Lesson 10. Linear Programming

M. Pellegrini*

February 18, 2004

1 Introduction

Material from [Cla95], [MR95, 262-275], [GW96] and [Pel01]. Unfortunately the notations used in these four papers/chapters differ slightly in such a way as to create sometimes a bit of confusion.

1.1 Linear algebraic formulation

We follow [GW96] and the original paper [Cla95]. For integer parameters $n, d > 0$, let $c \in R^d$, $A \in R^{d \times n}$ and $b \in R^n$, the LP problem in normal form is to find $x \in R^d$ that minimizes $c^T x$ and such that $x \geq 0$ and $Ax \leq b$.

1.2 Geometric formulation

The d inequalities $x \geq 0$ are called *nonnegativity constraints*: H_+. The n inequalities $Ax \leq b$ are called *matrix constraints*: H. We talk generically of *constraints* to indicate any of them.

Each constraint corresponds to an halfspace h_i. The Feasibility set is the set of points that satisfy all constraints, so $F = \bigcap_{i=1}^{n+d} h_i$ is a polyhedron in R^d.

If $F \neq \emptyset$ the problem is called feasible.

We simplify the treatment by assuming $c \geq 0$. This ensures that if $F \neq \emptyset$ then a solution point in F exists and is finite. The nonnegativity constraints force the solution to be a finite point.

1.3 Subsets of H

Take a subset $G \subseteq H$ and define $G^+ = G \cup H_+$. Clearly $H^+ = H \cup H_+$.

*Istituto di Informatica e Telematica del CNR, Via G. Moruzzi 1, 56100-Pisa (Italy). pellegrini@iit.cnr.it

Call $v(G^+)$ the vertex in $\bigcap_{i \in G^+} h_i$ that minimizes $c^T x$ and is the lexicographically minimum such vertex.

If $c \geq 0$ and $F \neq \emptyset$ it is easy to show that $v(G^+)$ is always defined and finite for any $G \subset H$.

1.4 Order of subsets

We define the following total order upon the subsets of H.

$G \prec F$ if and only if $(c^T v(F^+) < c^T v(G^+))$ or ($(c^T v(F^+) = c^T v(G^+))$ and $v(F^+)$ precedes lexicographically $v(G^+)$).

Also $G \approx F$ is and only if $v(F^+) = v(G^+)$. The combination of the two relations is $\preceq$.

1.5 Violations

For $G \subseteq H$ and $h \in H \setminus G$: h *violates* $v(G^+)$ if h does not contains $v(G^+)$.

1.6 Basis

A basis $B \subseteq G \subseteq H$ is minimal subset B of G such that $v(B^+) = v(G^+)$. If each vertex in the arrangements induced by H and H_+ is incident to d hyperplanes bounding the half-spaces, then the basis is unique.

1.7 Relationship of violators and basis

Because B is minimal for any $C \subset B$ we have $C \prec B$.

h violates G if and only if $G \prec G \cup \{h\}$.

h is *extreme* in G if an only if $G \setminus \{h\} \prec G$.

1.8 Some properties

(i) For $F \subseteq G \subseteq H$, $v(F^+) \preceq v(G^+)$.

(ii) If $v(F^+) = v(G^+)$ then $h \in H$ violates F if and only if it violates G.

(iii) Any basis for $G \subseteq H$ has at most d constraints and at most d extreme constraints.

2 Algorithms

2.1 Clarkson's algorithm I [Cla95]

This algorithm is useful when n is large compared to d, where large here means $n > 9d^2$. The goal of the algorithm is to compute $v(H^+)$, or more precisely to find a basis for H.

The main idea is as follows. Take a subset $G \subset H$ and the basis B of H (assumed unique) and the set of violators $V(G, H)$ for $v(G^+)$.

Lemma 1 *If $V(G, H) \neq \emptyset$ then there must be an element of $B \setminus (G \cup H_+)$ that is in $V = V(G, H)$.*

Proof. We assume that $V \neq \emptyset$ and $B \cap V = \emptyset$ and we derive a contradiction.

