CS 473: Algorithms

Ruta Mehta

University of Illinois, Urbana-Champaign

Spring 2018

Simplex and LP Duality

Lecture 19

March 29, 2018

Some of the slides are courtesy Prof. Chekuri

Outline

Simplex: Intuition and Implementation Details

- Computing starting vertex: equivalent to solving an LP!

Infeasibility, Unboundedness, and Degeneracy.

Duality: Bounding the objective value through *weak-duality*

Strong Duality, Cone view.

Part I

Recall

Feasible Region and Convexity

Canonical Form

Given $A \in R^{n \times d}$, $b \in R^{n \times 1}$ and $c \in R^{1 \times d}$, find $x \in R^{d \times 1}$

$$\begin{aligned} \mathbf{max}: \quad & c \cdot x \\ s.t. \quad & Ax \leq b \end{aligned}$$

Feasible Region and Convexity

Canonical Form

Given $A \in R^{n \times d}$, $b \in R^{n \times 1}$ and $c \in R^{1 \times d}$, find $x \in R^{d \times 1}$

$$\begin{aligned} \max : \quad & c \cdot x \\ s.t. \quad & Ax \leq b \end{aligned}$$

1. Each linear constraint defines a **halfspace**, a convex set.
2. Feasible region, which is an intersection of halfspaces, is a convex **polyhedron**.
3. Optimal value attained at a vertex of the polyhedron.

Simplex Algorithm

Simplex: Vertex hoping algorithm

Moves from a vertex to its neighboring vertex

Simplex Algorithm

Simplex: Vertex hoping algorithm

Moves from a vertex to its neighboring vertex

Questions

- Which neighbor to move to?
- When to stop?
- How much time does it take?

Observations

For Simplex

Suppose we are at a non-optimal vertex $\hat{x}$ and optimal is x^*, then $c \cdot x^* > c \cdot \hat{x}$.

Observations

For Simplex

Suppose we are at a non-optimal vertex $\hat{x}$ and optimal is x^*, then $c \cdot x^* > c \cdot \hat{x}$.

How does $(c \cdot x)$ change as we move from $\hat{x}$ to x^* on the line joining the two?

Observations

For Simplex

Suppose we are at a non-optimal vertex $\hat{x}$ and optimal is x^*, then $c \cdot x^* > c \cdot \hat{x}$.

How does $(c \cdot x)$ change as we move from $\hat{x}$ to x^* on the line joining the two?

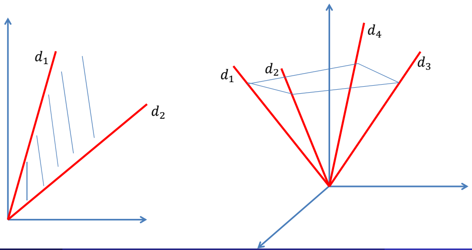
Strictly increases!

Cone

Definition

Given a set of vectors $D = \{d_1, \ldots, d_k\}$, the cone spanned by them is just their positive linear combinations, i.e.,

$$cone(D) = \{d \mid d = \sum_{i=1}^{k} \lambda_i d_i, \text{ where } \lambda_i \geq 0, \forall i\}$$

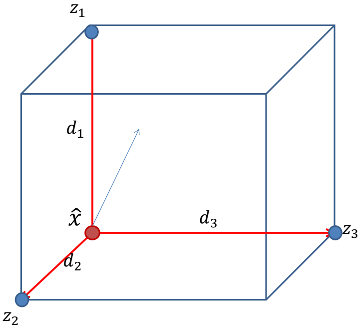


Cone at a Vertex

Let $z_1, \ldots, z_k$ be the neighboring vertices of $\hat{x}$. And let $d_i = z_i - \hat{x}$ be the direction from $\hat{x}$ to z_i.

Lemma

Any feasible direction of movement d from $\hat{x}$ is in the $cone(\{d_1, \ldots, d_k\})$.



Improving Direction Implies Improving Neighbor

Lemma

If $d \in cone(\{d_1, \ldots, d_k\})$ *and* $(c \cdot d) > 0$, *then there exists* d_i *such that* $(c \cdot d_i) > 0$.

Improving Direction Implies Improving Neighbor

Lemma

If $d \in cone(\{d_1, \ldots, d_k\})$ *and* $(c \cdot d) > 0$, *then there exists* d_i *such that* $(c \cdot d_i) > 0$.

Proof.

To the contrary suppose $(c \cdot d_i) \leq 0, \; \forall i \leq k$.
Since d is a positive linear combination of d_i's,

$$\begin{aligned} (c \cdot d) &= (c \cdot \textstyle\sum_{i=1}^k \lambda_i d_i) \\ &= \textstyle\sum_{i=1}^k \lambda_i (c \cdot d_i) \\ &\leq 0 \quad \text{A contradiction!} \end{aligned}$$

□

Improving Direction Implies Improving Neighbor

Lemma

If $d \in cone(\{d_1, \ldots, d_k\})$ *and* $(c \cdot d) > 0$, *then there exists* d_i *such that* $(c \cdot d_i) > 0$.

Proof.

To the contrary suppose $(c \cdot d_i) \leq 0, \ \forall i \leq k$.
Since d is a positive linear combination of d_i's,

$$\begin{aligned}(c \cdot d) &= (c \cdot \sum_{i=1}^{k} \lambda_i d_i) \\ &= \sum_{i=1}^{k} \lambda_i (c \cdot d_i) \\ &\leq 0 \quad \text{A contradiction!}\end{aligned}$$

□

Theorem

If vertex $\hat{x}$ *is not optimal then it has a neighbor where cost improves.*

How Many Neighbors a Vertex Has?

