3 Does the Simplex Algorithm Work?

In this section we carefully examine the simplex algorithm introduced in the previous chapter. Our goal is to either prove that it works, or to determine those circumstances under which it may fail. If the simplex does not always work, and we know why, then we might be able to devise a way to fix it. First we must define what we mean by saying that the simplex algorithm *works*. For example, we have already seen that some LPs can be infeasible and others unbounded. What does the algorithm do in these cases? Does it terminate somehow? Does it pivot forever? What does it mean to say that the simplex algorithm "successfully" terminated on such problems, and more importantly, will the procedure always "terminate successfully" on problems for which a solution exists?

We begin our study with a detailed analysis of the various components of the algorithm. Our investigation is broken into 3 parts:

1. Initialization: The simplex algorithm pivots between feasible dictionaries (equivalently, feasible tableaus). The pivoting process moves us from one BFS to another BFS having a greater objective value. Hence, in order to pivot, we need a feasible dictionary. How do we obtain the first feasible dictionary? Clearly, the initial dictionary which defines the slack and objective variables is not feasible if the LP does not have feasible origin, or equivalently, if any component of the vector b is negative. The problem of obtaining a first feasible dictionary is necessarily nontrivial since not every LP is feasible, and detecting infeasiblity can be a difficult task. We will soon see that the problems of determining feasibility and of obtaining an initial feasible dictionary is just as hard as solving an LP with feasible origin.

2. Iteration: Can we always choose variables to enter and leave the basis in an unambiguous way? Can there be multiple choices or no choice? If there are ambiguities, can they be resolved in a satisfactory manner?

3. Termination: Does the simplex algorithm terminate after a finite number of pivots? Does it terminate at a solution when a solution exists? Does it terminate when the problems is unbounded? Can it stall, or can it go on pivoting forever without ever *solving* the problem?

We delay the discussion of (1) until after we know that the method can be made to succeed on problems with feasible origin. We begin with (2) in order to obtain a better understanding of just how the method works. Hopefully, by understanding (2) we'll have a better understanding of how to address the questions in (3).

3.1 The Simplex Algorithm Iteration: Degeneracy and Cycling

Assume that we are given a feasible tableau or, equivalently, a feasible dictionary:

$$(D_B) \qquad \begin{aligned} x_i &= \widehat{b}_i - \sum_{j \in N} \widehat{a}_{ij} x_j \qquad\qquad i \in B \\ z &= \widehat{z} + \sum_{j \in N} \widehat{c}_j x_j \ , \end{aligned}$$

where $\widehat{b}_i \geq 0,\ i \in B$ (the non-negativity of $\widehat{b}_i,\ i \in B$ is equivalent to the feasibility of the dictionary D_B). Recall that the rule for choosing the variable to enter the basis is to choose *any* one of the nonbasic variables $x_{j_0},\ j_0 \in N$ for which $\widehat{c}_{j_0} > 0$. There may be many such nonbasic variables, but all of them have the potential to increase the value of the objective variable z.

Once we have chosen a variable, say $x_{j_0},\ j_0 \in N$ with $\widehat{c}_{j_0} > 0$, to enter the basis, then the variable that leaves the basis is chosen from among those that place the greatest restriction on increasing the value of the entering variable x_{j_0}. These restrictions are imposed by our desire to maintain feasibility, that is, the non-negativity of the basic variables. As we have seen, the leaving variable must be chosen from the indices that attain the minimum value among the ratios

$$\frac{\widehat{b}_i}{\widehat{a}_{ij_0}} \quad \text{for} \quad \widehat{a}_{ij_0} > 0 \qquad i \in B$$

(recall that if $\widehat{a}_{ij_0} \leq 0$, then the corresponding equation places no restriction on increases to the value of x_{j_0} since increasing the value of x_{j_0} in this instance does not decrease the value of x_i). Hence, if x_{i_0} is the leaving variable, then

$$\frac{\widehat{b}_{i_0}}{\widehat{a}_{i_0 j_0}} = \min \left\{ \frac{\widehat{b}_i}{\widehat{a}_{ij_0}} : i \in B,\ \widehat{a}_{ij_0} > 0 \right\}. \tag{3.1}$$

There are two potential problems with this selection rule for choosing the leaving variable.

(i) There is no $i \in B$ for which

$$\widehat{a}_{ij_0} > 0,$$

and

(ii) There is more than one $i_0 \in B$ for which (3.1) holds.

