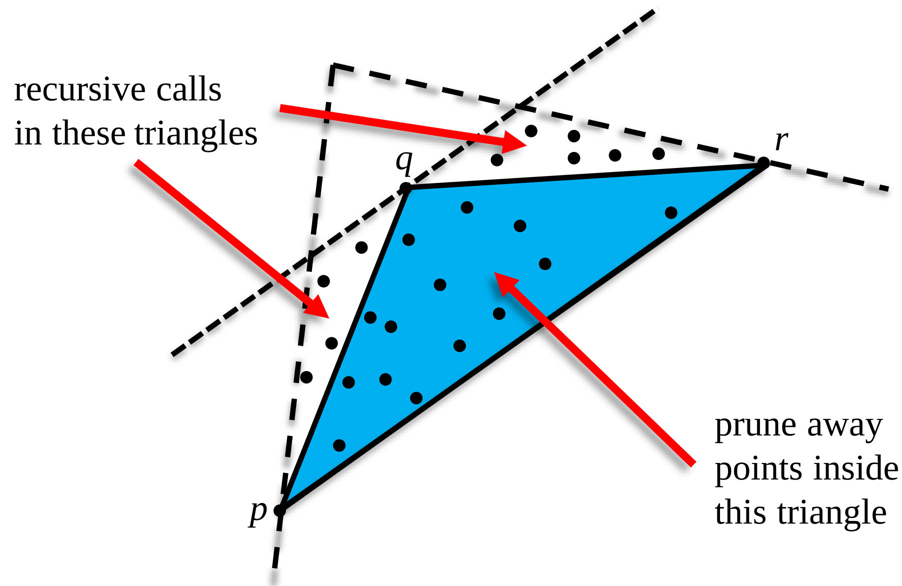




QuickHull Analysis

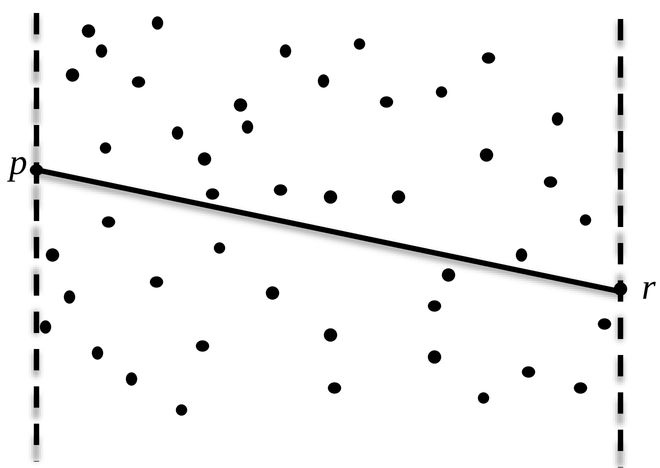
- Hopefully, we split evenly in each recursive call, which would lead to a running time of O(n log n), but this is not guaranteed.
- Worst-case running time: O(nh).
- Example: (x,x^2) for $x=2^i$, i=1,2,3,…