Geometric view...

$A \in R^{n \times d}$ $(n > d)$, $b \in R^n$, the constraints are: $Ax \leq b$

Geometry of faces

- r linearly independent hyperplanes forms $(d - r)$ dimensional face.

How Many Neighbors a Vertex Has?

Geometric view...

$A \in R^{n \times d}$ $(n > d)$, $b \in R^n$, the constraints are: $Ax \leq b$

Geometry of faces

- r linearly independent hyperplanes forms $(d - r)$ dimensional face.
- Vertex: 0-D face. formed by d L.I. hyperplanes.

How Many Neighbors a Vertex Has?

Geometric view...

$A \in R^{n \times d}$ $(n > d)$, $b \in R^n$, the constraints are: $Ax \leq b$

Geometry of faces

- r linearly independent hyperplanes forms $(d - r)$ dimensional face.
- Vertex: $\mathbf{0}$-D face. formed by d L.I. hyperplanes.
- Edge: $\mathbf{1}$-D face. formed by $(d - 1)$ L.I. hyperlanes.

How Many Neighbors a Vertex Has?

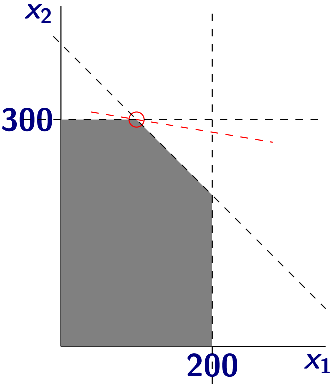
Geometric view...

$A \in R^{n \times d}$ $(n > d)$, $b \in R^n$, the constraints are: $Ax \leq b$

Geometry of faces

- r linearly independent hyperplanes forms $(d - r)$ dimensional face.
- Vertex: $\mathbf{0}$-D face. formed by d L.I. hyperplanes.
- Edge: $\mathbf{1}$-D face. formed by $(d-1)$ L.I. hyperlanes.

In 2-dimension $(d = 2)$



How Many Neighbors a Vertex Has?

Geometric view...

$A \in R^{n\times d}$ $(n > d)$, $b \in R^n$, the constraints are: $Ax \leq b$

Geometry of faces

- r linearly independent hyperplanes forms $(d - r)$ dimensional face.
- Vertex: 0-dimensional face. formed by d L.I. hyperplanes.
- Edge: 1-D face. formed by $(d - 1)$ L.I. hyperlanes.

In 3-dimension $(d = 3)$

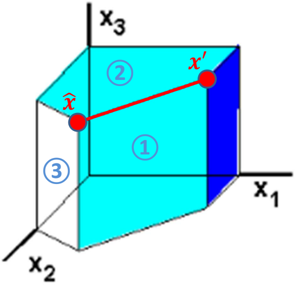


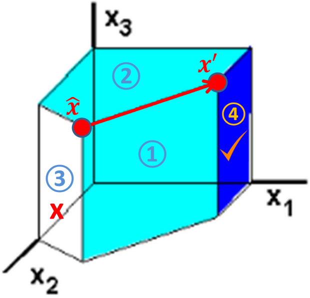
image source: webpage of Prof. Forbes W. Lewis

How Many Neighbors a Vertex Has?

Geometry view...

One neighbor per tight hyperplane. Therefore typically d.

- Suppose x' is a neighbor of $\hat{x}$, then on the edge joining the two $d-1$ constraints are tight.
- These $d-1$ are also tight at both $\hat{x}$ and x'.
- One more constraints, say i, is tight at $\hat{x}$. "Relaxing" i at $\hat{x}$ leads to x'.



Simplex Algorithm

Simplex: Vertex hoping algorithm

Moves from a vertex to its neighboring vertex

Questions + Answers

- Which neighbor to move to? One where objective value increases.

Simplex Algorithm

Simplex: Vertex hoping algorithm

Moves from a vertex to its neighboring vertex

Questions + Answers

- Which neighbor to move to? One where objective value increases.
- When to stop? When no neighbor with better objective value.

Simplex Algorithm

Simplex: Vertex hoping algorithm

Moves from a vertex to its neighboring vertex

Questions + Answers

- Which neighbor to move to? One where objective value increases.
- When to stop? When no neighbor with better objective value.
- How much time does it take? At most d neighbors to consider in each step.

Simplex in Higher Dimensions

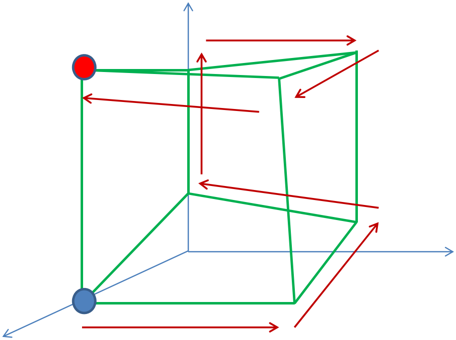
Simplex Algorithm

1. Start at a vertex of the polytope.
2. Compare value of objective function at each of the d "neighbors".
3. Move to neighbor that improves objective function, and repeat step 2.
4. If no improving neighbor, then stop.

Simplex is a greedy local-improvement algorithm! Works because a local optimum is also a global optimum — convexity of polyhedra.

Solving Linear Programming in Practice

1. Naïve implementation of Simplex algorithm can be very inefficient – Exponential number of steps!



Solving Linear Programming in Practice

1. Naïve implementation of Simplex algorithm can be very inefficient
   1. Choosing which neighbor to move to can significantly affect running time
   2. Very efficient Simplex-based algorithms exist
   3. Simplex algorithm takes exponential time in the worst case but works extremely well in practice with many improvements over the years
2. Non Simplex based methods like interior point methods work well for large problems.

