Algorithms (VI)

Duality and Simplex

Guoqiang Li

School of Software, Shanghai Jiao Tong University

Homework

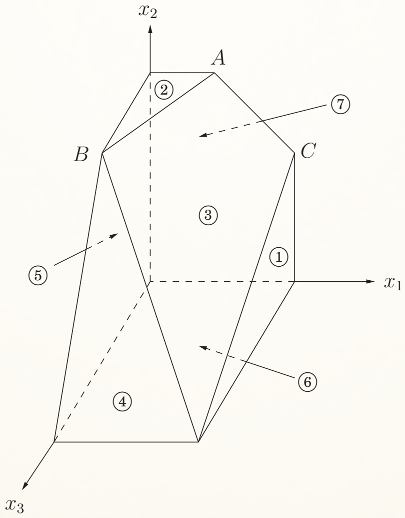
- Assignment 2 is announced! (deadline Apr. 13)

Linear Programming

LP

$$\begin{gathered}
\max x_1 + 6x_2 + 13x_3 \\
x_1 \leq 200 \\
x_2 \leq 300 \\
x_1 + x_2 + x_3 \leq 400 \\
x_2 + 3x_3 \leq 600 \\
x_1, x_2, x_3 \geq 0
\end{gathered}$$

The Example

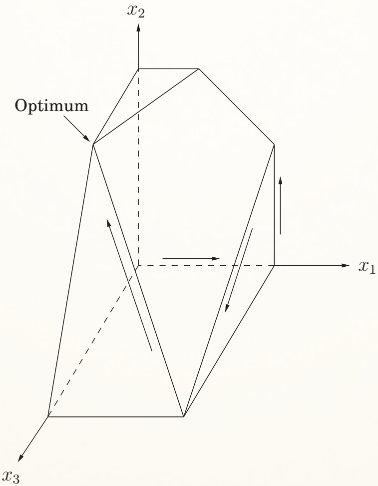


LP

- The point of final contact is the optimal vertex: $(0, 300, 100)$, with total profit $\$3100$.
- Q: How would the simplex algorithm behave on this modified problem?
- A possible trajectory

$$\frac{(0,0,0)}{\$0} \to \frac{(200,0,0)}{\$200} \to \frac{(200,200,0)}{\$1400} \to \frac{(200,0,200)}{\$2800} \to \frac{(0,300,100)}{\$3100}$$

The Example



Standard Form of LP

Therefore, we can reduce any LP (maximization or minimization, with both inequalities and equations, and with both nonnegative and unrestricted variables) into an LP of a much more constrained kind that we call the standard form:

- the variables are all nonnegative,
- the constraints are all equations,
- and the objective function is to be minimized.

$$\begin{array}{c} \max x_1 + 6x_2 \\ x_1 \le 200 \\ x_2 \le 300 \\ x_1 + x_2 \le 400 \\ x_1, x_2 \ge 0 \end{array} \quad \Longrightarrow \quad \begin{array}{c} \min -x_1 - 6x_2 \\ x_1 + s_1 = 200 \\ x_2 + s_2 = 300 \\ x_1 + x_2 + s_3 = 400 \\ x_1, x_2, s_1, s_2, s_3 \ge 0 \end{array}$$

Shortest Path in LP

In the shortest path problem, we are given a weighted, directed graph $G = (V, E)$, with weight function $w : E \to R$ mapping edges to real-valued weights, a source vertex s, and destination vertex t. We wish to compute the weight of a shortest path from s to t.

Shortest Path in LP

$$\max d_t$$

$$\begin{aligned} d_v \leq d_u + w(u,v) &\quad (u,v) \in E \\ d_s = 0 & \\ d_i \geq 0 &\quad i \in V \end{aligned}$$

Shortest Path in LP

Let $\mathcal{S} = \{S \subseteq V : s \in S, t \notin S\}$; that is, $\mathcal{S}$ is the set of all s-t cuts in the graph. Then we can model the shortest s-t path problem with the following integer program,

$$\min \textstyle\sum_{e \in E} w_e x_e$$

$$\begin{array}{rl} \sum_{e \in \delta(S)} x_e \geq 1 & S \in \mathcal{S} \\ x_e \in \{0, 1\} & e \in E \end{array}$$

where $\delta(S)$ is the set of all edges that have one endpoint in S and the other endpoint not in S.

