ORF 522: Lecture 3

Linear Programming: Chapter 3
Degeneracy

Robert J. Vanderbei

September 20, 2012

Slides last edited on September 19, 2012

Operations Research and Financial Engineering, Princeton University
http://www.princeton.edu/∼rvdb

Degeneracy

Definitions.

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

Example:

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

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.

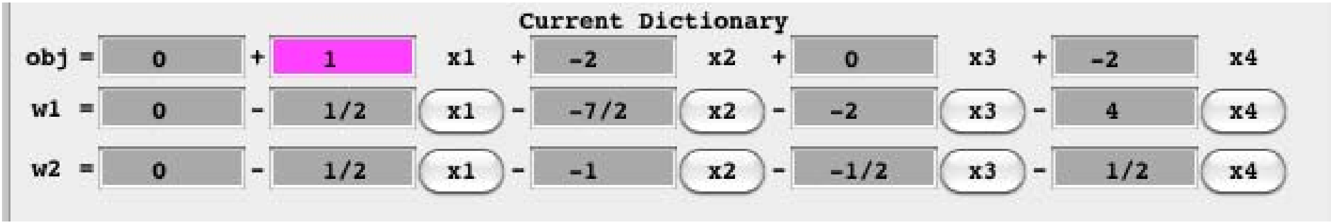
Hope Fades

An example that cycles using the following pivot rules:

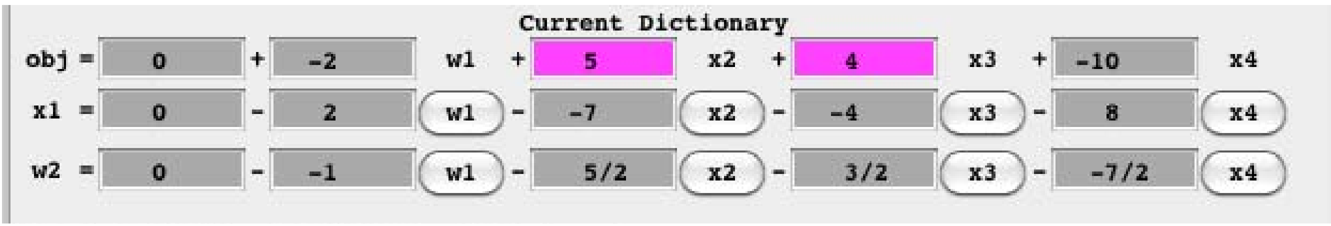
- entering variable: largest-coefficient rule.
- leaving variable: smallest-index rule.

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

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



$x_1 \Leftrightarrow w_1$:



$x_2 \Leftrightarrow w_2$:

$x_3 \Leftrightarrow x_1$:

$x_4 \Leftrightarrow x_2$:

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*:

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

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.

Theoretical Results

Cycling Theorem. If the simplex method fails to terminate, then it must cycle.

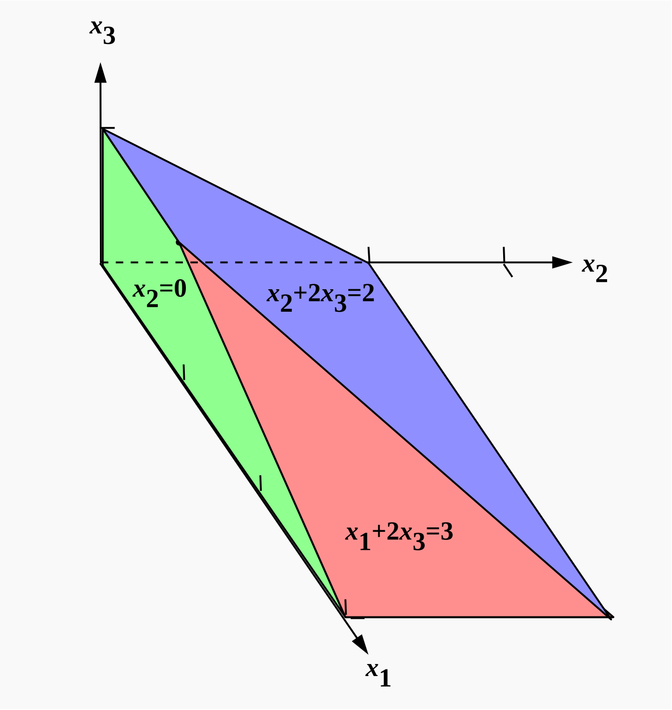
Why?

Fundamental Theorem of Linear Programming. For an arbitrary linear program in standard form, the following statements are true:

1. If there is no optimal solution, then the problem is either infeasible or unbounded.
2. If a feasible solution exists, then a basic feasible solution exists.
3. If an optimal solution exists, then a basic optimal solution exists.

Geometry

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



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