Polynomial time Algorithm for Linear Programming

Major open problem for many years: is there a polynomial time algorithm for linear programming?

Polynomial time Algorithm for Linear Programming

Major open problem for many years: is there a polynomial time algorithm for linear programming?
Leonid Khachiyan in 1979 gave the first polynomial time algorithm using the Ellipsoid method.

1. major theoretical advance
2. highly impractical algorithm, not used at all in practice
3. routinely used in theoretical proofs.

Polynomial time Algorithm for Linear Programming

Major open problem for many years: is there a polynomial time algorithm for linear programming?

Leonid Khachiyan in 1979 gave the first polynomial time algorithm using the Ellipsoid method.

1. major theoretical advance
2. highly impractical algorithm, not used at all in practice
3. routinely used in theoretical proofs.

Narendra Karmarkar in 1984 developed another polynomial time algorithm, the interior point method.

1. very practical for some large problems and beats simplex
2. also revolutionized theory of interior point methods

Polynomial time Algorithm for Linear Programming

Major open problem for many years: is there a polynomial time algorithm for linear programming?

Leonid Khachiyan in 1979 gave the first polynomial time algorithm using the Ellipsoid method.

1. major theoretical advance
2. highly impractical algorithm, not used at all in practice
3. routinely used in theoretical proofs.

Narendra Karmarkar in 1984 developed another polynomial time algorithm, the interior point method.

1. very practical for some large problems and beats simplex
2. also revolutionized theory of interior point methods

Following interior point method success, Simplex has been improved enormously and is the method of choice.

Issues

1. Starting vertex
2. The linear program could be infeasible: No point satisfy the constraints.
3. The linear program could be unbounded: Polygon unbounded in the direction of the objective function.
4. More than d hyperplanes could be tight at a vertex, forming more than d neighbors.

Computing the Starting Vertex

Equivalent to solving another LP!

Find an x such that $Ax \leq b$.
If $b \geq \mathbf{0}$ then trivial!

Computing the Starting Vertex

Equivalent to solving another LP!

Find an x such that $Ax \leq b$.
If $b \geq \mathbf{0}$ then trivial! $x = \mathbf{0}$. Otherwise.

Computing the Starting Vertex

Equivalent to solving another LP!

Find an x such that $Ax \leq b$.
If $b \geq 0$ then trivial! $x = 0$. Otherwise.

$$\begin{aligned} \textbf{min :} \quad & s \\ \textit{s.t.} \quad & \sum_j a_{ij} x_j - s \leq b_i, \quad \forall i \\ & s \geq 0 \end{aligned}$$

Trivial feasible solution:

Computing the Starting Vertex

Equivalent to solving another LP!

Find an x such that $Ax \leq b$.
If $b \geq \mathbf{0}$ then trivial! $x = \mathbf{0}$. Otherwise.

$$\begin{aligned} \textbf{min :} \quad & s \\ s.t. \quad & \textstyle\sum_j a_{ij} x_j - s \leq b_i, \quad \forall i \\ & s \geq 0 \end{aligned}$$

Trivial feasible solution: $x = \mathbf{0}$, $s = |\min_i b_i|$.

Computing the Starting Vertex

Equivalent to solving another LP!

Find an x such that $Ax \leq b$.
If $b \geq \mathbf{0}$ then trivial! $x = \mathbf{0}$. Otherwise.

$$\begin{aligned} \textbf{min}: \quad & s \\ s.t. \quad & \sum_j a_{ij}x_j - s \leq b_i, \quad \forall i \\ & s \geq 0 \end{aligned}$$

Trivial feasible solution: $x = \mathbf{0}$, $s = |\min_i b_i|$.

If $Ax \leq b$ feasible then optimal value of the above LP is $s = \mathbf{0}$.

Computing the Starting Vertex

Equivalent to solving another LP!

Find an x such that $Ax \leq b$.
If $b \geq 0$ then trivial! $x = 0$. Otherwise.

$$\begin{aligned} \textbf{min}: \quad & s \\ s.t. \quad & \textstyle\sum_j a_{ij}x_j - s \leq b_i, \quad \forall i \\ & s \geq 0 \end{aligned}$$

Trivial feasible solution: $x = 0$, $s = |\min_i b_i|$.

If $Ax \leq b$ feasible then optimal value of the above LP is $s = 0$.

Checks Feasibility!

Unboundedness: Example

$$\begin{aligned} \text{maximize} \quad & x_2 \\ x_1 + x_2 & \geq 2 \\ x_1, x_2 & \geq 0 \end{aligned}$$

Unboundedness depends on both constraints and the objective function.

Unboundedness: Example

$$\begin{aligned} \text{maximize} \quad & x_2 \\ x_1 + x_2 &\geq 2 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Unboundedness depends on both constraints and the objective function.

If unbounded in the direction of objective function, then the pivoting step in the simplex will detect it.

Degeneracy and Cycling

More than d constraints are tight at vertex $\hat{x}$. Say $d + 1$.

Suppose, we pick first d to form $\hat{A}$ such that $\hat{A}\hat{x} = \hat{b}$, and compute directions $d_1, \ldots, d_d$.

Degeneracy and Cycling

More than d constraints are tight at vertex $\hat{x}$. Say $d + 1$.

Suppose, we pick first d to form $\hat{A}$ such that $\hat{A}\hat{x} = \hat{b}$, and compute directions $d_1, \ldots, d_d$.

Then NextVertex($\hat{x}, d_i$) will encounter $(d + 1)^{th}$ constraint tight at $\hat{x}$ and return the same vertex. Hence we are back to $\hat{x}$!