- Can we relax the restriction $x_e \in \{0, 1\}$ to $0 \leq x_e \leq 1$?
- How about $x_e \geq 0$?

Duality

Product Planning Revisit

- Recall:

$$\begin{aligned} \max x_1 + 6x_2 \\ x_1 \leq 200 \\ x_2 \leq 300 \\ x_1 + x_2 \leq 400 \\ x_1, x_2, x_3 \geq 0 \end{aligned}$$

- Simplex declares the optimum solution to be $(x_1, x_2) = (100, 300)$, with objective value 1900.

Can this answer be checked somehow?

- We take the first inequality and add it to six times the second inequality:

$$x_1 + 6x_2 \leq 2000$$

Product Planning Revisit

- Multiplying the three inequalities by 0, 5, and 1, respectively, and adding them up yields

$$x_1 + 6x_2 \leq 1900$$

Multipliers

- Let's investigate the issue by describing what we expect of these three multipliers, call them y_1, y_2, y_3.

Multiplier	Inequality
y_1	$x_1 \leq 200$
y_2	$x_2 \leq 300$
y_3	$x_1 + x_2 \leq 400$

  These y_i's must be nonnegative, for otherwise they are unqualified to multiply inequalities.

- After the multiplication and addition steps, we get the bound:

$$(y_1 + y_3)x_1 + (y_2 + y_3)x_2 \leq 200y_1 + 300y_2 + 400y_3$$

- We want the left-hand side to look like our objective function $x_1 + 6x_2$ so that the right-hand side is an upper bound on the optimum solution.

Multipliers

$$x_1 + 6x_2 \leq 200y_1 + 300y_2 + 400y_3$$

if

$$y_1, y_2, y_3 \geq 0$$
$$y_1 + y_3 \geq 1$$
$$y_2 + y_3 \geq 6$$

The Dual Program

- We can easily find y's that satisfy the inequalities on the right by simply making them large enough, for example $(y_1, y_2, y_3) = (5, 3, 6)$.
- But these particular multipliers would tell us that the optimum solution of the LP is at most

$$200 \cdot 5 + 300 \cdot 3 + 400 \cdot 6 = 4300$$

  a bound that is far too loose to be of interest.
- What we want is a bound that is as tight as possible, so we should minimize

$$200y_1 + 300y_2 + 400y_3$$

  subject to the preceding inequalities. And this is a new linear program!

The Dual Program

$$\begin{gathered} \min 200y_1 + 300y_2 + 400y_3 \\ y_1 + y_3 \geq 1 \\ y_2 + y_3 \geq 6 \\ y_1, y_2, y_3 \geq 0 \end{gathered}$$

- By design, any feasible value of this dual LP is an upper bound on the original primal LP.
- So if we somehow find a pair of primal and dual feasible values that are equal, then they must both be optimal.
- Here is just such a pair:
  - Primal: $(x_1, x_2) = (100, 300)$;
  - Dual: $(y_1, y_2, y_3) = (0, 5, 1)$.
- They both have value 1900, and therefore they certify each other's optimality.

Matrix-Vector Form and Its Dual

Primal LP	Dual LP
$\max c^T\mathbf{x}$	$\min \mathbf{y}^T b$
$A\mathbf{x} \leq b$	$\mathbf{y}^T A \geq c^T$
$\mathbf{x} \geq 0$	$\mathbf{y} \geq 0$

Primal LP:

$$\begin{aligned} &\max\; c_1x_1 + \cdots + c_nx_n \\ &a_{i1}x_1 + \cdots + a_{in}x_n \leq b_i \quad \text{for } i \in I \\ &a_{i1}x_1 + \cdots + a_{in}x_n = b_i \quad \text{for } i \in E \\ &x_j \geq 0 \quad \text{for } j \in N \end{aligned}$$

