LP. Lecture 6: Ch. 6: the simplex method in matrix form, and Section 7.1: sensitivity analysis

- matrix notation
- simplex algorithm in matrix notation
- example
- negative transpose property: proof
- sensitivity analysis (section 7.1)

Matrix notation

Dictionary contra matrix form:

- dictionary best for understanding simplex algorithm and calculation by hand of minor examples
- in larger calculation the simplex algorithm in matrix form is used.
- matrix form is more efficient. Uses numerical linear algebra.
- important questions: (i) pricing, (ii) quick updating of basis, (iii) LU-factorization, (iv) exploiting sparsity

We will just explain the algorithm in matrix form, without discussing the numerical questions.

Consider LP problem in standard form

$$\begin{aligned} \max \quad & \textstyle\sum_{j=1}^{n} c_j x_j \\ \text{s.t.} \quad & \\ & \begin{aligned} \textstyle\sum_{j=1}^{n} a_{i,j} x_j &\le b_i \quad \text{for } i = 1, \dots, m \\ x_j &\ge 0 \quad \text{for } j = 1, \dots, n. \end{aligned} \end{aligned}$$

Converting to equations by using slack variables.

$$x_{n+i} = b_i - \textstyle\sum_{j=1}^{n} a_{i,j} x_j \quad \text{for } i = 1, \dots, m.$$

Matrix form:

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & \\ & \begin{aligned} Ax &= b, \\ x &\ge O \end{aligned} \end{aligned}$$

where

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} & 1 & 0 & \dots & 0 \\ a_{21} & \dots & a_{2n} & 0 & 1 & \dots & 0 \\ \vdots & & \vdots & & & \ddots & \\ a_{m1} & \dots & a_{mn} & 0 & 0 & & 1 \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix},$$

Note: A has full rowrank, which means that the rows in A are linearly independent.

$$c = \begin{bmatrix} c_1 \\ \vdots \\ c_n \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ x_{n+1} \\ \vdots \\ x_{n+m} \end{bmatrix}$$

The objective function is $c^T x = \sum_{j=1}^{n} c_j x_j$.

The simplex algorithm will in each iteration have split the variables into two groups: the basic and the nonbasic variables. Will as usual let B and N be the index sets of, respectively, the basic variables and the nonbasic variables.

We let A_B and A_N be the submatrices of A which corresponds to the columns with indices B and N, respectively. So, we have

$$A = \left[\begin{array}{cc} A_B & A_N \end{array} \right]$$

Note: here $B = \{1, \ldots, m\}$, but this is just to simplify the notation. In general the basis indices are spread out. Mathematically we can imagine permuting columns in A and elements in x (correspondingly) so that we have the form above.

Primal simplex algorithm

Split x and c similarly as

$$x = \begin{bmatrix} x_B \\ x_N \end{bmatrix}, \quad c = \begin{bmatrix} c_B \\ c_N \end{bmatrix}$$

With that

$$\begin{aligned} Ax &= \begin{bmatrix} A_B & A_N \end{bmatrix} \begin{bmatrix} x_B \\ x_N \end{bmatrix} = A_B\, x_B + A_N\, x_N, \\ c^T x &= \begin{bmatrix} c_B^T & c_N^T \end{bmatrix} \begin{bmatrix} x_B \\ x_N \end{bmatrix} = c_B^T x_B + c_N^T x_N. \end{aligned}$$

The set of equations $Ax = b$ is now

$$A_B x_B + A_N x_N = b$$

We now assume that A_B is nonsingular; A_B is then called a basis in A. The columns in A_B are then a basis for $\mathbb{R}^m$ (i.e., m linearly independent vectors in $\mathbb{R}^m$). Solve the set of equations:

$$x_B = A_B^{-1} b - A_B^{-1} A_N x_N \tag{1}$$

which expresses the basis variables x_B through the nonbasic variables x_N.

Note: *Any* solution of $Ax = b$ can be written in this form $x = \begin{bmatrix} x_B \\ x_N \end{bmatrix}$ where x_N is chosen appropriately and x_B is determined uniquely based on (1).

We now eliminate x_B from the objective function:

$$\begin{aligned} \eta &= c_B^T x_B + c_N^T x_N = \\ &= c_B^T (A_B^{-1} b - A_B^{-1} A_N x_N) + c_N^T x_N = \\ &= c_B^T A_B^{-1} b - ((A_B^{-1} A_N)^T c_B - c_N)^T x_N. \end{aligned}$$

We now have

$$\begin{array}{rcrcr} \eta & = & c_B^T A_B^{-1} b & + & (c_N - (A_B^{-1} A_N)^T c_B)^T x_N \\ \hline x_B & = & A_B^{-1} b & - & A_B^{-1} A_N x_N \end{array}$$

which is the dictionary we have used so far. Here

$$\begin{aligned} c_B^T A_B^{-1} b &= \bar{\eta} \\ (c_N - (A_B^{-1} A_N)^T c_B)^T &= [\bar{c}_j] \\ A_B^{-1} b &= [\bar{b}_i] \\ A_B^{-1} N &= [\bar{a}_{ij}]. \end{aligned}$$

where the vectors on the right-hand side have components indexed with $i \in B$ and $j \in N$.