If (i) occurs, then we can increase the value of the entering variable x_{j_0} as much as we want without violating feasibility since there are no restrictions. Since $\widehat{c}_{j_0} > 0$, this implies that we can increase the value of the objective variable z as much as we want without violating feasibility, that is, the problem is necessarily unbounded.

FACT: If there exists $j_0 \in N$ in the dictionary D_B for which $\widehat{c}_{j_0} > 0$ and $\widehat{a}_{ij_0} \leq 0$ for all $i \in B$, then the LP

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b,\ 0 \leq x \end{array}$$

is unbounded, i.e., the optimal value is $+\infty$.

Using this fact we are now able to detect unbounded LPs. We illustrate this with the following example:

$$\begin{array}{rrcrcrcl} \max & x_1 & + & x_2 & + & x_3 & & \\ \text{subject to} & 3x_1 & + & x_2 & - & 2x_3 & \leq & 5 \\ & 4x_1 & + & 3x_2 & & & \leq & 7 \\ & 0 \leq & x_1, & x_2, & x_3 & & & \end{array}$$

The initial tableau for this LP is

$$\left[\begin{array}{ccccc|c} 3 & 1 & -2 & 1 & 0 & 5 \\ 4 & 3 & 0 & 0 & 1 & 7 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right]$$

Note that all three variables x_1, x_2, and x_3 are candidates for entering the basis. However, in the column above x_3, there are no positive coefficients. Hence, if we choose x_3 to enter the basis, its value can be increased without bound and not violate the non-negativity of any of the other variables. Therefore, this problem is unbounded, that is, it is feasible and has $+\infty$ as its optimal value. In general, when pivoting one should scan the columns above all positive coefficients in the objective row. If one of these columns

Now consider the second possibility, that is there is more than one variable that can leave the basis. It turns out that this situation is very bad for the simplex algorithm and requires careful examination. Our next example which illustrates this case.

$$\begin{array}{rrrrcl} \max & 2x_1 & -x_2 & +8x_3 & & \\ \text{subject to} & 2x_1 & -4x_2 & +6x_3 & \leq & 3 \\ & -x_1 & +3x_2 & +4x_3 & \leq & 2 \\ & & & 2x_3 & \leq & 1 \\ & 0 \leq x_1, x_2, x_3 & & & & \end{array}$$

The simplex pivots are

$x = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	2	−4	6	1	0	0	3	Note that any one of these rows could serve as the pivot row!
	−1	3	4	0	1	0	2	
	0	0	②	0	0	1	1	
$z = 0$	2	−1	8	0	0	0	0	
$x = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{2} \end{pmatrix}$	②	−4	0	1	0	−3	0	Note that by pivoting on this tableau we do not change the objective value
	−1	3	0	0	1	−2	0	
	0	0	1	0	0	$\frac{1}{2}$	$\frac{1}{2}$	
$z = -4$	2	−1	0	0	0	−4	−4	
$x = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{2} \end{pmatrix}$	1	−2	0	$\frac{1}{2}$	0	$-\frac{3}{2}$	0	Note that we have not changed the point identified by this tableau
	0	①	0	$\frac{1}{2}$	1	$-\frac{7}{2}$	0	
	0	0	1	0	0	$\frac{1}{2}$	$\frac{1}{2}$	
$z = -4$	0	3	0	−1	0	−1	−4	
$x = \begin{pmatrix} 0 \\ 0 \\ \frac{1}{2} \end{pmatrix}$	1	0	0	$\frac{3}{2}$	2	$-\frac{17}{2}$	0	Again no change.
	0	1	0	$\frac{1}{2}$	1	$-\frac{7}{2}$	0	
	0	0	1	0	0	($\frac{1}{2}$)	$\frac{1}{2}$	
$z = -4$	0	0	0	$-\frac{5}{2}$	−3	$\frac{19}{2}$	−4	
$x = \begin{pmatrix} \frac{17}{2} \\ \frac{7}{2} \\ 0 \end{pmatrix}$	1	0	17	$\frac{3}{2}$	2	0	$\frac{17}{2}$	Finally, we break out to optimality.
	0	1	7	$\frac{1}{2}$	1	0	$\frac{7}{2}$	
	0	0	2	0	0	1	1	
$z = -\frac{27}{2}$	0	0	-19	$-\frac{5}{2}$	−3	0	$-\frac{27}{2}$	

Observations:

① If on a given pivot, there is more than one choice of variable to leave the basis, then the subsequent tableau will set one or more of the basic variables equal to zero in the associated basic feasible solution. A dictionary in which one or more of the basic variables is set to zero in the associated basic feasible solution is called a "degenerate dictionary". Correspondingly, a tableau in which one or more of the basic variables is set to zero in the associated basic feasible solution is called a "degenerate tableau".

