MA3231 Linear Programming
W. J. Martin
August 30, 2018

Summary of the Simplex Method

Overview

The Simplex Method is covered in our text (Chapter 2) and has been presented in class. But, as my notation differs slightly from Vanderbei's notation, I want to summarize the basic algorithm here in these notes for you. First we run through it mechanically and then we go through things again, introducing important notation and ideas and addressing the central issues in developing a working version of the basic method.

Mechanics of a Pivot

Suppose we are at some point in the algorithm where we have a feasible dictionary in front of us. The variables are partitioned into basic and non-basic variables and the previous pivot has left us with the dictionary

$$\begin{array}{rcrcl} \zeta & = & \bar{\zeta} & + & \displaystyle\sum_{j\in\mathcal{N}} \bar{c}_j x_j \\ \hline x_i & = & \bar{b}_i & - & \displaystyle\sum_{j\in\mathcal{N}} \bar{a}_{ij} x_j \quad (i\in\mathcal{B}) \end{array}$$

Based on this data, our pivot proceeds in four simple steps. We assume the dictionary is feasible and that our objective function is to be maximized over the feasible region.

ENTER:	Find $j\in\mathcal{N}$ such that $\bar{c}_j > 0$. (Say $j = r$ works.) This x_r is the "entering variable". If there is **no such** non-basic **variable**, then STOP: the current dictionary is OPTIMAL; by setting all non-basic variables to zero and solving for the basic variables ($x_i = \bar{b}_i$ for $i\in\mathcal{B}$), we obtain an optimal solution x^* with optimal objective value $\zeta^* = \bar{\zeta}$.
RATIOS:	Compute min $\left\{\frac{\bar{b}_i}{\bar{a}_{ir}} \mid i\in\mathcal{B},\ \bar{a}_{ir} > 0\right\}$; this is the upper feasible limit on the value of the entering variable x_r (If this set is **empty**, then stop; the problem is UNBOUNDED)
LEAVE:	Choose $h\in\mathcal{B}$ such that $\bar{b}_h/\bar{a}_{hr}$ is equal to the above minimum. (If there is a tie, I choose the one with smallest subscript h.) This variable x_h is the "leaving variable".
NEW DICTIONARY:	In the current dictionary, the equation $x_h = \bar{b}_h - \sum_k \bar{a}_{hk} x_k$ is called the "pivot equation". We solve this equation for the entering variable x_r and then substitute this expression for x_r into all other rows of the dictionary to obtain our new dictionary.

The last step is conceptual; the computation is already done since the dictionary tells us which variables are basic.

RE-PARTITION:	Since x_r enters the basis and x_h leaves the basis, we have a new basis $\hat{\mathcal{B}} = (\mathcal{B} \setminus \{h\}) \cup \{r\}$ and the new dictionary has non-basic variables $\hat{\mathcal{N}} = (\mathcal{N} \setminus \{r\}) \cup \{h\}$.

Since our goal in this course is not simply to robotically follow instructions, but to understand structure and ideas, we will shortly go through the entire sequence again, seeing why each step is as it is and seeing what might go wrong with such a simple approach. But first we need to think carefully about notation. How do we get to an initial feasible dictionary? How do we define unified notation that captures all situations? Let's first look at the transition from standard form to a first dictionary.

Notation and Set-Up

In our first version of the simplex method, we assume that an initial feasible dictionary is readily available. For example, assume we are first given an LP in standard form[1]:

$$\begin{array}{rrcl} \textbf{max} & d^\top x & & \\ \textbf{subject to} & Mx & \leq & b \\ & x & \geq & 0 \end{array}$$

Suppose this problem involves k variables $x_1, \ldots, x_k$ and m inequalities

$$\sum_{j=1}^{k} m_{ij} x_j \leq b_i \qquad (1 \leq i \leq m)$$

determined by matrix $M = [m_{ij}]_{i,j}$ and right-hand side vector $b = [b_i]_i$, I will immediately add a slack variable to each inequality constraint, thereby converting each to an equation[2]

$$\sum_{j=1}^{k} m_{ij} x_j \ + \ x_{k+i} = b_i \qquad (1 \leq i \leq m).$$

If we define $n = k + m$, then the total number of variables <u>plus</u> constraints in the original problem is n. Thus the equality form obtained by tossing in slack variables has exactly n variables. For some vector c of length n and some $m \times n$ matrix A, this new problem has form

$$\begin{array}{rl} \textbf{max} & c^\top x \\ \textbf{s.t.} & Ax = b \\ & x \geq 0 \end{array} \qquad (1)$$

[1]Sorry for using d and M in place of the usual c and A. But you'll see why in a moment.

