Linear Programming & the Simplex Algorithm

Part III – The Simplex Algorithm

Last Revision Oct 16, 2007

P&S Chapter 2

Recall $$\boxed{\bar{\mathbf{c}}_{\mathbf{j}} = \mathbf{c}_{\mathbf{j}} - \sum_{\mathbf{i}=1}^{\mathbf{m}} \mathbf{x}_{\mathbf{i},\mathbf{j}} \mathbf{c}_{\mathbf{B}(\mathbf{i})} = \mathbf{c}_{\mathbf{j}} - \mathbf{z}_{\mathbf{j}}}$$

c_j is the cost of one unit of x_j while z_j is the cost of the equivalent of one unit of x_j expressed using elements of basis $\mathcal{B}$.

$\bar{c}_j$ is therefore cost of bringing in one unit of x_j to replace the equivalent amount of basis variables. This is the *relative cost* of column j vis-a-vis basis $\mathcal{B}$.

Since $\bar{c}_j$ are so useful we would like to keep them in the tableau.This is usually done by adding a row 0 to the tableau storing the $\bar{c}_j$.

Before seeing how to maintain this, note that *relative cost* associated with *basis* column j is

$$\bar{c}_j = c_j - \sum_{i=1}^{m} x_{i,j} c_{B(i)} = 0$$

Now write the original 0th row of the tableau as

$$0 = -z + c_1 x_1 + \cdots + c_n x_n.$$

Suppose j is a basis column with $B(i) = j$.
To make the jth entry in row 0 be 0 we need to multiply row i by $-c_{B(i)} = -c_j$ and subtract from row 0.

Do this for every basic j. Note that for all nonbasic j the jth column of the 0th row will then be exactly the relative cost

$$\bar{c}_j = c_j - \sum_{i=1}^{m} x_{i,j} c_{B(i)}$$

Note too that the LHS of the equation will have value

$$-z_0 = -\sum_{i=1}^{m} x_{i,0} c_{B(i)}$$

which is exactly the $cost$ of the current basic solution.

0th row is then

$$-z_0 = -z + \sum_{i=1}^{n} \bar{c}_j x_j$$

(where, if j is basic, $x_j = 0$).

The 0th row is then

$$-z_0 = -z + \sum_{i=1}^{n} \bar{c}_j x_j$$

(where, if j is basic, $x_j = 0$).

Consider the LP as having $n+1$ variables instead of n, with the $(n+1)$st variable corresponding to $-z$.

The same pivot rules should work for the 0th row . That is, when column j enters the basis using pivot $x_{l,j}$ we need to zero c_j by using the same rules that we used for the other rows. Recall that, after pivoting, $x'_{l,j} = 1$. This means that, after all of the other pivoting operations, we need to multiply row l by $\bar{c}_j$ and subtract it from row 0.

Note that this gives us what we want since

- $\bar{c}'_j = 0$
- If $q \neq j$ in basis, $\bar{c}'_q = \bar{c}_q = 0$
- $-z'_0 = -z_0 - x'_{l,0}\bar{c}_j$ (new cost)
- If q not in new basis then $\bar{c}'_q = c_q - \sum_{i=1}^{m} x'_{i,q} c_{B'(i)}$ (prove this)
- $-z' = -z$ (so don't need extra column for $-z$)

We start with 0th row of tableau reflecting

$$-z_0 = -z + \sum_{i=1}^{n} \bar{c}_j x_j$$

and this equation is maintained with every pivot. Recall that we can only pivot on column j if $\bar{c}_j < 0$, and that if $\forall j, \bar{c}_j \geq 0$, then solution is optimal.

