we solved another revised simplex method example

This example is from p.26 in Chvátal:

max \ 3x_1 + 2x_2 + 4x_3 \quad \text{subject to}
\begin{align*}
x_4 &= 4 - x_1 - x_2 - 2x_3 \\
x_5 &= 5 - 2x_1 - 2x_3 \\
x_6 &= 7 - 2x_1 - x_2 - 3x_3 \\
\end{align*}
all variables \( \geq 0. \)

cutting stock problem

Each piece has length 91

We want

78 pieces of length 25 \( \frac{1}{2} \)
40 pieces of length 22 \( \frac{1}{2} \)
30 pieces of length 20
30 pieces of length 15

Let \( A \) be the matrix whose columns represent cutting patterns. A cutting pattern is a vector \( a = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} \) that satisfies the conditions

\[
25 \frac{1}{2} a_1 + 22 \frac{1}{2} a_2 + 20 a_3 + 15 a_4 \leq 91 \\
a_1, ..., a_4 \text{ integer and } \geq 0.
\]

For example, here are some cutting patterns:

\[
\begin{pmatrix}
3 & 0 & 0 & 1 \\
0 & 4 & 0 & 2 \\
0 & 0 & 4 & 3 \\
0 & 0 & 6 & 1 \\
\end{pmatrix}
\]

There are clearly only a finite number \( N \) of possible cutting patterns. The cutting stock problem is to minimize \( x_1 + ... + x_N \) (or maximize \( -x_1 - ... - x_N \)) subject to the constraint \( Ax = b \), where \( b \) is the demand vector. In our example, \( b = \begin{pmatrix} 78 \\ 40 \\ 30 \\ 30 \end{pmatrix} \). The variable \( x_i \) represents the number of times pattern \( i \) is used. It does not matter in what order all the possible patterns are placed in the matrix \( A \), so long as the \( x_i \) are numbered consistently.

In reality we will never compute the matrix \( A \); rather, we will compute new columns only as we need them.

Remark: This is really an integer programming problem since the only really interesting solutions are those for which all \( x_i \) are integral because it is not possible to use a pattern a fractional number of times. However, we will treat it as an LP. Keep in mind though, that any answers we get with fractions must be treated only as approximations,
rather than as true solutions.

solution of the example via revised simplex method

To start the algorithm, we need a basis (a set of 4 independent columns of $A$) and $x^*$ such that $Ax^*_B=b$ and $x^*_N=0$. For the basic columns we will use

$\begin{array}{cccc}
 1 & 2 & 3 & 4 \\
 1 & 3 & 0 & 0 \\
 2 & 0 & 4 & 0 \\
 3 & 0 & 0 & 4 \\
 4 & 0 & 0 & 6 \\
\end{array}$

Since $Ax^*_B=b$, we must have $x^*_B = \begin{pmatrix} \frac{78}{3} \\ 40/4 \\ 30/4 \\ 30/6 \end{pmatrix}$. Then you apply the revised simplex method.

Iteration 1:
First, solve $y'A_B=c_B=(-1,-1,-1,-1)$.

The solution is $y'=(-1/3,-1/4,-1/4,-1/6)$.

Next, look for a positive component of $c_N'-y'A_N$. This is the tricky part. We do not want to write down all the columns of $A_N$ because there are so many of them. Instead, we will try to generate a column $(a_1, a_2, a_3, a_4)$ for which the corresponding component of

$c_N'-y'A_N = (-1, ..., -1) - (-1/3, -1/4, -1/4, -1/6) A_N$

is positive. In other words, we want $(1/3, 1/4, 1/4, 1/6) \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} > 1$.

For the rest, please refer to the textbook, where this example is worked in detail (pp. 198- ).

February 9, 2000

eample, continued

In class today, we worked through two iterations of the above example. On the first iteration, we used $a^t(2,0,2,0)$ (call it column #5) as the entering column. To start the second iteration, we had:

$\begin{pmatrix} 16 \\ 10 \\ 15 \\ 5 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 2 \\ 4 \end{pmatrix}$ and $A_B = \begin{pmatrix} 3 & 0 & 2 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 6 & 0 \end{pmatrix}$.

As the entering column, we obtained $a^t(2,1,0,1)$ (call it column #6). To start the third iteration, we have:

$\begin{pmatrix} 24 \\ 4 \\ 15 \end{pmatrix} \begin{pmatrix} 6 \\ 2 \\ 4 \end{pmatrix}$.
Homework #20: Perform one more iteration (or show that the current iterate is optimal. The example is the same as the one on page 199-200 of Chvátal.