

A sunset scene over a body of water with palm trees in the foreground. The sun is low on the horizon, casting a warm glow over the sky and water. The palm trees are silhouetted against the darker background.

Lecture 1

Linear Optimization

Duality, Simplex Methods

Robert J. Vanderbei

April 14, 2012

Machine Learning Summer School
La Palma

<http://www.princeton.edu/~rvdb>

Linear Programming

Standard form:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

Dictionary. Add slack variables w and name the objective function:

$$\begin{array}{ll} \zeta & = c^T x \\ w & = b - Ax \\ & x, w \geq 0. \end{array}$$

Dictionary Solution. Set $x = 0$ and read off values of w 's: $w = b$.

Feasibility. If $b \geq 0$, then $w \geq 0$ and so dictionary solution is feasible.

Optimality. If $b \geq 0$ and $c \leq 0$, then the dictionary solution is optimal.

Primal Simplex Method (used when feasible)

Dictionary:

$$\begin{aligned}\zeta &= c^T x \\ w &= b - Ax\end{aligned}$$

$$x, w \geq 0.$$

Entering Variable. Choose an index j for which $c_j > 0$. Variable x_j is the entering variable.

Leaving Variable. Let x_j increase while holding all other x_k 's at zero. Figure out which slack variable hits zero first. Let's say it's w_i .

Pivot. Rearrange equations so that entering variable x_j is on left and leaving variable w_i is on right to get a new *dictionary*.

$$\begin{aligned}\zeta &= \zeta^* + \tilde{c}^T x_{\mathcal{N}} \\ x_{\mathcal{B}} &= \tilde{b} - \tilde{A}x_{\mathcal{N}}\end{aligned}$$

$$x, w \geq 0.$$

Here $x_{\mathcal{B}}$ denotes the vector of slack variables with w_i replaced with x_j , $x_{\mathcal{N}}$ denotes the set of original variables with x_j replaced with w_i , the tildes on A , b , and c denote the fact that the *constants* have changed, and ζ^* is the value of the objective function associated with this new dictionary. Continue until dictionary solution is optimal. Click [here](#) to try.

Unboundedness

Consider the following dictionary:

		Current Dictionary								
obj =	0.0	+	2.0	x1	+	-1.0	x2	+	1.0	x3
w1 =	4.0	-	-5.0	x1	-	3.0	x2	-	-1.0	x3
w2 =	10.0	-	-1.0	x1	-	-5.0	x2	-	2.0	x3
w3 =	7.0	-	0.0	x1	-	-4.0	x2	-	3.0	x3
w4 =	6.0	-	-2.0	x1	-	-2.0	x2	-	4.0	x3
w5 =	6.0	-	-3.0	x1	-	0.0	x2	-	-3.0	x3

- Could increase either x_1 or x_3 to increase obj.
- Consider increasing x_1 .
- Which basic variable decreases to zero first?
- Answer: none of them, x_1 can grow without bound, and obj along with it.
- This is how we detect *unboundedness* with the simplex method.

Initialization

Consider the following problem:

$$\begin{array}{llllll} \text{maximize} & -3x_1 & + & 4x_2 & & \\ \text{subject to} & -4x_1 & - & 2x_2 & \leq & -8 \\ & -2x_1 & & & \leq & -2 \\ & 3x_1 & + & 2x_2 & \leq & 10 \\ & -x_1 & + & 3x_2 & \leq & 1 \\ & & & -3x_2 & \leq & -2 \\ & & & & & x_1, x_2 \geq 0. \end{array}$$

Phase-I Problem

- Modify problem by subtracting a new variable, x_0 , from each constraint and
- replacing objective function with $-x_0$

Phase-I Problem

$$\begin{array}{ll} \text{maximize} & -x_0 \\ \text{subject to} & -x_0 - 4x_1 - 2x_2 \leq -8 \\ & -x_0 - 2x_1 \leq -2 \\ & -x_0 + 3x_1 + 2x_2 \leq 10 \\ & -x_0 - x_1 + 3x_2 \leq 1 \\ & -x_0 - 3x_2 \leq -2 \\ & x_0, x_1, x_2 \geq 0. \end{array}$$

- Clearly feasible: pick x_0 large, $x_1 = 0$ and $x_2 = 0$.
- If optimal solution has $\text{obj} = 0$, then original problem is feasible.
- Final phase-I basis can be used as initial *phase-II* basis (ignoring x_0 thereafter).
- If optimal solution has $\text{obj} < 0$, then original problem is infeasible.

Initialization—First Pivot

Applet depiction shows both the Phase-I and the Phase-II objectives:

		Current Dictionary						
obj =	0.0	+	0.0	x0 +	-3.0	x1 +	4.0	x2
	0.0	+	-1.0	x0 +	0.0	x1 +	0.0	x2
w1 =	-8.0	-	-1.0	x0 -	-4.0	x1 -	-2.0	x2
w2 =	-2.0	-	-1.0	x0 -	-2.0	x1 -	0.0	x2
w3 =	10.0	-	-1.0	x0 -	3.0	x1 -	2.0	x2
w4 =	1.0	-	-1.0	x0 -	-1.0	x1 -	3.0	x2
w5 =	-2.0	-	-1.0	x0 -	0.0	x1 -	-3.0	x2

- Dictionary is infeasible even for Phase-I.
- One pivot needed to get feasible.
- Entering variable is x_0 .
- Leaving variable is one whose current value is most negative, i.e. w_1 .
- After first pivot...

Initialization—Second Pivot

Going into second pivot:

		Current Dictionary								
obj =	0.0	+	0.0	w1	+	-3.0	x1	+	4.0	x2
	-8.0	+	-1.0	w1	+	4.0	x1	+	2.0	x2
x0 =	8.0	-	-1.0	w1	-	4.0	x1	-	2.0	x2
w2 =	6.0	-	-1.0	w1	-	2.0	x1	-	2.0	x2
w3 =	18.0	-	-1.0	w1	-	7.0	x1	-	4.0	x2
w4 =	9.0	-	-1.0	w1	-	3.0	x1	-	5.0	x2
w5 =	6.0	-	-1.0	w1	-	4.0	x1	-	-1.0	x2

- Feasible!
- Focus on the yellow highlights.
- Let x_1 enter.
- Then w_5 must leave.
- After second pivot...

Initialization—Third Pivot

Going into third pivot:

Current Dictionary										
obj =	-4.5	+	-0.75	w1	+	0.75	w5	+	3.25	x2
	-2.0	+	0.0	w1	+	-1.0	w5	+	3.0	x2
x0 =	2.0	-	0.0	w1	-	-1.0	w5	-	3.0	x2
w2 =	3.0	-	-0.5	w1	-	-0.5	w5	-	2.5	x2
w3 =	7.5	-	0.75	w1	-	-1.75	w5	-	5.75	x2
w4 =	4.5	-	-0.25	w1	-	-0.75	w5	-	5.75	x2
x1 =	1.5	-	-0.25	w1	-	0.25	w5	-	-0.25	x2

- x_2 must enter.
- x_0 must leave.
- After third pivot...

End of Phase-I

Current dictionary:

Current Dictionary							
obj =	$-7/3$	+	$-3/4$	w1 +	$11/6$	w5 +	0 x0
	0	+	0	w1 +	0	w5 +	0 x0
x2 =	$2/3$	-	0	w1 -	$-1/3$	w5 -	0 x0
w2 =	$4/3$	-	$-1/2$	w1 -	$1/3$	w5 -	0 x0
w3 =	$11/3$	-	$3/4$	w1 -	$1/6$	w5 -	0 x0
w4 =	$2/3$	-	$-1/4$	w1 -	$7/6$	w5 -	0 x0
x1 =	$5/3$	-	$-1/4$	w1 -	$1/6$	w5 -	0 x0

- Optimal for Phase-I (no yellow highlights).
- $obj = 0$, therefore original problem is feasible.