We can read the $\bar{c}_j$ information off the top row of the tableau. So, tableaus permit us to choose feasible pivot columns, *and* allow us to implement pivots easily.. Assuming nondegeneracy of bases, this gives us

procedure simplex
begin
    opt:='no', unbounded:='no';
    (**comment:** when either becomes 'yes', algorithm terminates)
    while opt = 'no' **and** unbounded = 'no' **do**
        if $\bar{c}_j \geq 0$ for all j **then** opt:='yes'
          else begin
              choose any j such that $\bar{c}_j < 0$;
              if $x_{ij} \leq 0$ for all i **then** unbounded:= 'yes'
              else
                  find $\theta_0 = \min_{\substack{i \\ x_{ij} > 0}} \left(\frac{x_{i0}}{x_{ij}} \right) = \frac{x_{k0}}{x_{kj}}$
                  and pivot on x_{kj}
              end
end

Suppose the original LP has cost function $z = x_1 + x_2 + x_3 + x_4 + x_5$ and in tableau form is

	x_1	x_2	x_3	x_4	x_5
0	1	1	1	1	1
1	3	2	1	0	0
3	5	1	1	1	0
4	2	5	1	0	1

Now make basis $\mathcal{B} = \{A_3, A_4, A_5\}$. When doing this don't forget to zero columns 3, 4 and 5 in row 0.

		x_1	x_2	x_3	x_4	x_5
$-z=$	-6	-3	-3	0	0	0
$x_3=$	1	3	2	1	0	0
$x_4=$	2	2	-1	0	1	0
$x_5=$	3	-1	3	0	0	1

The non-basis columns have $\bar{c}_1 = \bar{c}_2 = -3$ so they are both eligible to enter basis. We choose column 2. Pivot will then be on $x_{1,2}$ with $\theta_0 = 1/2$. After pivoting, basis will be $\mathcal{B}' = \{A_2, A_4, A_5\}$.

		x_1	x_2	x_3	x_4	x_5
$-z=$	$-9/2$	$3/2$	0	$3/2$	0	0
$x_2=$	$1/2$	$3/2$	1	$1/2$	0	0
$x_4=$	$5/2$	$7/2$	0	$1/2$	1	0
$x_5=$	$3/2$	$-11/2$	0	$-3/2$	0	1

$$
\begin{array}{r|r|rrrrr|}
 & & x_1 & x_2 & x_3 & x_4 & x_5 \\
\hline
-z= & -9/2 & 3/2 & 0 & 3/2 & 0 & 0 \\
\hline
x_2= & 1/2 & 3/2 & 1 & 1/2 & 0 & 0 \\
x_4= & 5/2 & 7/2 & 0 & 1/2 & 1 & 0 \\
x_5= & 3/2 & -11/2 & 0 & -3/2 & 0 & 1 \\
\hline
\end{array}
$$

We now have basis $\mathcal{B}' = \{A_2, A_4, A_5\}$. Note that the nonbasis columns both have $\bar{c}_j \geq 0$ so the BFS

$$(0, 1/2, 0, 5/2, 3/2)$$

with cost $z_0 = 1/2 + 5/2 + 3/2 = 9/2$, is optimal.

Column Selection

Q. The simplex algorithm allows choosing *any* non-basis column j with $\bar{c}_j < 0$ to move into the basis. In the case that more than one column has $\bar{c}_j < 0$ which column is *best* to choose?

A. No one knows. There are many possibilities.
Some common ones are

1. **Nonbasic gradient method:**
   Choose j with most negative c_j.
   Easy to implement.
   Might not give largest decrease in cost.

2. **Greatest increment method:**
   Choose j that results in largest decrease in cost.
   This is j that minimizes $\theta_0 \bar{c}_j$. Requires more work.

3. **All-variable gradient method**
   Choose j that minimizes

$$\frac{\bar{c}_j}{\sqrt{1 + \sum_{i=1}^{m} x_{i,j}^2}}$$

Q. For given column j how do we deal with ties in k? That is, suppose more than one row k has $x_{k,j} > 0$ and

$$\frac{x_{k0}}{x_{kj}} = \theta_0 = \min_{\substack{i \\ x_{ij}>0}} \left(\frac{x_{i0}}{x_{ij}} \right)$$

A. This can be a major issue. Note that if we pivot on $x_{k,j}$ with $x_{k,0} = 0$ then $\theta_0 = 0$ and we are not changing the cost at all. We are just moving from one basis to another of a degenerate BFS.

If we are not careful, then after a sequence of such non-improving "pivots" we can end up back at the original basis. This is called cycling and needs to be avoided.