Degeneracy and Cycling

More than d constraints are tight at vertex $\hat{x}$. Say $d+1$.

Suppose, we pick first d to form $\hat{A}$ such that $\hat{A}\hat{x} = \hat{b}$, and compute directions $d_1, \ldots, d_d$.

Then NextVertex($\hat{x}, d_i$) will encounter $(d+1)^{th}$ constraint tight at $\hat{x}$ and return the same vertex. Hence we are back to $\hat{x}$!

Same phenomenon will repeat!

Degeneracy and Cycling

More than d constraints are tight at vertex $\hat{x}$. Say $d+1$.

Suppose, we pick first d to form $\hat{A}$ such that $\hat{A}\hat{x} = \hat{b}$, and compute directions $d_1, \ldots, d_d$.

Then NextVertex($\hat{x}, d_i$) will encounter $(d+1)^{th}$ constraint tight at $\hat{x}$ and return the same vertex. Hence we are back to $\hat{x}$!

Same phenomenon will repeat!

This can be avoided by adding small random perturbation to b_is.

Feasible Solutions and Lower Bounds

Consider the program

$$\begin{array}{lrrl} \text{maximize} & 4x_1 + & 2x_2 & \\ \text{subject to} & x_1 + & 3x_2 & \leq 5 \\ & 2x_1 - & 4x_2 & \leq 10 \\ & x_1 + & x_2 & \leq 7 \\ & x_1 & & \leq 5 \end{array}$$

Feasible Solutions and Lower Bounds

Consider the program

$$
\begin{array}{lrrl}
\text{maximize} & 4x_1 + & 2x_2 & \\
\text{subject to} & x_1 + & 3x_2 & \leq 5 \\
 & 2x_1 - & 4x_2 & \leq 10 \\
 & x_1 + & x_2 & \leq 7 \\
 & x_1 & & \leq 5
\end{array}
$$

1. $(0, 1)$ satisfies all the constraints and gives value 2 for the objective function.

Feasible Solutions and Lower Bounds

Consider the program

$$\begin{array}{lrrl} \text{maximize} & 4x_1 + & 2x_2 & \\ \text{subject to} & x_1 + & 3x_2 & \leq 5 \\ & 2x_1 - & 4x_2 & \leq 10 \\ & x_1 + & x_2 & \leq 7 \\ & x_1 & & \leq 5 \end{array}$$

1. $(0, 1)$ satisfies all the constraints and gives value 2 for the objective function.
2. Thus, optimal value σ^* is at least 4.

Feasible Solutions and Lower Bounds

Consider the program

$$\begin{array}{lrrl} \text{maximize} & 4x_1 + & 2x_2 & \\ \text{subject to} & x_1 + & 3x_2 & \leq 5 \\ & 2x_1 - & 4x_2 & \leq 10 \\ & x_1 + & x_2 & \leq 7 \\ & x_1 & & \leq 5 \end{array}$$

1. $(0, 1)$ satisfies all the constraints and gives value 2 for the objective function.
2. Thus, optimal value σ^* is at least 4.
3. $(2, 0)$ also feasible, and gives a better bound of 8.

Feasible Solutions and Lower Bounds

Consider the program

$$\begin{array}{llllll} \text{maximize} & 4x_1 + & 2x_2 & & \\ \text{subject to} & x_1 + & 3x_2 & \leq & 5 \\ & 2x_1 - & 4x_2 & \leq & 10 \\ & x_1 + & x_2 & \leq & 7 \\ & x_1 & & \leq & 5 \end{array}$$

1. $(0, 1)$ satisfies all the constraints and gives value 2 for the objective function.
2. Thus, optimal value σ^* is at least 4.
3. $(2, 0)$ also feasible, and gives a better bound of 8.
4. How good is 8 when compared with σ^*?

Obtaining Upper Bounds

$$\begin{array}{lrrl}
\text{maximize} & 4x_1 + & 2x_2 & \\
\text{subject to} & x_1 + & 3x_2 & \leq 5 \\
& 2x_1 - & 4x_2 & \leq 10 \\
& x_1 + & x_2 & \leq 7 \\
& x_1 & & \leq 5
\end{array}$$

1. Let us multiply the first constraint by **2** and the and add it to second constraint

Obtaining Upper Bounds

$$\begin{array}{lrrl} \text{maximize} & 4x_1 + & 2x_2 & \\ \text{subject to} & x_1 + & 3x_2 & \leq 5 \\ & 2x_1 - & 4x_2 & \leq 10 \\ & x_1 + & x_2 & \leq 7 \\ & x_1 & & \leq 5 \end{array}$$

1. Let us multiply the first constraint by 2 and the and add it to second constraint

$$\begin{array}{rrrl} 2(& x_1 + & 3x_2 &) \leq 2(5) \\ +1(& 2x_1 - & 4x_2 &) \leq 1(10) \\ \hline & 4x_1 + & 2x_2 & \leq 20 \end{array}$$

Obtaining Upper Bounds

$$\begin{array}{lrrl} \text{maximize} & 4x_1+ & 2x_2 & \\ \text{subject to} & x_1+ & 3x_2 & \leq 5 \\ & 2x_1- & 4x_2 & \leq 10 \\ & x_1+ & x_2 & \leq 7 \\ & x_1 & & \leq 5 \end{array}$$

1. Let us multiply the first constraint by 2 and the and add it to second constraint

$$\begin{array}{rrrl} 2(& x_1+ & 3x_2 &) \leq 2(5) \\ +1(& 2x_1- & 4x_2 &) \leq 1(10) \\ \hline & 4x_1+ & 2x_2 & \leq 20 \end{array}$$

2. Thus, 20 is an upper bound on the optimum value!

Generalizing . . .

