Note: These lecture notes are based on and contain figures from the textbook "Computational Geometry" by Berg et al.*and lecture notes from [3], [1], [2]*

1 Halfplane Intersection Problem

We can represent lines in a plane by the equation

$$y = ax + b$$

where a is the slop and b the y-intercept. Note that this can not represent vertical lines, which have infinite slope (but these do not occur in general position)

Every non-vertical line defines two *halfplanes*, consisting of the points above and below the line.

- *lower halfplane:* $y \leq ax + b$
- *upper halfplane:* $y \geq ax + b$

1.1 Halfplane Intersection

Halfplane Intersection Problem: Given a collection $H = \{h_1, \ldots h_n\}$ of n closed halfplanes, compute their intersection.

But, what should the output look like? A halfplane is a convex set so the intersection of any number of them is also convex. Unlike the convex hull, the intersection of halfplanes may be empty or unbounded. A good way to represent the output would be to list the lines bounding the intersection in clockwise order. How big can this output be? By convexity, each halfplane can appear only once as a side. Hence the number of sides is $O(n)$.

Motivation We can use halfplane intersection to generate convex shape approximations. Also many (most?) optimizations problems (e.g. linear programming, gradient descent) often occur over intersection of halfplanes.

We'll discuss two algorithms for this problem: (1) divide-and-conquer; and (2) duality based. Is there a lowerbound? We can show via sorting it is $\Omega(n \log n)$. [**Jared:** *good exercise*]

2 Divide-and-Conquer

```
procedure DIVIDE-AND-CONQUER(H)
    If $n = |H| = 1$, return the halfplane
    Partition the halfplanes into $H_1$ and $H_2$ of size $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$
    Let $K_1$ and $K_2$ be the intersections of $H_1$ and $H_2$, computed recursively
    Return the intersection of $K_1$ and $K_2$
end procedure
```

If we can perform the intersection (last step) in $O(n)$ time then the algorithm will take $O(n \log n)$ time, which can be shown via solving a simple recurrence ($T(n) = 2T(n/2) + n$).

Thus, we must intersect two convex (possibly empty or unbounded) polygons in linear time

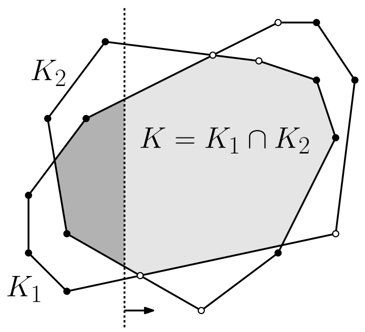

Figure 1. Plane Sweep Intersection of Polygons (from Berg et al.)

2.1 Plane Sweep

We compute $K_1 \cap K_2$ via a plane sweep.

First, break both polygons into upper and lower chains. The *upper chain* of a polygon is just the sequence of points that are above the line induced by the leftmost and rightmost point in the polygon. The *lower chain* is the sequence of points below this line. So this step can be done in $O(n)$ time.

Next, sweep! We sweep from left to right, computing the vertices and edges in $K_1 \cap K_2$ as we go. By convexity, the sweep line intersects each polygon in at most two points. Hence, there are at most 4 points to consider at any point during the sweep. For each of these 4 points on the sweep line, we keep track of (1) which upper/lower input hull that point is on; and (2) what is the slope of the line segment for that input hull containing the point. Our output will be the vertices of $K_1 \cap K_2$.

What are the total number of "events" that can happen as we sweep from left to right? The next event can be either the vertex of an input hull (4 possibilities), or the intersection of two edges of the input hull (4 possibilities). Note that we can compute upcoming intersections since we store the slopes for all 4 points on the sweep line.

Thus, we encounter $O(n)$ events; each event can be processed in $O(1)$ time (e.g. find the upper and/or lower vertex to output in any); and the next event can be computed in $O(1)$ time. Hence, total time is $O(n)$.

Higher Dimensions. This approach can be generalized to higher dimensions, where the sweep line is still across (say) the x axis, and the events that happen are intersections of *hyperplanes.*

But this ain't cheap in high dimensions. In particular, just *outputting* the facets in the envelope can be computational expensive for high d, since the number of facets in the intersection of n half-spaces in d-dimensions can be $\Theta(n^{\lfloor d/2 \rfloor})$. This motivates some dimensionality-reduction techniques we'll discus later in the class.

