Homework Assignment 1

Fall 2024

IE 523 Financial Computing Due on Tuesday 09/17 (at 11:59 pm)

In all of the following questions, you will need to use the `MyMatrix` class that we worked on during class. You may decide to create your own class, inspired from what we worked on in class, if you prefer.

As a reminder, during class (Lectures 2 and 3), we worked on a class that defines a two-dimensional matrix (let it be A) with elements $a_{ij}, 1 \leq i \leq m, 1 \leq j \leq n$. The class also allowed us to calculate the transpose of a matrix (A^T) and the determinant of a matrix. We now ask you to expand on that class in order to answer the following questions.

Question 1: Finding the rank of a matrix

Given a matrix $m \times n$ (that is, with m rows and n columns), we say that its *rank* is the maximum number of linearly independent columns (or rows) in the matrix. For example, matrix A_1 defined as

$$A_1 = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

has $\text{rank}(A_1) = 2$ because the last row (row 3) can be written as a linear combination of rows 1 and 2. Specifically, if a_i is the i-th row, then we can show that $a_3 = 2 \cdot a_2 - a_1$. On the other hand, matrix A_2 defined as

$$A_2 = \begin{bmatrix} 1 & 2 & 3 & 10 \\ 4 & 5 & 6 & 11 \\ 9 & 7 & 8 & 12 \end{bmatrix}$$

has $\text{rank}(A_2) = 3$. Hopefully it is easy to see that we always have $\text{rank}(A) \leq \min\{m, n\}$.

One possible way to find the rank of a matrix is to perform *Gaussian elimination*. The algorithm is summarized here:

1. Set $i = 1$. Also set rank $= n$ (the number of columns).
2. Select a_{ii}.
   (a) If $a_{ii} \neq 0$: find an appropriate multiplier so that you zero the elements in the i-th column. Update the elements in all of the following rows.
   (b) If $a_{ii} = 0$: check if there exists some element below a_{ii} in the i-th column that is nonzero. If there is, swap the rows and go back to Step 2. Otherwise, remove the column and swap in its place the last column in the matrix. Reduce the rank by 1. Go back to Step 2.
3. Set $i = i + 1$. If $i > rank$, stop. Otherwise, go back to Step 2.

Question 2: Solving systems of equations

Check our slides for this one!

One efficient way to solve a system of equations in the form of $Ax = b$ (where $A \in \mathbb{R}^{m \times m}$ and $b \in \mathbb{R}^{m \times 1}$) is to take the inverse of matrix A (termed A^{-1}) and set $x = A^{-1}b$.

Write the necessary code that will allow us to take the inverse of matrix A; additionally, you need to add code to your class that implements the multiplication between a two-dimensional matrix A and a suitable (one-dimensional) vector b.

Question 3: Enumerating all basic feasible solutions

In optimization, we mostly deal with systems of equations of the form $Ax = b$: that is, we look for *feasible* solutions that satisfy a series of constraints. In general, we have m rows (m constraints) and n columns (n variables). For example, consider the following system over four decision variables $x_1, x_2, x_3, x_4 \geq 0$:

$$x_1 + x_2 + x_3 = 6 \quad (1a)$$
$$x_2 + x_4 = 3 \quad (1b)$$

We can show that coefficient matrix A

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

has $\text{rank}(A) = 2$. Hence, the largest square submatrix that can produce a solution is $B \in \mathbb{R}^{2\times 2}$. We want to enumerate all basic feasible solutions, that is all possible solutions to the system that have at most $\text{rank}(A)$ non-zero decision variables (while the rest of the $n - \text{rank}(A)$ variables are zero).

Write the necessary code that will allow us to get all the basic feasible solutions. To achieve that, you will need a way to enumerate all possible square submatrices of dimensions $\text{rank}(A)\times\text{rank}(A)$ and check if they can produce a solution (that is, if the determinant is nonzero). Then, assuming the determinant is nonzero, check if the solution is nonnegative.

As an example, the system of equations in (1) (with rank equal to 2) has the following 6 submatrices of dimensions 2×2:

$$B_{12} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, B_{13} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}, B_{14} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B_{23} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, B_{24} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, B_{34} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

From these submatrices, B_{13} has determinant 0, so it cannot be a basic feasible solution. The rest produce:

$$B_{12} : x_B = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = B_{12}^{-1} b = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 3 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$B_{14} : x_B = \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = B_{14}^{-1} b = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 3 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \end{bmatrix}$$

$$B_{23} : x_B = \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} = B_{23}^{-1} b = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 6 \\ 3 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$B_{24} : x_B = \begin{bmatrix} x_2 \\ x_4 \end{bmatrix} = B_{24}^{-1} b = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 3 \end{bmatrix} = \begin{bmatrix} 6 \\ -3 \end{bmatrix}, \textbf{infeasible} \text{ as } x_4 < 0.$$

$$B_{34} : x_B = \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} = B_{34}^{-1} b = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 3 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \end{bmatrix}$$

Hence, our algorithm should return four points $sol_1 = (x_1, x_2, x_3, x_4) = (3, 3, 0, 0)$, $sol_2 = (x_1, x_2, x_3, x_4) = (6, 0, 0, 3)$, $sol_3 = (x_1, x_2, x_3, x_4) = (0, 3, 3, 0)$, and $sol_4 = (x_1, x_2, x_3, x_4) = (0, 0, 6, 3)$.