Optimization Methods in Management Science / Operations Research 15.053/058

Degeneracy in Linear Programming

Degeneracy in Linear Programming

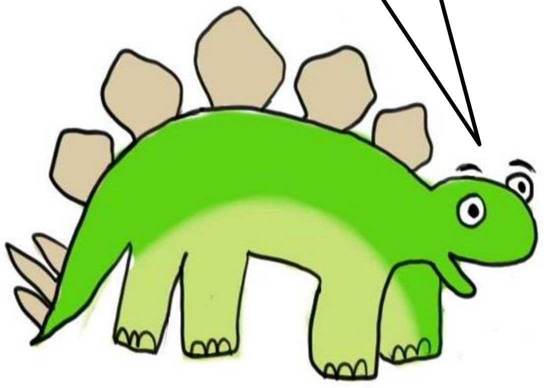
Degeneracy? Students at MIT shouldn't learn about degeneracy. And I heard that 15.053 students have already studied convicts' sex.

Sorry, Stan. But the topic is just as interesting. It's about degeneracy in Linear Programming.

I heard that today's tutorial is all about Ellen DeGeneres

Photo removed due to copyright restrictions.

Photo removed due to copyright restrictions.



Stan

Ellen DeGeneres

Pat Robertson

Actually, they studied convex sets.

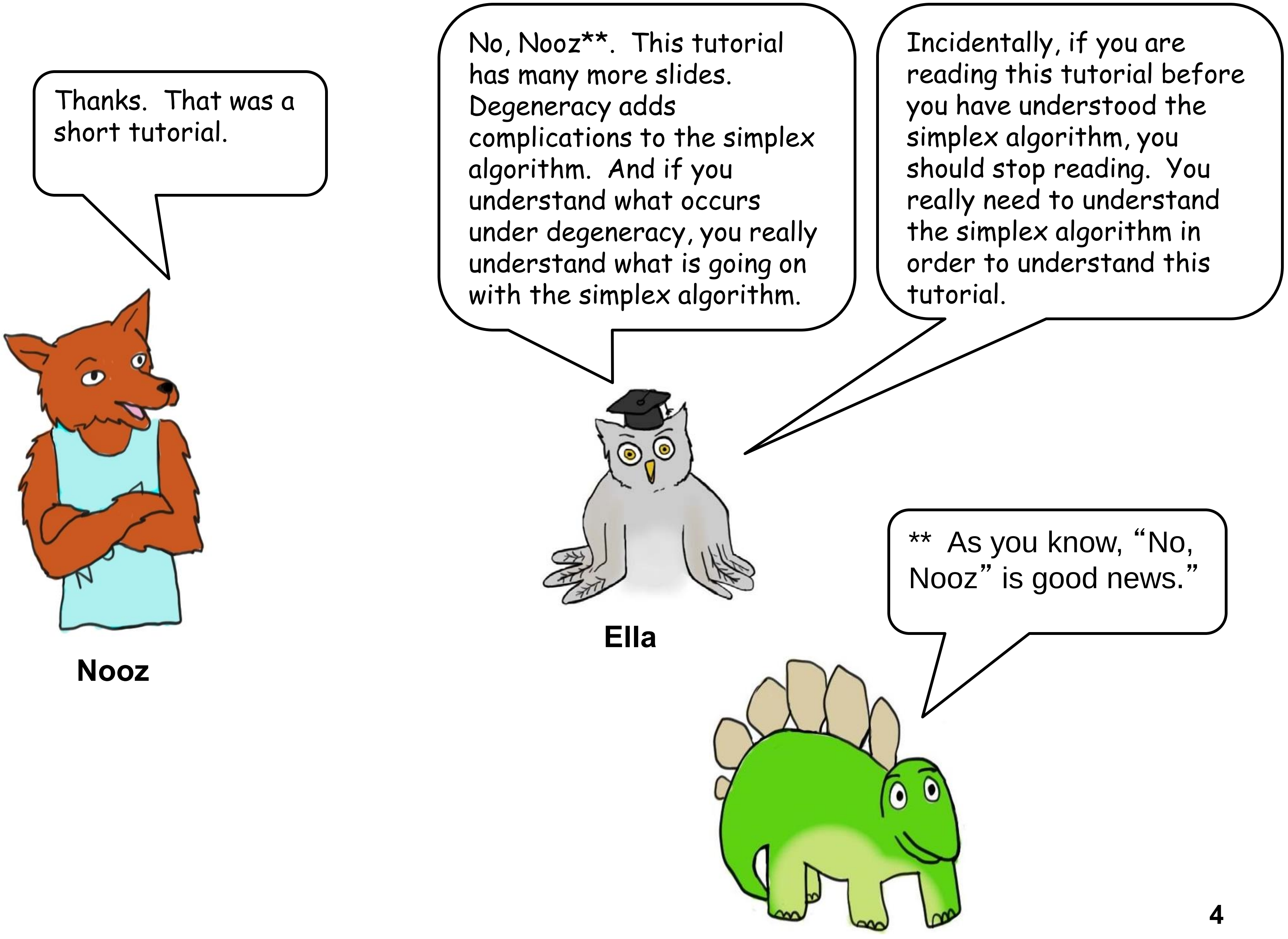
What is degeneracy?