Phase-II

Current dictionary:

Current Dictionary								
obj =	-7/3	+	-3/4	w1 +	11/6	w5 +	0	x0
	0	+	0	w1 +	0	w5 +	0	x0
x2 =	2/3	-	0	w1 -	-1/3	w5 -	0	x0
w2 =	4/3	-	-1/2	w1 -	1/3	w5 -	0	x0
w3 =	11/3	-	3/4	w1 -	1/6	w5 -	0	x0
w4 =	2/3	-	-1/4	w1 -	7/6	w5 -	0	x0
x1 =	5/3	-	-1/4	w1 -	1/6	w5 -	0	x0

For Phase-II:

- Ignore column with x_0 in Phase-II.
- Ignore Phase-I objective row.

w_5 must enter. w_4 must leave...

Optimal Solution

Current Dictionary										
obj =	-9/7	+	-5/14	w1	+	-11/7	w4	+	0	x0
	0	+	0	w1	+	0	w4	+	0	x0
x2 =	6/7	-	-1/14	w1	-	2/7	w4	-	0	x0
w2 =	8/7	-	-3/7	w1	-	-2/7	w4	-	0	x0
w3 =	25/7	-	11/14	w1	-	-1/7	w4	-	0	x0
w5 =	4/7	-	-3/14	w1	-	6/7	w4	-	0	x0
x1 =	11/7	-	-3/14	w1	-	-1/7	w4	-	0	x0

- Optimal!
- Click [here](#) to practice the simplex method on problems that may have infeasible first dictionaries.
- For instructions, click [here](#).

Degeneracy

Definitions.

A *dictionary* is degenerate if one or more “rhs”-value vanishes.

Example:

$$\begin{array}{r} \zeta = 6 + w_3 + 5x_2 + 4w_1 \\ \hline x_3 = 1 - 2w_3 - 2x_2 + 3w_1 \\ w_2 = 4 + w_3 + x_2 - 3w_1 \\ x_1 = 3 - 2w_3 \\ w_4 = 2 + w_3 - w_1 \\ w_5 = 0 - x_2 + w_1 \end{array}$$

A *pivot* is degenerate if the objective function value does not change.

Examples (based on above dictionary):

1. If x_2 enters, then w_5 must leave, pivot is degenerate.
2. If w_1 enters, then w_2 must leave, pivot is *not* degenerate.

Cycling

Definition.

A *cycle* is a sequence of pivots that returns to the dictionary from which the cycle began.

Note: Every pivot in a cycle must be degenerate. Why?

Pivot Rules.

Definition.

Explicit statement for how one chooses entering and leaving variables (when a choice exists).

Largest-Coefficient Rule.

A common pivot rule for entering variable:

Choose the variable with the largest coefficient in the objective function.

Hope.

Some pivot rule, such as the largest coefficient rule, will be proven never to cycle.

Hope Fades

An example that cycles using the following pivot rules:

- entering variable: largest-coefficient rule.
- leaving variable: smallest-index rule.

$$\begin{array}{r} \zeta = \\ \hline w_1 = \\ w_2 = \\ w_3 = \end{array} \begin{array}{r} x_1 - 2x_2 - 2x_4 \\ -0.5x_1 + 3.5x_2 + 2x_3 - 4x_4 \\ -0.5x_1 + x_2 + 0.5x_3 - 0.5x_4 \\ 1 - x_1 . \end{array}$$

Here's a demo of cycling (ignoring the last constraint)...

Current Dictionary

$$\begin{array}{r}
 \text{obj} = 0 + 1 x_1 + (-2) x_2 + 0 x_3 + (-2) x_4 \\
 w_1 = 0 - 1/2 x_1 - (-7/2) x_2 - (-2) x_3 - 4 x_4 \\
 w_2 = 0 - 1/2 x_1 - (-1) x_2 - (-1/2) x_3 - 1/2 x_4
 \end{array}$$

$x_1 \Leftrightarrow w_1$:

Current Dictionary

$$\begin{array}{r}
 \text{obj} = 0 + (-2) w_1 + 5 x_2 + 4 x_3 + (-10) x_4 \\
 x_1 = 0 - 2 w_1 - (-7) x_2 - (-4) x_3 - 8 x_4 \\
 w_2 = 0 - (-1) w_1 - 5/2 x_2 - 3/2 x_3 - (-7/2) x_4
 \end{array}$$

$x_2 \Leftrightarrow w_2$:

Current Dictionary

$$\begin{array}{r}
 \text{obj} = 0 + 0 w_1 + (-2) w_2 + 1 x_3 + (-3) x_4 \\
 x_1 = 0 - (-4/5) w_1 - 14/5 w_2 - 1/5 x_3 - (-9/5) x_4 \\
 x_2 = 0 - (-2/5) w_1 - 2/5 w_2 - 3/5 x_3 - (-7/5) x_4
 \end{array}$$

$x_3 \Leftrightarrow x_1$:

Current Dictionary

$$\begin{array}{r}
 \text{obj} = 0 + 4 w_1 + (-16) w_2 + (-5) x_1 + 6 x_4 \\
 x_3 = 0 - (-4) w_1 - 14 w_2 - 5 x_1 - (-9) x_4 \\
 x_2 = 0 - 2 w_1 - (-8) w_2 - (-3) x_1 - 4 x_4
 \end{array}$$

$x_4 \Leftrightarrow x_2$:

Current Dictionary													
obj =	0	+	1	w1	+	-4	w2	+	-1/2	x1	+	-3/2	x2
x3 =	0	-	1/2	w1	-	-4	w2	-	-7/4	x1	-	9/4	x2
x4 =	0	-	1/2	w1	-	-2	w2	-	-3/4	x1	-	1/4	x2

$w_1 \Leftrightarrow x_3$:

Current Dictionary													
obj =	0	+	-2	x3	+	4	w2	+	3	x1	+	-6	x2
w1 =	0	-	2	x3	-	-8	w2	-	-7/2	x1	-	9/2	x2
x4 =	0	-	-1	x3	-	2	w2	-	1	x1	-	-2	x2

$w_2 \Leftrightarrow x_4$:

Current Dictionary													
obj =	0	+	0	x3	+	-2	x4	+	1	x1	+	-2	x2
w1 =	0	-	-2	x3	-	4	x4	-	1/2	x1	-	-7/2	x2
w2 =	0	-	-1/2	x3	-	1/2	x4	-	1/2	x1	-	-1	x2

Cycling is rare! A program that generates random 2×4 fully degenerate problems was run more than *one billion* times and did not find one example!

Perturbation Method

Whenever a vanishing “rhs” appears perturb it.
If there are lots of them, say k , perturb them all.
Make the perturbations at different *scales*:

$$\text{other data} \gg \epsilon_1 \gg \epsilon_2 \gg \dots \gg \epsilon_k > 0.$$

An Example.