The basic solution Associated to the dictionary (the choice of basis B) is

$$x_N^* = O, \;\; x_B^* = A_B^{-1} b.$$

Will now look at the dual. Recall the correspondence

- the primal variable x_j corresponds to the dual slack variable z_j
- the primal slack variable w_i corresponds to the dual slack variable y_i

We say that x_j and z_j are complementary, and that w_i and y_i are complementary. Complementary variables have opposite roles in the equations: they are on the opposite sides.

This means that: a variable is in basis if and only if the complementary variable is out of basis.

Example:

$$
\text{(P)}\quad
\begin{array}{rcrcrcrcr}
\eta & = & 0 & + & 4x_1 & + & x_2 & + & 3x_3 \\
\hline
w_1 & = & 1 & - & x_1 & - & 4x_2 & & \\
w_2 & = & 3 & - & 3x_1 & + & x_2 & - & x_3
\end{array}
$$

$$
\text{(D)}\quad
\begin{array}{rcrcrcr}
-\xi & = & 0 & - & y_1 & - & 3y_2 \\
\hline
z_1 & = & -4 & + & y_1 & + & 3y_2 \\
z_2 & = & -1 & + & 4y_1 & - & y_2 \\
z_3 & = & -3 & & & + & y_2
\end{array}
$$

Initially x_1 is not in the basis in (P), while the complementary variable z_1 is in basis in (D) and so on. Pivot now in (P) by taking x_3 into basis and w_2 out of basis. By corresponding pivot in (D): z_3 (complementary of x_3) goes out of basis and y_2 (complementary of w_2) goes into basis.

In each pivot in (P) one basic variable and one nonbasic variable switch roles. By a corresponding pivot in (D) the complementary variables in (D) will also switch roles, but the opposite way.

This means that complementary variables still have opposite roles when it comes to being in basis. Because of this complementary property we choose to arrange the variables in the two problems like this:

$$\begin{array}{lcl} (x_1, \ldots, x_n, w_1, \ldots, w_m) & \longrightarrow & (x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+m}) \\ (z_1, \ldots, z_n, y_1, \ldots, y_m) & \longrightarrow & (z_1, \ldots, z_n, z_{n+1}, \ldots, z_{n+m}) \end{array}$$

So then x_j and z_j are complementary for $j = 1, \ldots, n+m$. In particular, the basic variables in (D) be z_N (not z_B!).

Because of the negative transpose property the dual dictionary (with basis B) is given by:

$$\begin{array}{rcrcr} -\xi & = & -c_B^T A_B^{-1} b & - & (A_B^{-1} b)^T z_B \\ \hline z_N & = & (A_B^{-1} A_N)^T c_B - c_N & + & (A_B^{-1} A_N)^T z_B. \end{array}$$

The corresponding dual basic solution for this dictionary is

$$z_B^* = O, \;\; z_N^* = (A_B^{-1} A_N)^T c_B - c_N.$$

We now introduce

$$\eta^* = c_B^T A_B^{-1} b$$

which is the value of the objective function η in (P) for the basis solution associated to B.

Conclusion: primal and dual dictionary for basis B now becomes

$$\begin{array}{rcrcr} \eta & = & \eta^* & - & (z_N^*)^T x_N \\ \hline x_B & = & x_B^* & - & A_B^{-1} A_N x_N \end{array} \qquad (2)$$

$$\begin{array}{rcrcr} -\xi & = & -\eta^* & - & (x_B^*)^T z_B \\ \hline z_N & = & z_N^* & + & (A_B^{-1} A_N)^T z_B \end{array} \qquad (3)$$

Note the negative transpose property.

simplex algorithm (primal) in short version:

- starts with a basis B so that x_B^* is feasible in (P)
- makes thereafter a sequence of pivots.
- each pivot is to find a neighboring basis (that equals the previous basis except for one index) so that η increases and also determine the corresponding primal- and dual basic solutions.

The simplex algorithm in matrix form will find the same solutions (in each iteration) as the dictionary approach. The difference is just that we are now going to operate with matrices and vectors. We use the notation from the dictionary form above.

An iteration in the simplex algorithm:

Step 1. Test optimality. If $z_N^* \geq 0$, stop. The present basic solution is optimal.

Step 2. Choose entering basic variable. Choose an index $j \in N$ where $z_j^* < 0$. Call x_j entering basis variable.