Dual LP:

$$\begin{aligned} &\min\; b_1y_1 + \cdots + b_my_m \\ &a_{1j}y_1 + \cdots + a_{mj}y_m \geq c_j \quad \text{for } j \in N \\ &a_{1j}y_1 + \cdots + a_{mj}y_m = c_j \quad \text{for } j \notin N \\ &y_i \geq 0 \quad \text{for } i \in I \end{aligned}$$

Matrix-Vector Form and Its Dual

$$\max x_1 + 6x_2$$
$$x_1 \leq 200$$
$$x_2 \leq 300$$
$$x_1 + x_2 \leq 400$$
$$x_1, x_2, x_3 \geq 0$$

$$\min 200y_1 + 300y_2 + 400y_3$$
$$y_1 + y_3 \geq 1$$
$$y_2 + y_3 \geq 6$$
$$y_1, y_2, y_3 \geq 0$$

Matrix-Vector Form and Its Dual

Theorem (Duality)

If a linear program has a bounded optimum, then so does its dual, and the two optimum values coincide.

Shortest Path in LP

In the shortest path problem, we are given a weighted, directed graph $G = (V, E)$, with weight function $w : E \to R$ mapping edges to real-valued weights, a source vertex s, and destination vertex t. We wish to compute the weight of a shortest path from s to t.

Shortest Path in LP

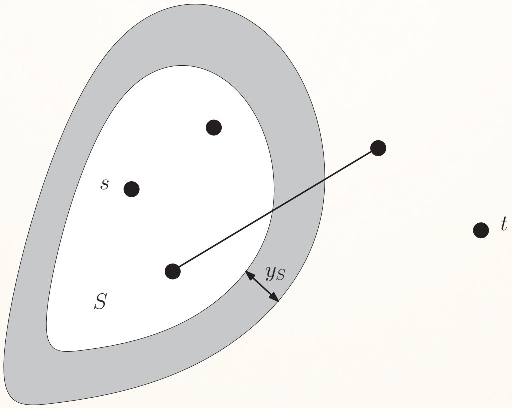
$$\min \sum_{e \in E} w_e x_e$$

$$\begin{aligned} \sum_{e \in \delta(S)} x_e &\geq 1 & S \in \mathcal{S} \\ x_e &\geq 0 & e \in E \end{aligned}$$

$$\max \sum_{S \in \mathcal{S}} y_S$$

$$\begin{aligned} \sum_{S \in \mathcal{S}, e \in \delta(S)} y_S &\leq w_e & e \in E \\ y_S &\geq 0 & S \in \mathcal{S} \end{aligned}$$

The Moat



Complementary Slackness

- The number of variables in the dual is equal to the number of constraints in the primal and the number of constraints in the dual is equal to the number of variables in the primal.
- An inequality constraint has slack if the slack variable is positive.
- The complementary slackness refers to a relationship between the slackness in a primal constraint and the slackness of the associated dual variable.

LP and Its Dual

$$\begin{aligned} \max x_1 + 6x_2 \\ x_1 \leq 200 \\ x_2 \leq 300 \\ x_1 + x_2 \leq 400 \\ x_1, x_2 \geq 0 \end{aligned}$$

$$\begin{aligned} \min 200y_1 + 300y_2 + 400y_3 \\ y_1 + y_3 \geq 1 \\ y_2 + y_3 \geq 6 \\ y_1, y_2, y_3 \geq 0 \end{aligned}$$

$$x_1 = 100, x_2 = 300$$

$$y_1 = 0, y_2 = 5, y_3 = 1$$

Complementary Slackness

Theorem

Assume LP problem (P) has a solution x^* and its dual problem (D) has a solution y^*.

1. *If $x_j^* > 0$, then the j-th constraint in (D) is binding.*
2. *If the j-th constraint in (D) is not binding, then $x_j^* = 0$.*
3. *If $y_i^* > 0$, then the i-th constraint in (P) is binding.*
4. *If the i-th constraint in (P) is not binding, then $y_i^* = 0$.*

Proof.