		Current Dictionary															
obj	=	0.0	+	0.0	e1	+	0.0	e2	+	0.0	e3	+	2.0	x1	+	4.0	x2
w1	=	0.0	+	1.0	e1	+	0.0	e2	+	0.0	e3	-	-1.0	x1	-	1.0	x2
w2	=	0.0	+	0.0	e1	+	1.0	e2	+	0.0	e3	-	-3.0	x1	-	1.0	x2
w3	=	0.0	+	0.0	e1	+	0.0	e2	+	1.0	e3	-	4.0	x1	-	-1.0	x2

Entering variable: x_2

Leaving variable: w_2

		Current Dictionary															
obj	=	0.0	+	0.0	e1	+	4.0	e2	+	0.0	e3	+	14.0	x1	+	-4.0	w2
w1	=	0.0	+	1.0	e1	+	-1.0	e2	+	0.0	e3	-	2.0	x1	-	-1.0	w2
x2	=	0.0	+	0.0	e1	+	1.0	e2	+	0.0	e3	-	-3.0	x1	-	1.0	w2
w3	=	0.0	+	0.0	e1	+	1.0	e2	+	1.0	e3	-	1.0	x1	-	1.0	w2

Perturbation Method—Example Con't.

Recall current dictionary:

		Current Dictionary															
obj	=	0.0	+	0.0	e1	+	4.0	e2	+	0.0	e3	+	14.0	x1	+	-4.0	w2
w1	=	0.0	+	1.0	e1	+	-1.0	e2	+	0.0	e3	-	2.0	x1	-	-1.0	w2
x2	=	0.0	+	0.0	e1	+	1.0	e2	+	0.0	e3	-	-3.0	x1	-	1.0	w2
w3	=	0.0	+	0.0	e1	+	1.0	e2	+	1.0	e3	-	1.0	x1	-	1.0	w2

Entering variable: x_1

Leaving variable: w_3

		Current Dictionary															
obj	=	0.0	+	0.0	e1	+	18.0	e2	+	14.0	e3	+	-14.0	w3	+	-18.0	w2
w1	=	0.0	+	1.0	e1	+	-3.0	e2	+	-2.0	e3	-	-2.0	w3	-	-3.0	w2
x2	=	0.0	+	0.0	e1	+	4.0	e2	+	3.0	e3	-	3.0	w3	-	4.0	w2
x1	=	0.0	+	0.0	e1	+	1.0	e2	+	1.0	e3	-	1.0	w3	-	1.0	w2

Other Pivot Rules

Smallest Index Rule.

Choose the variable with the smallest index (the x variables are assumed to be “before” the w variables).

Note: Also known as *Bland's rule*.

Random Selection Rule.

Select at random from the set of possibilities.

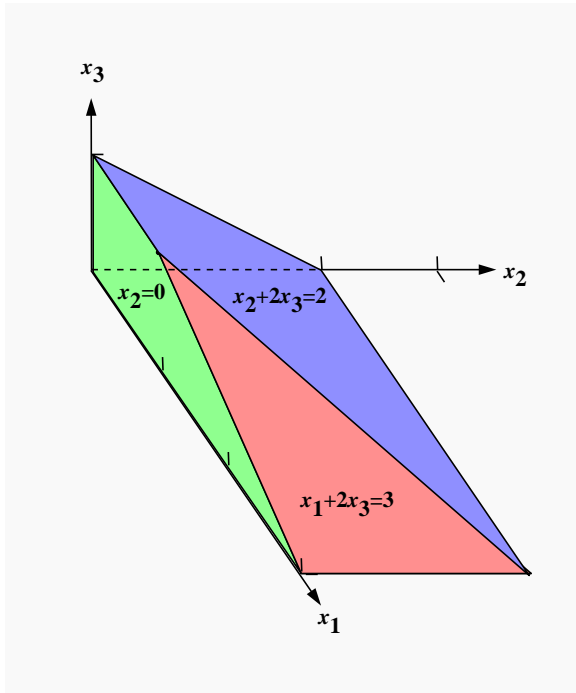
Greatest Increase Rule.

Pick the entering/leaving pair so as to maximize the increase of the objective function over all other possibilities.

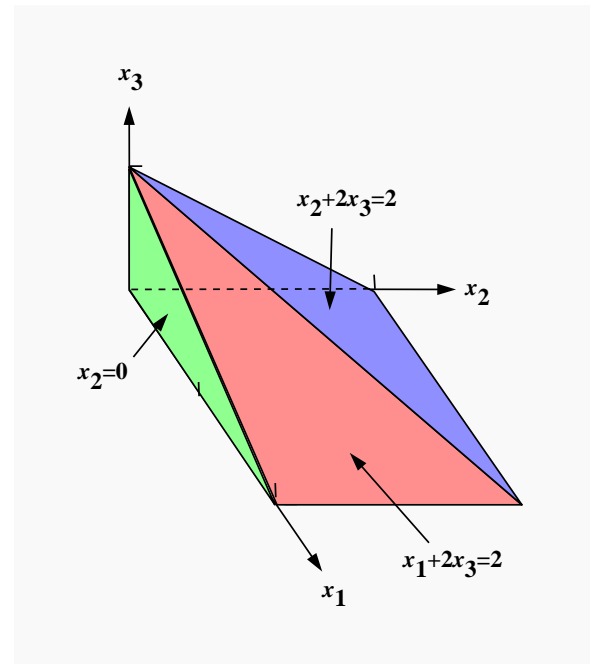
Note: Too much computation.

Geometry

$$\begin{aligned} &\text{maximize} && x_1 + 2x_2 + 3x_3 \\ &\text{subject to} && x_1 + 2x_3 \leq 3 \\ & && x_2 + 2x_3 \leq 2 \\ & && x_1, x_2, x_3 \geq 0. \end{aligned}$$



$$\begin{aligned} &\text{maximize} && x_1 + 2x_2 + 3x_3 \\ &\text{subject to} && x_1 + 2x_3 \leq 2 \\ & && x_2 + 2x_3 \leq 2 \\ & && x_1, x_2, x_3 \geq 0. \end{aligned}$$



Efficiency

Question:

Given a problem of a certain size, how long will it take to solve it?

Two Kinds of Answers:

- *Average Case*. How long for a *typical* problem.
- *Worst Case*. How long for the *hardest* problem.

Average Case.

- Mathematically difficult.
- Empirical studies.

Worst Case.

- Mathematically tractable.
- Limited value.

Measures

Measures of Size

- Number of constraints m and/or number of variables n .
- Number of data elements, mn .
- Number of nonzero data elements.
- Size, in bytes, of AMPL formulation (model+data).

Measuring Time

Two factors:

- Number of iterations.
- Time per iteration.

Klee–Minty Problem (1972)

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^n 2^{n-j} x_j \\ \text{subject to} \quad & 2 \sum_{j=1}^{i-1} 2^{i-j} x_j + x_i \leq 100^{i-1} \quad i = 1, 2, \dots, n \\ & x_j \geq 0 \quad j = 1, 2, \dots, n. \end{aligned}$$

Example $n = 3$:

$$\begin{aligned} \text{maximize} \quad & 4 x_1 + 2 x_2 + x_3 \\ \text{subj. to} \quad & x_1 \leq 1 \\ & 4 x_1 + x_2 \leq 100 \\ & 8 x_1 + 4 x_2 + x_3 \leq 10000 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

A Distorted Cube

Constraints represent a “minor” distortion to an n -dimensional hypercube:

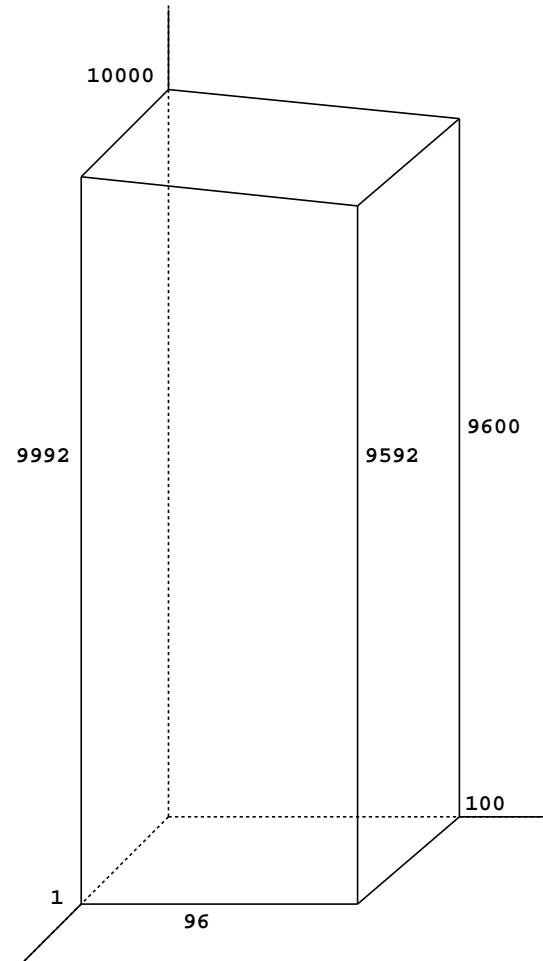
$$0 \leq x_1 \leq 1$$

$$0 \leq x_2 \leq 100$$

\vdots

$$0 \leq x_n \leq 100^{n-1}.$$

Case $n = 3$:



Analysis

Replace

$$1, 100, 10000, \dots,$$

with

$$1 = b_1 \ll b_2 \ll b_3 \ll \dots$$

Then, make following replacements to rhs:

$$b_1 \longrightarrow b_1$$

$$b_2 \longrightarrow 2b_1 + b_2$$

$$b_3 \longrightarrow 4b_1 + 2b_2 + b_3$$

$$b_4 \longrightarrow 8b_1 + 4b_2 + 2b_3 + b_4$$

⋮

Hardly a change!

Make a similar constant adjustment to objective function.

Look at the pivot tool version...

Case $n = 3$:

obj	=	-2	b1	+	-1	b2	+	0	b3	+	4	x1	+	2	x2	+	1	x3
w1	=	1	b1	+	0	b2	+	0	b3	-	1	x1	-	0	x2	-	0	x3
w2	=	2	b1	+	1	b2	+	0	b3	-	4	x1	-	1	x2	-	0	x3
w3	=	4	b1	+	2	b2	+	1	b3	-	8	x1	-	4	x2	-	1	x3

Now, watch the pivots...

Exponential

Klee–Minty problem shows that:

Largest-coefficient rule can take $2^n - 1$ pivots to solve a problem in n variables and constraints.

For $n = 70$,

$$2^n = 1.2 \times 10^{21}.$$

At 1000 iterations per second, this problem will take 40 billion years to solve. The age of the universe is estimated at 15 billion years.

Yet, problems with 10,000 to 100,000 variables are solved routinely every day.

Worst case analysis is just that: worst case.

Complexity

n	n^2	n^3	2^n
1	1	1	2
2	4	8	4
3	9	27	8
4	16	64	16
5	25	125	32
6	36	216	64
7	49	343	128
8	64	512	256
9	81	729	512
10	100	1000	1024
12	144	1728	4096
14	196	2744	16384
16	256	4096	65536
18	324	5832	262144
20	400	8000	1048576
22	484	10648	4194304
24	576	13824	16777216
26	676	17576	67108864
28	784	21952	268435456
30	900	27000	1073741824

Sorting: fast algorithm = $n \log n$,
slow algorithm = n^2

Matrix times vector: n^2

Matrix times matrix: n^3

Matrix inversion: n^3

Simplex Method:

- Worst case: $n^2 2^n$ operations.
- Average case: n^3 operations.
- Open question:

Does there exist a variant of the simplex method whose worst case performance is polynomial?

Linear Programming:

- *Theorem*: There exists an algorithm whose worst case performance is $n^{3.5}$ operations.

Average Case

Matlab Program

Define a random problem:

```
m = ceil(exp(log(400)*rand()));
n = ceil(exp(log(400)*rand()));

A = round(sigma*(randn(m,n)));
b = round(sigma*rand(m,1));

y = round(sigma*rand(1,m));
z = round(sigma*rand(1,n));
c = y*A - z;

A = -A;
```

Initialize a few things:

```
iter = 0;
opt = 0;
```

The Main Loop:

```
while max(c) > eps || min(b) < -eps,
    % pick largest coefficient
    [cj, col] = max(c);
    Acol = A(:,col);

    % select leaving variable
    [t, row] = max(-Acol./b);
    if t < eps,
        opt = -1; % unbounded
        'unbounded'
        break;
    end
    Arow = A(row,:);

    a = A(row,col); % pivot element

    .
    .
    .

    iter = iter+1;
end
```



