18.409 The Behavior of Algorithms in Practice 9 Apr. 2002

Lecture 14

Lecturer: Dan Spielman *Scribe: Brian Sutton*

In this lecture, we study the worst-case complexity of the simplex method. We conclude by stating, without proof, one of the major theorems in the smoothed analysis of the simplex method.

1 More common geometric view of the simplex method

The animation that we saw in an earlier lecture is not the most common way of viewing the simplex method geometrically. Here we give a more common interpretation.

Start with the LP formulation

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x} \\ &\text{s.t. } \mathbf{a_i}^T\mathbf{x} \le b_i. \end{aligned}$$

Each constraint $\mathbf{a_i}^T\mathbf{x} \le \mathbf{b_i}$ defines a halfspace, so the complete set of constraints defines a polytope P (assuming that the LP is feasible and finite). Solving the LP is equivalent to finding the vertex of this polytope that lies farthest in the direction of $\mathbf{c}$.

The simplex method is

1. Find some vertex of P.
2. Move to an adjacent vertex in the direction of $\mathbf{c}$, i.e. which improves the objective function, and iterate.

Definition 1. *A* basic feasible solution *of an LP is a feasible* $\mathbf{x}$ *for which exactly* d *constraints are tight. Equivalently* $\mathbf{x}$ *is a vertex of the feasible polytope.*

Note that this definition is equivalent to the definition given in an earlier lecture.

Observation 2. *If the* $\mathbf{a_i}$*'s and* b_i*'s are in general position, then*

1. *No* $\mathbf{x}$ *satisfies* $d+1$ *constraints with equality.*

2. The feasible polytope P is simple.

3. Exactly d planes meet at any vertex.

4. Every vertex has exactly d neighbors.

One can show that under a perturbation of the b_i's, the feasible polytope is simple with high probability.

2 Worst-case complexity of the simplex method

We will assume that P is a simple polytope. Our definition of the simplex method,

1. Find some vertex of P,
2. Move to one of the d neighbors improving the objective function, and iterate,

is incomplete. It does not specify how to find the initial vertex, nor how to choose the next neighbor in the sequence. Finding the initial vertex will be addressed in a later lecture. Now we look at *pivot rules*, which specify which neighbor to choose in step (2) of the algorithm. Chvatl's book is a good reference on pivot rules.

2.1 Pivot rules

1. Greedy: Choose the vertex giving the maximal improvement in the objective function.
2. Bland's rule: Index the constraints. The current vertex $\mathbf{v}$ satisfies exactly d constraints with equality, i.e. it lies at the intersection of d faces of the polytope. Any adjacent vertex $\mathbf{w}$ lies on $d-1$ of these faces. So when moving from $\mathbf{v}$ to $\mathbf{w}$, exactly one constraint is loosened from tightness. Choose $\mathbf{w}$ so that the constraint that is loosened has least index.
3. Dantzig's largest coefficient: Similar to Bland's.
4. Steepest edge: Make the most progress per distance travelled.
5. Random.

2.2 Time complexity of deterministic pivot rules

The simplex method with any of the above four deterministic pivot rules takes exponential time in worst-case. There are no known polynomial time pivot rules. We have the belief, "Almost all deterministic pivot rules take exponential time in the worst case."

A powerful tool for generating polytopes that produce slow behavior for a particular pivot rule was provided by Amenta and Ziegler.

The following question came from the audience: Is the performance bad because the polytope has large diameter, or because the algorithm chooses a poor path?

It seems that the polytope of an LP in d dimensions with n constraints has polynomial diameter in n and d. Our best proven result is

Theorem 3 (Kalai-Kleitman). *The diameter of the polytope is $\leq n^{\log_2 n}$.*

But we have the conjecture

Conjecture 4 (Hirsch). *The diameter of the polytope is $\leq n - d + 1$. (Actually, the conjecture might be $n - d - 1$. We don't remember.)*

2.3 Time complexity of random pivot rules

We know very little about the time complexity of random pivot rules. We do know that there exist random pivot rules that take $n^{O(\sqrt{d})}$ steps. This result was proved independently by Kalai and Matoušek-Sharir-Welzl.

2.4 Worst-case complexity of Bland's rule

The initial paper on constructing worst-case inputs for a pivot rule was by Klee-Minty.

We will construct a deformed hypercube that will cause Bland's rule to visit every vertex when started at a particular vertex.

In d dimensions, a hypercube is defined by the $2d$ constraints

$$\begin{gathered} 0 \le x_1 \le 1 \\ 0 \le x_2 \le 1 \\ \cdots \\ 0 \le x_i \le 1 \\ \cdots \end{gathered}$$

Our feasible set is

$$P := \begin{array}{rcccl} 0 & \le & x_1 & \le & 1 \\ \varepsilon x_1 & \le & x_2 & \le & 1 - \varepsilon x_i \\ & & \cdots & & \\ \varepsilon x_{i-1} & \le & x_i & \le & 1 - \varepsilon x_{i-1} \\ & & \cdots, & & \end{array}$$

with $\varepsilon = 1/3$. We will index the constraints as follows:

$$\begin{array}{rcccl} 0 & \overset{1}{\le} & x_1 & \overset{2}{\le} & 1 \\ & & \cdots & & \\ \varepsilon x_{i-1} & \overset{2i-1}{\le} & x_i & \overset{2i}{\le} & 1 - \varepsilon x_{i-1} \\ & & \cdots & & \end{array}$$

Claim: $\mathbf{x}$ is a vertex of P if and only if inequality $2i-1$ or $2i$ is tight for each i. To prove the claim, observe that it is impossible to satisfy inequalities $2i-1$ and $2i$ simultaneously. Thus, in order to make d inequalities tight, we must choose one inequality from each pair.