② It is possible that a pivot on a degenerate dictionary (or tableau) does not change the associated basic feasible solution and the value of the objective variable z. Such a pivot is called a "degenerate pivot".

Observation ② is particularly troublesome since it opens the door to the possibility of an infinite sequence of degenerate pivots never terminating with optimality. Unfortunately, this *can* occur leading to the failure of the method. An example of the phenomenon is given

in the below. Our goal is to understand how such a pathological situation can occur and then to devise methods to overcome the problem.

EXAMPLE: **Cycling**
Choose the pivot column to be the column associated with the largest coefficient in the objective row. If there is a tie, choose the variable with the smallest index to enter the basis. The tie breaking rule for the choice of pivot row (or the leaving variable) is to choose that row that appears higher up in the tableau. In the following example, due to H. W. Kuhn, this tie breaking rule leads to cycling.

$$\begin{array}{rll} \text{maximize} & 2x_1 + 3x_2 - x_3 - 12x_4 & \\ \text{subject to} & -2x_1 - 9x_2 + x_3 + 9x_4 & \leq 0 \\ & \frac{1}{3}x_1 + x_2 - \frac{1}{3}x_3 - 2x_4 & \leq 0 \\ & 2x_1 + 3x_2 - x_3 - 12x_4 & \leq 2 \\ & x_1,\ x_2,\ x_3,\ x_4 & \geq 0 \end{array}$$

In the following discussion we assume that the simplex algorithm is operating with an iron clad pivoting rule so that any non-optimal feasible dictionary is associated with a unique pivot. One example of an iron clad pivoting rule is given in the Cycling Example above. For another example, suppose we are given the feasible dictionary

$$(D_B) \qquad \begin{aligned} x_i &= \widehat{b}_i - \sum_{j \in N} \widehat{a}_{ij} x_j \qquad\qquad i \in B \\ z &= \widehat{z} + \sum_{j \in N} \widehat{c}_j x_j \ . \end{aligned}$$

The so called *largest-coefficient largest-subscript rule* provides such an iron clad pivot rule.

Largest-Coefficient Largest-Subscript Rule

Choice of Entering Variable:

Among all those variables x_j with $j \in N$ such that $\widehat{c}_j = \max\{\widehat{c}_k : k \in B\}$ choose x_{j_0} so that j_0 is largest.

Choice of Leaving Variable:

Among all those variables x_i with $i \in B$ such that $\frac{\widehat{b}_i}{\widehat{a}_{ij_0}} = \min\left\{\frac{\widehat{b}_k}{\widehat{a}_{kj_0}} : k \in B, \widehat{a}_{kj_0} > 0\right\}$ choose x_{i_0} so that i_0 is largest.

Recall that each dictionary is associated with a set of basic indices (the indices of the basic variables). We now examine the relationship between dictionaries and their bases. Is this relationship one-to-one? That is, does the basis uniquely determine the dictionary, or

can there be multiple dictionaries having the same basis. To answer this, we first ask a simpler question. How many possible ways are there to choose a set of basic indices? Since every basis must contain m variables and there are only $n+m$ variables altogether, the maximum number of possible sets of basic indices therefore equals the number of possible ways to choose m distinct elements from a collection of $n+m$ objects. This number is given by the binomial coefficient

$$\left(\begin{array}{c} n+m \\ m \end{array} \right) = \frac{(n+m)!}{m!n!}.$$

Consequently, there are only a finite number of potential bases. Now if it were true that each basis was associated with a unique dictionary, then, due to our assumption of an iron clad pivoting rule, if the same basis appears a second time, then the exactly the same pivot will be taken and so we produce exactly the same sequence of pivots until we return to this basis again and the process starts over again. That is, the sequence of bases and corresponding dictionaries begins to cycle.

Recapping, we have observed that since there are only finitely many bases, the only way for there to be an infinite sequence of pivots is if there is at least one dictionary, say D_1, whose basis appears twice. Let

$$D_1, \ldots, D_N, D_{N+1}, D_{N+2}, \ldots$$

be the sequence of pivots that start at D_1 and end at D_N where D_1 and D_{N+1} have the same basis. Now if each basis is associated with a unique dictionary, then $D_1 = D_{N+1}$. Returning to our assumption that the pivoting rule uniquely determines the next basis, we must have $D_2 = D_{N+2}, D_3 = D_{N+3}, \cdots, D_N = D_{2N}, D_1 = D_{2N+1}, \cdots$. That is, the same sequence of dictionaries appear over and over again. If this occurs we say that the sequence of dictionaries cycles. These observations motivate the following result.