3 Duality

3.1 Duality Definition

- Given any point $p = (a, b)$, let p^* be the line $ax - b$.
- Given any line $\ell = ax + b$, let ℓ^* be the point $(a, -b)$

Primal and Dual planes

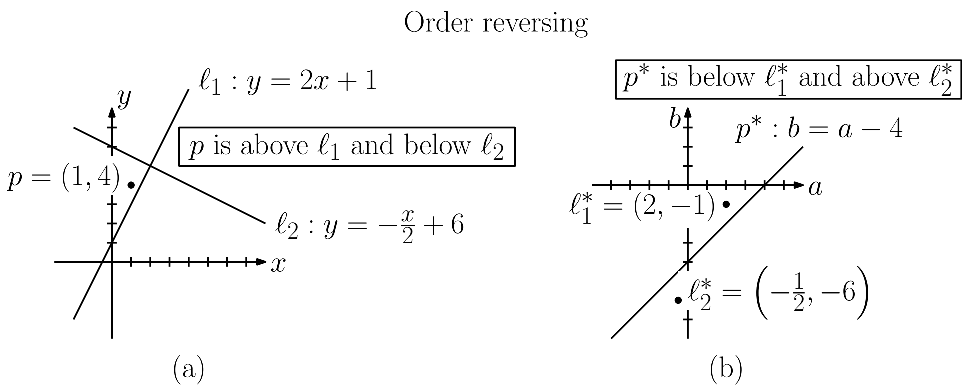
Order reversing
$\ell_1 : y = 2x + 1$
y
p is above ℓ_1 and below ℓ_2
$p = (1, 4)$
$\ell_2 : y = -\frac{x}{2} + 6$
x
(a)
p^* is below ℓ_1^* and above ℓ_2^*
b
$p^* : b = a - 4$
a
$\ell_1^* = (2, -1)$
$\ell_2^* = \left(-\frac{1}{2}, -6\right)$
(b)

Figure 2. Order-Reversing Property of Duality (from Berg et al.)

- **Primal plane:** Original (x,y) plane
- **Dual plane:** The new (a,b) plane

3.2 Duality Properties

- **Self-Inverse:** $p^{**} = p$
- **Order Reversing:** p is above/on/below ℓ in the primal plane iff p^* is below/on/above ℓ^* in the dual plane.
- **Preserves Vertical Distances:** The vertical distance between p and ℓ equals the vertical distance between p^* and ℓ^*
- **Intersection preservation:** Lines ℓ_1 and ℓ_2 intersect at point p in the primal plane iff the dual line p^* passes through points ℓ_1^* and ℓ_2^*
- **Colinearity/Coincidence:** Three points are collinear in the primal plane iff their dual lines intersect at the same point.

Order Reversal Property To show point p is above line ℓ iff line p^* is below point ℓ^*

- Point $p = (p_x, p_y)$ is above the line $\ell = mx + b$
- $\iff p_y \geq mp_x + b$
- $\iff p_y - mp_x \geq b$
- $\iff$ line $p^* = p_x x - p_y$ is below point $\ell^* = (m, -b)$

3.3 A geometric View

Let $\mathcal{U}$ be the parabola $y = x^2/2$. Consider some point (p_x, p_y) on $\mathcal{U}$. Note that $p^* = p_x x - p_y$. Then:

- p^* has same slope as the tangent line, since the derivative of $\mathcal{U}$ at p is p_x.

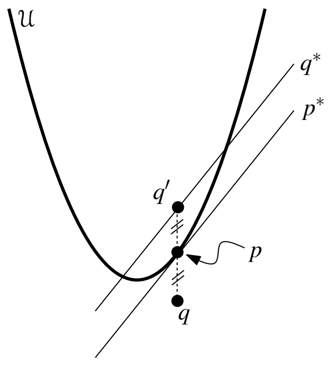

Figure 3. Duality: a geometric view ([2])

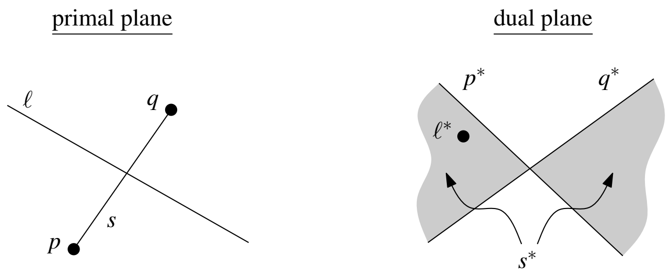

Figure 4. Dual of a line segment is a wedge ([2])

- p^* has same y-intercept as tangent line, since the tangent line intersects the y-axis at $(0, p_x^2/2)$.
  - the tangent line equation is: $p_y = p_x(p_x) - b$, which implies that $b = p_x^2 - p_y = p_x^2 - p_x^2/2 = p_x^2/2$.