Theorem 5. *Bland's rule visits every vertex of P when solving*

1. $\max x_d$ *s.t.* $\mathbf{x} \in P$ *if started at* $(0, 0, \ldots, 0)$.
2. $\min x_d$ *s.t.* $\mathbf{x} \in P$ *if started at* $(0, \ldots, 0, 1)$.

Proof. The proof is by induction. Assume that the theorem is true for $d-1$ dimensions. We will prove part (1) of the theorem. The proof of part (2) is very similar.

The walk around the polytope begins with a sequence of points satisfying $x_d = \varepsilon x_{d-1}$. Because Bland's rule will not loosen this equality until it is impossible to loosen any equality of lower index (without decreasing the objective function), the walk starts by maximizing x_{d-1} over the polytope in $d-1$ dimensions that is the projection of P to coordinates

$x_1, \dots, x_{d-1}$. Thus, by induction (part (1)), the walk visits every vertex of P satisfying $x_d = \varepsilon x_{d-1}$.

The next step is to $(0, 0, \dots, 0, 1, 1 - \varepsilon)$. Now, x_d gets larger as x_{d-1} gets smaller, so that the walk visits every vertex satisfying $x_d = 1 - \varepsilon x_{d-1}$ by induction (part (2)). $\square$

2.5 Pivot rules under perturbation

There are two types of perturbations one might consider:

1. Absolute perturbations: Write the feasible set as $A\mathbf{x} \leq \mathbf{b}$, and suppose A is subject to absolute perturbations. This model seems strange, since A can be so sparse. (In our worst-case instance for Bland's rule, A contained only one nonzero entry in any row.)

2. Zero-preserving perturbations: Our proof for the exponential running time of Bland's rule holds up under zero-preserving perturbations. What about problem instances for the greedy pivot rule? The first worst-case instance, provided by Jeroslow, does not hold up under perturbation. It is not known whether the instance provided by Amenta and Ziegler holds up under perturbation.

3 Smoothed analysis of the simplex method

Our smoothed analysis of the simplex method will take advantage of some nice properties of the shadow-vertex pivot rule.

3.1 Definition of the shadow-vertex pivot rule

The inspiration for this rule comes from the fact that linear programming is easy in two dimensions. The idea is to project the feasible polytope to a convex polygon, i.e. "take a shadow," that satisfies

1. The optimal vertex maps to the exterior,

2. The start vertex maps to the exterior,

and then optimize over the shadow. To construct such a shadow, find a linear function $\mathbf{t}$ that is optimized by the current vertex and map each vertex $\mathbf{v}$ of the polytope to $(\langle \mathbf{v}, \mathbf{c}\rangle, \langle \mathbf{v}, \mathbf{t}\rangle)$.

Claim: If a vertex $\mathbf{v}$ of the polytope projects to the exterior of the shadow, then one can compute which neighbor of $\mathbf{v}$ will be visited next on the shadow.

Thus, algorithmically, the shadow-vertex pivot rule is reasonable. The rule's major analytical advantage is the fact that we can succinctly describe the vertices that are visited as follows. Let $\mathrm{opt}_{\mathbf{z}}(A, \mathbf{b})$ denote the vertex optimizing $\mathbf{z}^T\mathbf{x}$ s.t. $A\mathbf{x} \le \mathbf{b}$. Then the vertices encountered when optimizing $\mathbf{c}$ are

$$\bigcup_{0\le\lambda\le1} \mathrm{opt}_{(1-\lambda)\mathbf{t}+\lambda\mathbf{c}}(A, \mathbf{b}).$$

3.2 Worst-case complexity of the shadow-vertex pivot rule

The shadow-vertex pivot rule has exponential worst-case complexity. The following instance induces exponential running time:

$$\begin{array}{rcccl} 0 & \le & x_1 & \le & 1 \\ \varepsilon x_1 & \le & x_2 & \le & 1-\varepsilon x_i \\ & & \cdots & & \\ \varepsilon(x_{i-1}-\gamma x_{i-2}) & \le & x_i & \le & 1-\varepsilon(x_{i-1}-\gamma x_{i-2}) \\ & & \cdots, & & \end{array}$$

where $\varepsilon = 1/3$ and $\gamma = 1/12$.

3.3 Smoothed analysis of the shadow-vertex pivot rule

We will prove the following theorem in upcoming lectures.

Theorem 6. *Let $\mathbf{t}$ and $\mathbf{c}$ be arbitrary vectors, $b_i \in \{1, -1\}$, and $\hat{\mathbf{a}}_{\mathbf{1}}, \ldots, \hat{\mathbf{a}}_{\mathbf{n}}$ vectors of norm at most 1. If $\mathbf{a_1}, \ldots, \mathbf{a_n}$ are Gaussian random vectors of variance σ^2 centered at $\hat{\mathbf{a}}_{\mathbf{i}}$, then*

$$E\left[\left|\bigcup_{0\le\lambda\le1} \mathrm{opt}_{(1-\lambda)\mathbf{t}+\lambda\mathbf{c}}(\mathbf{a_1}, \ldots, \mathbf{a_n}, b_1, \ldots, b_n)\right|\right] \le poly(n, d, \frac{1}{\sigma}).$$

It is an open problem to prove a similar result for perturbations that preserve feasibility (or unfeasibility).