Consider $v(G^+)$, since the elements of B are not violating it we may add them to G without changing the solution. Therefore $v(G^+) = v((G \cup B)^+)$. Since $B \subseteq (G \cup B)$ we have $v(B^+) \preceq v((G \cup B)^+)$. On the other hand by definition $v(B^+) = v(H^+)$.

Since $G \subseteq H$ we have $v(G^+) \preceq v(H^+)$.

Putting together all relations:

$$v(H^+) = v(B^+) \preceq v((G \cup B)^+) = v(G^+) \preceq v(H^+)$$

Because there is only one point that has rank both lower-equal and higher-equal than $v(H^+)$, it follows that $v(G^+) = v(H^+)$ and therefore the violating set $V(G, H)$ must be empty.

$\blacksquare$

Next we show that when we take random samples the expected size of the conflict list (set of violators) is small.

Lemma 2 *Let H be a set of n constraints in R^d. Let $G \subseteq H$ be a subset of H, and $R \subseteq H$ be a random subset of H of size r, with $1 \leq r \leq n$. Then the expects number of violators for $v((G \cup R)^+)$ is bounded by*

$$\frac{d(n-r)}{r+1}.$$

Proof. Consider the set of subsets of size r of H: $\binom{H}{r}$.

Define the indicator function: $\chi_{G^+}(R, h)$ that has value 1 if $h \in H$ violates $v((G \cup R)^+)$, 0 otherwise.

By definition:

$$E[|V|] = \frac{1}{\binom{n}{r}} \sum_{R \in \binom{H}{r}} \sum_{h \in H \setminus R} \chi_{G^+}(R, h)$$

We use now backward analysis by taking all $r+1$ subsets $\binom{H}{r+1}$, and considering $R = Q \setminus \{h\}$, thus:

$$\chi_{G^+}(R,h) = \chi_{G^+}(Q \setminus \{h\}, h)$$

for all choices of Q and $h \in Q$.

$$\sum_{R \in \binom{H}{r}} \sum_{h \in H \setminus R} \chi_{G^+}(R,h) = \sum_{Q \in \binom{H}{r+1}} \sum_{h \in Q} \chi_{G^+}(Q \setminus \{h\}, h)$$

Now $\chi_{G^+}(Q \setminus \{h\}, h) = 1$ only if h is extremal for Q, and there are at most d extremal constraints in any set.

$$\leq \sum_{Q \in \binom{H}{r+1}} d = d\binom{n}{r+1}$$

Finally:

$$E[|V|] = d\frac{\binom{n}{r+1}}{\binom{n}{r}} = \frac{d(n-r)}{r+1}.$$

∎.

Observation: Choosing $r = d\sqrt{n}$ we get $E[|V|] \leq \sqrt{n}$. Thus by Markov inequality the $Prob[|V| > 2\sqrt{n}\} < 1/2$.

So after d additions to G, G must contain the basis of H and the cycle stop. On average the number of times the cycle is performed in $2d$ (This can be seen seeing the success of the test on $|V|$ as a negative Bernoulli random variable).

2.2 Algorithm Clarkson I.

(see slides).

2.3 Complexity analysis

In the main loop we perform for each iteration, n violations test, each of cost $O(d)$. The loop is executed an expected number of times $2d$. The total costs of all operations except calls to simplex and recursive calls is $O(nd^2)$.

Since G is initially empty and at each iteration a set of size $2\sqrt{n}$ is added over d iterations, we have that $|R \cup G|$ is at most $3d\sqrt{n}$.

Calling $S(m, d)$ the cost of the simplex algorithm over m constraints in dimension d, and $T(n)$ the expected cost of Clarkson's algorithm we get:

for $n \geq 9d^2$:

$$T(n) \leq O(nd^2) + 2dT(3d\sqrt{n})$$

for $n < 9d^2$

$$T(n) = S(n, d)$$

.

Note that the threshold for which there is no gain in the problem size is when $n = 3d\sqrt{n}$, that happens when $n = 9d^2$.

We can use the above recursion to estimate separately the number of simplex calls and the total extra costs.

A direct calculation gives a recursion depth of $\log \log n$ and a number of calls to simplex is $(\log n)^{\log d+1}$.

Next we will see a second scheme with a better performance.