Thus for p on $\mathcal{U}$, p^* is exactly the tangent line at p. What about for any point $q = (q_x, q_y)$ not necessarily on $\mathcal{U}$?

- q^* has the same slope as the tangent line at q_x
- q^* intersects the point $(q_x, q_x^2/2 + (q_x^2/2 - q_y))$

3.4 Some Observations

- Duality can be applied to objects other than lines and points. For example, we can take the dual of a line segment.
- The dual of a segment is a double wedge. See Figure 4.
- Q: What line would dualize to a point in the right side of the figure?

4 Duality: Convex Hull = Half-Plane Intersection!

Lower and Upper Envelope We are given n lines $\{\ell_1, \dots \ell_n\}$. The *lower envelope* is the intersection of their lower halfplanes. The *upper envelope* is the intersection of their upper halfplanes.

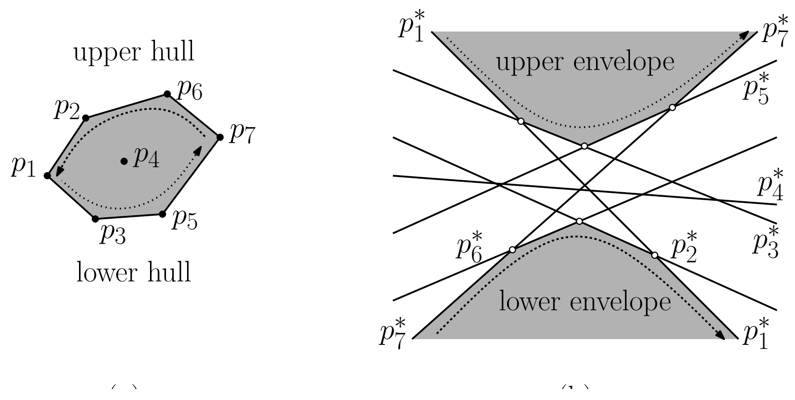

Figure 5. Envelope/Convex Hull Duality (from Berg et al.)

Note that we can perform half-plane intersection via computing the upper and lower envelopes as follows. Given a set of half-planes to intersect, first find the upper envelope of the half-planes that are open to the top, then find the lower envelope of the half-planes that are open to the bottom. Then merge these two envelopes (in $O(1)$ time), by computing the leftmost and rightmost intersection points.

Lemma: Let P be a set of points in the plane. Then the ccw order of the upper convex hull of P equals the left to right order of the lines on the lower envelope of the dual P^*. Symmetrically the order in the lower convex hull of P is the left-to-right order of lines in the upper envelope.

Proof: Assume no three points are co-linear. Consider two consecutive points q and r on the upper hull and let ℓ be the line through these two points. By definition of the hull, ℓ is above all points in P. Thus, by the order reversing property, all dual lines of P^* pass *above* point ℓ^*. This means that point ℓ^* lies on the lower envelope. Next, consider the dual lines q^* and r^*. By the incidence property, the dual point ℓ^* is the intersection of these two lines. (By general position, we assume the two points have different x-coordinates and so q^* and r^* have different slopes). Thus, the edges in the lower-envelope associated with the point ℓ^* are determined by the lines q^* and r^*.

It remains only to show that the points in the lower envelope are ordered left-to-right. Note that as we move ccw along the upper hull (i.e. right-to-left), the slopes of the edges increase. Since the slope of the line in the primal plane determines the x-coordinate of the dual point, this means that we visit the lower envelope from left-to-right.

The proof for the upper envelope follows by symmetry. □

5 Arrangements

We'll soon see some more applications of duality. But, before discussion the new problems, a brief interlude to discuss *arrangements*, which are a kind of data structure to understand lines in the plane.

Once we introduce this data structure, we'll see how to use it and duality to efficiently address the following problem. Given a set of n points, are there 3 points that are colinear? The naive algorithm takes $O(n^3)$ time. But a clever algorithm using duality and *arrangements* takes just $O(n^2)$. The idea is that detecting 3 points on a line dualizes to detecting 3 lines intersecting at a point!