- **As you know, the simplex algorithm starts at a corner point and moves to an adjacent corner point by increasing the value of a non-basic variable x_s with a positive cost coefficient.**
- **Typically, the entering variable x_s does increase in value, and the objective value z improves. But it is possible that x_s does not increase at all. This situation can occur when one of the RHS coefficients is 0.**
- **In this case, the objective value and solution does not change, but there is an exiting variable. This situation is called degeneracy.**

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

A basic feasible solution is called ***degenerate*** if one of its RHS coefficients (excluding the objective value) is 0.

This bfs is degenerate.


Thanks. That was a short tutorial.
No, Nooz**. This tutorial has many more slides. Degeneracy adds complications to the simplex algorithm. And if you understand what occurs under degeneracy, you really understand what is going on with the simplex algorithm.
Incidentally, if you are reading this tutorial before you have understood the simplex algorithm, you should stop reading. You really need to understand the simplex algorithm in order to understand this tutorial.
** As you know, "No, Nooz" is good news."
Nooz
Ella

Degeneracy and Basic Feasible Solutions

- We may think that every two distinct bases lead to two different solutions. This would be true if there was no degeneracy. But with degeneracy, we can have two different bases, and the same feasible solution.

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

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

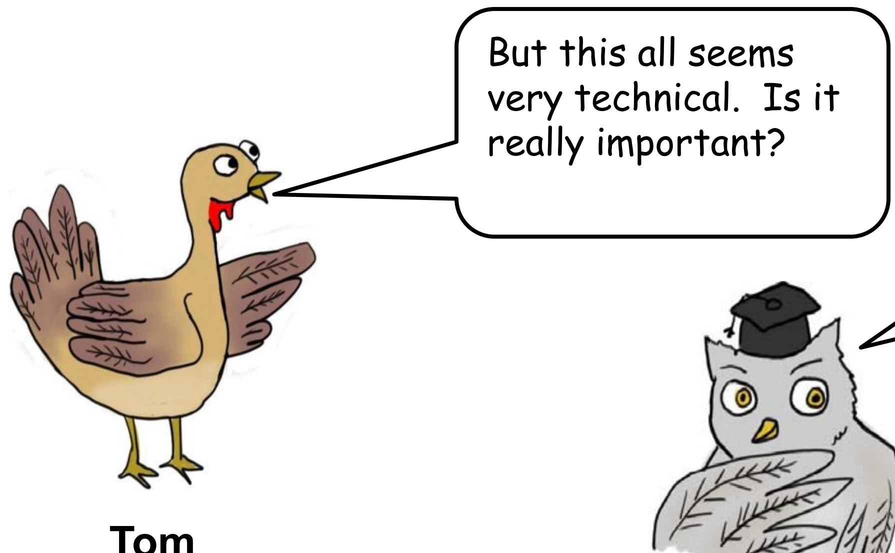
We now pivot on the "2" in Constraint 2 and obtain a second tableau.