1. Multiply first equation by y_1, second by y_2, third by y_3 and fourth by y_4 ($y_1, y_2, y_3, y_4 \geq 0$) and add

$$
\begin{array}{rrrl}
y_1(& x_1 + & 3x_2 &) \leq y_1(5) \\
+y_2(& 2x_1 - & 4x_2 &) \leq y_2(10) \\
+y_3(& x_1 + & x_2 &) \leq y_3(7) \\
+y_4(& x_1 & &) \leq y_4(5) \\
\hline
\multicolumn{4}{l}{(y_1 + 2y_2 + y_3 + y_4)x_1 + (3y_1 - 4y_2 + y_3)x_2 \leq \ldots}
\end{array}
$$

Generalizing . . .

1. Multiply first equation by y_1, second by y_2, third by y_3 and fourth by y_4 ($y_1, y_2, y_3, y_4 \geq 0$) and add

$$
\begin{array}{rrrl}
y_1(& x_1 + & 3x_2 &) \leq y_1(5) \\
+y_2(& 2x_1 - & 4x_2 &) \leq y_2(10) \\
+y_3(& x_1 + & x_2 &) \leq y_3(7) \\
+y_4(& x_1 & &) \leq y_4(5) \\
\hline
\multicolumn{4}{l}{(y_1 + 2y_2 + y_3 + y_4)x_1 + (3y_1 - 4y_2 + y_3)x_2 \leq \ldots}
\end{array}
$$

2. $5y_1 + 10y_2 + 7y_3 + 5y_4$ is an upper bound,

Generalizing . . .

1. Multiply first equation by y_1, second by y_2, third by y_3 and fourth by y_4 $(y_1, y_2, y_3, y_4 \geq 0)$ and add

$$
\begin{array}{rrrl}
y_1(& x_1 + & 3x_2 &) \leq y_1(5) \\
+y_2(& 2x_1 - & 4x_2 &) \leq y_2(10) \\
+y_3(& x_1 + & x_2 &) \leq y_3(7) \\
+y_4(& x_1 & &) \leq y_4(5) \\
\hline
\multicolumn{4}{c}{(y_1 + 2y_2 + y_3 + y_4)x_1 + (3y_1 - 4y_2 + y_3)x_2 \leq \ldots}
\end{array}
$$

2. $5y_1 + 10y_2 + 7y_3 + 5y_4$ is an upper bound, provided coefficients of x_i are same as in the objective function $(4x_1 + 2x_2)$,

$$y_1 + 2y_2 + y_3 + y_4 = 4 \quad 3y_1 - 4y_2 + y_3 = 2$$

Generalizing . . .

1. Multiply first equation by y_1, second by y_2, third by y_3 and fourth by y_4 $(y_1, y_2, y_3, y_4 \geq 0)$ and add

$$\begin{array}{rrrl} y_1(& x_1 + & 3x_2 &) \leq y_1(5) \\ +y_2(& 2x_1 - & 4x_2 &) \leq y_2(10) \\ +y_3(& x_1 + & x_2 &) \leq y_3(7) \\ +y_4(& x_1 & &) \leq y_4(5) \\ \hline \end{array}$$

$$(y_1 + 2y_2 + y_3 + y_4)x_1 + (3y_1 - 4y_2 + y_3)x_2 \leq \ldots$$

2. $5y_1 + 10y_2 + 7y_3 + 5y_4$ is an upper bound, provided coefficients of x_i are same as in the objective function $(4x_1 + 2x_2)$,

$$y_1 + 2y_2 + y_3 + y_4 = 4 \quad 3y_1 - 4y_2 + y_3 = 2$$

3. Subject to these constrains, the best upper bound is $\min : 5y_1 + 10y_2 + 7y_3 + 5y_4$!

Dual LP: Example

Thus, the optimum value of program

$$\begin{aligned} \text{maximize} \quad & 4x_1 + 2x_2 \\ \text{subject to} \quad & x_1 + 3x_2 \leq 5 \\ & 2x_1 - 4x_2 \leq 10 \\ & x_1 + x_2 \leq 7 \\ & x_1 \leq 5 \end{aligned}$$

is upper bounded by the optimal value of the program

$$\begin{aligned} \text{minimize} \quad & 5y_1 + 10y_2 + 7y_3 + 5y_4 \\ \text{subject to} \quad & y_1 + 2y_2 + y_3 + y_4 = 4 \\ & 3y_1 - 4y_2 + y_3 = 2 \\ & y_1, y_2 \geq 0 \end{aligned}$$

Dual Linear Program

Given a linear program Π in canonical form

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^{d} c_j x_j \\ \text{subject to} & \sum_{j=1}^{d} a_{ij} x_j \le b_i \quad i = 1, 2, \ldots n \end{array}$$

the dual $\mathbf{Dual}(\Pi)$ is given by

$$\begin{array}{lll} \text{minimize} & \sum_{i=1}^{n} b_i y_i & \\ \text{subject to} & \sum_{i=1}^{n} y_i a_{ij} = c_j & j = 1, 2, \ldots d \\ & y_i \ge 0 & i = 1, 2, \ldots n \end{array}$$

Dual Linear Program

Given a linear program Π in canonical form

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^{d} c_j x_j \\ \text{subject to} & \sum_{j=1}^{d} a_{ij} x_j \leq b_i \quad i = 1, 2, \ldots n \end{array}$$

the dual $\mathrm{Dual}(\Pi)$ is given by

$$\begin{array}{lll} \text{minimize} & \sum_{i=1}^{n} b_i y_i & \\ \text{subject to} & \sum_{i=1}^{n} y_i a_{ij} = c_j & j = 1, 2, \ldots d \\ & y_i \geq 0 & i = 1, 2, \ldots n \end{array}$$

Proposition

$\mathrm{Dual}(\mathrm{Dual}(\Pi))$ *is equivalent to* Π

Dual Linear Program

Succinct representation..