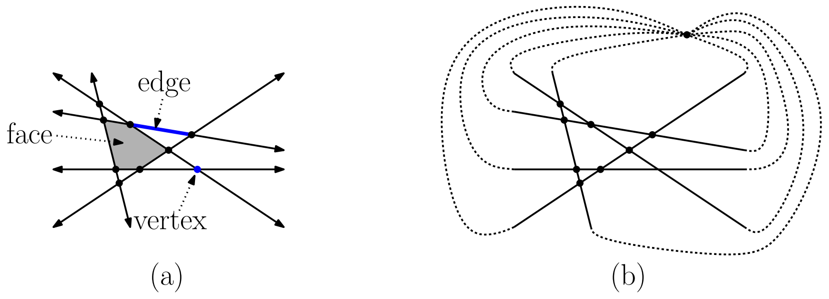

Figure 6. Basic elements of an arrangement. Adding a vertex at infinity creates a proper planar graph ([1])

5.1 Arrangement Definition and Complexity

An *arrangement* of lines consists of faces, edges and vertices that are created via a set of lines in the plane (See Figure 6). The following lemma gives the combinatorial complexity of an arrangement.

Lemma: An arrangement in the plane for n lines in general position has:

- $\binom{n}{2}+1$ vertices;
- n^2 edges;
- $\binom{n}{2}+n+1$ faces.

Proof: First note that the number of vertices equals the number of intersection points of lines plus 1 (the vertex at infinity). Assuming general position, this is $\binom{n}{2+1}$ vertices.

Next we prove the bound on the edges by induction. Base case is trivial. Inductive step: When we add a new line to an arrangement of $n-1$ lines, we start with $(n-1)^2$ edges by the IH. Then the new line splits exactly one edge from each of the existing lines. Also, the new line is split into n edges by its intersection with the other lines. Thus, the total number of edges in the arrangement with the new line is $(n-1)^2+(n-1)+n=n^2$

The number of faces follows by Euler's formula $v-e+f=2$. (One can prove Euler's theorem by induction on the number of lines in an arrangement. Note that when a new line intersects a face, it creates: 1 new face and 1 new edge which is the edge supported by the line inside the face. Additionally, whenever the new line intersects an old edge, it creates both a new vertex and a new edge. Thus, when adding a new line, the increase in $v+f$ equals the increase in e, which establishes the inductive step.) $\square$

Note that we can represent the arrangement fully as:

- The vertex and edge graph as shown in Figure 6 (right);
- A pointer for each edge to the two faces that edge is in; and
- For each face, a linked list that maintains the edges of the face in CCW order.

Thus for $d=2$, the complexity of an arrangement is $O(n^2)$. More generally, for all $d\geq 2$, the complexity of an arrangement of $d-1$ dimensional hyperplanes is $O(n^d)$.

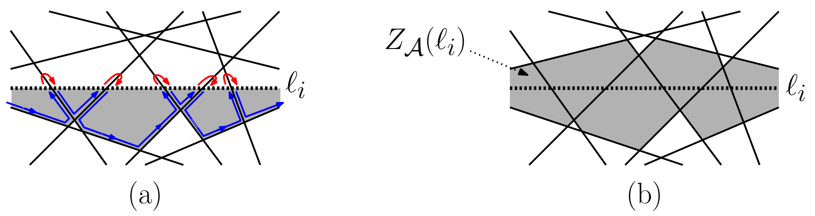

Figure 7. (a) adding ℓ_i to an arrangement; (b) the zone of line ℓ_i ([1])

5.2 Building an Arrangement

We build an arrangement by adding one line after another. Let $L = \{\ell_1, \dots \ell_n\}$ denote the set of lines. We show that line ℓ_i can be added in $O(i)$ time. This gives a $O(n^2)$ time algorithm for building the entire arrangement. Suppose the first $i-1$ lines have been inserted and consider the insertion of ℓ_i.

First, determine the leftmost unbounded face containing ℓ_i. Assume that at $x = \infty$, the lines are sorted by increasing slope. Then we can find in $O(\log i)$ time where ℓ_i falls in terms of slope. This determines the leftmost face containing ℓ_i.

Next, the line ℓ_i cuts through a bunch of edges and faces that we need to split up. To determine which edges are cut, we "walk" the line through the arrangement from one face to the next. When we enter a face, we need to determine through which edge ℓ_i exits that face. We do this in a brute-force manner by just walking the edges in ccw order till we find the edge at which ℓ_i exits the face. We then jump to the face on the other side of that edge. See Figure 7(a).