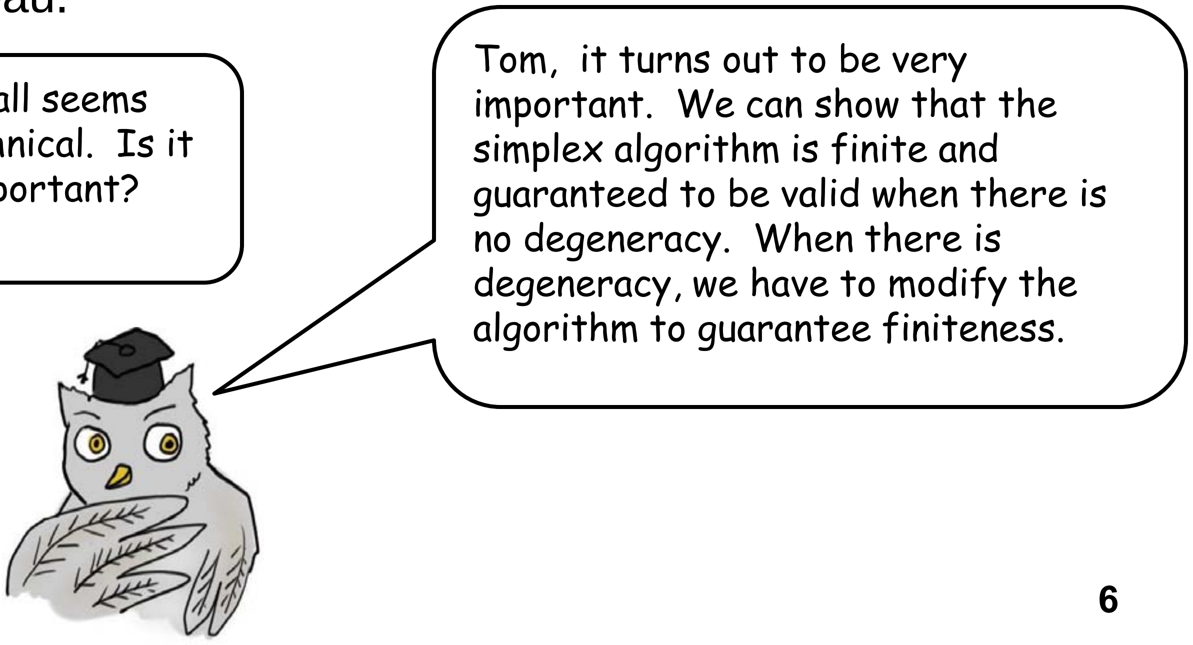
Both tableaus correspond to the same feasible solution with $z = 2$, $x_1 = x_2 = x_4 = 0$; $x_3 = 6$. But the basic variables and the coefficients of the two tableaus are different.

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

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

Not only are the two tableaus different, but the second tableau satisfies the optimality conditions. This means that the first basic feasible solution was optimal, even though we were not aware that this solution was optimal when we looked at the first tableau.




Tom, it turns out to be very important. We can show that the simplex algorithm is finite and guaranteed to be valid when there is no degeneracy. When there is degeneracy, we have to modify the algorithm to guarantee finiteness.

I'm now going to reveal a secret. The definition of degeneracy given here is slightly different than the one given in the lecture on geometry. To the right is a picture of what I said in that lecture.

It wasn't that I was misinforming you. There just wasn't a better way of describing the situation during that lecture.

When a corner point is the solution of two different sets of equality constraints, then this is called degeneracy. This will turn out to be important for the simplex algorithm.

From Lecture 3.

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

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

Whenever two different bases correspond to the same basic solution, then some basic variable must be 0. (If variable x_s is pivoted in, its value will change unless the RHS is 0.)

But it is possible for the RHS to be 0, and for there still to be only one basis for that solution. The tableau to my right illustrates this point.

-z	x_1	x_2	x_3		
1	-2	0	0	=	-2
0	-3/2	0	1	=	6
0	0	1	0	=	0

We say that we say that a basis is degenerate if a basic variable is 0. This is far simpler than the alternative definition in which one needs to check if there are two bases corresponding to the same solution..

The Finiteness of the Simplex Algorithm when there is no degeneracy

Recall that the simplex algorithm tries to increase a non-basic variable x_s. If there is no degeneracy, then x_s will be positive after the pivot, and the objective value will improve.

Recall also that each solution produced by the simplex algorithm is a basic feasible solution with m basic variables, where m is the number of constraints. There are a finite number of ways of choosing the basic variables. (An upper bound is $n! / (n-m)!\, m!$, which is the number of ways of selecting m basic variables out of n.)

So, the simplex algorithm moves from bfs to bfs. And it never repeats a bfs because the objective is constantly improving. This shows that the simplex method is finite, so long as there is no degeneracy.

Cycling

- If a sequence of pivots starting from some basic feasible solution ends up at the exact same basic feasible solution, then we refer to this as "cycling." If the simplex method cycles, it can cycle forever.

- Klee and Minty [1972] gave an example in which the simplex algorithm really does cycle. Here is their example, with the pivot elements outlined.