Assignment ! $\square$

The Simplex Algorithm

General Description

let v be any *vertex* of the feasible region
while there is a *neighbor* v' of v with better objective value:
    set $v = v'$

- Say there are n variables, $x_1, \ldots, x_n$.
- Any setting of the x_i's can be represented by an n-tuple of real numbers and plotted in n-dimensional space.
- A linear equation involving the x_i's defines a hyperplane in this same space $\mathbb{R}^n$, and the corresponding linear inequality defines a half-space, all points that are either precisely on the hyperplane or lie on one particular side of it.
- Finally, the feasible region of the linear program is specified by a set of inequalities and is therefore the intersection of the corresponding half-spaces, a convex polyhedron.

Vertices and Neighbors

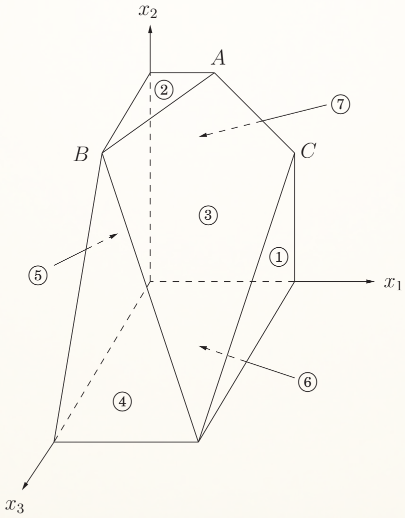
Definition
Each vertex is the unique point at which some subset of hyperplanes meet.

Pick a subset of the inequalities. If there is a unique point that satisfies them with equality, and this point happens to be feasible, then it is a vertex.

Each vertex is specified by a set of n inequalities (say there are n variables).

Definition
Two vertices are neighbors if they have $n - 1$ defining inequalities in common.

The Example



The Algorithm

On each iteration, simplex has two tasks:

1. Check whether the current vertex is optimal (and if so, halt).
2. Determine where to move next.

As we will see, both tasks are easy if the vertex happens to be at the origin. And if the vertex is elsewhere, we will transform the coordinate system to move it to the origin!

The Convenience for the Origin

- Suppose we have some generic LP:

$$\max c^T \mathbf{x}$$
$$A\mathbf{x} \leq b$$
$$\mathbf{x} \geq 0$$

where $\mathbf{x}$ is the vector of variables, $\mathbf{x} = (x_1, \ldots, x_n)$.

- Suppose the origin is feasible. Then it is certainly a vertex, since it is the unique point at which the n inequalities

$$\{x_1 \geq 0, \ldots, x_n \geq 0\}$$

are tight.

Task 1 in the Origin

Lemma

The origin is optimal if and only if all $c_i \leq 0$.

Proof

- If all $c_i \leq 0$, then considering the constraints $x \geq 0$, we can't hope for a better objective value.
- Conversely, if some $c_i > 0$, then the origin is not optimal, since we can increase the objective function by raising x_i.

Task 2 in the Origin

- We can move by increasing some x_i for which $c_i > 0$.
- Q: How much can we increase it?
- Until we hit some other constraint.
- That is, we release the tight constraint $x_i \geq 0$ and increase x_i until some other inequality, previously loose, now becomes tight.
- At that point, we again have exactly n tight inequalities, so we are at a new vertex.

An Example

$$\max 2x_1 + 5x_2$$

$$\begin{aligned} 2x_1 - x_2 &\leq 4 \\ x_1 + 2x_2 &\leq 9 \\ -x_1 + x_2 &\leq 3 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

What If Current Vertex u is Elsewhere?

- The trick is to transform u into the origin, by shifting the coordinate system from the usual $(x_1, \ldots, x_n)$ to the "local view" from u.
- These local coordinates consist of (appropriately scaled) distances $y_1, \ldots, y_n$ to the n hyperplanes (inequalities) that define and enclose u.
- Specifically, if one of these enclosing inequalities is $a_i \cdot x \leq b_i$, then the distance from a point x to that particular "wall" is

$$y_i = b_i - a_i \cdot x$$

- The n equations of this type, one per wall, define the y_i's as linear functions of the x_i's, and this relationship can be inverted to express the x_i's as a linear function of the y_i's.

