# CO 602/CM 740: Fundamentals of Optimization
# Problem Set 7

H. Wolkowicz

Fall 2011.
Handed out: 2011-Nov-13
Due: Wed, 2011-Nov-23, by 3PM.

## Contents



## 1 Consistent Matrix Balancing

Consider the matrices

$$B = \begin{bmatrix} 9.4400 & 8.6600 & 8.0400 \\ 6.2600 & 5.4100 & 8.4800 \end{bmatrix}$$

$$\bar{B} = \begin{bmatrix} 2.2900 & 1.5200 & 5.3800 & 0.7800 \\ 9.1300 & 8.2600 & 9.9600 & 4.4300 \end{bmatrix}$$

Let $e$ be the vector of ones of appropriate length, and let $R = Be, C = B^T e$ denote the row, column sums of $B$, respectively.

1. Use a network flow approach and formulate a mathematical model for the problem of consistent rounding of $B$, i.e. the problem is to round the elements of $B$ (up or down) in order to obtain the rounded (up or down) row and column sums of $B$,

   $$\text{round}(B)e = \text{round}(Be), \qquad \text{round}(B^T)e = \text{round}(B^T e),$$

   where round$(v)$ refers to the specific rounding process used on the elements of the vector or matrix $v$. Write down both a mathematical model, call it (P), and the corresponding directed graph. Your model should also guarantee consistency between the sum of the row sums and the sum of the column sums,

$$e^T \text{round}(Be) = e^T \text{round}(B^T e).$$

2. Transform the model for Item 1 (if needed) to formulate the mathematical model as a *max-flow* or *network flow* problem.

3. Write a MATLAB code that takes as input a given $m \times n$ matrix $B$ with elements correct to single precision (say 8 decimal accuracy), and outputs round($B$) with consistently rounded elements. (You can assume that the numbers are *nice enough* so that your computer can round the elements and sums correctly.) Use linprog to solve the linear programming model. Your MATLAB code should be able to:

   (a) emphasize rounding up so that the total sum $e^T \text{round}(B)e$ is a maximum;

   (b) emphasize rounding down so that the total sum $e^T \text{round}(B)e$ is a minimum;

   (c) (BONUS-1) find a consistently rounded matrix for each of the possible values of the total sum $e^T \text{round}(B)e$.

   Test the MATLAB code on the given matrices $B, \bar{B}$ above.

4. Use the F-F algorithm or network simplex method with phase I to solve problem (P) with the given matrix $B$, above.

5. **(BONUS-2)** Suppose that $A$ is a general $m \times n$ matrix with elements correct to single precision. (You can assume again that the numbers are *nice enough* so that your computer can round the elements and sums correctly.) Prove that the model (P) for consistent rounding from Item 1, above, always has a feasible solution; or, provide a counterexample.

## 2 Max-flow, Min-cut via Duality

For the maximum flow problem, let $\lambda_i$ be a (shadow) price variable for the conservation of flow constraint associated with node $i$. Let $\omega_{ij}$ be a (shadow) price variable associated with the capacity constraint for arc $ij$. Use the *game theory* approach introduced in class to derive a dual problem to the maximum flow problem. Prove the max-flow min-cut theorem with the aid of this dual problem.

## 3 Permutation and Doubly Stochastic Matrices

*(Birkhoff-von Neumann Theorm)* A nonnegative $n \times n$ matrix $A$ is *doubly stochastic* if $Ae = A^T e = e$, the vector of ones. A $\{0,1\}$ $n \times n$ matrix $P$ is a *permutation matrix* if each row and column has exactly one element equal to 1.

1. Let $P_i, i = 1, \ldots, k$ be permutation matrices and let $\lambda_i, i = 1, \ldots, k$ be nonnegative scalars with sum equal to 1. Prove that $\sum_{i=1}^{n} \lambda_i P_i$ is a doubly stochastic matrix.

2. Let $A$ be a doubly stochastic matrix. Show that there exist permutation matrices $P_1, \ldots, P_k$ and nonnegative scalars $\lambda_1, \ldots, \lambda_k$ that sum to 1, such that $A = \sum_{i=1}^{n} \lambda_i P_i$. Conclude that $A$ is a doubly stochastic matrix if, and only if, $A$ is a (finite) convex combination of permutation matrices. (*Hint:* Consider the assignment problem.)

## 4 Convergence of the Bellman-Ford Algorithm

*(Text Section. 7.9)* Assume that every cycle in the graph in nonnegative.

1. Prove that $p(t) \geq p^*, \forall t$, and conclude that $p(t)$ has a limit.

2. Prove that $p(t)$ can take only a finite number of values and therefore converges after a finite number of steps.

3. Prove that the limit satisfies Bellman's equation.

4. Prove that the algorithm converges to $p^*$.