-z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	3/4	-20	1/2	-6	0	0	0	-3
0	$\boxed{1/4}$	-8	-1	9	1	0	0	0
0	1/2	-12	- 1/2	3	0	1	0	0
0	0	0	1	0	0	0	1	1

Initial tableau

-z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	0	4	3 1/2	-33	-3	0	0	-3
0	1	-32	-4	36	4	0	0	0
0	0	$\boxed{4}$	1 1/2	-15	-2	1	0	0
0	0	0	1	0	0	0	1	1

After 1 pivot

Cycling Example Continued

After 2 pivots

$-z$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	0	0	2	-18	-1	-1	0	-3
0	1	0	$\boxed{8}$	-84	-12	8	0	0
0	0	1	3/8	-3 3/4	- 1/2	1/4	0	0
0	0	0	1	0	0	0	1	1

After 3 pivots

$-z$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	- 1/4	0	0	3	2	-3	0	-3
0	1/8	0	1	-10 1/2	-1 1/2	1	0	0
0	- 3/64	1	0	$\boxed{3/16}$	1/16	- 1/8	0	0
0	- 1/8	0	0	10 1/2	1 1/2	-1	1	1

After 4 pivots

$-z$	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	1/2	-16	0	0	1	-1	0	-3
0	-2 1/2	56	1	0	$\boxed{2}$	-6	0	0
0	- 1/4	5 1/3	0	1	1/3	- 2/3	0	0
0	2 1/2	-56	0	0	-2	6	1	1

Cycling Example Continued

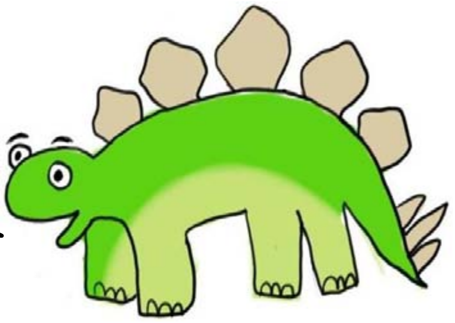
-z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	1 3/4	-44	- 1/2	0	0	2	0	-3
0	-1 1/4	28	1/2	0	1	-3	0	0
0	1/6	-4	- 1/6	1	0	$\boxed{1/3}$	0	0
0	0	0	1	0	0	0	1	1