Proposition 4 *Every basis uniquely determines its associated dictionary. Consequently, the simplex algorithm fails to terminate if and only if it cycles. The simplex algorithms can only cycle between degenerate dictionaries (or tableaus) with each dictionary (or tableau) in the cycle being associated with the same basic feasible solution and objective value.*

REMARK: This proposition tells us that each basis has a unique dictionary associated with it. But BFS can be associated with many dictionaries. Consequently, any BFS associated with two or more distinct dictionaries is necessarily degenerate. This follows since in this case there must be a variable that is basic in one of the dictionaries but nonbasic in the other dictionary. When it is non-basic this variable takes the value zero in the associated basic feasible solution and so when it is basic it must also take the value zero, which implies that this BFS is degenerate.

PROOF: Clearly, if the simplex algorithm cycles, then it cannot terminate. Next let us suppose that the simplex algorithm fails to terminate. We need to show that it must cycle. By the discussion prior to the statement of this result, we know that if each basis is associated

with a unique dictionary, then the simplex algorithm must cycle. Thus, we establish the result by showing that each basis yields a unique dictionary. To this end let

$$(D_1) \qquad \begin{aligned} x_i &= \widehat{b}_i - \sum_{j \in N} \widehat{a}_{ij} x_j, \qquad i \in B \\ z &= \widehat{z}_i + \sum_{j \in N} \widehat{c}_j x_j \end{aligned}$$

and

$$(D_2) \qquad \begin{aligned} x_i &= b_i^* - \sum_{j \in N} a_{ij}^* x_j, \qquad i \in B \\ z &= z^* + \sum_{j \in N} c_j^* x_j \end{aligned}$$

be two dictionaries associated with the same basis B. Recall from the definition of a dictionary that D_1 and D_2 have identical solution sets. Let $j_0 \in N$ and consider the solution obtained by setting $x_{j_0} = t$ and $x_j = 0$ for $j \in N$, $j \neq j_0$. From this solution we see that

$$\begin{aligned} \widehat{b}_i - \widehat{a}_{ij_0} t &= x_i = b_i^* - a_{ij_0}^* t \qquad \text{for } i \in B \\ \widehat{z} + \widehat{c}_{j_0} t &= z = z^* + c_{j_0}^* t. \end{aligned}$$

Setting $t = 0$, we have

$$\widehat{b}_i = b_i^* \qquad i \in B$$

and

$$\widehat{z} = z^*.$$

Then, setting $t = 1$, we have

$$\widehat{a}_{ij_0} = a_{ij_0}^* \qquad \text{for } i \in B.$$

By repeating this argument for all $j \in N$, we find that D_1 and D_2 are identical dictionaries. Thus, each basis gives rise to a unique dictionary.

We now show that any cycle must be a cycle of degenerate pivots and that each dictionary in a cycle yields one in the same associated basic feasible solution. If $D_1, D_2, \ldots, D_N$ is the cycle of dictionaries in question. We have that the value of z in D_1 and D_{N+1} is the same. If we let z_k be the value of z in D_k, we have

$$z_1 \leq z_2 \leq \cdots \leq z_N \leq z_{N+1} = z_1.$$

Therefore, $z_i = z_j$ for $1 \leq i < j \leq N$. Hence each of the pivots taking D_i to D_j is necessarily degenerate for $1 \leq i < j \leq N$. Therefore, each of the dictionaries $D_1, \ldots, D_N$ must be degenerate. Since degenerate pivots do not alter the basic feasible solution, it must be the case that each $D_i : i = 1, \ldots, N$ identifies one in the same basic feasible solution. ■

We have established that the simplex algorithm can only fail to terminate if it cycles, and that it can only cycle in the presence of degeneracy. In order to assure that the simplex algorithm successfully terminates we need to develop a pivoting rule that avoids cycling. Over the years a number of anti-cycling pivoting rules have been proposed. In these notes, we present one such rule known as the smallest subscript rule, or Bland's Rule (named after its discoverer, Robert Bland of Cornell University). Recall that given the basis $B \subset \{1, 2, \ldots, n+m\}$, our notation for the associated dictionary is

$$(D_B) \qquad \begin{array}{rcll} x_i & = & \widehat{b}_i - \sum_{j \in N} \widehat{a}_{ij} x_j & i \in B \\ z & = & \widehat{z} + \sum_{j \in N} \widehat{c}_j x_j \ . & \end{array}$$