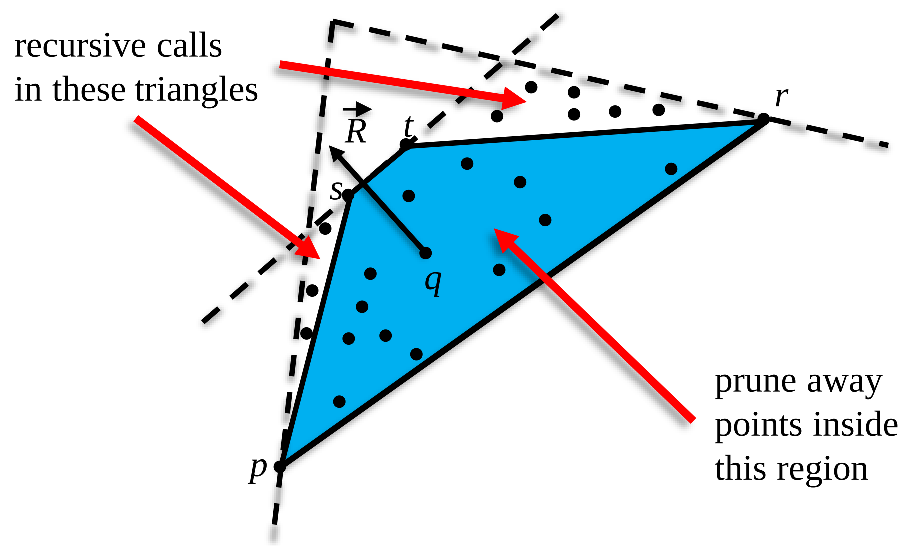




Randomized QuickHull

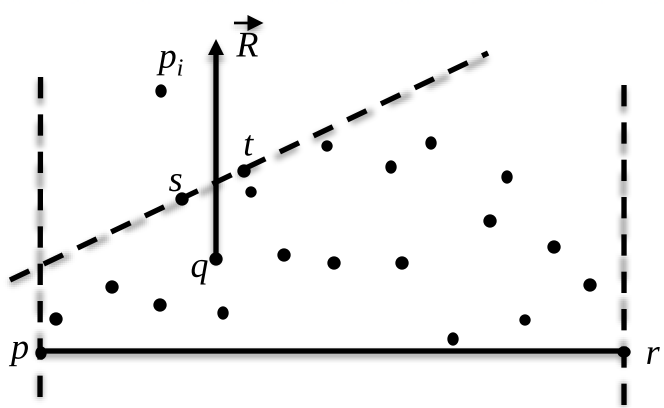
- Find a point, p ∈ S, with minimum x-coordinate, and a point, r ∈ S, with maximum x-coordinate. Clearly, p and r are on the convex hull of S.
- For each recursive call, choose a point, q, uniformly at random.
- Shoot a ray perpendicular to pr and find the bridge edge, (s,t), intersected by this ray.
- Partition the remaining points of S′ into those above ps and qt and in bounding triangles and we recursively solve the problem for each of these subsets if they are nonempty.

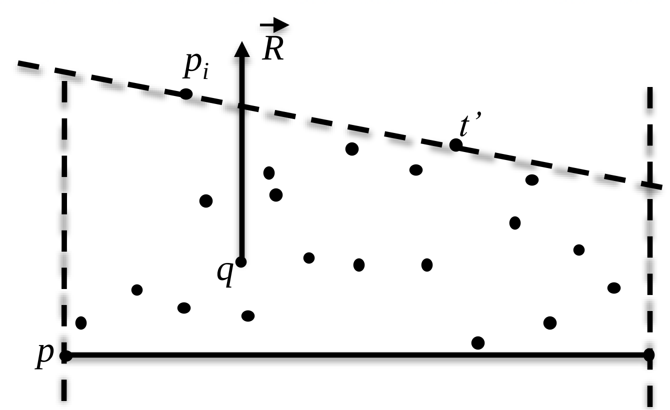




Ray Shooting – Randomized

Let $(s, t) \leftarrow (q, q)$ be our initial candidate convex hull (degenerate) bridge edge.
Let (s, t) initially define a horizontal line, $\overline{st}$, through q.
Let R_l denote the left halfplane defined by the vertical line through q.
Let R_r denote the right halfplane defined by the vertical line through q.
Let $S_l = S_r = \{q\}$ be the set of points processed so far that are respectively in R_l and R_r.
Randomly permute the points in $S = \{p_1, p_2, \ldots, p_n\}$.
for $i \leftarrow 1$ **to** n **do**
    if p_i is above the line st **then**
        if p_i is in R_l **then**
            Find the point, t', in S_r such that $p_i t'$ minimizes the angle with the x-axis.
            Let $(s, t) = (p_i, t')$.
        else
            Find the point, s', in S_l such that $s' p_i$ minimizes the angle with the x-axis.
            Let $(s, t) = (s', p_i)$.
    if $p_i \in R_l$ **then** add p_i to S_l.
    if $p_i \in R_r$ **then** add p_i to S_r.





Ray Shooting – Analysis

Theorem 2. *Algorithm RayShoot performs at most $2n$ orientation tests in expectation.*

Proof: The running time analysis follows by a simple backwards analysis. Let X_i be a 0-1 random variable that is 1 if and only if the condition in line 8 in the ray-shooting algorithm is true. Since the searching operations in lines 10 and 13 use an orientation test for each member of S_r (resp., S_l), the total number of orientation tests performed by RayShoot is at most