[2]The fact that the unmodified constraint is of "$\leq$" form is then captured by a new requirement that the corresponding slack variable must be non-negative.

where, now, $x = (x_1, \ldots, x_n)$. In fact, we know c and A; we can express them in terms of the data we already have using the notation of partitioned matrices from linear algebra:

$$c = \begin{bmatrix} d \\ 0 \end{bmatrix}, \qquad A = \begin{bmatrix} M \,\vdots\, I \end{bmatrix}.$$

That identity submatrix on the right is very important to the simplex method.

If we partition the variables into two sets

$$\mathcal{B} = \{k+1, k+2, \ldots, n\}, \qquad \mathcal{N} = \{1, 2, \ldots, k\},$$

then we can write $x^\top = [x_\mathcal{N}^\top \,\vdots\, x_\mathcal{B}^\top]$ and $A_\mathcal{N} = M$, $A_\mathcal{B} = I$ so that

$$A = \begin{bmatrix} A_\mathcal{N} \,\vdots\, A_\mathcal{B} \end{bmatrix}.$$

and the problem (1) can be expressed (check!)

$$\begin{array}{rcl} \textbf{max} \;\; c^\top x_\mathcal{N} + 0^\top x_\mathcal{B} & & \\ A_\mathcal{N} x_\mathcal{N} + A_\mathcal{B} x_\mathcal{B} & = & b \\ x_\mathcal{N} \geq 0, \;\; x_\mathcal{B} \geq 0 & & \end{array}$$

(It makes a lot of sense to call this initial basis/non-basis pair $\mathcal{S}$ and $\mathcal{D}$ instead of $\mathcal{B}$ and $\mathcal{N}$ since the initial set of basic variables are exactly the <u>slack</u> variables and the <u>decision</u> variables, respectively.) Since $A_\mathcal{B}$ is an identity matrix, it is easy to solve the set of constraint equations for the basic variables

$$x_\mathcal{B} = I x_\mathcal{B} = b - A_\mathcal{N} x_\mathcal{N} \; .$$

This gives us our initial dictionary for the simplex method:

$$\begin{array}{cccccc} \zeta & = & 0 & + & c^\top x_\mathcal{N} \\ \hline x_\mathcal{B} & = & b & - & A_\mathcal{N} x_\mathcal{N} \end{array}$$

The Basics of Pivoting

As discussed in class and in the text, our simplex algorithm has us not only move along the boundary of the feasible region by increasing one profitable non-basic variable as much as is feasibly possible, but also has us migrate from one dictionary to the next. Since the data at some arbitrary point in the algorithm may be quite different from the initial data, we use bars on all the values. That is how our above description of a simplex pivot captures both the original data ($\bar{c}_j = c_j$, etc.) but also any crazy dictionary that comes our way as the algorithm progresses.

Suppose we are at some point in the algorithm where the set $\{1, \ldots, n\}$ of variable indices is partitioned into basic and non-basic variables. While it is more inituitive to define $\mathcal{B}$ and

$\mathcal{N}$ to be sets of variables, the notation leads us to instead define them as sets of indices. For example, while we may prefer to think that $\mathcal{B} = \{x_4, x_1, x_5\}$, the algorithm below uses $\mathcal{B} = \{4, 1, 5\}$.

Partitioned Matrices
This is where it is so crucial to forget the convention that the columns of matrix A come in some specific order. This habit, forced upon us by our two-dimensional view of things, makes understanding various algorithms like the simplex method so much harder. So please, for the remainder of these notes, just view a matrix (or vector) having its columns indexed by some set such as $\{1, 2, \ldots, n\}$ and not in any particular order. The same goes for the rows, but these are going to be indexed by a subset $\mathcal{B}$ of the variables in just a moment.