An Example

$$\max 2x_1 + 5x_2$$

$$\begin{aligned} 2x_1 - x_2 &\leq 4 \\ x_1 + 2x_2 &\leq 9 \\ -x_1 + x_2 &\leq 3 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

$$\max 15 + 7y_1 - 5y_2$$

$$\begin{aligned} y_1 + y_2 &\leq 7 \\ 3y_1 - 2y_2 &\leq 3 \\ y_2 &\geq 0 \\ y_1 &\geq 0 \\ -y_1 + y_2 &\leq 3 \end{aligned}$$

An Example

$$\max 15 + 7y_1 - 5y_2$$

$$y_1 + y_2 \leq 7$$
$$3y_1 - 2y_2 \leq 3$$
$$y_2 \geq 0$$
$$y_1 \geq 0$$
$$-y_1 + y_2 \leq 3$$

$$\max 22 - 7/3z_1 - 1/3z_2$$

$$-1/3z_1 + 5/3z_2 \leq 6$$
$$z_1 \geq 0$$
$$z_2 \geq 0$$
$$1/3z_1 - 2/3z_2 \leq 1$$
$$1/3z_1 + 1/3z_2 \leq 4$$

Rewriting the LP

- Thus we can rewrite the entire LP in terms of the $\mathbf{y}$'s.
- This doesn't fundamentally change it (for instance, the optimal value stays the same), but expresses it in a different coordinate frame.
- The revised "local" LP has the following three properties:
  1. It includes the inequalities $\mathbf{y} \geq 0$, which are simply the transformed versions of the inequalities defining u.
  2. u itself is the origin in $\mathbf{y}$-space.
  3. The cost function becomes $\max c_u + \tilde{c}^T \mathbf{y}$, where c_u is the value of the objective function at u and $\tilde{c}$ is a transformed cost vector.

Loose End

The Starting Vertex

- In a general LP, the origin might not be feasible and thus not a vertex at all.
- However, it turns out that finding a starting vertex can be reduced to an LP and solved by simplex!
- Start with any linear program in standard form:

$$\min c^T \mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} = b \text{ and } x \geq 0 \quad .$$

- We first make sure that the right-hand sides of the equations are all nonnegative: if $b_i < 0$, just multiply both sides of the i-th equation by -1.

The Starting Vertex

- Then we create a new LP as follows:
  - Create m new artificial variables $z_1, \ldots, z_m \geq 0$, where m is the number of equations.
  - Add z_i to the left-hand side of the i-th equation.
  - Let the objective, to be minimized, be $z_1 + z_2 + \ldots + z_m$.

An Example

$$\begin{gathered} \min -x_1 - 6x_2 \\ x_1 + s_1 = 200 \\ x_2 + s_2 = 300 \\ x_1 + x_2 + x_3 = 400 \\ x_1, x_2, x_3 \geq 0 \end{gathered}$$

$$\begin{gathered} \min z_1 + z_2 + z_3 \\ x_1 + s_1 + z_1 = 200 \\ x_2 + s_2 + z_2 = 300 \\ x_1 + x_2 + x_3 + z_3 = 400 \\ x_1, x_2, x_3 \geq 0 \\ z_1, z_2, z_3 \geq 0 \end{gathered}$$

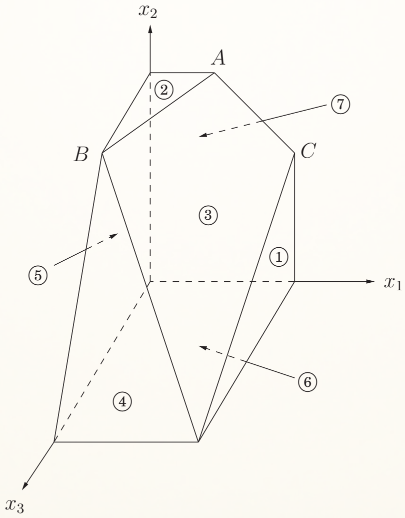
The Starting Vertex

- For this new LP, it's easy to come up with a starting vertex, namely, the one with $z_i = b_i$ for all i and all other variables zero.
- Therefore we can solve it by simplex, to obtain the optimum solution.
- There are two cases:
  1. If the optimum value of $z_1 + \ldots + z_m$ is zero, then all z_i's obtained by simplex are zero, and hence from the optimum vertex of the new LP we get a starting feasible vertex of the original LP, just by ignoring the z_i's.
  2. If the optimum objective turns out to be positive: We tried to minimize the sum of the z_i's, but simplex decided that it cannot be zero. But this means that the original linear program is infeasible: it needs some nonzero z_i's to become feasible.

