1 Simplex Method Cycling

There is a kind of special case where the algorithm can cycle and not produce a global optimal solution. We will see this in the following example.

Example 1.1. Suppose we are solving the following LP:

\[
\begin{align*}
\text{max} & \quad 10x + 3y \\
\text{s.t.} & \quad x + y + s_1 = 4 \\
& \quad 5x + 2y + s_2 = 11 \\
& \quad y + s_3 = 4 \\
& \quad x, y, s_1, s_2, s_3 \geq 0
\end{align*}
\]

where the current basic feasible solution is \( \bar{x}^t = [0, 4, 0, 3, 0] \) with basis \( \{y, s_1, s_2\} \). Let's run the simplex method with this. We have two directions

\[
\bar{d}^x = [1, a, b, c, 0]
\]

and

\[
\bar{d}^{s_3} = [0, a, b, c, 1].
\]

We solve for these directions and get

\[
\bar{d}^x = [1, 0, -1, -5, 0]
\]

and

\[
\bar{d}^{s_3} = [0, -1, 1, 2, 1].
\]

Now check the reduced costs

\[
\hat{c}^x = 10
\]

and

\[
\hat{c}^{s_3} = -2.
\]

So our improving direction is \( \bar{d}^x \). Let's now compute the step size:

\[
\lambda_{\text{max}} = \min \left\{ \frac{x_3}{-d_3^t}, \frac{x_4}{-d_4^t} \right\} = \min \{0/1, 3/5\} = 0.
\]

Now the step size is ZERO. If we go this much in this direction we don't go anywhere so \( \bar{x}^{t+1} = \bar{x}^t + 0 \bar{d}^x = \bar{x}^t \) **BUT** now we have changed the basis to \( \{y, x, s_2\} \).
Note: In the example the program still “thinks” that the “new” solution is new. And it will continue to cycle back to this solution. Actually if there is a case of degeneracy with the nonnegativity constraints then its possible that the simplex method without any assistance will never stop and give an optimal solution. To get around this we do the following:

**Bland’s Rule to Prevent Cycling:**
1) Select the entering variable by choosing the smallest index nonbasic variable whose simplex direction is improving.
2) Select the leaving variable as the smallest index from those that define the maximal step size.

This keeps the algorithm from cycling between different basic feasible solutions. But it does not keep it from cycling between the same basic feasible solution. There is a nice way to stop that from happening:

- if you get a step size of zero then stop you are cycling.

Now Bland’s rule and this last trick keep the simplex from cycling.

### 2 Simplex Method Initial Solutions

Now we focus on how to find an initial basic feasible solution for the simplex method to run with.

We revisit the plan if we had an LP in standard form:

\[
\begin{align*}
\max \ & \mathbf{c}^\top \mathbf{x} \\
\text{s.t.} \ & A\mathbf{x} \leq \mathbf{b} \\
\ & \mathbf{x} \geq 0
\end{align*}
\]

with \( \mathbf{b} \geq 0 \) then we can put this in canonical form by letting \( \mathbf{s} = [s_1, \ldots, s_{n-m}] \) and then the canonical form LP is

\[
\begin{align*}
\max \ & \mathbf{c}^\top \mathbf{x} \\
\text{s.t.} \ & A\mathbf{x} + I_{d_m}\mathbf{s} = \mathbf{b} \\
\ & \mathbf{x}, \mathbf{s} \geq 0.
\end{align*}
\]

Then a “canonical” basic feasible solution is \((\mathbf{x}, \mathbf{s}) = (\mathbf{0}, \mathbf{b}).\)

Now, if we had one of the inequalities that was pointed in the opposite direction then we do not have the identity matrix on the right side. But here’s the trick: make the Phase I linear program:

1. Make an new LP with a new variable \( a_i \) for each constraint that is \( \geq \) or \( = \) (that is for each surplus or equality constraint in the original program).
2. Add each of these new artificial variables to the corresponding constraints.

3. Make the objective function the sum of these new artificial variables.

The result is that if the original problem was feasible then there will be a global optimal solution to this new problem where the sum of the $a_i$ are all zero. Since this optimal solution is an extreme point it will also be an extreme point of the original LP.

3 The Complete Simplex Method Algorithm

<table>
<thead>
<tr>
<th>Simplex Method Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase I</td>
</tr>
<tr>
<td>1 Put original LP in canonical form with $\vec{b} \geq 0$.</td>
</tr>
<tr>
<td>2 Make phase I LP from this with initial b.f.s. slack and artificial $= \vec{b}$.</td>
</tr>
<tr>
<td>3 Run the simplex algorithm on it. If $\vec{a} &gt; 0$ then the original LP is infeasible. Otherwise we have an initial b.f.s. call it $\vec{x}$ for the original canonical form LP.</td>
</tr>
<tr>
<td>Phase II</td>
</tr>
<tr>
<td>1 Set $t = 0$ let $\vec{x}$ be our initial basic feasible solution with basis $B^t$.</td>
</tr>
<tr>
<td>2 while $\vec{x}$ is not a local optimum do</td>
</tr>
<tr>
<td>3 For each nonbasic variable $x_k$ construct simplex directions $\vec{d}_k$</td>
</tr>
<tr>
<td>4 If no direction is improving then we are at a local optimum and Stop.</td>
</tr>
<tr>
<td>5 Otherwise, find the largest improving simplex direction $\vec{d}_k$.</td>
</tr>
<tr>
<td>6 If $d_k \geq 0$ then STOP the LP is unbounded.</td>
</tr>
<tr>
<td>7 Compute the step size $\lambda_{max}$.</td>
</tr>
<tr>
<td>8 If $\lambda_{max} = 0$ stop otherwise it will cycle and follow Bland’s rule.</td>
</tr>
<tr>
<td>9 else The new solution is $\vec{x}^{t+1} = \vec{x} + \lambda_{max} \vec{d}$.</td>
</tr>
<tr>
<td>10 And update the new basis $B^{t+1} = {x_k} \cup B^t \setminus {x_j}$ this is called a pivot, keeping track of the entering and leaving variables.</td>
</tr>
<tr>
<td>11 Set $t = t + 1$.</td>
</tr>
<tr>
<td>12 end while</td>
</tr>
</tbody>
</table>