CS 525 - Spring 2015 - Homework 9

assigned 11/4/15 - due in class Friday 11/13/15

1. We see in the book how to express the assignment problem (in which we seek the optimal matching between two sets of objects $\mathcal{N}_1$ and $\mathcal{N}_2$ of equal size) as a linear program. Rather than force all the data objects into the usual notation $\mathcal{A}$, b, p, however, we can state this problem elegantly if we define the unknown to be an $n \times n$ *matrix* X, where at the solution, we have $X_{ij} = 1$ if there is a "match" between element i of the set $\mathcal{N}_1$ and element j of the set $\mathcal{N}_2$, and zero otherwise. Accordingly, we can define the "preference matrix" C to be an $n \times n$ matrix for which C_{ij} denotes the desirability of matching element i of the set $\mathcal{N}_1$ with element j of the set $\mathcal{N}_2$. (Lower values of C indicate preferred matches.)

   In this notation, we can express the assignment problem as a linear program, as follows:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij} \text{ subject to } X^T e = e, \; Xe = e, \; 0 \le X \le 1,$$

   where $e = (1, 1, \ldots, 1)^t$ as usual. The final bound constraint indicates that all elements of X need to lie between 0 and 1. The other constraints mean that all the rows and all the columns of X need to sum to 1. As mentioned above, this is equivalent to the formulation discussed in class, but more graceful.

   Write a code using Matlab and CVX to solve the assignment problem using the formulation above. Apply it to the following preference matrix C:

$$C = \begin{bmatrix} 3 & 2.1 & 2 & 5 & 6 \\ 2 & 2 & 1.2 & 4 & 1.9 \\ 1.4 & 1.3 & 2 & 2.1 & 6 \\ 1.8 & 2.5 & 3.8 & 1.2 & 5.1 \\ 0.7 & 2.1 & 2.5 & 4.3 & 4.2 \end{bmatrix}.$$

Have your code write out the optimal matchings in a format like this:

$$(1, 4), (2, 3), (3, 1), (4, 2), (5, 5)$$

and print out the final objective value.

In addition, generate random preference matrices of size $n \times n$ with elements from a uniform $[0, 1]$ distribution. (Use Matlab's `rand()` command.) Use your code to solve these systems for $n = 5$, $n = 10$, $n = 20$.

2. For the assignment problem above, prove that the optimal matching does not change if we add an arbitrary constant number α to all elements of the matrix C (that is, we replace C by $C + \alpha ee^T$).

3. Do Exercise 5-2-3. You can use the code `rsm()`, but you need to choose a feasible initial basis B. (Data for this problem can be loaded from the file `ex5-2-3.mat`.)

4. Do Exercise 5-2-8. You can use the code `rsmbdd()`. (Data for this problem can be loaded from the file `ex5-2-8.mat`.)