At any point in the simplex algorithm, our n variables are partitioned into basic and non-basic variables:

$$\mathcal{B} \cup \mathcal{N} = \{1, 2, \ldots, n\}, \qquad \mathcal{B} \cap \mathcal{N} = \emptyset \ .$$

As in the above case where slack variables are introduced, the columns of matrix A and rows of vectors c and x, each indexed by the variables, are partitioned accordingly:

$$\begin{aligned} c^\top &= [c_{\mathcal{B}}^\top \vdots c_{\mathcal{N}}^\top] \\ x^\top &= [x_{\mathcal{B}}^\top \vdots x_{\mathcal{N}}^\top] \\ A &= [A_{\mathcal{B}} \vdots A_{\mathcal{N}}] \\ A &= [B \vdots A_{\mathcal{N}}] \end{aligned}$$

You'll notice that I have introduced a new symbol; since we use the submatrix $A_{\mathcal{B}}$ so much, we call it simply B. (Vanderbei also abbreviates $A_{\mathcal{N}}$ to N, and you are free to do that, but I prefer $A_{\mathcal{N}}$.) It is extremely important to keep track of where we are: the submatrix B of our constraint matrix A depends on the basis $\mathcal{B}$ and changes with each pivot. In fact

Definition: A subset $\mathcal{B}$ of the variables (indices) is a basis for the problem $\max / \min c^\top x$ subject to $Ax = b$, $x \geq 0$ if the corresponding submatrix B of the constraint matrix A is invertible (hence square).

It is now clear that all bases for a given problem have the same size; the size of a basis is equal to the number of rows of A. If A has m rows, then not *all* sets of m variables are bases[3] and, as we have seen in class, not all bases are feasible. The linear algebra student cannot resist to comment that a subset $\mathcal{B}$ of $\{1, 2, \ldots, n\}$ is a "basis" if and only if the corresponding columns of A form a basis for the column space of A. (It is a good idea to try to prove this for yourself right now.)

Assume at our current point in the algorithm, we have partitioned the variables into $\mathcal{B}$

[3]In fact, if the matrix A itself does not have full row rank, then there are no bases at all! We would first row reduce in this case to eliminate redundant constraints.

and $\mathcal{N}$ and have partitioned A and c as above. Then the problem (1) can be expressed

$$\begin{aligned} \mathbf{max}\ \ & c_{\mathcal{B}}^{\top} x_{\mathcal{B}} + c_{\mathcal{N}}^{\top} x_{\mathcal{N}} \\ & B x_{\mathcal{B}} + A_{\mathcal{N}} x_{\mathcal{N}} = b \\ & x_{\mathcal{B}},\quad x_{\mathcal{N}} \geq 0 \end{aligned} \tag{2}$$

When we discuss a given dictionary, it is convenient to have both notation for its individual entries and expressions for these entries in terms of the original data. As in the introduction, we use the "bar" notation to refer to the numbers before us as we look at the dictionary:

$$\begin{array}{ccccl} \zeta & = & \bar{\zeta} & + & \sum_{j\in\mathcal{N}} \bar{c}_j x_j \\ \hline x_i & = & \bar{b}_i & - & \sum_{j\in\mathcal{N}} \bar{a}_{ij} x_j \quad (i \in \mathcal{B}) \end{array}$$

There are $m + 1 = |\mathcal{B}| + 1$ equations here: the objective function above the separating line and one for each basic variable below this line.

But how do we compute all of these values $\bar{b}_i$, $\bar{c}_j$, $\bar{a}_{ij}$ from the original data? Using the constraint equation from (2), we solve for the set of basic variables:

$$\begin{aligned} B x_{\mathcal{B}} + A_{\mathcal{N}} x_{\mathcal{N}} &= b \\ B x_{\mathcal{B}} &= b - A_{\mathcal{N}} x_{\mathcal{N}} \\ x_{\mathcal{B}} &= B^{-1} b - B^{-1} A_{\mathcal{N}} x_{\mathcal{N}} \end{aligned}$$

which is valid since we are assuming that B is an invertible matrix.