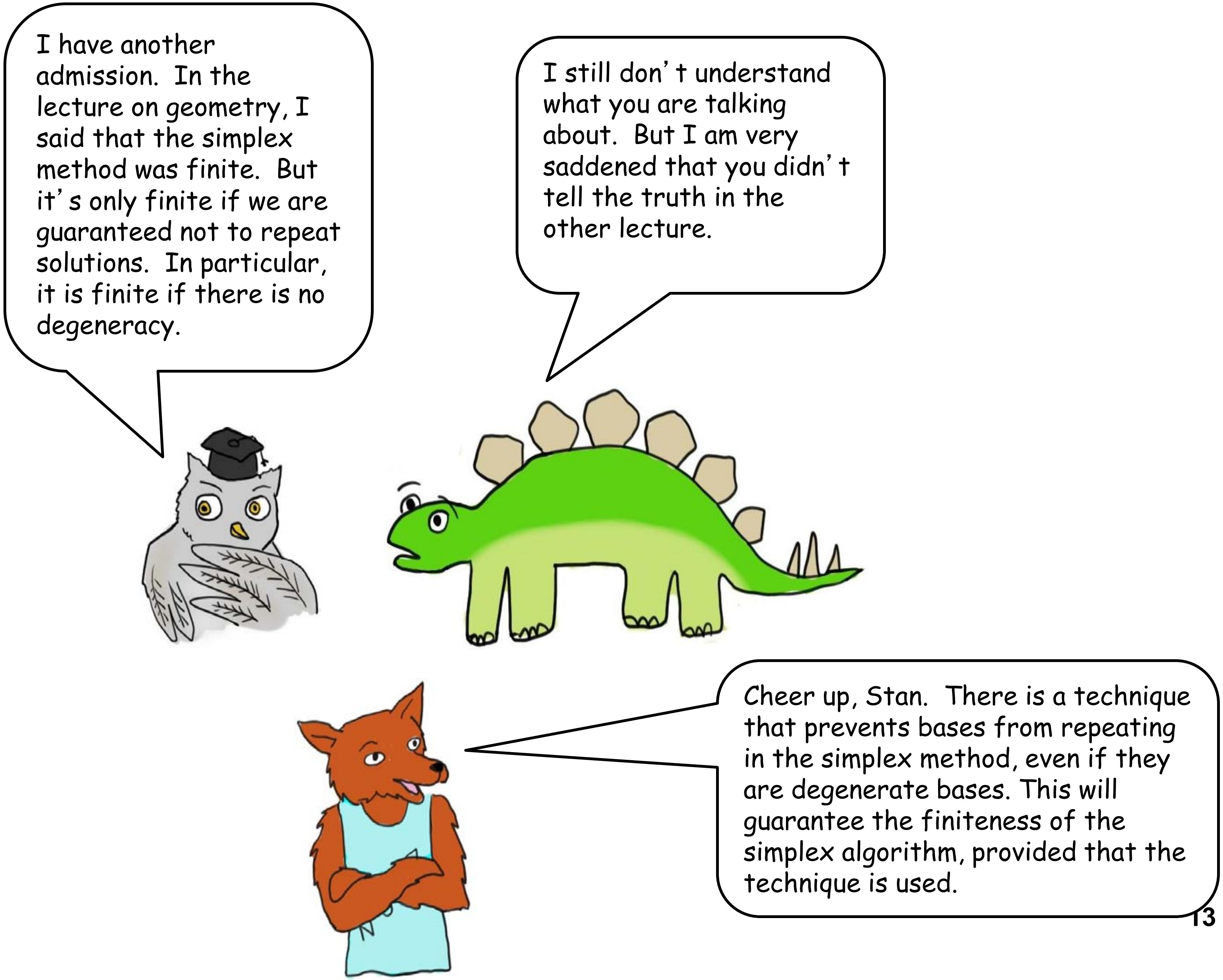
After 5 pivots

-z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	RHS
1	3/4	-20	1/2	-6	0	0	0	-3
0	1/4	-8	-1	9	1	0	0	0
0	1/2	-12	- 1/2	3	0	1	0	0
0	0	0	1	0	0	0	1	1

After 6 pivots

And Klee and Minty said "The first shall be last and the last shall be first", and they saw that their example of cycling was good.




I have another admission. In the lecture on geometry, I said that the simplex method was finite. But it's only finite if we are guaranteed not to repeat solutions. In particular, it is finite if there is no degeneracy.
I still don't understand what you are talking about. But I am very saddened that you didn't tell the truth in the other lecture.
Cheer up, Stan. There is a technique that prevents bases from repeating in the simplex method, even if they are degenerate bases. This will guarantee the finiteness of the simplex algorithm, provided that the technique is used.

Is the simplex method finite?

So, how do we know that the simplex method will terminate if there is degeneracy?
There are several approaches to guaranteeing that the simplex method will be finite, including one developed by Professors Magnanti and Orlin. And there is the perturbation technique that entirely avoids degeneracy. But we're going to show you Bland's rule, developed by Bob Bland. It's the simplest rule to guarantee finiteness of the simplex method.
NOOZ

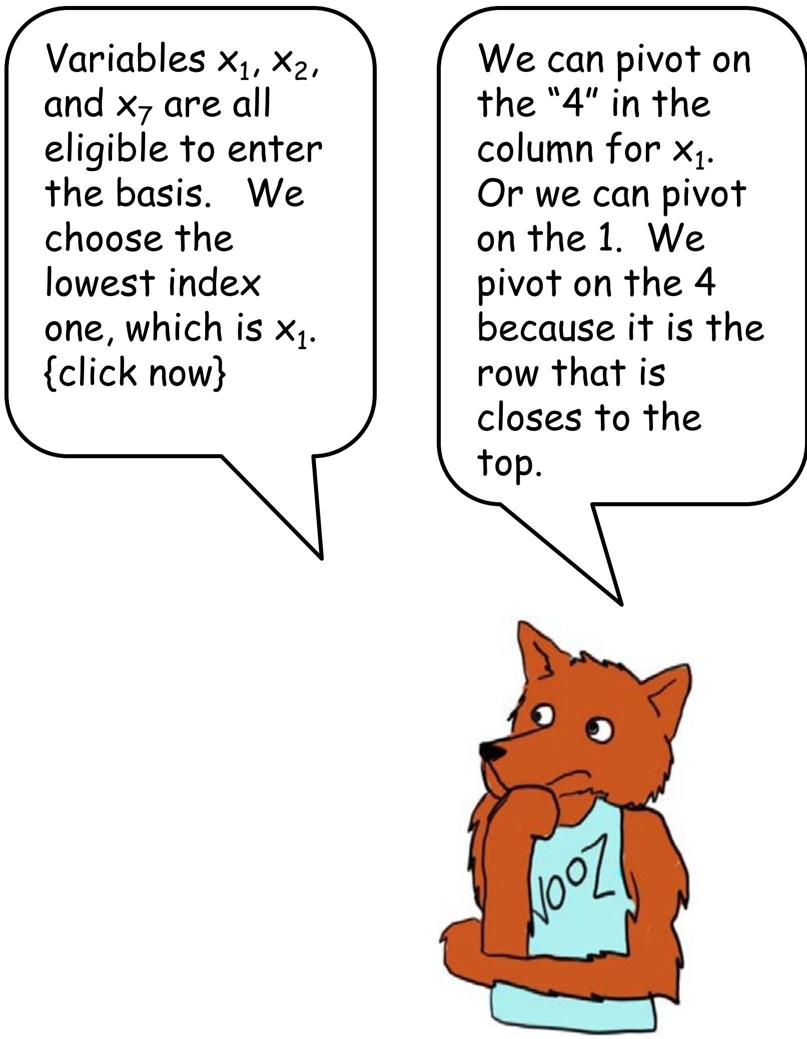
Photo of Bob Bland removed due to copyright restrictions.