The Smallest Subscript Rule

Whenever a degenerate pivot is possible choose the entering and leaving variables as follows:

Choice of entering variable: x_{j_0} for $j_0 \in N$ is the entering variable if $\widehat{c}_{j_0} > 0$ and

$$j_0 \leq j \text{ whenever } \widehat{c}_j > 0.$$

Choice of leaving variable: x_{i_0} for $i_0 \in B$ is the leaving variable if

$$\frac{\widehat{b}_{i_0}}{\widehat{a}_{i_0 j_0}} = \min \left\{ \frac{\widehat{b}_i}{\widehat{a}_{ij_0}} : i \in B, \widehat{a}_{ij_0} > 0 \right\}$$

and

$$i_0 \leq i \ \text{ whenever } \ \frac{\widehat{b}_{i_0}}{\widehat{a}_{i_0 j_0}} = \frac{\widehat{b}_i}{\widehat{a}_{ij_0}} \quad i \in B.$$

In short, we always choose the variable with the smallest subscript whenever there is a tie in the choice of either entering or leaving variable.

Theorem 4.1 (R.G. Bland (1977)) *The simplex algorithm terminates as long as the choice of variable to enter or leave the basis is made according to the smallest subscript rule.*

PROOF: We need only show that the smallest subscript rule prevents cycling. To this end, we assume to the contrary that there is a case where cycling occurs even though the smallest subscript rule has been implemented. Denote this cycle of dictionaries by $D_0, D_1, \ldots, D_N = D_0$. Since it is a cycle, every dictionary in the cycle is degenerate and identifies the same BFS. Note that in pivoting from D_k to D_{k+1} some variable must leave the basis and another must enter. But since the leaving variable is in the basis in D_k while the entering variable is not, the leaving variable must entered the basis (at least once) somewhere else in the cycle while the entering variable must leave the basis (at least once) elsewhere in the cycle. Hence, in the cycle $D_0, D_1, \ldots, D_N$, there are variables that leave and return to the basis at least once during the cycle. We will call these the *fickle* variables. Note that *any* variable that

either enters or leaves a basis in the cycle $D_0, D_1, \ldots, D_N$ is necessarily fickle. In addition, when the fickle variables are nonbasic their value in the associated BSF is zero. Since the BFS identified remains fixed in a cycle, all fickle variables take the value zero in this BFS and so remain zero throughout the cycle.

Among the fickle variables there is one x_ℓ having the largest subscript. Let

$$\begin{array}{lll} D & x_i & = b_i - \sum_{j \notin B} a_{ij} x_j \,, \qquad i \in B \\ & z & = v + \sum_{j \notin B} c_j x_j \end{array}$$

be a dictionary in the cycle where x_ℓ is leaving the basis with say x_e entering. Since x_e is also fickle, $e < \ell$. Let D^* be a dictionary in the cycle with x_ℓ entering the basis. Since each dictionary in the cycle identifies the same BFS, the value v stays constant throughout the cycle. Therefore, we can write the objective row of D^* as

$$z = v + \sum_{j=1}^{m+n} c_j^* x_j, \tag{4.2}$$

where $c_j^* = 0$ if x_j is basic in D^*. Since the solution set of every dictionary in the cycle is identical, every solution to D must satisfy the objective equation (3.2). In particular, the solution to D obtained by setting $x_e = t,\ x_j = 0\ (j \notin B,\ j \neq e)$ giving $x_i = b_i - a_{ie}t\ (i \in B)$ and $z = v + c_e t$, must also satisfy (3.2) for every choice of t. That is,

$$v + c_e t = v + c_e^* t + \sum_{i \in B} c_i^* (b_i - a_{ie} t) \qquad \text{for all } t.$$

By grouping terms in this expression and re-arranging we obtain

$$\left(c_e - c_e^* + \sum_{i \in B} c_i^* a_{ie} \right) t = \sum_{i \in B} c_i^* b_i \qquad \text{for all } t.$$

Since the right hand side of this expression is constant, it must be zero as is the coefficient on the left, i.e.