We then substitute this expression for $x_{\mathcal{B}}$ into the first equation from (2) and collect like terms to arrive at the standard matrix form of a simplex dictionary:

$$\boxed{\begin{array}{ccccc} \zeta & = & c_{\mathcal{B}}^{\top} B^{-1} b & + & \left(c_{\mathcal{N}}^{\top} - c_{\mathcal{B}}^{\top} B^{-1} A_{\mathcal{N}}\right) x_{\mathcal{N}} \\ \hline x_{\mathcal{B}} & = & B^{-1} b & - & B^{-1} A_{\mathcal{N}} x_{\mathcal{N}} \end{array}}$$

These two expressions for a dictionary are very important in this course. In principal, we could memorize each entry of the dictionary in terms of the original data, but most of these are tedious:

- current objective value $\bar{\zeta} = c_{\mathcal{B}}^{\top} B^{-1} b$
- "reduced costs" $\bar{c}_j = c_j - \sum_{i\in\mathcal{B}} \sum_{h=1}^{m} c_i \left(B^{-1}\right)_{ih} a_{hj}$
- RHS values $\bar{b}_i = \sum_{h=1}^{m} \left(B^{-1}\right)_{ih} b_h$

- $\bar{a}_{ij} = \sum_{h=1}^{m} (B^{-1})_{ih} a_{hj}$
  (Observe: this is the same as with a matrix A and its row-reduced version $B^{-1}A$ in MA2071)

Recovering the Inverse
As the above unpleasant formulas suggest, it is important to know how to recover the matrix B^{-1} from a given dictionary, just by knowing the basis and the original data. In the case where the original problem was in standard form and the slack variables $x_{\mathcal{S}}$ formed the first basis. We look at the derivation of our dictionary in two ways:

$$\begin{aligned} Bx_{\mathcal{B}} + A_{\mathcal{N}}x_{\mathcal{N}} &= b \\ Bx_{\mathcal{B}} &= b - A_{\mathcal{N}}x_{\mathcal{N}} \\ x_{\mathcal{B}} &= B^{-1}b - B^{-1}A_{\mathcal{N}}x_{\mathcal{N}} \end{aligned}$$

$$\begin{aligned} Mx_{\mathcal{D}} + Ix_{\mathcal{S}} &= b \\ B^{-1}Mx_{\mathcal{D}} + B^{-1}x_{\mathcal{S}} &= B^{-1}b \end{aligned}$$

We stop calculating here because the next step is better seen without messy notation. When we move the non-basic variables to the right-hand side, some of these may be slack variables and some of them may be decision variables. Nevertheless, each slack variable appearing in the current dictionary carries with it a column of the matrix B^{-1}. We never want to invert a large matrix when we don't have to – such calculations are computationally taxing.

Here's an example. Suppose we begin with the problem

$$\begin{array}{rrcrcrl} \textbf{maximize} & 50x_1 & - & 70x_2 & + & 90x_3 & \\ \textbf{subject to} & -x_1 & & & + & 2x_3 & \le 18 \\ & x_1 & - & 2x_2 & + & 2x_3 & \le 12 \\ & & & 3x_2 & - & 6x_3 & \le 21 \\ & 3x_1 & - & 4x_2 & + & x_3 & \le 15 \\ & x_1, & & x_2, & & x_3 & \ge 0 \end{array}$$

We introduce slack variables x_4, x_5, x_6, x_7 as usual and obtain the initial dictionary

```
zeta =   0  +  50 x1  -  70 x2  +  90 x3
------------------------------------------
 x4  =  18  +     x1           -   2 x3
 x5  =  12  -     x1  +   2 x2  -   2 x3
 x6  =  21            -   3 x2  +   6 x3
 x7  =  15  -   3 x1  +   4 x2  -     x3
```

