15.053 February 14, 2002

- Putting Linear Programs into standard form
- Introduction to Simplex Algorithm

- Note: this presentation is designed with animation to be viewed as a slide show.

Linear Programs in Standard Form

1. Non-negativity constraints for all variables.
2. All remaining constraints are expressed as equality constraints.
3. The right hand side vector, *b*, is non-negative.

An LP not in Standard Form

maximize $3x_1 + 2x_2 - x_3 + x_4$

$x_1 + 2x_2 + x_3 - x_4 \leq 5$; — not equality

$-2x_1 - 4x_2 + x_3 + x_4 \leq -1$; — not equality

$x_1 \geq 0, x_2 \geq 0$ — x_3 may be negative

Converting Inequalities into Equalities Plus Non-negatives

Before	After
$x_1 + 2x_2 + x_3 - x_4 \leq 5$	$x_1 + 2x_2 + x_3 - x_4 + s = 5$
	$s \geq 0$

s is called a ***slack variable***, which measures the amount of "unused resource."

Note that $s = 5 - x_1 - 2x_2 - x_3 + x_4$.

To covert a "≤" constraint to an equality, add a slack variable.

Converting "≥" constraints

- Consider the inequality $-2x_1 - 4x_2 + x_3 + x_4 \leq -1$;
- Step 1. Eliminate the negative RHS

  $2x_1 + 4x_2 - x_3 - x_4 \geq 1$
- Step 2. Convert to an equality

  $2x_1 + 4x_2 - x_3 - x_4 - s = 1$

  $s \geq 0$
- The variable added will be called a "*surplus variable.*"

To covert a "≥" constraint to an equality, subtract a surplus variable.

More Transformations

How can one convert a maximization problem to a minimization problem?

Example: Maximize $3W + 2P$

Subject to "constraints"

Has the same optimum solution(s) as

Minimize $-3W - 2P$

Subject to "constraints"

The Last Transformations (for now)

Transforming variables that may take on negative values.

maximize $3x_1 + 4x_2 + 5x_3$

subject to $2x_1 - 5x_2 + 2x_3 = 17$

other constraints

$x_1 \leq 0$, x_2 is unconstrained in sign, $x_3 \geq 0$

Transforming x_1: replace x_1 by $y_1 = -x_1$; $y_1 \geq 0$.

max $-3y_1 + 4x_2 + 5x_3$

$-2y_1 - 5x_2 + 2x_3 = 17$

$y_1 \geq 0$, x_2 is unconstrained in sign, $x_3 \geq 0$

One can recover x_1 from y_1.

Transforming variables that may take on negative values.

$\max -3y_1 + 4x_2 + 5x_3$
$-2y_1 - 5x_2 + 2x_3 = 17$
$y_1 \geq 0$, x_2 is unconstrained in sign, $x_3 \geq 0$

Transforming x_2: replace x_2 by $x_2 = y_3 - y_2$; $y_2 \geq 0, y_3 \geq 0$.

$\max -3y_1 + 4(y_3 - y_2) + 5x_3$
$-2y_1 - 5y_3 + 5y_2 + 2x_3 = 17$
all vars ≥ 0

One can recover x_2 from y_2, y_3.

Another Example

- Exercise: transform the following to standard form (maximization):

Minimize $x_1 + 3x_2$
Subject to $2x_1 + 5x_2 \leq 12$
$x_1 + x_2 \geq 1$
$x_1 \geq 0$

Perform the transformation with your partner

Preview of the Simplex Algorithm

<u>maximize</u> $z = -3x_1 + 2x_2$
subject to $-3x_1 + 3x_2 + x_3 = 6$
$-4x_1 + 2x_2 + x_4 = 2$
$x_1, x_2, x_3, x_4 \geq 0$

-z	x_1	x_2	x_3	x_4		
1	-3	2	0	0	=	0
0	-3	3	1	0	=	6
0	-4	2	0	1	=	2

LP Canonical Form = LP Standard Form + Jordan Canonical Form

-z	x_1	x_2	x_3	x_4		
1	-3	2	0	0	=	0
0	-3	3	1	0	=	6
0	-4	2	0	1	=	2

z is not a decision variable

The basic variables are x_3 and x_4.

The non-basic variables are x_1 and x_2.

The basic feasible solution is $x_1 = 0, x_2 = 0, x_3 = 6, x_4 = 2$

For each constraint there is a basic variable

-z	x_1	x_2	x_3	x_4			
1	-3	2	0	0	=	0	
0	-3	3	1	0	=	6	Constraint 1
0	-4	2	0	1	=	2	Constraint 2

Constraint 1: basic variable is x_3

Constraint 2: basic variable is x_4

The basis consists of variables x_3 and x_4

Optimality Conditions Preview

Note:
$z = -3x_1 + 2x_2$

-z	x_1	x_2	x_3	x_4		
1	-3	2	0	0	=	0
0	-3	3	1	0	=	6
0	-4	2	0	1	=	2

Obvious Fact: If one can improve the current basic feasible solution x, then x is not optimal.

Idea: assign a small value to just one of the non-basic variables, and then adjust the basic variables.

The current basic feasible solution (bfs) is not optimal!

$-z$	x_1	x_2	x_3	x_4		
1	-3	2	0	0	=	0
0	-3	3	1	0	=	6
0	-4	2	0	1	=	2

If there is a positive coefficient in the z row, the basis is not optimal**

Recall: $z = -3x_1 + 2x_2$

Increase x_2 to $\Delta > 0$. Let x_1 stay at 0.

What happens to x_3, x_4 and z? .