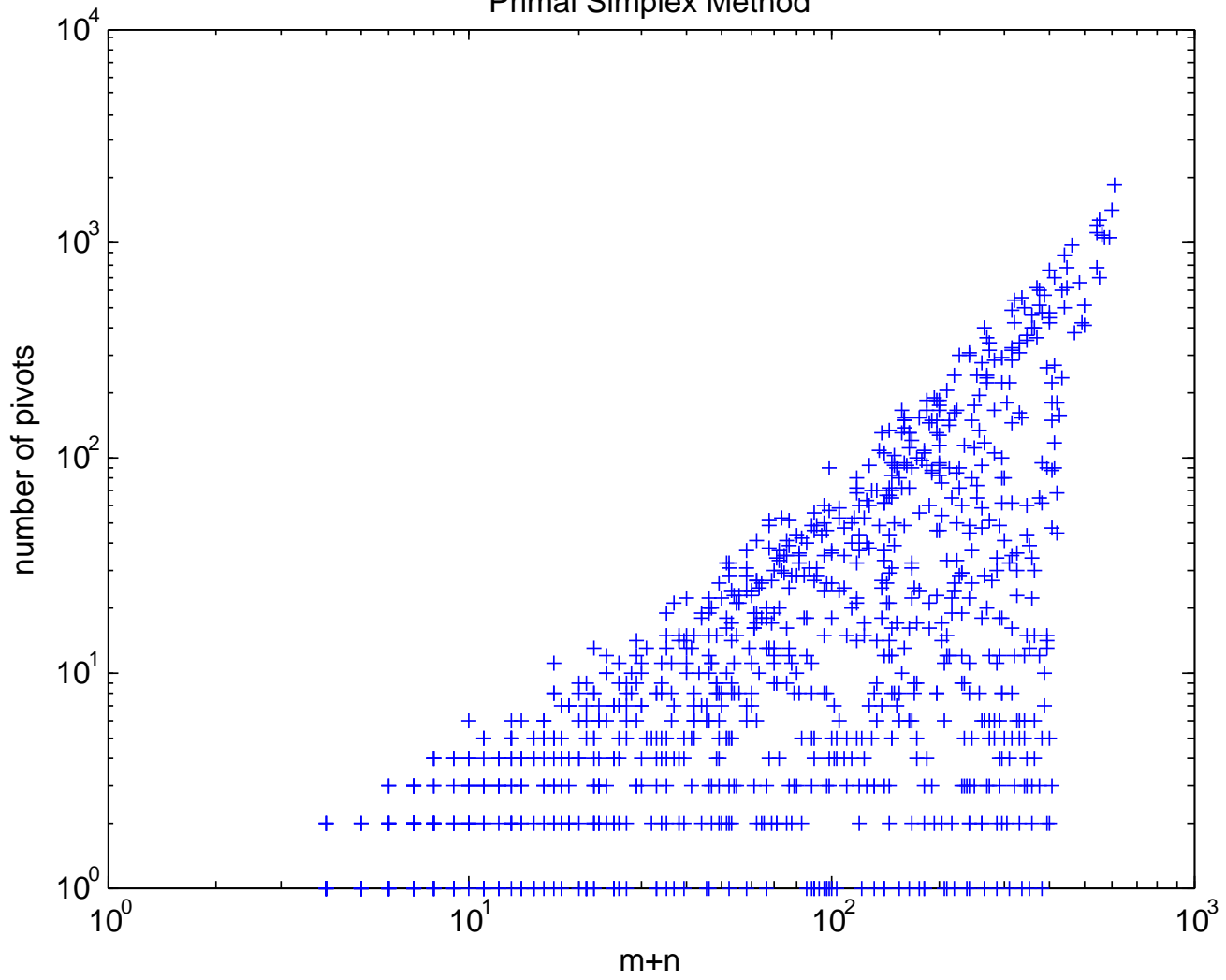
The code for a pivot:

```
A = A - Acol*Arow/a;
A(row,:) = -Arow/a;
A(:,col) = Acol/a;
A(row,col) = 1/a;

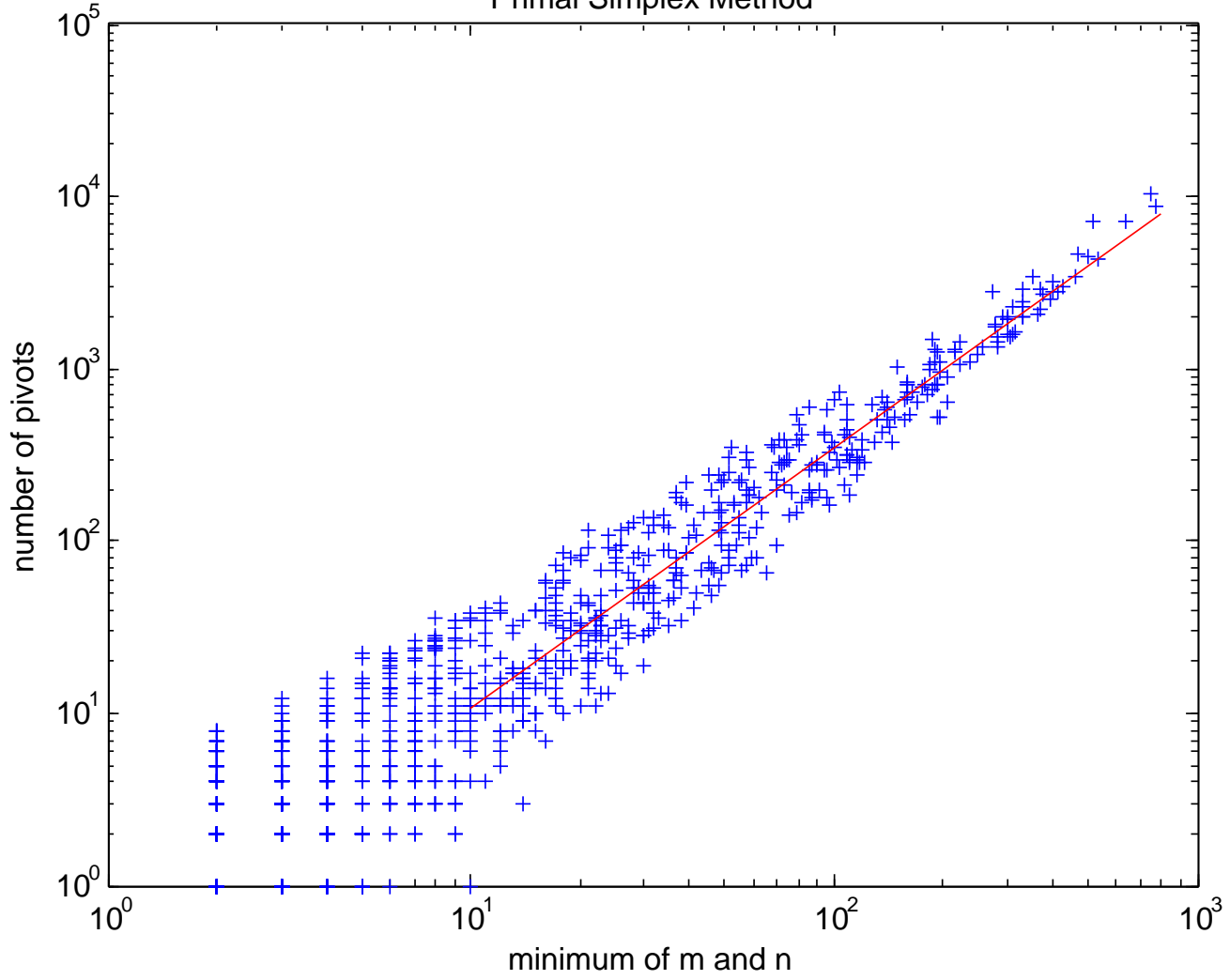
brow = b(row);
b = b - Acol*brow/a;
b(row) = -brow/a;

ccol = c(col);
c = c - ccol*Arow/a;
c(col) = ccol/a;
```