After a few pivots, we arrive at the dictionary

```
zeta = 558.0  + 26.0 x2  - 44.0 x5  -  2.0 x7
-------------------------------------------------
 x4  =  13.2  +  0.4 x2  +  1.4 x5  -  0.8 x7
 x3  =   4.2  +  0.4 x2  -  0.6 x5  +  0.2 x7
 x6  =  46.2  -  0.6 x2  -  3.6 x5  +  1.2 x7
 x1  =   3.6  +  1.2 x2  +  0.2 x5  -  0.4 x7
```

This allows us to piece together the four columns of B^{-1}, two coming from the LHS (the first and third column spacially, labeled 4 and 6 respectively) and two coming from the data on the RHS:

$$\mathrm{c}ol_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathrm{c}ol_5 = \begin{bmatrix} -1.4 \\ 0.6 \\ 3.6 \\ -0.2 \end{bmatrix}, \quad \mathrm{c}ol_6 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathrm{c}ol_7 = \begin{bmatrix} 0.8 \\ -0.2 \\ -1.2 \\ 0.4 \end{bmatrix}.$$

Of course, B is just the square submatrix of the original constraint matrix formed by columns $4, 3, 6, 1$. So we have, for this example,

$$B = \begin{bmatrix} 1 & 2 & 0 & -1 \\ 0 & 2 & 0 & 1 \\ 0 & -6 & 1 & 0 \\ 0 & 1 & 0 & 3 \end{bmatrix}, \qquad B^{-1} = \begin{bmatrix} 1 & -7/5 & 0 & 4/5 \\ 0 & 3/5 & 0 & -1/5 \\ 0 & 18/5 & 1 & -6/5 \\ 0 & -1/5 & 0 & 2/5 \end{bmatrix}.$$

In general, when the equality-form problem is obtained from an LP in standard form by introducing one slack variable for each constraint, the slack variables become the initial basis and we apply this technique to recover B^{-1}, column by column, from the final dictionary — the i^{th} column of B^{-1} is the column of the final dictionary corresponding to the i^{th} slack variable, where we multiply by -1 in the case that the variable is non-basic and the column is simply the i^{th} column of the identity matrix if the slack variable is basic in row i of the dictionary.

Seeing the Pivot in Matrix Form

Now let us finish this discussion where we started. We have before us a dictionary

$$\begin{array}{ccccc} \zeta & = & c_{\mathcal{B}}^{\top} B^{-1} b & + & \left(c_{\mathcal{N}}^{\top} - c_{\mathcal{B}}^{\top} B^{-1} A_{\mathcal{N}}\right) x_{\mathcal{N}} \\ \hline x_{\mathcal{B}} & = & B^{-1} b & - & B^{-1} A_{\mathcal{N}} x_{\mathcal{N}} \end{array}$$

and we choose as entering variable some non-basic variable x_r with strictly positive $\bar{c}_r$:

$$c_r > c_{\mathcal{B}}^{\top} B^{-1} \mathbf{a}_r$$

where we use $\mathbf{a}_r$ to denote column r of the submatrix $A_{\mathcal{N}}$.

Next we move in direction $-B^{-1}\mathbf{a}_r$ until we violate feasibility:

$$x_{\mathcal{B}} = B^{-1}b - tB^{-1}\mathbf{a}_r\ , \qquad x_r = t\ .$$

The "ratios" computation simply finds the largest value of t which keeps all these entries non-negative.

It is possible that, as this parameter t increases, all entries of $x_{\mathcal{B}}$ remain non-negative, either increasing along with t or staying fixed independent of t. In this case, the problem is unbounded; we can increase "profit" to arbitrarily large values without compromising feasibility. But is it more typical that, when we maximize this parameter t, at least one variable will become zero. Since x_r is entering the basis and the basis must stay the same size, one of these variables must leave the basis to become non-basic (and thus have value zero) in the next dictionary. It is a nice exercise to prove that this swap ensures that the next submatrix B will again be invertible.

There is a lot more to say here, but that will have to wait for another day. It is so very important that you be able to derive the matrix form of a dictionary as displayed in the box above. If you cannot derive it, at least memorize it. All of this material will come in handy later when we discuss duality, sensitivity analysis, and interior point methods.