$$c_e - c_e^* = - \sum_{i \in B} c_i^* a_{ie} \ .$$

Since x_e enters the basis in D, $c_e > 0$. Since x_ℓ enters the basis in D^* and $e < \ell$, the smallest subscript rule implies that $c_e^* \leq 0$. Therefore, $c_e - c_e^* > 0$. Consequently, $\sum_{i \in B} c_i^* a_{ie} < 0$, so for some $s \in B$, $c_s^* a_{se} < 0$. Since $s \in B$, x_s is basic in D, and since $c_s^* \neq 0$, x_s is nonbasic in D^*, so x_s is fickle which implies that $s \leq \ell$. We claim that $s < \ell$. Indeed, since x_ℓ is leaving the basis in D and x_e is entering, we know that $a_{\ell e} > 0$, and since x_ℓ enters the basis in D^*, $c_\ell^* > 0$, so that $c_\ell^* a_{\ell e} > 0$. Consequently, we cannot have $s = \ell$, and so $s < \ell$. Now since $s < \ell$, x_s cannot be a candidate to enter the basis in D^*, that is, $c_s^* \leq 0$. But $c_s^* \neq 0$, so

$c_s^* < 0$. But $c_s^* a_{se} < 0$, so $a_{se} > 0$. Now x_s is fickle and $s \in B$, so its value in the BFS is zero, i.e. $b_s = 0$. But since $b_s = 0$ and $a_{se} > 0$, x_s is a candidate for leaving the basis in D. But x_ℓ leaves the basis in D with $s < \ell$. This violates the smallest subscript rule contradicting the supposition that it was employed. Hence no such cycle can exist. ■

With the introduction of an anti-cycling rule, such as Bland's Rule, we now know that if we are given an initial feasible dictionary, then we can apply the simplex algorithm to either obtain an optimal basic feasible solution or determine that the LP is unbounded.

4.1 Initialization

We now turn to the problem of finding an initial basic feasible solution. Again consider an LP in standard form,

$$\mathcal{P} \qquad \begin{array}{ll} \text{maximize} & c^T x \\ \text{subject tp} & Ax \le b, \quad 0 \le x. \end{array}$$

We associate with this LP an auxiliary LP of the form

$$\mathcal{Q} \qquad \begin{array}{ll} \text{minimize} & x_0 \\ \text{subject to} & Ax - x_0 \mathsf{e} \le b, \quad 0 \le x_0, x \ . \end{array}$$

where $\mathsf{e} \in \mathsf{R}^m$ is the vector of all ones. The ith row of the system of inequalities $Ax - x_0\mathsf{e} \le b$ takes the form

$$a_{i1}x_1 \ + \ a_{i2}x_2 \ + \ \ldots \ + \ a_{in}x_n \ - \ x_0 \ \le \ b_i \ .$$

The system of inequalities can also be written in block matrix form as

$$\left[\ -\mathsf{e} \quad A \ \right] \begin{pmatrix} x_0 \\ x \end{pmatrix} \le b \ .$$

Note that if the optimal value in the auxiliary problem is zero, then at the optimal solution $(\tilde{x}_0, \tilde{x})$ we have $\tilde{x}_0 = 0$. If we plug this into the inequality $Ax - x_0\mathsf{e} \le b$, we get $A\tilde{x} \le b$. That is, $\tilde{x}$ is feasible for the original LP $\mathcal{P}$. Corresponding, if $\hat{x}$ is feasible for $\mathcal{P}$, then $(\hat{x}_0, \hat{x})$ with $\hat{x}_0 = 0$ is feasible for $\mathcal{A}$, in which case $(\hat{x}_0, \hat{x})$ must be optimal for $\mathcal{A}$. Therefore, $\mathcal{P}$ is feasible if and only if the optimal value in $\mathcal{A}$ is zero. In particular the feasibility of $\mathcal{P}$ can be determined by solving the LP $\mathcal{A}$.

The auxiliary problem $\mathcal{A}$ is also called the *Phase I* problem since solving it is the first phase of a two phase process of solving general LPs. In Phase I we solve the auxiliary problem to obtain an initial feasible tableau for the original problem, and in Phase II we solve the original LP starting with the feasible tableau provided in Phase I.