Problem: Naively, we can say that we encounter $i-1$ lines, and hence pass through i faces. Since each face is bounded by at most i lines, we take $O(i^2)$ time. But this is slower than what we hope for. To improve the analysis, we need the zone theorem.

5.3 The Zone Theorem

Given an arrangement $\mathcal{A}(L)$ and a line ℓ not in L, the *zone* of ℓ in $\mathcal{A}(L)$, denoted $Z_{\mathcal{A}(\ell)}$, is the set of faces of the arrangement that are intersected by ℓ. The Zone theorem below states that the complexity of the zone (i.e. number of edges of all faces in the zone) is $O(n)$. The proof below is based on that of [1].

Theorem: Given an arrangement $\mathcal{A}(\mathcal{L})$ of n lines in the plane, and given any line ℓ in the plane, the total number of edges in all the faces that ℓ intersects is at most $6n$.

Proof: For simplicity, rotate the plane so that ℓ is horizontal. We call an edge a *left bounding edge* for the face lying to the left of it (Figure 8(a)), and a *right bounding edge* for the face lying to the right. We will show that there are at most $3n$ left bounding edges in the zone of ℓ in the arrangement $\mathcal{A}(\mathcal{L})$. A symmetrical argument works for the right-bounding edges in the zone.

Base Case: $n = 1$, there is exactly one left bounding edge in ℓ's zone, and $1 \le 3n = 3$.

Inductive Step: Consider an arrangement of n lines and let ℓ_1 be the line that intersect ℓ at the rightmost point (Figure 8(b)). By the inductive hypothesis, there are at most $3(n-1)$ left-bounding edges in the zone for ℓ in the arrangement $\mathcal{A}(L - \ell_1)$.

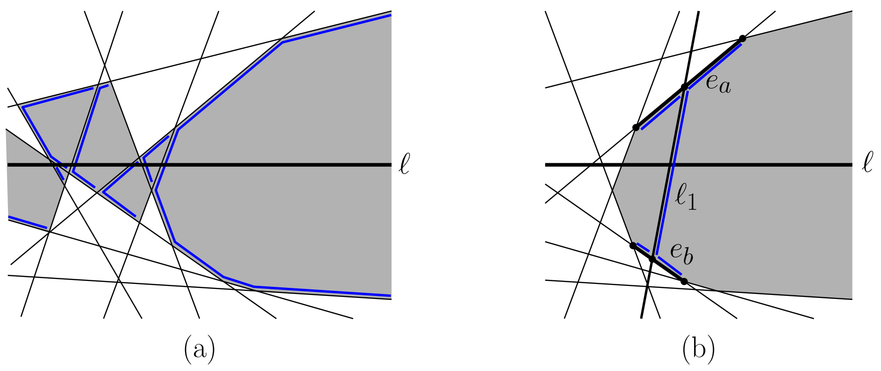

Figure 8. (a) all left bounding edges of ℓ; (b) rightmost face intersected by ℓ and the left bounding edges created by ℓ_1.([1])

Consider the rightmost face intersected by ℓ and note that all edges of this face are left-bounding edges. Line ℓ_1 intersects ℓ within this face and, by convexity, it intersects the boundary of this face in two edges, call them e_a and e_b. The insertion of ℓ_1 into this face creates a new left bounding edge along ℓ, and also splits e_a and e_b into two new left bounding edges for a net increase of 3 edges (Figure 8(b)). Note that ℓ_1 can not contribute any other left-bounding edges to the zone: to the left of ℓ_1, the edges induced by ℓ_1 can form right-bounding edges only; to the right of ℓ_1, all other faces possibly touched by ℓ_1 are shielded from ℓ by either the line supporting e_a or the line supporting e_b.

Thus, the total number of left bounding edges on the zone of ℓ in $\mathcal{A}(L)$ is at most $3(n-1)+3 \leq 3n$. □

6 Some Applications

6.1 Three Points on a line?

Given a set P of n points, in $O(n^2)$, we can determine if there are 3 points on a line (i.e. are the points in general position?).

Algorithm:

1. Iteratively create the arrangement for the lines in P^*
2. If 3 lines in P^* intersect at the same vertex, then the points associated with those 3 lines in the primal plane are co-linear.