A cycling example

Consider

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

Suppose we use the rule that

- Always select nonbasis column with most negative $\bar{c}_j$ to enter the basis.
- In case of a tie, always select basic variable with smallest subscript to leave basis.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	$-3/4$	$+20$	$-1/2$	$+6$	0	0	0
0	**1/4**	-8	-1	9	1	0	0
0	1/2	-12	$-1/2$	3	0	1	0
1	0	0	1	0	0	0	1

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	0	-4	$-7/2$	33	3	0	0
0	1	-32	-4	36	4	0	0
0	0	**4**	3/2	-15	-2	1	0
1	0	0	1	0	0	0	1

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	0	0	-2	18	1	1	0
0	1	0	**8**	-84	-12	8	0
0	0	1	3/8	$-15/4$	$-1/2$	1/4	0
1	0	0	1	0	0	0	1

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	1/4	0	0	-3	-2	3	0
0	1/8	0	1	$-21/2$	$-3/2$	1	0
0	$-3/64$	1	0	**3/16**	1/16	$-1/8$	0
1	$-1/8$	0	0	21/2	3/2	-1	1

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	$1/4$	0	0	-3	-2	3	0
0	$1/8$	0	1	$-21/2$	$-3/2$	1	0
0	$-3/64$	1	0	**$3/16$**	$1/16$	$-1/8$	0
1	$-1/8$	0	0	$21/2$	$3/2$	-1	1

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	$-1/2$	16	0	0	-1	1	0
0	$-5/2$	56	1	0	**2**	-6	0
0	$-1/4$	$16/3$	0	1	$1/3$	$-2/3$	0
1	$5/2$	-56	0	0	-2	6	1

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
3	$-7/4$	44	$1/2$	0	0	-2	0
0	$-5/4$	28	$1/2$	0	1	-3	0
0	$-1/6$	-4	$-1/6$	1	0	**$1/3$**	0
1	0	0	1	0	0	0	1

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

We will now see **Bland's anticycling rule** that guarantees that cycles won't occur.

Theorem:

Suppose in the simplex algorithm we choose the column to enter the basis by

$$j = \text{min}\{j : c_j - z_j < 0\}$$

(choose the lowest numbered favorable column), and the row by

$$B(i) = \text{min}\left\{B(i) : x_{i,j} > 0 \text{ and } \frac{x_{i,0}}{x_{ij}} \leq \frac{x_{k,0}}{x_{k,j}} \text{ for every } k \text{ with } x_{k,j} > 0\right\}$$

(choose in case of tie the lowest numbered column to leave the basis). Then the algorithm terminates after a finite number of pivots.

Note: See proof in P&S textbook.

The “Real” Simplex Algorithm

Q. Until now always assumed we were given a BFS. In reality this isn’t always provided, so how do we find one?

A. If we begin with $Ax \le b$

then slack variables form a BFS.

Otherwise, we can use the artificial variable or two-phase method. This works by adding m new artificial variables x_i^a, $i = 1, \ldots, m$ to the LP.

Starting BFS will be $x_i^a = b_i$.

Note” Might have to multiply some of the original equations by -1 *to ensure* $b \ge 0$.

$$\begin{array}{c|ccc|ccc} & x_1^a & \cdots & x_m^a & x_1 & \cdots & x_n \\ \hline & 1 & & \mathbf{0} & & & \\ b & & \ddots & & & A & \\ & \mathbf{0} & & 1 & & & \end{array} \qquad \begin{array}{ll} x_j \ge 0 & j = 1, \ldots, n \\ x_i^a \ge 0 & i = 1, \ldots, m \end{array}$$

$$\begin{array}{c|ccc|ccc} & x_1^a & \cdots & x_m^a & x_1 & \cdots & x_n \\ \hline & 1 & & \mathbf{0} & & & \\ b & & \ddots & & & A & \\ & \mathbf{0} & & 1 & & & \end{array} \qquad \begin{array}{ll} x_j \geq 0 & j = 1, \ldots, n \\ x_i^a \geq 0 & i = 1, \ldots, m \end{array}$$

Phase I. Minimize cost function $\xi = \sum_{i=1}^{m} x_i^a$

There are three possible outcomes:

1. $\xi = 0$ and all x_i^a have been driven out of the basis. Current basis is then a BFS of original problem and we can continue. (best case!).
   When we continue it is with Phase II, which uses original cost function.