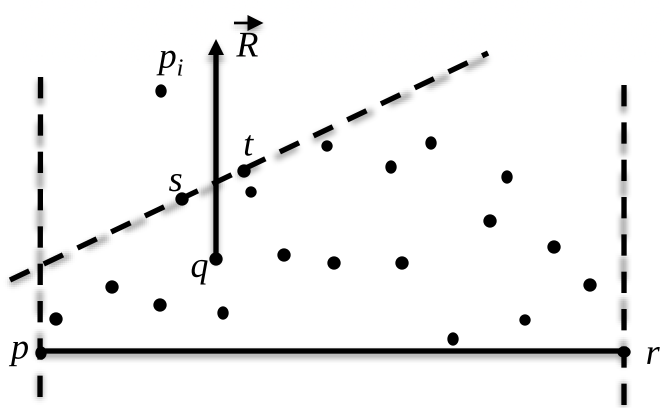
$$\sum_{i=1}^{n} iX_i.$$

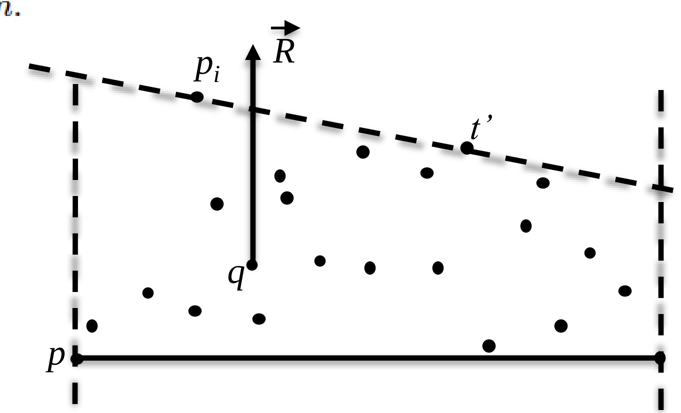
By the linearity of expectation,

$$E\left[\sum_{i=1}^{n} iX_i\right] = \sum_{i=1}^{n} iP_i,$$

where P_i is the probability that the point p_i is above the line st. Now consider the iterations of RayShoot backwards, and note that p_i will satisfy the condition in line 8 if it is one of the two points that defines the edge of the convex hull of $q \cup \{p_1, p_2, \ldots, p_i\}$ intersecting $\vec{R}$. Thus, $P_i \leq 2/i$, which implies that the expected number of orientation tests performed by RayShoot is

$$\sum_{i=1}^{n} i \cdot \frac{2}{i} = 2n.$$





Theorem 3. *Given a set, S, of n points in the plane, the randomized Ray-shooting Quickhull algorithm constructs the convex hull of S in $O(n \log h)$ expected time, where h is the number of points of S on the convex hull.*

Proof: The proof is an adaptation of an analysis of the expected running time of the Quicksort algorithm [1,19,21]. Let $T(n,h)$ denote the expected running time of the randomized Ray-shooting Quickhull algorithm on an instance of size $n \geq 2$ with hull size $h \geq 2$. Also, to simplify the notation, let $T(0,h) = 0$ and $T(1,h) = 0$. Then, by the way a problem instance in the randomized Ray-shooting Quickhull algorithm is divided, there is a constant $c \geq 1$, such that the general case is as follows:

$$T(n,h) \leq cn + \frac{1}{n} \sum_{i=0}^{n-1} \max_{h_1+h_2=h} \{T(i,h_1) + T(n-i-1,h_2)\},$$

where, by Theorem 2, $c = 2$ if we are focused on counting orientation tests. We claim that there is a constant, $d \geq 1$, such that $T(n,h) \leq dn$, for $n \geq 2$ and $h = 1,2$, and $T(n,h) \leq dn \log h$ otherwise;[1] hence, by this induction hypothesis,

$$T(n,h) \leq cn + \frac{1}{n} \left(2d(n-1)\log(h-1) + \sum_{i=1}^{n-2} \max_{h_1+h_2=h} \{di \log h_1 + d(n-i-1)\log h_2\} \right).$$

By elementary calculus, the righthand side is maximized with $h_1 = ih/n$ and $h_2 = (n-i-1)h/n$. Thus,

$$\begin{aligned}
T(n,h) &\leq cn + \frac{d}{n} \left(2(n-1)\log(h-1) + \sum_{i=2}^{n-2} (i \log(ih/n) + (n-i-1)\log((n-i-1)h/n)) \right) \\
&\leq cn + \frac{2d}{n} \sum_{i=1}^{n-1} i \log(ih/n) \\
&= cn + \frac{2d}{n} \sum_{i=1}^{n-1} i \log i + \frac{2d}{n} (\log h) \sum_{i=1}^{n-1} i - \frac{2d}{n} (\log n) \sum_{i=1}^{n-1} i.
\end{aligned}$$

By another application of calculus,

$$\sum_{i=1}^{n-1} i \log i \leq \int_1^n (x \log x)dx \leq (n^2/2)\log n - n^2/4 + 1/4.$$

Also, it is well-known that $\sum_{i=1}^{n-1} i = n(n-1)/2$. Therefore,

$$\begin{aligned}
T(n,h) &\leq cn + dn\log n - dn/2 + d/(2n) + d(n-1)\log h - d(n-1)\log n \\
&\leq dn \log h,
\end{aligned}$$

for $d = 2c + 1$. ■