Given a $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}^d$, linear program Π

$$\begin{aligned} \text{maximize} \quad & c \cdot x \\ \text{subject to} \quad & Ax \leq b \end{aligned}$$

the dual $\mathrm{Dual}(\Pi)$ is given by

$$\begin{aligned} \text{minimize} \quad & y \cdot b \\ \text{subject to} \quad & yA = c \\ & y \geq 0 \end{aligned}$$

Proposition

$\mathrm{Dual}(\mathrm{Dual}(\Pi))$ *is equivalent to* Π

Duality Theorem

Theorem (Weak Duality)

If x *is a feasible solution to* Π *and* y *is a feasible solution to* $\mathrm{Dual}(\Pi)$ *then* $c \cdot x \leq y \cdot b$.

Duality Theorem

Theorem (Weak Duality)

If x *is a feasible solution to* Π *and* y *is a feasible solution to* $\mathbf{Dual}(\Pi)$ *then* $c \cdot x \leq y \cdot b$.

Theorem (Strong Duality)

If x^* *is an optimal solution to* Π *and* y^* *is an optimal solution to* $\mathbf{Dual}(\Pi)$ *then* $c \cdot x^* = y^* \cdot b$.

Many applications! Maxflow-Mincut theorem can be deduced from duality.

Weak Duality

Theorem (Weak Duality)

If $\mathbf{x}$ *is a feasible solution to* $\mathbf{\Pi}$ *and* $\mathbf{y}$ *is a feasible solution to* $\mathbf{Dual(\Pi)}$ *then* $\mathbf{c} \cdot \mathbf{x} \leq \mathbf{y} \cdot \mathbf{b}$*.*

We already saw the proof by the way we derived it but we will do it again formally.

Proof.

Since $\mathbf{y'}$ is feasible in $\mathbf{Dual(\Pi)}$: $\mathbf{y'A = c}$

Weak Duality

Theorem (Weak Duality)

If x *is a feasible solution to* Π *and* y *is a feasible solution to* $\mathrm{Dual}(\Pi)$ *then* $c \cdot x \leq y \cdot b$.

We already saw the proof by the way we derived it but we will do it again formally.

Proof.

Since y' is feasible in $\mathrm{Dual}(\Pi)$: $y'A = c$

Therefore $c \cdot x' = y'Ax'$

Weak Duality

Theorem (Weak Duality)

If x is a feasible solution to Π and y is a feasible solution to $\mathrm{Dual}(\Pi)$ then $c \cdot x \leq y \cdot b$.

We already saw the proof by the way we derived it but we will do it again formally.

Proof.

Since y' is feasible in $\mathrm{Dual}(\Pi)$: $y'A = c$

Therefore $c \cdot x' = y'Ax'$

Since x' is feasible in Π, $Ax' \leq b$ and hence,

$$c \cdot x' = y'Ax' \leq y' \cdot b$$

Strong Duality and Complementary Slackness

$$\begin{array}{ll} \textbf{maximize}: & c \cdot x \\ \text{subject to} & Ax \leq b \end{array} \quad \xrightarrow{\textit{Dual}} \quad \begin{array}{ll} \textbf{minimize}: & y \cdot b \\ \text{subject to} & yA = c \\ & y \geq 0 \end{array}$$

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \Rightarrow (Ax)_i = b_i$$

Strong Duality and Complementary Slackness

$$\begin{array}{ll} \textbf{maximize}: & c \cdot x \\ \text{subject to} & Ax \leq b \end{array} \quad \xrightarrow{\textit{Dual}} \quad \begin{array}{ll} \textbf{minimize}: & y \cdot b \\ \text{subject to} & yA = c \\ & y \geq 0 \end{array}$$

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\text{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \;\Rightarrow\; (Ax)_i = b_i$$

Geometric Interpretation: c is in the cone of the normal vectors of the tight hyperplanes at x.

Strong Duality and Complementary Slackness

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \;\Rightarrow\; (Ax)_i = b_i$$

Theorem

(x^*, y^*) satisfies complementary Slackness if and only if strong duality holds, i.e., $c \cdot x^* = y^* \cdot b$.

Proof.

$$\begin{aligned} c \cdot x^* &= (y^* A) \cdot x^* \\ &= y^* \cdot (Ax^*) \end{aligned}$$

$(\Rightarrow)$

Strong Duality and Complementary Slackness

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \;\Rightarrow\; (Ax)_i = b_i$$

Theorem

(x^*, y^*) satisfies complementary Slackness if and only if strong duality holds, i.e., $c \cdot x^* = y^* \cdot b$.

Proof.

$(\Rightarrow)$

$$\begin{aligned} c \cdot x^* &= (y^* A) \cdot x^* \\ &= y^* \cdot (Ax^*) \\ &= \sum_{i=1}^{n} y_i^* (Ax^*)_i \\ &= \sum_{i:y_i>0} y_i^* (Ax^*)_i \end{aligned}$$

Strong Duality and Complementary Slackness

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \;\Rightarrow\; (Ax)_i = b_i$$

Theorem

(x^*, y^*) satisfies complementary Slackness if and only if strong duality holds, i.e., $c \cdot x^* = y^* \cdot b$.

Proof.