Solving $\mathcal{Q}$ by the simplex algorithm yields an initial feasible dictionary for $\mathcal{P}$. However, to solve $\mathcal{Q}$ we need an initial feasible dictionary for $\mathcal{Q}$. But if $\mathcal{P}$ does not have feasible origin neither does $\mathcal{Q}$! Fortunately, an initial feasible dictionary for $\mathcal{Q}$ is easily constructed. Observe that if we set $x_0 = -\min\{b_i : i = 0, \ldots, n\}$, then $b + x_0\mathsf{e} \ge 0$ since

$$\begin{array}{rl} & \min\{b_i + x_0 : i = 1, \ldots, m\} \\ = & \min\{b_i : i = 1, \ldots, m\} - \min\{b_i : i = 0, \ldots, m\} \ge 0. \end{array}$$

Hence, by setting $x_0 = -\min\{b_i : i = 0, \ldots, m\}$ and $x = 0$ we obtain a feasible solution for $\mathcal{Q}$. It is also a basic feasible solution. To see this, consider the initial dictionary for $\mathcal{Q}$

$$\begin{aligned} x_{n+i} &= b_i + x_0 - \sum_{j=1}^{m} a_{ij} x_j \\ z &= -x_0. \end{aligned}$$

Let i_0 be any index such that

$$b_{i_0} = \min\{b_i : i = 0, 1, \ldots, m).$$

If $i_0 = 0$, then this LP has feasible origin and so the initial dictionary is optimal. If $i_0 > 0$, then pivot on this row bringing x_0 into the basis. This yields the dictionary

$$\begin{aligned} x_0 &= -b_{i_0} + x_{n+i_0} + \sum_{j=1}^{m} a_{i_0 j} x_j \\ x_{n+i} &= b_i - b_{i_0} + x_{n+i_0} - \sum_{j=1}^{m} (a_{ij} - a_{i_0 j}) x_j, \quad i \neq i_0 \\ z &= b_{i_0} - x_{n+i_0} - \sum_{j=1}^{m} a_{i_0 j} x_j. \end{aligned}$$

But $b_{i_0} \leq b_i$ for all $i = 1, \ldots, m$, so

$$0 \leq b_i - b_{i_0} \text{ for all } i = 1, \ldots, m.$$

Therefore this dictionary is feasible. We illustrate this initialization procedure by example.

Consider the LP

$$\begin{array}{lrcrcrcr} \max & x_1 & - & x_2 & + & x_3 & & \\ \text{s.t.} & 2x_1 & - & x_2 & + & 2x_3 & \leq & 4 \\ & 2x_1 & - & 3x_2 & + & x_3 & \leq & -5 \\ & -x_1 & + & x_2 & - & 2x_3 & \leq & -1 \end{array}$$

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

This LP does not have feasible origin since the right hand side vector $b = (4, -5, -1)^T$ is not componentwise non-negative. Hence the initial dictionary D_I is not feasible. Therefore, we must first solve the auxiliary problem $\mathcal{Q}$ to obtain a feasible dictionary. The auxiliary problem has the form

$$\begin{array}{lrcrcrcrcr} \max & -x_0 & & & & & & & & \\ \text{s.t.} & -x_0 & + & 2x_1 & - & x_2 & + & 2x_3 & \leq & 4 \\ & -x_0 & + & 2x_1 & - & 3x_2 & + & x_3 & \leq & -5 \\ & -x_0 & - & x_1 & + & x_2 & - & 2x_3 & \leq & -1 \end{array}$$

$$0 \leq x_0, x_1, x_2, x_3 \ .$$

The initial tableau for this LP has the following form:

$$\begin{array}{rrrrrrr|r} -1 & 2 & -1 & 2 & 1 & 0 & 0 & 4 \\ -1 & 2 & -3 & 1 & 0 & 1 & 0 & -5 \\ -1 & -1 & 1 & -2 & 0 & 0 & 1 & -1 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

However, instead of pivoting on this tableau we will pivot on a somewhat different tableau which differs in one respect. In the Phase I tableau we also include the objective row for the original LP. This is done to save the effort of having to compute the proper coefficients for this row after solving the auxiliary problem. Having these coefficients in hand at the end of Phase I allows one to immediately begin Phase II. This is illustrated in the following example. Here we denote the objective variable for the Phase I problem $\mathcal{Q}$ as w.