Step 3. Calculate the primal search direction. Will now let $x_N = te_j$ where e_j is the j'th unit vector; this determines the change of (primal) nonbasic variables. The primal basic variables will then be given by (see (2))

$$x_B = x_B^* - A_B^{-1} A_N t e_j = x_B^* - t \cdot \Delta x_B \tag{4}$$

where the search direction is given by

$$\Delta x_B = A_B^{-1} A_N e_j.$$

(Δx_B contains the coordinates of the jth column in A_N expressed in basis A_B.)

Step 4. Calculate primal step length. We choose t as large as possible so that x_B still is nonnegative. From equation (4) we get that the new value of the basic variable x_i is

$$x_i = x_i^* - t \cdot \Delta_i.$$

So if $\Delta_i \leq 0$ for all i, the problem (P) is unbounded. Otherwise the maximal t is given by

$$t = \min\{x_i^*/\Delta_i : \Delta_i > 0\}. \tag{5}$$

Based on steps 3 and 4 we can determine the new primal solution (see Step 8).

Step 5. Choose the leaving basic variable. Choose an index i where the minimum occurred in (5), and let x_i be the leaving basic variable.

Step 6. Calculate the dual search direction. It still remains to find the change in the dual variables (we need these to find the new coefficients in the objective function in (P)). This is determined by the choice of i and j over. Since x_i leaves basis in (P), the complementary variable z_i will go into basis in (D), so it has to be increased from zero to a certain value s. The dual basic variables are then given by (see (3))

$$z_N = z_N^* + (A_B^{-1}A_N)^T s e_i = z_N^* - s \cdot \Delta z_N \qquad (6)$$

where the search direction is given by

$$\Delta z_N = -(A_B^{-1}A_N)^T e_i.$$

Step 7. Calculate the dual step length. We can determine the dual step length s based on that z_j leaves basis (which happens because the complementary variable x_j goes into primal basis). Since z_j becomes zero we get from (6) that

$$s = z_j^* / \Delta_j.$$

Step 8. Update primal and dual solution. Primal solution is updated by

$$x_j^* := t, \;\; x_B^* := x_B^* - t \cdot \Delta x_B$$

and dual solution is updated by

$$z_i^* := s, \;\; z_N^* := z_N^* - s \cdot \Delta z_N.$$

Step 9. Update basis. Finally the basis is updated by

$$B := (B \setminus \{i\}) \cup \{j\}.$$

Final comments:

- example: see section 6.3 in Vanderbei's book
- dual simplex in matrix form: see section 6.4
- summary: see slide number 2.

Negative transpose property

Consider the primal LP problem (P)

$$\max \quad c^T x \quad \text{s.t.} \quad Ax + w = b, \quad x, w \geq O.$$

and the dual (D)

$$\min \quad b^T y \quad \text{s.t.} \quad A^T y - z = c, \quad y, z \geq O.$$

Alternatively: (P) is

$$\max \quad \bar{c}^T \bar{x} \quad \text{s.t.} \quad \bar{A}\bar{x} = \bar{b}, \quad \bar{x} \geq O.$$

and the dual (D)

$$\min \quad \hat{b}^T \hat{y} \quad \text{s.t.} \quad \hat{A}^T \hat{y} = \hat{c}, \quad \hat{y} \geq O.$$

Here

$$\bar{A} = \left[\begin{array}{cc} A & I \end{array} \right], \quad \bar{c} = \left[\begin{array}{c} c \\ O \end{array} \right], \quad \bar{x} = \left[\begin{array}{c} x \\ w \end{array} \right],$$

and

$$\hat{A} = \left[\begin{array}{cc} -I & A^T \end{array} \right], \quad \hat{b} = \left[\begin{array}{c} O \\ b \end{array} \right], \quad \hat{y} = \left[\begin{array}{c} z \\ y \end{array} \right],$$

Complementary - primal and dual basis: column j is in basis in $\bar{A}$ if and only if column j is *not* in basis in $\hat{A}$.

In the beginning the m last columns in $\bar{A}$ are in basis, and the n first columns in $\hat{A}$ are in basis.