Primal Simplex Method

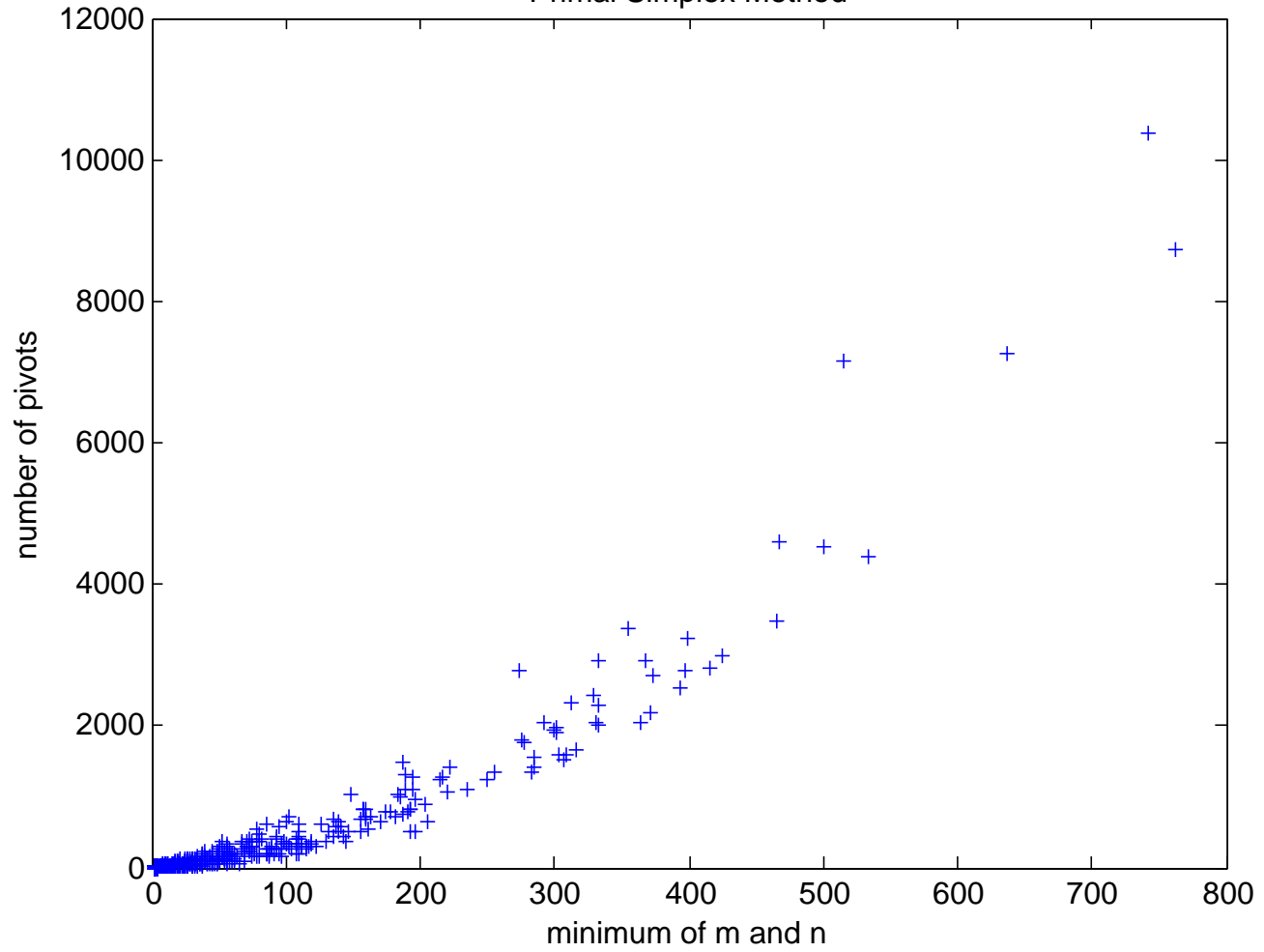


Primal Simplex Method



$$\text{iters} = 0.6893 \min(m, n)^{1.3347} \approx \frac{2}{3} \min(m, n)^{4/3}$$

Primal Simplex Method



Average Case—AMPL Version

Declare parameters:

```
param eps := 1e-9;
param sigma := 30;
param niters := 1000;
param size := 400;

param m;
param n;
param AA {1..size, 1..size};
param bb {1..size};
param cc {1..size};
param A {1..size, 1..size};
param b {1..size};
param c {1..size};
param x {1..size};
param z {1..size};
param y {1..size};
param w {1..size};

param iter;
param opt;
param stop;
param mniters {1..niters, 1..3};
param maxc;
param minbovera;
param col;
param row;
```

```
param Arow {1..size};
param Acol {1..size};
param a;
param brow;
param ccol;
param ii;
param jj;
```

Define a random problem:

```
let m := ceil(exp(log(size)*Uniform01()));
let n := ceil(exp(log(size)*Uniform01()));
let {i in 1..m, j in 1..n} A[i,j] := round(sigma*Normal01());
let {i in 1..m} y[i] := round(sigma*Uniform01());
let {j in 1..n} z[j] := round(sigma*Uniform01());
let {i in 1..m} b[i] := round(sigma*Uniform01());
let {j in 1..n} c[j] := sum {i in 1..m} y[i]*A[i,j] - z[j];
let {i in 1..m, j in 1..n} A[i,j] := -A[i,j];
let {i in 1..m, j in 1..n} AA[i,j] := A[i,j];
let {i in 1..m} bb[i] := b[i];
let {j in 1..n} cc[j] := c[j];
```

The Simplex Method (Phase 2)

```
repeat while (max {j in 1..n} c[j]) > eps {
  let maxc := 0;
  for {j in 1..n} {
    if (c[j] > maxc) then {
      let maxc := c[j];
      let col := j;
    }
  }
  let minbovera := 1/eps;
  for {i in 1..m} {
    if (A[i,col] < -eps) then {
      if (-b[i]/A[i,col] < minbovera) then {
        let minbovera := -b[i]/A[i,col];
        let row := i;
      }
    }
  }
  if minbovera >= 1/eps then {
    let opt := -1; # unbounded
    display "unbounded";
    break;
  }
  .
  .
  .
}
```

The code for a pivot:

```
let {j in 1..n} Arow[j] := A[row,j];
let {i in 1..m} Acol[i] := A[i,col];
let a := A[row,col];
let {i in 1..m, j in 1..n}
  A[i,j] := A[i,j] - Acol[i]*Arow[j]/a;
let {j in 1..n} A[row,j] := -Arow[j]/a;
let {i in 1..m} A[i,col] := Acol[i]/a;
let A[row,col] := 1/a;

let brow := b[row];
let {i in 1..m}
  b[i] := b[i] - brow*Acol[i]/a;
let b[row] := -brow/a;

let ccol := c[col];
let {j in 1..n}
  c[j] := c[j] - ccol*Arow[j]/a;
let c[col] := ccol/a;
```



The AMPL code can be found here:

<http://orfe.princeton.edu/rvdb/307/lectures/primalsimplex.txt>

Duality

Every Problem:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

\implies

Has a Dual:

$$\begin{array}{ll} \text{minimize} & b^T y \\ \text{subject to} & A^T y \geq c \\ & y \geq 0. \end{array}$$

Original problem is called the *primal problem*.

A problem is defined by its data (notation used for the variables is arbitrary).

Dual in “Standard” Form:

$$\begin{array}{ll} \text{—maximize} & -b^T y \\ \text{subject to} & -A^T y \leq -c \\ & y \geq 0. \end{array}$$

Dual is *negative transpose* of primal.

Theorem 1 *Dual of dual is primal.*

Weak Duality Theorem

Theorem 2 *If x is feasible for the primal and y is feasible for the dual, then*

$$c^T x \leq b^T y.$$

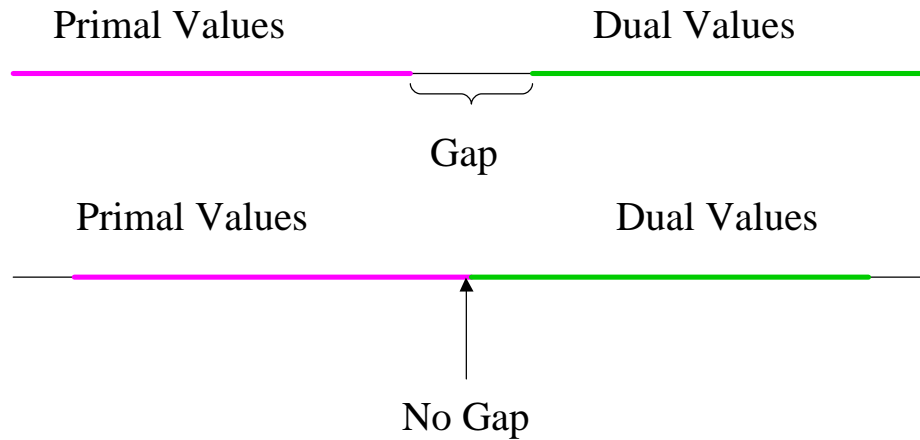
Proof.

$$\begin{aligned} c^T x &\leq (A^T y)^T x && \text{(because } c \leq A^T y \text{ and } x \geq 0) \\ &= y^T Ax \\ &= (Ax)^T y \\ &\leq b^T y && \text{(because } Ax \leq b \text{ and } y \geq 0). \end{aligned}$$

Gap or No Gap?

An important question:

Is there a gap between the **largest primal value** and the **smallest dual value**?



Answer is provided by the Strong Duality Theorem (coming later).

Simplex Method and Duality

A Primal Problem:

$$\begin{aligned} \text{obj} &= 0 + (-3)x_1 + 2x_2 + 1x_3 \\ w_1 &= 0 - 0x_1 - (-1)x_2 - 2x_3 \\ w_2 &= 3 - (-3)x_1 - 4x_2 - 1x_3 \end{aligned}$$

Its Dual:

$$\begin{aligned} \text{obj} &= 0 + 0y_1 + (-3)y_2 \\ z_1 &= 3 - 0y_1 - 3y_2 \\ z_2 &= -2 - 1y_1 - (-4)y_2 \\ z_3 &= -1 - (-2)y_1 - (-1)y_2 \end{aligned}$$

Notes:

- Dual is negative transpose of primal.
- Primal is feasible, dual is not.