Takes just $O(n^2)$ time to compute the arrangement!

6.2 Levels of an Arrangement

We say that a point is at *level* k (denoted $\mathcal{L}_{||}$) in an arrangement if there are at most $k-1$ lines above the point and at most $n-k$ lines below. See Figure 9(a). The upper envelope of the lines is level 1 of the arrangement, and the bottom envelope is level n. Note that each point in the arrangement is generally on two levels, and that the levels are piecewise linear.

We can compute the levels of an arrangement in $O(n^2)$ time as follows. First we sort all the lines via slope. Next, for each line, ℓ in the arrangement, we first compute the level of the leftmost

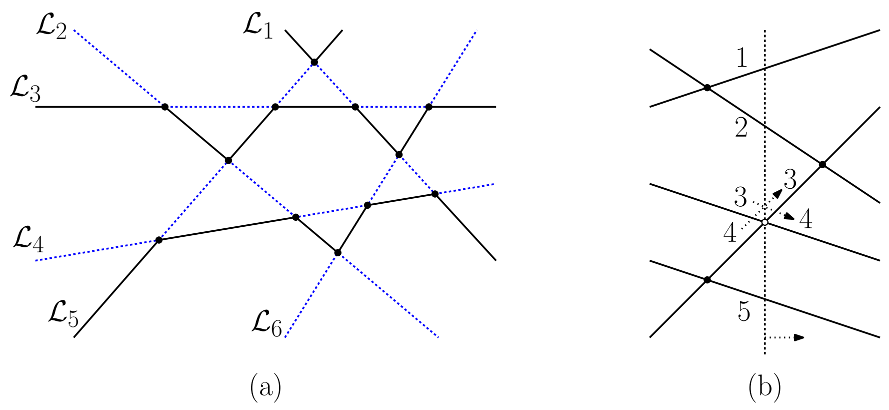

Figure 9. (a) Levels of an arrangement. (b) rightmost face intersected by ℓ and the left bounding edges created by ℓ_1.([1])

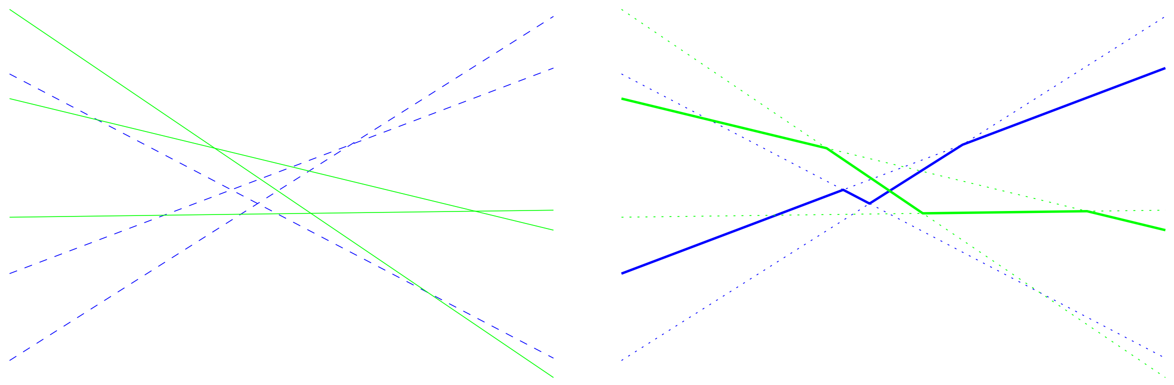

Figure 10. An arrangement of three green lines (solid) and three blue lines (dashed). The median levels for the green and blue are marked in bold on the right. ([3])

vertex ($x = -\infty$) of ℓ in this sorted order in $O(n)$ time. Next we walk along ℓ from left to right using the edge lists. We maintain the level as we walk. The level only changes at a vertex of the arrangement. Its change can be computed by inspecting edges incident to that vertex. By the zone theorem, we can do the work associated with every line in $O(n)$ time, for a total runtime of $O(n^2)$.

Alternatively, we can use plane-sweep from left to right to do this in $O(n^2 \log n)$ time (keeping "events" in a priority queue - hence the $\log n$ factor). Whenever, we come to the next vertex, we swap the level numbers associated with the two lines at the intersection. See Figure 9(b).[1]

6.3 Ham-sandwich algorithm

We now discuss a discrete version of the celebrated "ham-sandwich" theorem.

