Consider the following bipartite transportation problem (from Luenberger, Linear and Nonlinear Programming). The nodes on the left represent supplies, those on the right are demands. All arcs are directed from left to right; and there is an arc from each supply node to each demand node. The \( ij \) entry of the cost matrix \( C \) specifies the cost of the arc from supply node \( i \) to demand node \( j \). The flows indicated below constitute an initial feasible tree solution that you can use to start the network simplex algorithm.

\[
C = \begin{pmatrix}
3 & 4 & 6 & 8 & 9 \\
2 & 2 & 4 & 5 & 5 \\
2 & 2 & 2 & 3 & 2 \\
3 & 3 & 2 & 4 & 2 \\
\end{pmatrix}
\]

Any flow can be represented as a 4x5 matrix. For example, the feasible tree flow shown above can be represented by the matrix at right. (Note that each column represents a demand node and each row represents a supply node).

We can circle the entries of the cost matrix that correspond to tree arcs and then propagate the potential through the tree. (Can you see how this is done?) This produces the orange numbers shown.

Continue in this manner to solve the transportation problem, always working with the matrix, and never directly with the network. (Although, you may wish to work secretly with the network, just to keep things clear in your mind.) Issues to consider are: how to recognize a "bad arc" by looking at the matrices, how to recognize the cycle that a bad arc forms with the tree, how to increase flow around a cycle, and so forth.

When you are done, verify that \( c'x = b'p \).