Use primal to choose pivot: x_2 enters, w_2 leaves.
Make analogous pivot in dual: z_2 leaves, y_2 enters.

Second Iteration

After First Pivot:

Primal (feasible):

$$\begin{array}{rcl} \text{obj} & = & \boxed{3/2} + \boxed{-3/2} x_1 + \boxed{-1/2} w_2 + \boxed{1/2} x_3 \\ w_1 & = & \boxed{3/4} - \boxed{-3/4} x_1 - \boxed{1/4} w_2 - \boxed{9/4} x_3 \\ x_2 & = & \boxed{3/4} - \boxed{-3/4} x_1 - \boxed{1/4} w_2 - \boxed{1/4} x_3 \end{array}$$

Dual (still not feasible):

$$\begin{array}{rcl} \text{obj} & = & \boxed{-3/2} + \boxed{-3/4} y_1 + \boxed{-3/4} z_2 \\ z_1 & = & \boxed{3/2} - \boxed{3/4} y_1 - \boxed{3/4} z_2 \\ y_2 & = & \boxed{1/2} - \boxed{-1/4} y_1 - \boxed{-1/4} z_2 \\ z_3 & = & \boxed{-1/2} - \boxed{-9/4} y_1 - \boxed{-1/4} z_2 \end{array}$$

Note: *negative transpose property intact.*

Again, use primal to pick pivot: x_3 enters, w_1 leaves.

Make analogous pivot in dual: z_3 leaves, y_1 enters.

After Second Iteration

Primal:

- Is *optimal*.

$$\begin{aligned} \text{obj} &= \boxed{5/3} + \boxed{-4/3} x_1 + \boxed{-5/9} w_2 + \boxed{-2/9} w_1 \\ x_3 &= \boxed{1/3} - \boxed{-1/3} x_1 - \boxed{1/9} w_2 - \boxed{4/9} w_1 \\ x_2 &= \boxed{2/3} - \boxed{-2/3} x_1 - \boxed{2/9} w_2 - \boxed{-1/9} w_1 \end{aligned}$$

Dual:

- Negative transpose property remains intact.
- Is *optimal*.

$$\begin{aligned} \text{obj} &= \boxed{-5/3} + \boxed{-1/3} z_3 + \boxed{-2/3} z_2 \\ z_1 &= \boxed{4/3} - \boxed{1/3} z_3 - \boxed{2/3} z_2 \\ y_2 &= \boxed{5/9} - \boxed{-1/9} z_3 - \boxed{-2/9} z_2 \\ y_1 &= \boxed{2/9} - \boxed{-4/9} z_3 - \boxed{1/9} z_2 \end{aligned}$$

Conclusion

Simplex method applied to primal problem (two phases, if necessary), solves both the primal and the dual.

Dual Simplex Method (used when dual is feasible)

Primal Dictionary:

$$\begin{aligned}\zeta &= 0 + c^T x \\ w &= b - Ax\end{aligned}$$

Dual Dictionary:

$$\begin{aligned}-\xi &= 0 - b^T y \\ z &= -c + A^T y\end{aligned}$$

Entering Variable. Choose an index i for which $b_i < 0$. Variable y_i is the entering variable.

Leaving Variable. Let y_i increase while holding all other y_k 's at zero. Figure out which slack variable hits zero first. Let's say it's z_j .

Dual Pivot. In dual problem, y_i is the entering variable and z_j is the leaving variable.

Primal Pivot. The corresponding pivot in primal problem has w_i as the leaving variable and x_j as the entering variable.

After the pivot, the primal and dual are still negative transposes of each other.

Continue improving the dual problem until reaching optimality.

At optimality, $\tilde{b} \geq 0$ and $\tilde{c} \leq 0$.

Hence, solution associated with primal dictionary is optimal too.

Dual Simplex Method

When: dual feasible, primal infeasible (i.e., pinks on the left, not on top).

An Example. Showing both primal and dual dictionaries:

obj	=	0.0	+	-2.0	x1	+	-4.0	x2	+	0.0	x3	+	-6.0	x4
w1	=	-3.0	-	-1.0	x1	-	2.0	x2	-	0.0	x3	-	-1.0	x4
w2	=	-5.0	-	2.0	x1	-	-3.0	x2	-	0.0	x3	-	-2.0	x4
w3	=	8.0	-	2.0	x1	-	3.0	x2	-	3.0	x3	-	2.0	x4

obj	=	0.0	+	3.0	y1	+	5.0	y2	+	-8.0	y3
z1	=	2.0	-	1.0	y1	-	-2.0	y2	-	-2.0	y3
z2	=	4.0	-	-2.0	y1	-	3.0	y2	-	-3.0	y3
z3	=	0.0	-	0.0	y1	-	0.0	y2	-	-3.0	y3
z4	=	6.0	-	1.0	y1	-	2.0	y2	-	-2.0	y3

Looking at dual dictionary: y_2 enters, z_2 leaves.

On the primal dictionary: w_2 leaves, x_2 enters.

After pivot...

Dual Simplex Method: Second Pivot

Going in, we have:

obj	=	-6.6667	+	-4.6667	x1	+	-1.3333	w2	+	0.0	x3	+	-3.3333	x4
w1	=	-6.3333	-	0.3333	x1	-	0.6667	w2	-	0.0	x3	-	-2.3333	x4
x2	=	1.6667	-	-0.6667	x1	-	-0.3333	w2	-	0.0	x3	-	0.6667	x4
w3	=	3.0	-	4.0	x1	-	1.0	w2	-	3.0	x3	-	0.0	x4

obj	=	6.6667	+	6.3333	y1	+	-1.6667	z2	+	-3.0	y3
z1	=	4.6667	-	-0.3333	y1	-	0.6667	z2	-	-4.0	y3
y2	=	1.3333	-	-0.6667	y1	-	0.3333	z2	-	-1.0	y3
z3	=	0.0	-	0.0	y1	-	0.0	z2	-	-3.0	y3
z4	=	3.3333	-	2.3333	y1	-	-0.6667	z2	-	0.0	y3

Looking at dual: y_1 enters, z_4 leaves.

Looking at primal: w_1 leaves, x_4 enters.

Dual Simplex Method Pivot Rule

obj =	-6.6667	+	-4.6667	x1	+	-1.3333	w2	+	0.0	x3	+	-3.3333	x4
w1 =	-6.3333	-	0.3333	x1	-	0.6667	w2	-	0.0	x3	-	-2.3333	x4
x2 =	1.6667	-	-0.6667	x1	-	-0.3333	w2	-	0.0	x3	-	0.6667	x4
w3 =	3.0	-	4.0	x1	-	1.0	w2	-	3.0	x3	-	0.0	x4

Referring to the primal dictionary:

- Pick leaving variable from those rows that are *infeasible*.
- Pick entering variable from a box with a negative value and which can be increased the least (on the dual side).

Next primal dictionary shown on next page...

Dual Simplex Method: Third Pivot

Going in, we have:

obj	=	-15.7143	+	-5.1429	x1	+	-2.2857	w2	+	0.0	x3	+	-1.4286	w1
x4	=	2.7143	-	-0.1429	x1	-	-0.2857	w2	-	0.0	x3	-	-0.4286	w1
x2	=	-0.1429	-	-0.5714	x1	-	-0.1429	w2	-	0.0	x3	-	0.2857	w1
w3	=	3.0	-	4.0	x1	-	1.0	w2	-	3.0	x3	-	0.0	w1

Which variable must leave and which must enter?

See next page...