Given: R and B are two finite sets of points.

Goal: Output a line that *bisects* both R and B. That is in either open half-plane defined by the line, there are no more than $|R|/2$ points from R and no more than $|B|/2$ points from B.

Theorem: We can always find such a line in $O(n^2)$ time.

[1]The $\log n$ factor can be eliminated via topological plane sweep.

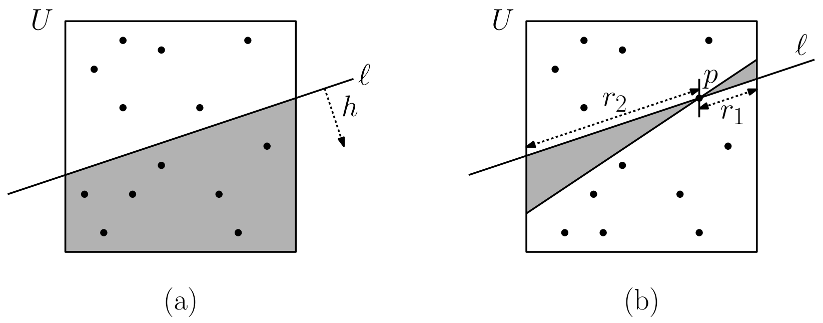

Figure 11. Computing Discrepancy [1]

Proof: Assume $|R|$ and $|B|$ are both odd. If either set is not, remove an arbitrary point from it. Then, a bisector of the remaining sets will also bisect the original sets.

Now create an arrangement of both $\mathcal{A}(R^*)$ and $\mathcal{A}(B^*)$, and compute the levels for both the red and blue points. Let the *median* level for the red points be the level $(|R| + 1)/2$, and note that it defines bisecting lines for R. The leftmost and rightmost segment of this median level are defined by the same line, ℓ_r, since slope alone determines the ordering of the lines at $\pm\infty$. Similarly, there is a line ℓ_b that determines the leftmost and rightmost segment of the median level for $\mathcal{A}(B^*)$.

Note that ℓ_r and ℓ_b have different slopes since no two points in $R \cup B$ have the same x coordinate.[2] Thus, since the median levels for $\mathcal{A}(R^*)$ and $\mathcal{A}(B^*)$ are piecewise linear functions, these median levels must also intersect somewhere in $\mathcal{A}(R^* \cup B^*)$ (see Figure 10(right)). In the primal plane, this intersection point represents a half-plane that bisects R and B. □

6.4 Red Blue Matching

Lemma: Given n red and n blue points in the plane, we can match them up using non-intersecting line segments.

Proof: The following recursive algorithm constructs the line segments.

1. If $|P| = 2$, match up the two points with a line segment
2. If $|P| > 2$
   (a) Compute a ham-sandwich cut that bisects P
   (b) If $|P|$ is odd, match the red and blue point on the bisecting line
   (c) If $|P|$ is even, rotate or translate the cutting line slightly so that (1) no points are on the line; and (2) the line is still a bisecting cut
   (d) Recursively compute a matching for both P_1 and P_2, where P_1 and P_2 are the points on either side of the cut.

It's now pretty easy to show (inductively!) that the above algorithm produces line segments that match up each red point to a unique blue point, and that these line segments do not intersect. □

6.5 Halfplane Discrepancy

Discrepancy is a problem closely related to sampling. It measures how well a discrete sample can approximate a continuous structure. Initially, it was important in computer graphics for determining how well a collection of random rays would be spread over a continuous surface. But it has new applications in big data: finding how well a small sample represents a larger population; and also probability and randomized algorithms: discretizing continuous probability distributions.

We are given a set P of n points in the unit square U, and we want for any halfplane h that the fraction of points lying in the halfplane is close to the area of $h \cap U$. If we define $\mu(h)$ to be the area of $h \cap U$ and $\mu_P(h) = |P \cap h|/|P|$, then we define the *discrepancy* for a point set P and a halfplane h as:

$$\Delta_p(h) = |\mu_h - \mu_P(h)|.$$

We define the *halfplane discrepancy* of P as supremum (least upperbound) of this quantity over all h:

$$\Delta(P) = \sup_h |\mu_h - \mu_P(h)|.$$

The Discrepancy Problem:
Given: A set of n points P lying in the unit square.
Goal: Compute $\Delta(P)$ and return a halfplane that achieves this discrepancy.

6.5.1 The Solution

The following lemma helps us compute $\Delta(P)$.