$(\Rightarrow)$

$$\begin{aligned} c \cdot x^* &= (y^* A) \cdot x^* \\ &= y^* \cdot (Ax^*) \\ &= \sum_{i=1}^{n} y_i^* (Ax^*)_i \\ &= \sum_{i:y_i>0} y_i^* (Ax^*)_i \\ &= \sum_i y_i^* b_i = y^* \cdot b \end{aligned}$$

□

Strong Duality and Complementary Slackness

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \;\Rightarrow\; (Ax)_i = b_i$$

Theorem

(x^*, y^*) *satisfies complementary Slackness if and only if strong duality holds, i.e.,* $c \cdot x^* = y^* \cdot b$.

Proof.

$(\Leftarrow)$

Strong Duality and Complementary Slackness

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1..n, \quad y_i > 0 \Rightarrow (Ax)_i = b_i$$

Theorem

(x^*, y^*) *satisfies complementary Slackness if and only if strong duality holds, i.e.,* $c \cdot x^* = y^* \cdot b$.

Proof.

$(\Leftarrow)$ **Exercise** □

Duality for another canonical form

$$
\begin{array}{llllll}
\text{maximize} & 4x_1 + & x_2 + & 3x_3 & & \\
\text{subject to} & x_1 + & 4x_2 & & \leq & 2 \\
 & 2x_1 - & x_2 + & x_3 & \leq & 4 \\
 & & & x_1, x_2, x_3 & \geq & 0
\end{array}
$$

Duality for another canonical form

$$\begin{array}{llllll} \text{maximize} & 4x_1 + & x_2 + & 3x_3 & & \\ \text{subject to} & x_1 + & 4x_2 & & \leq & 2 \\ & 2x_1 - & x_2 + & x_3 & \leq & 4 \\ & & & x_1, x_2, x_3 & \geq & 0 \end{array}$$

Choose non-negative y_1, y_2 and multiply inequalities

$$\begin{array}{llllll} \text{maximize} & 4x_1 + & x_2 + & 3x_3 & & \\ \text{subject to} & y_1(x_1 + & 4x_2 &) & \leq & 2y_1 \\ & y_2(2x_1 - & x_2 + & x_3) & \leq & 4y_2 \\ & & & x_1, x_2, x_3 & \geq & 0 \end{array}$$

Duality for another canonical form

Choose non-negative y_1, y_2 and multiply inequalities

$$\begin{array}{llllll} \text{maximize} & 4x_1 + & x_2 + & 3x_3 & & \\ \text{subject to} & y_1(x_1 + & 4x_2 &) & \leq & 2y_1 \\ & y_2(2x_1 - & x_2 + & x_3) & \leq & 4y_2 \\ & & & x_1, x_2, x_3 & \geq & 0 \end{array}$$

Adding the inequalities we get an inequality below that is valid for any feasible x and any non-negative y:

$$(y_1 + 2y_2)x_1 + (4y_1 - y_2)x_2 + y_2x_3 \leq 2y_1 + 4y_2$$

Duality for another canonical form

Choose non-negative y_1, y_2 and multiply inequalities

$$\begin{array}{llrrrl} \text{maximize} & & 4x_1+ & x_2+ & 3x_3 & \\ \text{subject to} & y_1(& x_1+ & 4x_2 &) & \leq 2y_1 \\ & y_2(& 2x_1- & x_2+ & x_3) & \leq 4y_2 \\ & & & & x_1, x_2, x_3 & \geq 0 \end{array}$$

Adding the inequalities we get an inequality below that is valid for any feasible x and any non-negative y:

$$(y_1 + 2y_2)x_1 + (4y_1 - y_2)x_2 + y_2 x_3 \leq 2y_1 + 4y_2$$

Suppose we choose y_1, y_2 such that
$y_1 + 2y_2 \geq 4$ and $4y_2 - y_2 \geq 1$ and $y_2 \geq 3$
Then, since $x_1, x_2, x_3 \geq 0$, we have $4x_1 + x_2 + 3x_3 \leq 2y_1 + 4y_2$

Duality for another canonical form

$$\begin{array}{lrrrl} \text{maximize} & 4x_1 + & x_2 + & 3x_3 & \\ \text{subject to} & x_1 + & 4x_2 & & \leq 2 \\ & 2x_1 - & x_2 + & x_3 & \leq 4 \\ & & & x_1, x_2, x_3 & \geq 0 \end{array}$$

is upper bounded by

$$\begin{array}{lrrl} \text{minimize} & 2y_1 + & 4y_2 & \\ \text{subject to} & y_1 + & 2y_2 & \geq 4 \\ & 4y_1 - & y_2 & \geq 1 \\ & y_2 & & \geq 3 \\ & & y_1, y_2 & \geq 0 \end{array}$$

Duality for another canonical form

Compactly, for the primal LP Π

$$\begin{array}{ll} \max & c \cdot x \\ \text{subject to} & Ax \leq b, \ x \geq \mathbf{0} \end{array}$$

the dual LP is $\mathrm{Dual}(\Pi)$

$$\begin{array}{ll} \min & y \cdot b \\ \text{subject to} & yA \geq c, \ y \geq \mathbf{0} \end{array}$$

Duality for another canonical form

Compactly, for the primal LP Π

$$\begin{array}{ll} \max & c \cdot x \\ \text{subject to} & Ax \leq b, \; x \geq 0 \end{array}$$

the dual LP is $\mathrm{Dual}(\Pi)$

$$\begin{array}{ll} \min & y \cdot b \\ \text{subject to} & yA \geq c, \; y \geq 0 \end{array}$$