Bob Bland

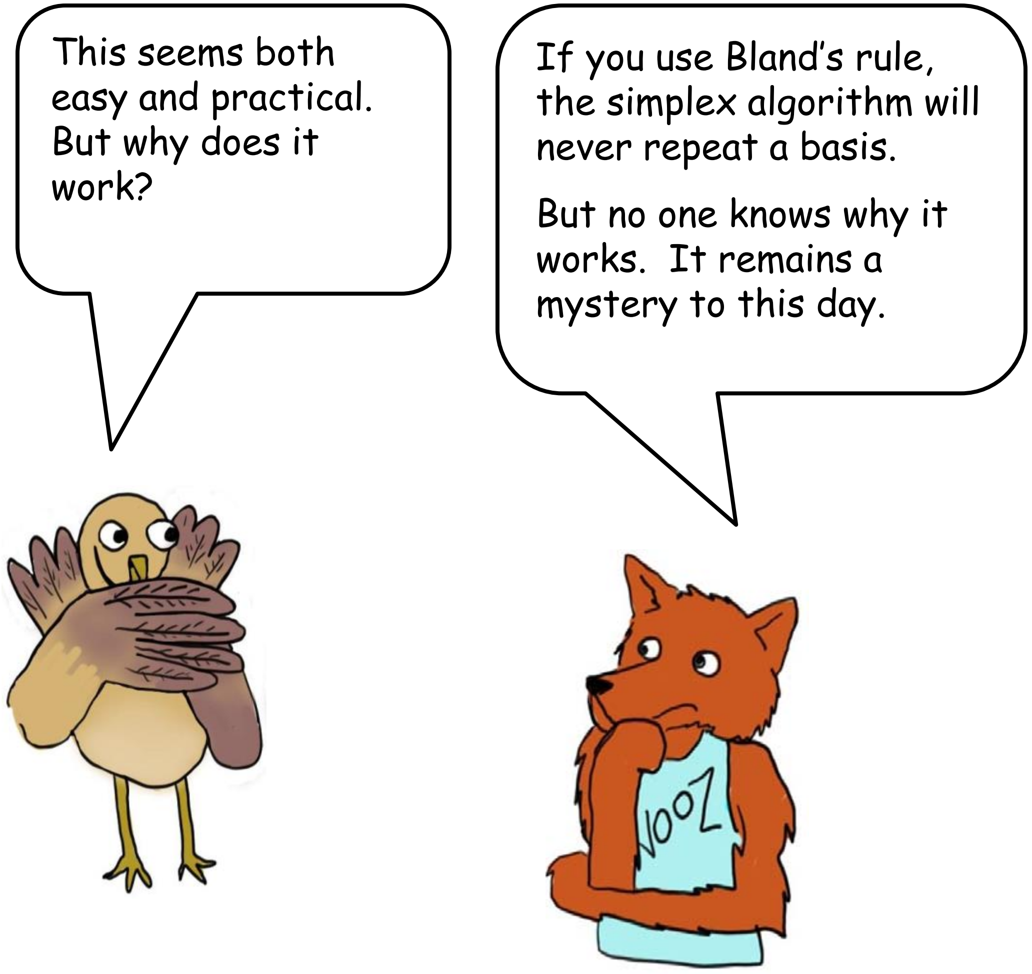
Bland's Rule.

1. The entering variable should be the lowest index variable with positive reduced cost.
2. The leaving variable (in case of a tie in the min ratio test) should be the lowest index row. (It is the row closest to the top, regardless of the leaving variable.)

-z	x_1	x_2	x_3	x_4	x_5	x_6		
1	1	2	0	7	0	0	=	-20
0	3	3	0	2	1	0	=	6
0	$\textcircled{4}$	2	0	-1	0	1	=	0
0	1	3	1	1	0	0	=	0


Variables x_1, x_2, and x_7 are all eligible to enter the basis. We choose the lowest index one, which is x_1. {click now}
We can pivot on the "4" in the column for x_1. Or we can pivot on the 1. We pivot on the 4 because it is the row that is closes to the top.
NOOZ

More on Bland's rule


This seems both easy and practical. But why does it work?
If you use Bland's rule, the simplex algorithm will never repeat a basis.

But no one knows why it works. It remains a mystery to this day.
NOOZ

Nooz is not being serious. I proved that using the rule will prevent bases from repeating. I know why it works, and so does anyone who has read my proof.

But the proof is too complicated to present here. So, please take our word for it.

