# Computational Geometry: Polytopes

Don Sheehy

April 29, 2010

## 1 Recap of last time

Last time, we gave a geometric interpretation of Linear Programming. That is, how do we solve the problem:

$$\text{maximize } c^T x \text{ subject to } Ax \leq b. \tag{1}$$

We saw that the case were the desired vector $x$ is 2-dimensional, the problem can be solved in time linear in the number of constraints (i.e. the number of rows of $A$).

## 2 Two types of polytopes

Today will be our last proper lecture of the course, and we will be covering some details of the theory of polytopes. We will be going back through the various topics seen throughout this course and will emphasize the important role that polytopes have played in every single topic. This might even make you wonder why we wait until the very end to explore this important unifying object in more detail. The reason for this ordering is related to the relative effectiveness of out hindsight versus our foresight.

**Polytopes from Convex Hulls** We encountered poloytopes on the second day of class when we discussed the convex hull problem. We described the convex closure of a finite set of points as a polytope. In the plane, we simply called this a convex polygon.

**Polytopes from Linear Programs** We also encountered polytopes as the set of feasible solutions to a linear system. In that case, the object under consideration was the common intersection of a collection of halfspaces. There was a major difference between these polytopes and the ones that we saw with convex hulls, namely that these polytopes may not be bounded. That is, might not be any box gig enough to contain the whole thing.

Modulo this business with the unbounded polytopes, which can be handled with some more technical definitions, these two classes of polytopes are the same.

**Theorem 2.1** (Main Theorem of Polytopes)**.** *Every bounded polytope can be represented both as the convex hull of a finite point set or as the intersection of a finite collection of halfspaces.*

The intuition here is the convex closure of a finite point set is the intersection of all the support halfspaces and intersection of halfspaces is just the convex closure of all the vertices (intersections of $d$ bounding hyperplanes).

This theorem doesn't really tell the whole story, especially, when we think about these polytopes in terms of general position. Given a set of points in general position, the support hyperplanes of the convex closure are **not** in general position. This might seem surprising, so let's take a look at a simple example.

Consider 6 points in general position in $\mathbb{R}^3$. If we take their convex hull, we might get something that looks like a skewed octahedron. Now, the fact that they are in general position mean that no 4 are coplanar or cocircular. It also mean that if we wiggle the points, we still get the same set of faces. That is, it will still look like an octahedron, even if we perturb the points.

Let's look now at the support hyperplanes. If we wiggle these planes, what happens to the polytope? To make it easier to visualize, what happens if we wiggle a single plane? The corresponding face might not be a triangle anymore.

Can you think of a 3-polytope that doesn't change its face structure if you wiggle the support planes? How about a cube? What is different about a cube? Are the vertices of the cube in general position?

We can now introduce the two crucial definitions that separate these two cases.

**Definition 2.1.** *A polytope is* simplicial *if every facet is a simplex.*

**Definition 2.2.** *A polytope is* simple *if every vertex is incident to d facets.*

The key observation is that if the halfspaces are in general position, then their intersection will be a *simple* polytope. On the other hand, if we have a set of points in general position, their convex closure will be a *simplicial* polytope.

It is not coincidental that I chose the octahedron and the cube as my primary example for illustrating these two classes of polytopes. Recall that the cube and the octahedron are polar duals of each other. The duality between the definition of simple and simplicial polytopes is also not coincidental. If a facet is a simplex then it has $d$ vertices. Its dual is a vertex incident to $d$ facets.

**Theorem 2.2.** *The polar of a simple polytope is simplicial and the polar of a simplicial polytope is simple.*

**Question 2.1.** *Is there a* 3*-polytope that is both simple and simplicial?*

We have also seen this duality between simple and simplicial polytopes before, when we studied the relationship between Voronoi diagrams nand Delaunay Triangulations. Recall that the Delaunay triangulation can be constructed by lifting the vertex set of onto the paraboloid and then projecting the lower hull. In fact, the Voronoi diagram can also be constructed as the projection of the

lower hull of a polytope. After lifting the points onto the paraboloid, we take the tangent planes of the paraboloid at those points and intersect the halfspaces bounded by these planes containing the paraboloid to get the desired

# 3 The complexity of polytopes in higher dimensions

Say $f_p(P)$ is the number of $k$-faces of a polytope $P$. That is $f_0(P)$ is the number of vertices and $f_1(P)$ is the number of edges, etc. For 3-polytopes, we made ample use of Euler's formula. It was helpful because it guaranteed that the numbers of edges and faces were both linear in the number of vertices. Or, in the notation we have just introduced, $f_1(P) = O(f_0(P))$ and $f_2(P) = O(f_0(P))$. Unfortunately, this does not work as we increase the dimension of our polytopes.