	x_0								
	$\downarrow$								
	-1	2	-1	2	1	0	0	4	
	$\textcircled{-1}$	2	-3	1	0	1	0	-5	$\leftarrow$ most negative
	-1	-1	1	-2	0	0	1	-1	
z	0	1	-1	1	0	0	0	0	
w	-1	0	0	0	0	0	0	0	
	0	0	2	1	1	-1	0	9	
	1	-2	3	-1	0	-1	0	5	
	0	-3	$\textcircled{4}$	-3	0	-1	1	4	
z	0	1	-1	1	0	0	0	0	
w	0	-2	3	-1	0	-1	0	5	
	0	$\frac{3}{2}$	0	$\frac{5}{2}$	1	$-\frac{1}{2}$	$-\frac{1}{2}$	7	
	1	$\frac{1}{4}$	0	$\textcircled{\frac{5}{4}}$	0	$-\frac{1}{4}$	$-\frac{3}{4}$	2	
	0	$-\frac{3}{4}$	1	$-\frac{3}{4}$	0	$-\frac{1}{4}$	$\frac{1}{4}$	1	
z	0	$\frac{1}{4}$	0	$\frac{1}{4}$	0	$-\frac{1}{4}$	$\frac{1}{4}$	1	
w	0	$\frac{1}{4}$	0	$\frac{5}{4}$	0	$-\frac{1}{4}$	$-\frac{3}{4}$	2	
	-2	1	0	0	1	0	1	3	Auxiliary problem
	$\frac{4}{5}$	$\frac{1}{5}$	0	1	0	$-\frac{1}{5}$	$-\frac{3}{5}$	$\frac{8}{5}$	solved.
	$\frac{3}{5}$	$-\frac{3}{5}$	1	0	0	$-\frac{2}{5}$	$-\frac{1}{5}$	$\frac{11}{5}$	Extract an initial
z	$-\frac{1}{5}$	$\frac{4}{20}$	0	0	0	$-\frac{4}{20}$	$\frac{8}{20}$	$\frac{3}{5}$	feasible tableau.
w	-1	0	0	0	0	0	0	0	

We have solved the Phase I problem. The optimal value in the Phase I problem is zero. Hence the original problem is feasible. In addition, we can extract from the tableau above an initial feasible tableau for the the original LP.

1	0	0	1	0	①	3
$\frac{1}{5}$	0	1	0	$-\frac{1}{5}$	$-\frac{3}{5}$	$\frac{8}{5}$
$-\frac{3}{5}$	1	0	0	$-\frac{2}{5}$	$-\frac{1}{5}$	$\frac{11}{5}$
$\frac{1}{5}$	0	0	0	$-\frac{1}{5}$	$\frac{2}{5}$	$\frac{3}{5}$
1	0	0	1	0	1	3
$\frac{4}{5}$	0	1	$\frac{3}{5}$	$-\frac{1}{5}$	0	$\frac{17}{5}$
$-\frac{2}{5}$	1	0	$\frac{1}{5}$	0	0	$\frac{14}{5}$
$-\frac{1}{5}$	0	0	$-\frac{2}{5}$	$-\frac{1}{5}$	0	$-\frac{3}{5}$

Hence the optimal solution is

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 2.8 \\ 3.4 \end{pmatrix}$$

with optimal value $z = .6$.

Recapping, we have

The Two Phase Simplex Algorithm

Phase I Formulate and solve the auxiliary problem. Two outcomes are possible:

(i) The optimal value in the auxiliary problem is positive. In this case the original problem is infeasible.

(ii) The optimal value is zero and an initial feasible tableau for the original problem is obtained.

Phase II If the original problem is feasible, apply the simplex algorithm to the initial feasible tableau obtained from Phase I above. Again, two outcomes are possible:

(i) The LP is determined to be unbounded.

(ii) An optimal basic feasible solution is obtained.

Clearly the two phase simplex algorithms can be applied to solve any LP. This yields the following theorem.

THEOREM: [THE FUNDAMENTAL THEOREM OF LINEAR PROGRAMMING] *Every LP has the following three properties:*

(i) If it has no optimal solution, then it is either infeasible or unbounded.

(ii) If it has a feasible solution, then it has a basic feasible solution.

(iii) If it is feasible and bounded, then it has an optimal basic feasible solution.

PROOF: **Part (i):** Suppose an LP has no solution. This LP is either feasible or infeasible. Let us suppose it is feasible. In this case, the first phase of the two-phase simplex algorithm produces a basic feasible solution. Hence, the second phase of the two-phase simplex algorithm either discovers that the problem is unbounded or produces an optimal basic feasible solution. By assumption, the LP has no solution so it must be unbounded. Therefore, the LP is either infeasible or unbounded.

Part (ii): If an LP has a feasible solution, then the first phase of the two-phase simplex algorithm produces a basic feasible solution.

Part (iii): Suppose an LP is bounded. In particular, this implies that the LP is feasible, and, so by Part (ii), it has a basic feasible solution. The second phase of the two-phase simplex algorithm either discovers that the problem is unbounded or produces an optimal basic feasible solution. Since the LP is bounded, the second phase produces an optimal basic feasible solution. ■