2. Reach optimality with $\xi > 0$.
   Then there is no feasible solution to original problem and we stop.

3. $\xi = 0$ but some artificial variables remain at zero level in basis.

Case 3 (cont). $\xi = 0$ but some artificial variables remain at zero level.

In this case suppose ith basis column at end of Phase I belongs to artificial variable and $x_i^a = 0$.
Then "pivot" on *any* nonzero (not necessarily positive) $x_{i,j}$ where j corresponds to a non-artificial variable. This drives the artificial variable out of the basis. Note that since $x_i^a = 0$ we have $\theta_0 = 0$ so no infeasibility occurs and $\xi = 0$ does not change.

Also note that this might not be what we usually call a pivot, since it's possible that $\bar{c}_j \geq 0$ or $x_{i,j} < 0$. Since $\theta_0 = 0$, this does not matter.

Keep on doing this until we get a basis composed of only non-artificial variables. Only way that this can fail is if a row has zero values in the columns corresponding to *all* non-artificial variables. This implies that we've created a zero row in *original* matrix using elementary row operations, i.e, that the original matrix was not of rank m.

In this case deleting that row (it's useless) contributes nothing to original LP. So, delete it and continue with Phase I. In this way we must at some point end up at case 1.

Case 3 (cont). $\xi = 0$ but some artificial variables remain at zero level.

We just saw that in this case we can always drive artificial variables out of the basis and replace them with real variables unless at some point we've created a zero row in *original* matrix using elementary row operations, i.e, that the original matrix was not of rank m.

In this case deleting that row (it's useless) contributes nothing to original LP. So, delete it and continue with phase I. In this way we must at some point end up at case 1.

Note that if A was not of rank m then it is impossible to drive all of the artificial variables out of the basis (why?), so, if there is a feasible solution, we must end up at case 3.

Therefore, if there exists a feasible solution, the first phase of the two phase algorithm will always correctly find a BFS from which to start the second, *even if the original matrix was not of full rank.*

We have therefore just seen how to get rid of the original assumption that A is of full rank.

```
        procedure two-phase
        begin
          infeasible := 'no', redundant := 'no';
          (comment: Phase I may set these to 'yes')
Phase I:  introduce an artificial basis, x_i^a;
          call simplex with cost ξ = Σ x_i^a;
          if ξ_opt > 0 in Phase I then infeasible := 'yes'
            else begin
              if an artificial variable is in the basis and
                can not be driven out then redundant := 'yes',
                and omit the corresponding row;
Phase II:   call simplex with original cost
            end
        end
```

We now have the final version of simplex algorithm.

Note that all of our original assumptions are no longer necessary.

- If A is not of rank m we find out at the end of Phase I and throw out enough rows so that it becomes of full rank.

- If the original problem is infeasible, we find out at the end of Phase I.

- If the original problem is unbounded, we find out in Phase II.

In order to implement the two-phase method, we can have two row 0s; the first corresponding to the real cost and the second corresponding to the artificial variable cost. While pivoting during Phase I, we will maintain both of these rows.

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5	
$-z=$	0	0	0	0	1	1	1	1	1	row $0'$
$-\xi=$	0	1	1	1	0	0	0	0	0	row 0
	1	1	0	0	3	2	1	0	0	
	3	0	1	0	5	1	1	1	0	
	4	0	0	1	2	5	1	0	1	

Starting with the above, we subtract rows 1,2,3 from the ξ cost row to set the appropriate $\bar{c}_j = 0$.

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z=$	0	0	0	0	1	1	1	1	1
$-\xi=$	-8	0	0	0	-10	-8	-3	-1	-1
$x_1^a=$	1	1	0	0	**3**	2	1	0	0
$x_2^a=$	3	0	1	0	5	1	1	1	0
$x_3^a=$	4	0	0	1	2	5	1	0	1

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z=$	0	0	0	0	1	1	1	1	1
$-\xi=$	-8	0	0	0	-10	-8	-3	-1	-1
$x_1^a=$	1	1	0	0	$\mathbf{3}$	2	1	0	0
$x_2^a=$	3	0	1	0	5	1	1	1	0
$x_3^a=$	4	0	0	1	2	5	1	0	1