Dual Simplex Method: Third Pivot—Answer

Answer is: x_2 leaves, x_1 enters.

Resulting dictionary is OPTIMAL:

obj	=	-17.0	+	-9.0	x_2	+	-1.0	w_2	+	0.0	x_3	+	-4.0	w_1
x_4	=	2.75	-	-0.25	x_2	-	-0.25	w_2	-	0.0	x_3	-	-0.5	w_1
x_1	=	0.25	-	-1.75	x_2	-	0.25	w_2	-	0.0	x_3	-	-0.5	w_1
w_3	=	2.0	-	7.0	x_2	-	0.0	w_2	-	3.0	x_3	-	2.0	w_1

Dual-Based Phase I Method

Example:

obj	=	0.0		+	-4.0	x1	+	2.0	x2	+	3.0	x3	
w1	=	0.0	+	1.0	-	2.0	x1	-	-1.0	x2	-	3.0	x3
w2	=	0.0	+	1.0	-	3.0	x1	-	-3.0	x2	-	-4.0	x3
w3	=	-3.0	+	1.0	-	-1.0	x1	-	-1.0	x2	-	1.0	x3
w4	=	-1.0	+	1.0	-	-2.0	x1	-	0.0	x2	-	0.0	x3

Notes:

- Two objective functions: the true objective (on top), and a fake one (below it).
- For *Phase I*, use the fake objective—it's dual feasible.
- Two right-hand sides: the real one (on the left) and a fake (on the right).
- Ignore the fake right-hand side—we'll use it in another algorithm later.

Phase I—First Pivot: w_3 leaves, x_1 enters.

After first pivot...

Dual-Based Phase I Method—Second Pivot

Current dictionary:

obj	=	-12.0	+		+	-4.0	w3	+	6.0	x2	+	-1.0	x3
w1	=	-6.0	+	3.0	-	2.0	w3	-	-3.0	x2	-	5.0	x3
w2	=	-9.0	+	4.0	-	3.0	w3	-	-6.0	x2	-	-1.0	x3
x1	=	3.0	+	-1.0	-	-1.0	w3	-	1.0	x2	-	-1.0	x3
w4	=	5.0	+	-1.0	-	-2.0	w3	-	2.0	x2	-	-2.0	x3

Dual pivot: w_2 leaves, x_2 enters.

After pivot:

obj	=	-3.0	+		+	-1.0	w3	+	1.0	w2	+	-2.0	x3
w1	=	-1.5	+	1.0	-	0.5	w3	-	-0.5	w2	-	5.5	x3
x2	=	1.5	+	-0.6667	-	-0.5	w3	-	-0.1667	w2	-	0.1667	x3
x1	=	1.5	+	-0.3333	-	-0.5	w3	-	0.1667	w2	-	-1.1667	x3
w4	=	2.0	+	0.3333	-	-1.0	w3	-	0.3333	w2	-	-2.3333	x3

Dual-Based Phase I Method—Third Pivot

Current dictionary:

obj	=	-3.0	+	-1.0	w3	+	1.0	w2	+	-2.0	x3		
w1	=	-1.5	+	1.0	-	0.5	w3	-	-0.5	w2	-	5.5	x3
x2	=	1.5	+	-0.6667	-	-0.5	w3	-	-0.1667	w2	-	0.1667	x3
x1	=	1.5	+	-0.3333	-	-0.5	w3	-	0.1667	w2	-	-1.1667	x3
w4	=	2.0	+	0.3333	-	-1.0	w3	-	0.3333	w2	-	-2.3333	x3

Dual pivot:

w_1 leaves,
 w_2 enters.

After pivot:

obj	=	0.0	+	0.0	w3	+	2.0	w1	+	9.0	x3		
w2	=	3.0	+	-2.0	-	-1.0	w3	-	-2.0	w1	-	-11.0	x3
x2	=	2.0	+	-1.0	-	-0.6667	w3	-	-0.3333	w1	-	-1.6667	x3
x1	=	1.0	+	0.0	-	-0.3333	w3	-	0.3333	w1	-	0.6667	x3
w4	=	1.0	+	1.0	-	-0.6667	w3	-	0.6667	w1	-	1.3333	x3

It's *feasible!*

Fourth Pivot—Phase II

Current dictionary:

obj	=	0.0			+	0.0	w3	+	2.0	w1	+	9.0	x3
					+	-1.0	w3	+	0.0	w1	+	-2.0	x3
w2	=	3.0	+	-2.0	-	-1.0	w3	-	-2.0	w1	-	-11.0	x3
x2	=	2.0	+	-1.0	-	-0.6667	w3	-	-0.3333	w1	-	-1.6667	x3
x1	=	1.0	+	0.0	-	-0.3333	w3	-	0.3333	w1	-	0.6667	x3
w4	=	1.0	+	1.0	-	-0.6667	w3	-	0.6667	w1	-	1.3333	x3

It's feasible.

Ignore fake objective.

Use the real thing (top row).

Primal pivot: x_3 enters, w_4 leaves.

After pivot:

obj	=	6.75		+	4.5	w3	+	-2.5	w1	+	-6.75	w4	
				+	-2.0	w3	+	1.0	w1	+	1.5	w4	
w2	=	11.25	+	6.25	-	-6.5	w3	-	3.5	w1	-	8.25	w4
x2	=	3.25	+	0.25	-	-1.5	w3	-	0.5	w1	-	1.25	w4
x1	=	0.5	+	-0.5	-	0.0	w3	-	0.0	w1	-	-0.5	w4
x3	=	0.75	+	0.75	-	-0.5	w3	-	0.5	w1	-	0.75	w4

Problem is *unbounded!*

Strong Duality Theorem

Conclusion on previous slide is the essence of the *strong duality theorem* which we now state:

Theorem 3 *If the primal problem has an optimal solution,*

$$x^* = (x_1^*, x_2^*, \dots, x_n^*),$$

then the dual also has an optimal solution,

$$y^* = (y_1^*, y_2^*, \dots, y_m^*),$$

and

$$\sum_j c_j x_j^* = \sum_i b_i y_i^*.$$

Paraphrase:

If primal has an optimal solution, then there is no duality gap.

Duality Gap

Four possibilities:

- Primal optimal, dual optimal (no gap).
- Primal unbounded, dual infeasible (no gap).
- Primal infeasible, dual unbounded (no gap).
- Primal infeasible, dual infeasible (infinite gap).

Example of *infinite gap*:

$$\begin{array}{ll} \text{maximize} & 2x_1 - x_2 \\ \text{subject to} & x_1 - x_2 \leq 1 \\ & -x_1 + x_2 \leq -2 \\ & x_1, x_2 \geq 0. \end{array}$$

Complementary Slackness

Theorem 4 *At optimality, we have*

$$\begin{aligned}x_j z_j &= 0, & \text{for } j = 1, 2, \dots, n, \\w_i y_i &= 0, & \text{for } i = 1, 2, \dots, m.\end{aligned}$$

Proof

Recall the proof of the Weak Duality Theorem:

$$\begin{aligned}\sum_j c_j x_j &\leq \sum_j (c_j + z_j) x_j = \sum_j \left(\sum_i y_i a_{ij} \right) x_j = \sum_{ij} y_i a_{ij} x_j \\ &= \sum_i \left(\sum_j a_{ij} x_j \right) y_i = \sum_i (b_i - w_i) y_i \leq \sum_i b_i y_i,\end{aligned}$$

The inequalities come from the fact that

$$\begin{aligned}x_j z_j &\geq 0, & \text{for all } j, \\ w_i y_i &\geq 0, & \text{for all } i.\end{aligned}$$

By Strong Duality Theorem, the inequalities are equalities at optimality.