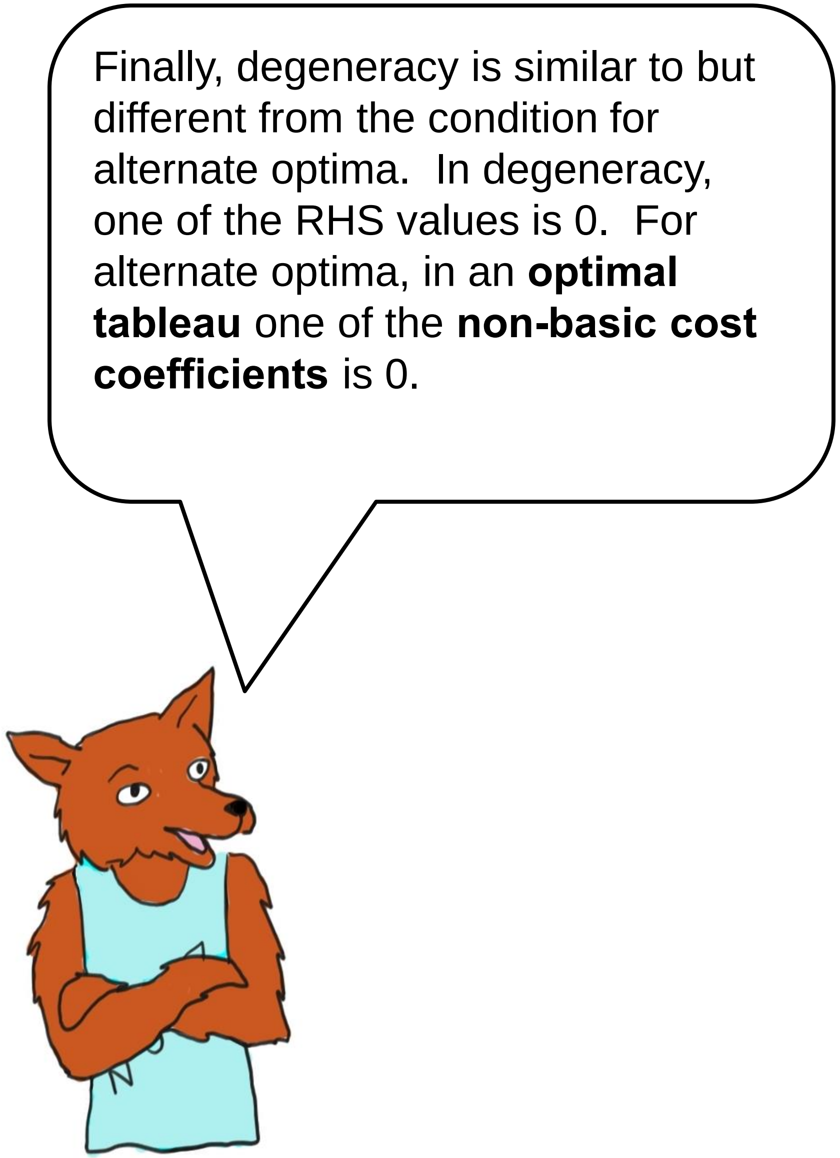
Photo of Bob Bland removed due to copyright restrictions.

Bob Bland

Degeneracy and the Simplex Algorithm

The simplex method without degeneracy	The simplex method with degeneracy
The solution changes after each pivot. The objective value strictly improves after a pivot.	The solution may stay the same after a pivot. The objective value may stay the same.
The simplex method is guaranteed to be finite.	The simplex method may cycle and be infinite. But it becomes finite if we use the Bland's rule or several other approaches.
Two different tableaus in canonical form give two different solutions.	It is possible that there are two different sets of basic variables that give the same solution.

Degeneracy vs. Alternative Optima


Finally, degeneracy is similar to but different from the condition for alternate optima. In degeneracy, one of the RHS values is 0. For alternate optima, in an **optimal tableau** one of the **non-basic cost coefficients** is 0.