In $R^4$, it is possible to find ourselves with a polytope with $O(n^2)$ edges (where $n = f_0(P)$). If we are interested in applying geometric methods that will require us to actually represent all of the faces, then we will start to see a blow up in the size of our data structure. The next logical question to ask is: **what is the worst case blowup?**

If we start limiting our attention to the number of edges in a 4-polytope, there is an obvious upper bound of $\binom{n}{2}$ edges, one for each pair of vertices. In fact, there are polytopes that achieve this bound. These are polytopes for which every pair of vertices is connected by an edge. Let's see how to construct one.

Let $x : \mathbb{R} \to \mathbb{R}^d$ be defined as

$$x(t) = \begin{pmatrix} t \\ t^2 \\ \vdots \\ t^d \end{pmatrix}$$

This is the parametrization of a curve known as *the moment curve*.

For any set of real numbers $\{t_1, \ldots, t_n\}$, the *cyclic polytope* is defined as $C_d = CC\{t_1, \ldots, t_n\}$. The face numbers, $f_k(C_d)$ are independent of the choice of the $t_i$'s. The cyclic polytope is the worst case for face numbers of any $d$-polytope.

**Theorem 3.1.** *For all $d$-polytopes $P$ and all $k$, $f_k(P) \leq f_k(C_d)$.*

This theorem is known as the Upper Bound Theorem for polytopes. It establishes the worst case but does not tell us how bad that worst case is. For that, we'll start with another definition.

**Definition 3.1.** *A polytope $P$ is* $k$-neighborly *if every set of $k$ or fewer vertices is the vertex set of a face of $P$.*

In particular, a polytope that is 2-neighborly has a 1-skeleton that forms the complete graph.

So, how many faces can a polytope have?

**Theorem 3.2.** *The cyclic $d$-polytope $C_d$ is $\lfloor \frac{d}{2} \rfloor$-neighborly.*

This theorem, first proven by Gale in 1950, immediately gives us worst-case bounds for the number of $k$-faces of a $d$-polytope for $k \leq \frac{d}{2}$. It implies that for such $k$,

$$f_k(C_d) = \binom{n}{k}.$$

Now, it might seem like we are introducing a lot slack into the bounds by not counting faces of dimension greater than $d/2$. This is not the case, because if there were more than $\binom{n}{\frac{d}{2}}$ faces of some dimension $k > \frac{d}{2}$, then the polar would have more then $\binom{n}{k}$ faces of dimension $d - k$, which is impossible for $k > \frac{d}{2}$.

As computer scientists, we usually just want to know how this number of faces behaves asymptotically. For that it suffices to look at the largest face number to get that the number of faces in a polytope is at most $O(n^{\lfloor \frac{d}{2} \rfloor})$.

# 4 Review of this course (in light of polytopes)

## 4.1 Sorting

The Delaunay triangulation and the voronoi diagram are both projections of polytopes. We focused more on the Delaunay triangulation but the Voronoi diagram can also be constructed by projecting a polytope. In that case, we we take the points on the paraboloid and replace them with tangent hyperplanes. The intersection of the halfspaces bounded by these hyperplanes projects to the Delaunay triangulation.

Even in 2D, you can see that every vertex of the Voronoi diagram has only 3 faces around it. This corresponds both to the fact that the dual face is a triangle and the fact that it is the projection of a simple polytope.

## 4.2 Graphs

We talked a lot about polytopes in our section on graphs. In particular, we were interested in 3-polytopes. At first, it was just convenient to think about the graphs in terms of both stresses and liftings to poytopes. The Maxwell-Cremona correspondence made this correspondence concrete. Later, we saw Steinitz's Theorem, the equivalence of 3-polytopal graphs and simple, planar, 3-connected graphs. Unfortunately, there is no equally nice generalization of Steinitz's Theorem to higher dimensions.

In the plane, we were regularly using Euler's formula as a way to bound the complexity of the objects we were working with, be it a triangulation of an arrangement of lines. Although there is a generalization of Euler's formula to general spaces (even non-euclidean ones), it doesn't give us the nice combinatorial properties that we get in the plane. In particular, it doesn't guarantee that the number of edges, faces, or other high dimensional faces must remain linear in the number of vertices. Let's see how this plays out for polytopes.

## 4.3 Selection

When discussing centerpoints, we saw that the halfspace depth produced a nested sequence of polytopes call the depth levels. In fact, many different notions of depth give rise to convex level sets. Some, do not. The wedge depth that we saw in class and the simplicial depth that we heard about with the projects are two examples.

## 4.4 Search

When discussing halfspace range search, we used a dual representation that allowed us to keep around a whole complex of polytopes representing equivalence classes of searches.

## 4.5 Optimization

Finally, this brings us full circle to optimization. Here we saw how linear programming has a natural interpretation in terms of polyopes. Moreover, we saw that random incremental construction and backwards analysis allow for a linear time algorithm. This required that we could do some geometric operations in constant time and in fact, the dependence on the dimension was $O(d!)$, but still $O(n)$ when $d$ is a constant.