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z=$	$-1/3$	$-1/3$	0	0	0	$1/3$	$2/3$	1	1
$-\xi=$	$-14/3$	$10/3$	0	0	0	$-4/3$	$1/3$	-1	-1
$x_1^a=$	$1/3$	$1/3$	0	0	1	$\mathbf{2/3}$	$1/3$	0	0
$x_2^a=$	$4/3$	$-5/3$	1	0	0	$-7/3$	$-2/3$	1	0
$x_3^a=$	$10/3$	$-2/3$	0	1	0	$11/3$	$1/3$	0	1

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z=$	$-1/2$	$-1/2$	0	0	$-1/2$	0	$1/2$	1	1
$-\xi=$	-4	4	0	0	2	0	1	-1	-1
$x_1^a=$	$1/2$	$1/2$	0	0	$3/2$	1	$1/2$	0	0
$x_2^a=$	$5/2$	$-1/2$	1	0	$7/2$	0	$1/2$	$\mathbf{1}$	0
$x_3^a=$	$3/2$	$-15/6$	0	1	$-11/2$	0	$-3/2$	0	1

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z=$	-3	0	-1	0	-4	0	0	0	1
$-\xi=$	$-3/2$	$7/2$	1	0	$11/2$	0	$3/2$	0	-1
$x_1^a=$	$1/2$	$1/2$	0	0	$3/2$	1	$1/2$	0	0
$x_2^a=$	$5/2$	$-1/2$	1	0	$7/2$	0	$1/2$	1	0
$x_3^a=$	$3/2$	$-5/2$	0	1	$-11/2$	0	$-3/2$	0	$\mathbf{1}$

		x_1^a	x_2^a	x_3^a	x_1	x_2	x_3	x_4	x_5
$-z=$	$-9/2$	$5/2$	-1	-1	$3/2$	0	$3/2$	0	0
$-\xi=$	0	1	1	1	0	0	0	0	0
$x_1^a=$	$1/2$	$1/2$	0	0	$3/2$	1	$1/2$	0	0
$x_2^a=$	$5/2$	$-1/2$	1	0	$7/2$	0	$1/2$	1	0
$x_3^a=$	$3/2$	$-5/2$	0	1	$-11/2$	0	$-3/2$	0	1

The previous example was only implementing Phase I.

At the end of phase I, $\xi = 0$, and all of the artificial variables have been driven out of the basis.

Note, though, that as we start Phase II, we have $\bar{c}_j \geq 0$ for all of the non-artificial variables, so the starting BFS of Phase II is already optimal.

Consider the following sequence of pivots in a LP

-34	-1	-14	-6	0	0	0	0
4	1	1	1	1	0	0	0
2	**1**	0	0	0	1	0	0
3	0	0	1	0	0	1	0
6	0	3	1	0	0	0	1

-32	0	-14	-6	0	1	0	0
2	0	1	**1**	1	-1	0	0
2	1	0	0	0	1	0	0
3	0	0	1	0	0	1	0
6	0	3	1	0	0	0	1