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

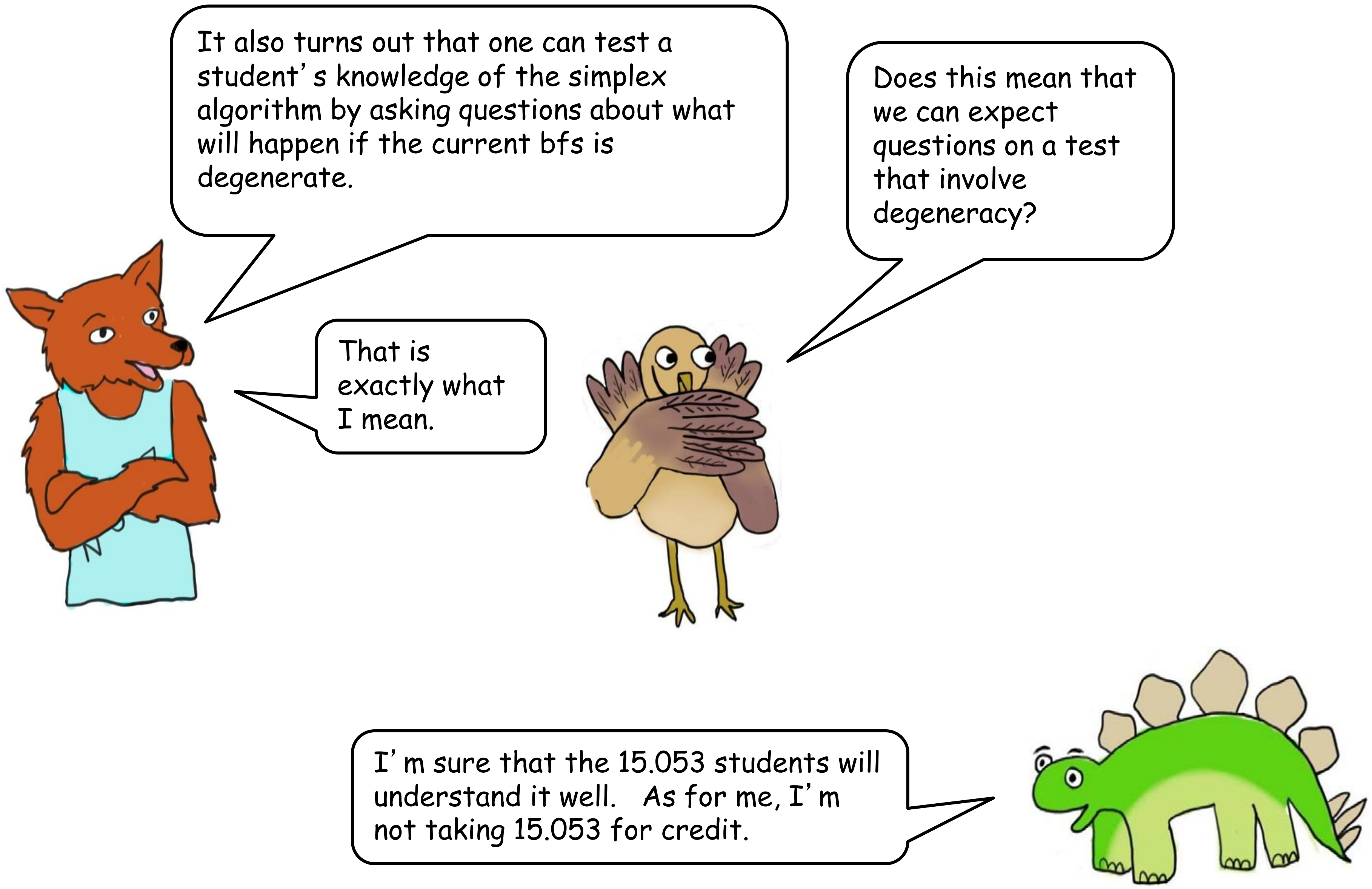
The optimal solution is: z = 8, $x_1 = x_2 = 0$, $x_3 = 6$, and $x_4 = 2$. x_1 is nonbasic, and its cost coefficient is 0. Increasing x_1 (and adjusting x_3 and x_4) does not change z, and so the solution value remains optimal.

Summary

- Degeneracy is important because we want the simplex method to be finite, and the generic simplex method is not finite if bases are permitted to be degenerate.

- In principle, cycling can occur if there is degeneracy. In practice, cycling does not arise, but no one really knows why not. Perhaps it does occur, but people assume that the simplex algorithm is just taking too long for some other reason, and they never discover the cycling.

- Researchers have developed several different approaches to ensure the finiteness of the simplex method, even if the bases can be degenerate. Bob Bland developed a very simple rule that prevents cycling.


It also turns out that one can test a student's knowledge of the simplex algorithm by asking questions about what will happen if the current bfs is degenerate.
Does this mean that we can expect questions on a test that involve degeneracy?
That is exactly what I mean.
I'm sure that the 15.053 students will understand it well. As for me, I'm not taking 15.053 for credit.

Last Slide

And that is all for our tutorial on degeneracy. We hope to see you again for our next tutorial.

We thank Stan, Tom, Ella, and Nooz for sharing all of their insights on degeneracy. I also want to thank Ellen DeGeneres for making a cameo appearance.

Amit

Mita