After a few pivots

$$\bar{A} = \left[\begin{array}{cc} A & I \end{array} \right] = \left[\begin{array}{cc} \bar{A}_N & \bar{A}_B \end{array} \right] P$$

for a permutation matrix P. The columns in $\bar{A}$ are permuted. Since the corresponding pivots occur in the dual

$$\hat{A} = \begin{bmatrix} -I & A^T \end{bmatrix} = \begin{bmatrix} \hat{A}_B & \hat{A}_N \end{bmatrix} P$$

But $P^{-1} = P^T$, so $PP^T = I$. Which means that

$$\bar{A}\hat{A}^T = \begin{bmatrix} \bar{A}_N & \bar{A}_B \end{bmatrix} PP^T \begin{bmatrix} \hat{A}_B^T \\ \hat{A}_N^T \end{bmatrix} = \bar{A}_N \hat{A}_B^T + \bar{A}_B \hat{A}_N^T$$

and in addition we have that

$$\bar{A}\hat{A}^T = \begin{bmatrix} A & I \end{bmatrix} \begin{bmatrix} -I \\ A \end{bmatrix} = -A + A = O.$$

So:

$$\bar{A}_N \hat{A}_B^T + \bar{A}_B \hat{A}_N^T = O$$

By some algebra we get that

$$\bar{A}_B^{-1} \bar{A}_N = -(\hat{A}_B^{-1} \hat{A}_N)^T$$

This shows the negative transpose property.

Sensitivity analysis

Sensitivity analysis (section 7.1): what happens with the solutions when the parameters are changed?

Look at a question like this for LP: given an optimal basis, how much can each coefficient in the objective function be altered without the present basic solution becoming non optimal?

We can find the answer via duality!

Recall that when A_B is the optimal basis we have

$$\begin{aligned} x_B^* &= A_B^{-1} b, \\ y_N^* &= (A_B^{-1} A_N)^T c_B - c_N, \\ \eta^* &= c_B^T A_B^{-1} b. \end{aligned}$$

So: assume that only c is altered (among the data). Then y_N^* is altered, but not x_B^*. So if c is not altered too much, such that the new vector y_N^* is nonnegative, then x_B^* will still be optimal!

Assume that c is altered to $c + t \cdot \Delta c$, where t is a number and Δc is an "perturbation vector" (often a unity vector). Then y_N^* is altered to $y_N^* + t \cdot \Delta y_N$ where

$$\Delta y_N = (A_B^{-1} A_N)^T \Delta c_B - \Delta c_N.$$

So the present basis will still be optimal (after the perturbation in c) if

$$(*) \quad y_N^* + t \cdot \Delta y_N \geq O.$$

The sensitivity analysis boils down to determining the smallest and the largest value of t so that $(*)$ holds!!

Example:

$$
\begin{array}{lllll}
\max & 5x_1+ & 4x_2+ & 3x_3 & \\
\text{s.t.} & & & & \\
\text{(i)} & 2x_1+ & 3x_2+ & x_3 & \leq 5 \\
\text{(ii)} & 4x_1+ & x_2+ & 2x_3 & \leq 11 \\
\text{(iii)} & 3x_1+ & 4x_2+ & 2x_3 & \leq 8
\end{array}
$$

$$x_1, x_2, x_3 \geq 0.$$

Optimal dictionary, where $B = \{3, 1, 5\}$ and $N = \{4, 2, 6\}$

$$
\begin{array}{llrrrrrr}
\eta & = 13 & - & x_4 & - & 3x_2 & - & x_6 \\
\hline
x_3 & = 1 & + & 3x_4 & + & x_2 & - & 2x_6 \\
x_1 & = 2 & - & 2x_4 & - & 2x_2 & + & x_6 \\
x_5 & = 1 & + & 2x_4 & + & 5x_2 & &
\end{array}
$$

Note: be aware of the order of the variables in the matrix calculations!

We want to look at a change of the coefficient 3 to x_3 in the obj.func.
Therefore, let $\Delta c_B = (1, 0, 0)^T$ and $\Delta c_N = (0, 0, 0)^T$. The matrix $A_B^{-1} A_N$ is found from the optimal dictionary like this:

$$-A_B^{-1} A_N = \begin{bmatrix} 3 & 1 & -2 \\ -2 & -2 & 1 \\ 2 & 5 & 0 \end{bmatrix} \quad \text{which gives}$$

$$\Delta y_N = (A_B^{-1} A_N)^T \Delta c_B - \Delta c_N =$$

$$\begin{bmatrix} -3 & -1 & 2 \\ 2 & 2 & -1 \\ -2 & -5 & 0 \end{bmatrix}^T \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -3 \\ -1 \\ 2 \end{bmatrix}.$$

So: B will be the optimal basis if

$$(*) \quad (y_N^* + t \cdot \Delta y_N)^T = (1,3,1) + t \cdot (-3,-1,2) \geq O$$

$$\text{i.e.} \quad 1 - 3t \geq 0,\ 3 - t \geq 0,\ 1 + 2t \geq 0.$$

This gives $-1/2 \leq t \leq 1/3$. So the coefficient of x_3 (which was 3) can vary between $3 - 1/2 = 5/2$ and $3 + 1/3 = 10/3$.
Finally: note what happens if we use $\Delta c_B = O$!

This sensitivity analysis shows how important dictionaries (or the concept of a basis) are to understand linear programming!

Further themes are:

- some game theory
- convexity (geometrical aspects of LP), and
- network flow problems.