Definition (Complementary Slackness)

x feasible in Π and y feasible in $\mathrm{Dual}(\Pi)$, s.t.,

$$\forall i = 1, \ldots, n, \quad y_i > 0 \;\Rightarrow\; (Ax)_i = b_i$$
$$\forall j = 1, \ldots, d, \quad x_j > 0 \;\Rightarrow\; (yA)_j = c_j$$

In General...

from Jeff's notes

Primal	Dual
$\max c \cdot x$	$\min y \cdot b$
$\sum_j a_{ij}x_j \le b_i$	$y_i \ge 0$
$\sum_j a_{ij}x_j \ge b_i$	$y_i \le 0$
$\sum_j a_{ij}x_j = b_i$	–
$x_j \ge 0$	$\sum_i y_i a_{ij} \ge c_j$
$x_j \le 0$	$\sum_i y_i a_{ij} \le c_j$
–	$\sum_i y_i a_{ij} = c_j$
$x_j = 0$	–

Primal	Dual
$\min c \cdot x$	$\max y \cdot b$
$\sum_j a_{ij}x_j \le b_i$	$y_i \le 0$
$\sum_j a_{ij}x_j \ge b_i$	$y_i \ge 0$
$\sum_j a_{ij}x_j = b_i$	–
$x_j \le 0$	$\sum_i y_i a_{ij} \ge c_j$
$x_j \ge 0$	$\sum_i y_i a_{ij} \le c_j$
–	$\sum_i y_i a_{ij} = c_j$
$x_j = 0$	–

Figure H.4. Constructing the dual of an arbitrary linear program.

Some Useful Duality Properties

Assume primal LP is a maximization LP.

- For a given LP, Dual is another LP. The variables in the dual correspond to "tight" primal constraints and vice-versa.

Some Useful Duality Properties

Assume primal LP is a maximization LP.

- For a given LP, Dual is another LP. The variables in the dual correspond to "tight" primal constraints and vice-versa.
- Dual of the dual LP give us back the primal LP.

Some Useful Duality Properties

Assume primal LP is a maximization LP.

- For a given LP, Dual is another LP. The variables in the dual correspond to "tight" primal constraints and vice-versa.
- Dual of the dual LP give us back the primal LP.
- Weak and strong duality theorems.

Some Useful Duality Properties

Assume primal LP is a maximization LP.

- For a given LP, Dual is another LP. The variables in the dual correspond to "tight" primal constraints and vice-versa.
- Dual of the dual LP give us back the primal LP.
- Weak and strong duality theorems.
- If primal is unbounded (objective achieves infinity) then dual LP is infeasible. Why? If dual LP had a feasible solution it would upper bound the primal LP which is not possible.

Some Useful Duality Properties

Assume primal LP is a maximization LP.

- For a given LP, Dual is another LP. The variables in the dual correspond to "tight" primal constraints and vice-versa.
- Dual of the dual LP give us back the primal LP.
- Weak and strong duality theorems.
- If primal is unbounded (objective achieves infinity) then dual LP is infeasible. Why? If dual LP had a feasible solution it would upper bound the primal LP which is not possible.
- If primal is infeasible then dual LP is unbounded.

Some Useful Duality Properties

Assume primal LP is a maximization LP.

- For a given LP, Dual is another LP. The variables in the dual correspond to "tight" primal constraints and vice-versa.
- Dual of the dual LP give us back the primal LP.
- Weak and strong duality theorems.
- If primal is unbounded (objective achieves infinity) then dual LP is infeasible. Why? If dual LP had a feasible solution it would upper bound the primal LP which is not possible.
- If primal is infeasible then dual LP is unbounded.
- Primal and dual optimum solutions satisfy complementary slackness conditions (discussed soon).

Part II

Examples of Duality

Max matching in bipartite graph as LP

Input: $\mathrm{G} = (V = L \cup R, \mathrm{E})$

$$
\begin{aligned}
\textbf{max} \quad & \sum_{uv \in \mathrm{E}} x_{uv} & \\
s.t. \quad & \sum_{uv \in \mathrm{E}} x_{uv} \leq 1 & \forall v \in V. \\
& x_{uv} \geq 0 & \forall uv \in \mathrm{E}
\end{aligned}
$$

When one writes combinatorial problems as LPs one is writing a single formulation in an abstract way that applies to all instances. In the above, for each fixed graph G one gets a fixed LP and hence the above is sometimes called a “formulation”.

Max matching in bipartite graph as LP

Input: $G = (V = L \cup R, E)$

$$
\begin{aligned}
\max \quad & \sum_{uv \in E} x_{uv} & \\
s.t. \quad & \sum_{uv \in E} x_{uv} \leq 1 & \forall v \in V. \\
& x_{uv} \geq 0 & \forall uv \in E
\end{aligned}
$$

Dual LP has one variable y_v for each vertex $v \in V$.

$$
\begin{aligned}
\min \quad & \sum_{v \in V} y_v & \\
s.t. \quad & y_u + y_v \geq 1 & \forall uv \in E \\
& y_v \geq 0 & \forall v \in V
\end{aligned}
$$

Network flow

s-t flow in directed graph $G = (V, E)$ with capacities c. Assume for simplicity that no incoming edges into s.

$$\begin{aligned}
\max \quad & \sum_{(s,v)\in E} x(s,v) \\
& \sum_{(u,v)\in E} x(u,v) - \sum_{(v,w)\in E} x(v,w) = 0 \quad \forall v \in V \setminus \{s,t\} \\
& x(u,v) \leq c(u,v) \quad \forall (u,v) \in E \\
& x(u,v) \geq 0 \quad \forall (u,v) \in E.
\end{aligned}$$

Dual of Network Flow