3 Clarkson's algorithm II

The idea is to choose random samples, but this time we use a biased probability distribution where each constraint is weighted. The initial weight is 1 for each constraint. In the algorithm every time a constraint violates the current solution its weight is doubled. Intuitively the elements of the basis for H will receive weights so high that they will be chosen with high probability.

Let $\mu : H \to N$ be the counter for constraints in H.

Initially $\mu(h) = 1$ for all $h \in H$. For any subset $G \subset H$, extend the notation to $\mu(G) = \sum_{h \in G} \mu(h)$.

Taking samples in H with bias μ is equivalent to considering H a multi-set where each element h is replicated $\mu(h)$ times.

We can prove the sampling lemma for multi-sets:

Lemma 3 *Let H be a multi-set of m constraints in R^d, and $R \subseteq H$ be a random multi-subset of H of size r, with $1 \leq r \leq m$. Then the expects number of violators for $v(R^+)$ is bounded by*

$$\frac{d(m-r)}{r+1}.$$

Proof. . Analogous to the case of sets. Omitted.

Choosing $r = 6d^2$ we have $E[\mu(V)] \leq \frac{m}{6d} = \frac{\mu(H)}{6d}$.

So By Markov inequality the total weight $\mu(V)$ is $\frac{\mu(H)}{3d}$ with probability at least 1/2. The expected number of iterations before the condition holds is 2. We call an iteration in which the condition holds a *successful* one.

Now we show that the cycle terminates within a finite number of iterations, more precisely after $3d \ln n$ successful iterations (thus on average after $6d \ln n$ iterations).

Lemma 4 *Let k be a positive integer. After kd successful iterations for the Basis B of H it holds: $2^k \leq \mu(B) \leq ne^{k/3}$.*

Proof. Clearly since $B \subseteq H$ at any time: $\mu(B) \leq \mu(H)$. At each successful iteration the weight of H is increased by at most $(1/3d)\mu(H)$, so after kd iterations;

$$\mu(B) \leq \mu(H) \leq n(1 + \frac{1}{3d})^{kd} < ne^{k/3}$$

On the other hand if $V \neq 0$ in each successful iteration at least one of the at most d elements of B gets its weight doubled. Therefore after dk iterations one element of B will have weight at least 2^k. This is a lower bound also for $\mu(B)$.

∎.

Take $k = 3 \ln n$. We have $2^k = n^{3ln2} \approx n^{2.0794415}$. We have $ne^{3 \ln n/3} = n^2$. Thus the lower bound is actually higher than the upper bound. This is a contradiction, therefore the algorithm must terminate before $3d \ln n$ successful iterations.

3.1 Complexity analysis

The algorithm performs an expected number $O(d^2 n \log n)$ of operations in testing violations, and an expected number $6d \ln n$ invocations to simplex with at most $6d^2 + d$ constraints. In short: $O(nd^2 \log n + (d \log n)S(6d^2 + d, d))$.

3.2 A mixed algorithm

Take algorithm Clarkson I and substitute each recursive call to calls to Algorithm Clarkson II. In this way we can reduce the complexity to: $O(d^2 n + (d^2 \ln n)S(d^2 + d, d) + d^4 \sqrt{n} \log n)$.

This for constant d this algorithm is linear in n.

4 Notes

At present the worst case complexity of the simplex is exponential in d. At present the best expected worst case bound for a Combinatorial randomized algorithm is subexponential: $O(nd^2 + exp(O(\sqrt{d \log d})))$ in [MSW96].

References

[Cla95] Kenneth L. Clarkson. Las vegas algorithms for linear and integer programming when the dimension is small. *J. ACM*, 42(2):488–499, 1995.

[GW96] Bernd Gärtner and Emo Welzl. Linear programming - randomization and abstract frameworks. In *Proc. 13th Ann. Symp. on Theoret. Aspects of Comput. Sci.(STACS)*, volume 1046 of *Lect. Notes Comp. Sci.*, pages 669–687, 1996.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[MSW96] J. Matousek, M. Sharir, and E. Welzl. A. subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, October/November 1996.

[Pel01] M. Pellegrini. Randomized combinatorial algorithms for linear programming when the dimension is moderately high. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 101–108, 2001.