Degeneracy

- A vertex is degenerate if it is the intersection of more than n faces of the polyhedron, say $n + 1$.
- Algebraically, it means that if we choose any one of n sets of $n + 1$ inequalities and solve the corresponding system of these linear equations in n unknowns, we'll get the same solution in all $n + 1$ cases.

An Example



Degeneracy

- This is a serious problem: simplex may return a suboptimal degenerate vertex simply because all its neighbors are identical to it and thus have no better objective.
- And if we modify simplex so that it detects degeneracy and continues to hop from vertex to vertex despite lack of any improvement in the cost, it may end up looping forever.

Degeneracy

- One way to fix this is by a perturbation:

  change each b_i by a tiny random amount to $b_i \pm \varepsilon_i$.

- This doesn't change the essence of the LP since the ε_i's are tiny, but it has the effect of differentiating between the solutions of the linear systems.

Unboundedness

- In some cases an LP is unbounded, in that its objective function can be made arbitrarily large (or small, if it's a minimization problem).
- If this is the case, simplex will discover it:
  - In exploring the neighborhood of a vertex, it will notice that taking out an inequality and adding another leads to an underdetermined system of equations that has an infinity of solutions.
  - And in fact (this is an easy test) the space of solutions contains a whole line across which the objective can become larger and larger, all the way to ∞.
- In this case simplex halts and complains.

An Example

$$\max x_1 + x_2$$
$$x_1 - x_2 \geq 0$$
$$x_1, x_2 \geq 0$$

The Running Time of Simplex

The Running Time of Simplex

- Q: What is the running time of simplex, for a generic linear program:
$$\max c^T\mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} \leq 0 \text{ and } \mathbf{x} \geq 0$$
where there are n variables and $\mathbb{A}$ contains m inequality constraints?
- It is an iterative algorithm that proceeds from vertex to vertex. Let u be the current vertex, i.e., unique point at which n inequality constraints are satisfied with equality.
- Each of its neighbors shares $n-1$ of these inequalities, so u can have at most $n \cdot m$ neighbors.

The Running Time of Simplex

- A naive way for an iteration:
  1. check each potential neighbor to see whether it really is a vertex of the polyhedron,
  2. determine its cost.
- Finding the cost is quick, just a dot product.
- Checking whether it is a true vertex involves: solve a system of n equations and check whether the result is feasible.
- By Gaussian elimination this takes $O(n^3)$ time, giving total $O(mn^4)$ per iteration.

The Running Time of Simplex

- A much better way: the mn^4 can be improved to mn.
- Recall the local view from vertex u. The per-iteration overhead of rewriting the LP in terms of the current local coordinates is just $O((m+n)n)$.
- The local view changes only slightly between iterations, in just one of its defining inequalities.

The Running Time of Simplex

- Next, to select the best neighbor, we recall that the (local view of) the objective function is of the form

$$\max c_u + \tilde{c} \cdot \mathbf{y}$$

  where c_u is the value of the objective function at u.

- This immediately identifies a promising direction to move: we pick any $\tilde{c}_i > 0$ (if there is none, then the current vertex is optimal and simplex halts).
- Since the rest of the LP has now been rewritten in terms of the $\mathbf{y}$-coordinates, it is easy to determine how much y_i can be increased before some other inequality is violated. (And if we can increase y_i indefinitely, we know the LP is unbounded.)

The Running Time of Simplex

- Q: How many iterations could there be?
- At most $\binom{m+n}{n}$, i.e., the number of vertices.
- It is exponential in n.
- And in fact, there are examples of LPs for which simplex does indeed take an exponential number of iterations.
- Simplex is an exponential-time algorithm.
- However, such exponential examples do not occur in practice, and it is this fact that makes simplex so valuable and so widely used.

Referred Materials

- Content of this lecture comes from Section 7.4 and 7.6 in [DPV07] and Section 7.3 in [WS11].