LP Solver With Simplex

I used Python as programming language and the Revised Simplex method to solve the given LP problem. Source code is well commented that every function, its parameters and return values are described elaborately.

Firstly I took **3 elementary row operations'** functions and **matrix reversing functions** from the previous project of ours. After that, I wrote a function to parse the input and create the corresponding matrix, rows and columns. I use a 2 dimensional list (list of lists of floats) to implement the matrix. While parsing the data for the first time, I place the slack variables as an Identity Matrix at the end of A matrix. I stored data in the **matrix_A, c_Row, b_Col** and got the values of **m** and **n**.

I wrote several helper functions in order to handle this problem in a modular way. **identify()** is to add slack variable matrix to the A matrix. **rowMult()** function to multiply two given rows. **getCol()** and **getColT()** to get specific column as a column or as a row which is transposed from column. **matrixMult()** to apply matrix multiplication. **getMatrix()** to retrieve wanted columns of a given matrix in a matrix form.

In order to reach the solution, I wrote a function namely **solve()** which uses Revised Simplex algorithm. In the function, I mimicked the exact definition in the lecture notes. First I determine basic variables and nonbasic variables sets. Then I started to iterate in the **while()** structure. I construct the needed rows and matrices as **cb**, **matrixB** and **B inverse** and **cb.bInverse .** Then I applied the **pricing out** process to determine which variable will enter the basis.

If there is no element to enter the basic set, that means I are in the result position of simplex. In that case, regarding values are calculated and after required controls, results will be printed in the console. In this state I can determine whether our optimum point is feasible or do I have multiple optima. I state those.

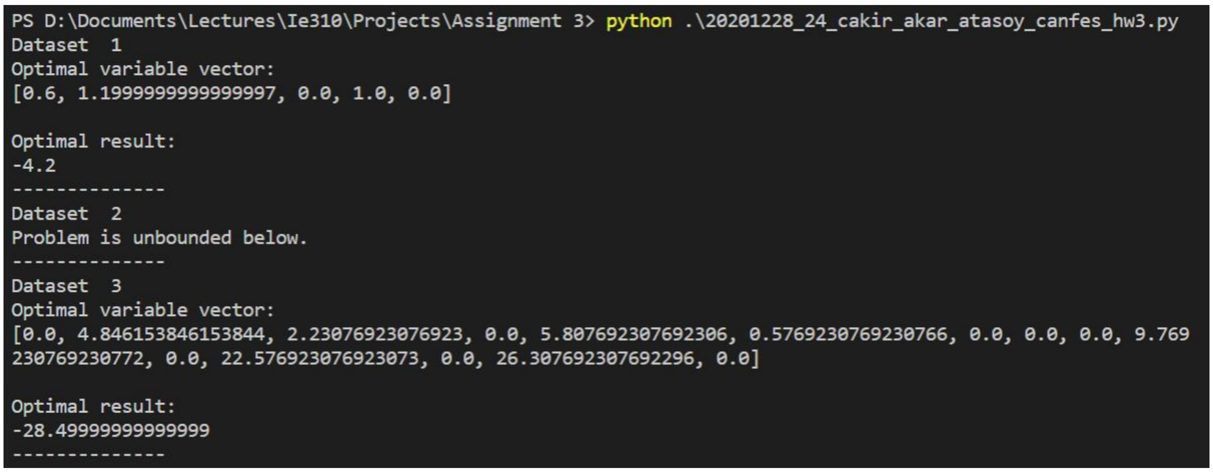
If there is a variable to enter the basic set, I continue the revised simplex algorithm. I now use the **ratio test** to determine the row in which corresponding variable should enter the basis. If there is no valid variable to enter the basic set, that means I have an unbounded below case.

After finding entering and leaving variables, I should update basic and nonbasic variables sets and proceed with iteration.

Our code is written in the most modular way and can be executed with any size of input matrix. No specific bugs are encountered. I haven't used any additional linear algebra library. Just pure Python code is written.

I didn't round the result values but it is a very easy operation.

Example output is below:



```
PS D:\Documents\Lectures\Ie310\Projects\Assignment 3> python .\20201228_24_cakir_akar_atasoy_canfes_hw3.py
Dataset  1
Optimal variable vector:
[0.6, 1.1999999999999997, 0.0, 1.0, 0.0]

Optimal result:
-4.2
--------------
Dataset  2
Problem is unbounded below.
--------------
Dataset  3
Optimal variable vector:
[0.0, 4.846153846153844, 2.23076923076923, 0.0, 5.807692307692306, 0.5769230769230766, 0.0, 0.0, 0.0, 9.769
230769230772, 0.0, 22.576923076923073, 0.0, 26.307692307692296, 0.0]

Optimal result:
-28.49999999999999
--------------
```