Lemma: Let h be the halfplane that maximizes discrepancy with respect to P. Let ℓ denote the line that bounds h. Then either (1) ℓ passes through one point of P and this point is the midpoint of the line segment $\ell \cap U$; or (2) ℓ passes through two points of P.

Proof: Consider a halfplane induced by a line ℓ that does not intersect any points. Then the discrepancy can be increased by either moving ℓ up or down (whichever direction increases discrepancy) until it hits a point. (See Figure 11 (a)). Thus ℓ must intersect at least one point.

Now consider the case where ℓ intersects just one point, but that this point does *not* form the midpoint of the line segment $\ell \cap U$. In this case, there are two line segments r_1 and r_2 of different length that are induced by the point (See Figure 11 (b)).

Assume $r_1 < r_2$, and consider the rotation of h about p by a small angle ϕ. Then the asymptotic approximation to the gain due to the area on the right is $r_1^2\phi/2$, since this triangle can be approximated by an sector of angle ϕ in a circle of radius r_1.[3] The loss due to the area on the left is similarly asymptotically $r_2^2\phi/2$.

Thus this rotation will decrease the area of the region lying below h. Similarly a rotation in the opposite direction will increase the area of the region lying below h. Hence, it is again possible to increase the discrepancy via rotation in the appropriate direction. □

[2]True by general position assumption. We can rotate the plane slightly to prevent this if it's not true.

[3]Technically, we can say that as ϕ goes to 0, the area of the triangle is $r_1^2\phi/2+o(r_1)$, since the error in approximating the shape as a sector angle versus a triangle is $o(r_1)$

Given this lemma, we can solve the problem in $O(n^2)$ time as follows. Let a *type 1* line be one that passes through one point of P and this point is the midpoint of the line segment $\ell \cap U$. Let a *type 2* line be one that passes through two points of P. Note that there are only $O(n)$ type 1 lines, so we can easily compute the discrepancy of the halfplanes induced by these lines in $O(n^2)$ time.

For the type 2 lines, we use arrangements and levels. In particular, we know that any type 2 line will be a vertex in the arrangement in the dual of P^*, i.e. in $\mathcal{A}(\mathcal{P}^*)$. For each such vertex $p \in \mathcal{A}(\mathcal{P}^*)$, we can compute the area of the two halfplanes induced by the line p^* in the bounding box U in constant time. Thus we can compute μ_h in constant time per type-2 halfplane h.

Moreover, we can compute the levels of all the points in $\mathcal{A}(\mathcal{P}^*)$ in $O(n^2)$ time as described in the Section 6.2. This allows us to compute in constant time the value $\mu_P(h)$ in constant time per type-2 halfplane h.

Since the total number of type-2 halfplanes is $O(n^2)$ and we can compute the discrepancy induced by each in constant time, we can also find the maximum discrepancy of these type-2 halfplanes in $O(n^2)$.

7 Duality and Arrangements in Higher Dimensions

In $\mathbb{R}^d$, identify the y axis with the d-th coordinate. Then a point can be written as $p = (x_1, \ldots, x_{d-1}, y)$ and a $d-1$ dimensional hyperplane, h is $y = \sum_{i=1}^{d-1} a_i x_i - b$. The dual of the hyperplane is the point $h^* = (a_1, \ldots, a_{d-1}, b)$ and the dual of the point is the hyperplane $b = \sum_{i=1}^{d-1} x_i a_i - y$. All the properties generalize to point/hyperplane relationships under the assumption that the y (i.e. b) axis is "vertical".

An arrangement of d dimensional hyperplanes has combinatorial complexity $O(n^d)$.

References

[1] David Mount. Computational Geometry. http://www.cs.umd.edu/class/fall2016/cmsc754/Lects/cmsc754-fall16-lects.pdf, 2016.

[2] Alper Ungor. Lecture 11: Arrangements and Duality. https://www.cise.ufl.edu/class/cot5520sp15/CG_ArrangementsCh8.pdf, 2016. https://www.cise.ufl.edu/class/cot5520sp15/CG_ArrangementsCh8.pdf.

[3] Emo Welzl and Prof. Bernd Gartner. Chapter 11: Line Arrangements. https://www.ti.inf.ethz.ch/ew/Lehre/CG13/lecture/Chapter https://www.ti.inf.ethz.ch/ew/Lehre/CG13/lecture/Chapter%2011.pdf.