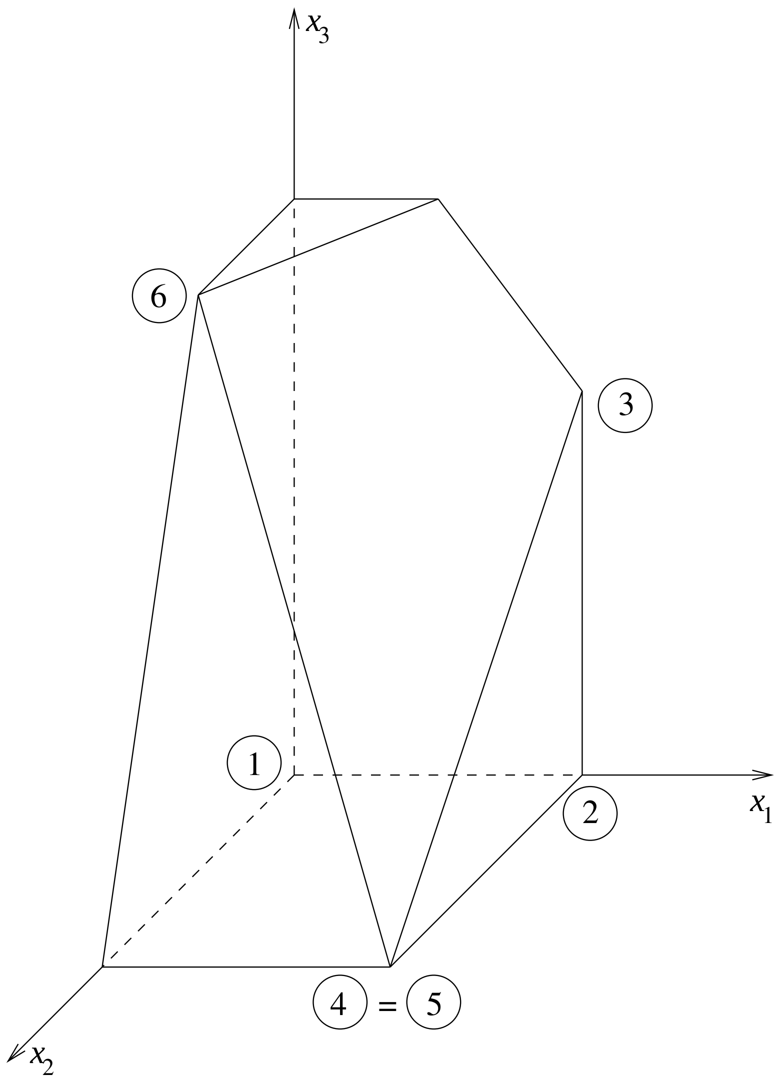
-20	0	-8	0	6	-5	0	0
2	0	**1**	1	1	-1	0	0
2	1	0	0	0	1	0	0
1	0	-1	0	-1	1	1	0
4	0	2	0	-1	1	0	1

-4	0	0	8	14	-13	0	0
2	0	1	1	1	-1	0	0
2	1	0	0	0	1	0	0
3	0	0	1	0	0	1	0
0	0	0	-2	-3	**3**	0	1

-4	0	0	$-2/3$	1	0	0	$13/3$
2	0	1	$1/3$	0	0	0	$1/3$
2	1	0	$\mathbf{2/3}$	1	0	0	$-1/3$
3	0	0	1	0	0	1	0
0	0	0	$-2/3$	-1	1	0	$1/3$

-2	1	0	0	2	0	0	4
1	$-1/2$	1	0	$-1/2$	0	0	$1/2$
3	$3/2$	0	1	$3/2$	0	0	$-1/2$
0	$-3/2$	0	0	$-3/2$	0	1	$1/2$
2	1	0	0	0	1	0	0

They actually correspond to moving in the given order, from vertex to vertex in the following polytope:



In the previous example simplex we saw that simplex moved along edges from vertex to vertex. We now see that this always happens.

Two *vertices* $\widehat{x}$, $\widehat{y}$ of a polytope are called *adjacent* if line segment $[\widehat{x}, \widehat{y}]$ is an edge of the polytope.

Two *BFSs* of LP $Ax = b$, $x \geq 0$ are called *adjacent* if there exist bases $\mathcal{B}_x$ $\mathcal{B}_y$ such that

$$\mathcal{B}_y = \left(\mathcal{B}_x - \{A_j\}\right) \cup \{A_k\}$$

and

$$x = B_x^{-1} b, \quad \text{and} \quad y = B_y^{-1} b.$$

Theorem:
Let P be a polytope,
$F = \{x : Ax = b, x \geq 0\}$ the corresponding feasible set, and
$\widehat{x} = (x_1, \dots, x_{n-m})$, $\widehat{y} = (y_1, \dots, y_{n-m})$
be distinct vertices of P.

Then the following are equivalent.

(a) The segment $[\widehat{x}, \widehat{y}]$ is an edge of P.

(b) For every $\widehat{z} \in [\widehat{x}, \widehat{y}]$,
if $\widehat{z} = \lambda \widehat{z}' + (1 - \lambda)\widehat{z}''$ with
$0 < \lambda < 1$ and $\widehat{z}'$, $\widehat{z}'' \in P$,
then $\widehat{z}'$, $\widehat{z}'' \in [\widehat{x}, \widehat{y}]$.

(c) The corresponding vectors x, y of F are adjacent bfs's.

Proof: In the P&S textbook.