$x_3 = 6 - 3\Delta.$
$x_4 = 2 - 2\Delta.$
$z = 2\Delta.$

Optimality Conditions
(note that the data is different here)

$-z$	x_1	x_2	x_3	x_4		
1	-2	-4	0	0	=	-8
0	-3	3	1	0	=	6
0	-4	2	0	1	=	2

Important Fact. **If there is no positive coefficient in the z row, the basic feasible solution is optimal!**

$z = -2x_1 - 4x_2 + 8.$

Therefore $z \le 8$ for all feasible solutions.

But z = 8 in the current basic feasible solution

This basic feasible solution is optimal!

Let $x_2 = \Delta$. How large can Δ be?
What is the solution after changing x_2?

$-z$	x_1	x_2	x_3	x_4		
1	-3	2	0	0	=	0
0	-3	3	1	0	=	6
0	-4	2	0	1	=	2

$x_1 = 0$
$x_2 = 1$
$x_3 = 3$
$x_4 = 0$
$z = 2$

What is the value of Δ that maximizes z, but leaves a feasible solution?

$\Delta = 1$.

Fact. The resulting solution is a basic feasible solution for a different basis.

Pivoting to obtain a better solution

New Solution: basic variables are x_2 and x_3. Nonbasics: x_1 and x_4.

$-z$	x_1	x_2	x_3	x_4		
1	1	0	0	-1	=	-2
0	3	0	1	-1.5	=	3
0	-2	1	0	.5	=	1

$x_1 = 0$
$x_2 = 1$
$x_3 = 3$
$x_4 = 0$
$z = 2$

If we pivot on the coefficient 2, we obtain the new basic feasible solution.

Summary of Simplex Algorithm

- **Start in canonical form with a basic feasible solution**

1. **Check for optimality conditions**
2. **If not optimal, determine a non-basic variable that should be made positive**
3. **Increase that non-basic variable, and perform a pivot, obtaining a new bfs**
4. **Continue until optimal (or unbounded).**

OK. Let's iterate again. $z = x_1 - x_2 + 2$

$-z$	x_1	x_2	x_3	x_4		
1	1	0	0	-1	=	-2
0	3	0	1	-1.5	=	3
0	-2	1	0	.5	=	1

$x_1 = \Delta$
$x_2 = 1 + 2\Delta$
$x_3 = 3 - 3\Delta.$
$x_4 = 0$
$z = 2 + \Delta$

The cost coefficient of x_1 is positive.

Set $x_1 = \Delta$ and $x_4 = 0$.

How large can Δ be?

A Digression: What if we had a problem in which Δ could increase to infinity?

$z = x_1 - x_2 + 2$

-z	x_1	x_2	x_3	x_4		
1	1	0	0	-1	=	-2
0	-3	0	1	-1.5	=	3
0	-2	1	0	.5	=	1

$x_1 = \Delta$
$x_2 = 1 + 2\Delta$
$x_3 = 3 + 3\Delta.$
$x_4 = 0$
$z = 2 + \Delta$

Suppose we change the 3 to a –3.
Set $x_1 = \Delta$ and $x_4 = 0$.
How large can Δ be?

If the non-cost coefficients in the entering column are ≤ 0, then the solution is ***unbounded***

End Digression: Perform another pivot

-z	x_1	x_2	x_3	x_4		
1	0	0	-1/3	-1/2	=	-3
0	1	0	1/3	-1/2	=	1
0	0	1	2/3	1/2	=	3

$x_1 = 1$
$x_2 = 3$
$x_3 = 0$
$x_4 = 0$
$z = 3$

What is the largest value of Δ ? Δ = 1

Variable x_1 becomes basis, x_3 becomes nonbasic.

Pivot on the coefficient with a 3.

Check for optimality

$z = -x_1/3 - x_2/2 + 3$

-z	x_1	x_2	x_3	x_4		
1	0	0	-1/3	-1/2	=	-3
0	1	0	1/3	-1/2	=	1
0	0	1	2/3	1/2	=	3

$x_1 = 1$
$x_2 = 3$
$x_3 = 0$
$x_4 = 0$
$z = 3$

There is no positive coefficient in the z-row.

The current basic feasible solution is optimal!

Summary of Simplex Algorithm Again

- Start in canonical form with a basic feasible solution

1. Check for optimality conditions
   - Is there a positive coefficient in the cost row?
2. If not optimal, determine a non-basic variable that should be made positive
   - Choose a variable with a positive coef. in the cost row.
3. increase that non-basic variable, and perform a pivot, obtaining a new bfs
   - We will review this step, and show a shortcut
4. Continue until optimal (or unbounded).

Performing a "Pivot". Towards a shortcut.

$z = 2x_1 + 3$

-z	x_1	x_2	x_3	x_4		
1	2	0	0	0	=	-3
0	3	0	1	0	=	7
0	-2	1	0	0	=	1
0	2	0	0	1	=	5

Exercise: to do with your partner.

$x_1 = \Delta$
$x_2 = 1 + 2\Delta$
$x_3 = 7 - 3\Delta.$
$x_4 = 5 - 2\Delta$
$z = 3 + \Delta$

1. Determine how large Δ can be.
2. Determine the next solution.
3. Determine what coefficient should be pivoted on.
4. See if there is a shortcut for finding the coefficient.

More on performing a pivot

- To determine the column to pivot on, select a variable with a positive cost coefficient
- To determine a row to pivot on, select a coefficient according to a minimum ratio rule
- Carry out a pivot as one does in solving a system of equations.

Next Lecture

- Review of the simplex algorithm
- Formalizing the simplex algorithm
- How to find an initial basic feasible solution, if one exists
- A proof that the simplex algorithm is finite (assuming non-degeneracy)