Lab 0 - simplex optimization

Yury Malyshev

September 24, 2019

Step 1 - import needed libraries. numpy is used for matrix manipulations. math is used for defining infinity csv is used for reading text files

[1]:
```
import numpy as np;
import math;
import csv;
```

[2]:
```
class Simplex:

    initialMatrix = [];
    initialDirectives = [];
    solutionMatrix = [];

    def __init__(self, filename):
        print("Init. Filename = " + filename);
        self.initialDirectives, self.initialMatrix = Simplex.readFile(filename);
        print(self.initialDirectives);
        Simplex.printArray(self.initialMatrix);


    def solve(self):
        print("Solving using simplex...");
        t_matrix, t_directives = self.checkObjective();
        t_matrix, t_directives = self.correctSigns(t_matrix, t_directives);
        self.solutionMatrix = self.correctSurplus( t_matrix, t_directives );
        self.solutionMatrix = self.maximize(self.solutionMatrix);
        print("="*25 + "Final Result" + "="*25 + "\n");
        Simplex.printArray(self.solutionMatrix);
        print("="*62);

    def checkObjective(self):
        matrix = np.copy(self.initialMatrix);
        directives = np.copy(self.initialDirectives);
        if directives[-1] == "min":
            for i in range(matrix.shape[0]):
                if directives[i] == "leq":
                    matrix[i] = np.multiply(matrix[i], -1.);
            temp = [];
            matrix = np.transpose(matrix);
            for i in range(1,matrix.shape[0]):
                temp.append("leq");
            temp.append("max");
            directives = temp;
        return matrix, directives;

    def correctSigns(self, matrix, directives):
        matrix[-1] = np.multiply(matrix[-1], -1.);
        for i in range(matrix.shape[0]-1):
            if matrix[i, -1] < 0.:
                if directives[i] == "geq":
                    directives[i] = "leq";
                elif directives[i] == "leq":
                    directives[i] = "geq";
                matrix[i] = np.multiply(matrix, -1.);
        return matrix, directives;

    def correctSurplus(self, matrix, directives):
        s = np.zeros((matrix.shape[0], matrix.shape[0]-1), dtype=np.float32);
        for i in range(matrix.shape[0]-1):
            if directives[i] == "leq":
                s[i, i] = 1;
            else:
                s[i,i] = -1;
        matrix = np.insert(matrix, [matrix.shape[1]-1], s, axis=1);
        return matrix;

    @staticmethod
    def divideArrays(array1, array2): #divides array1 by array2. If value in␣
↪array2 <= 0 -> result = infinity
        temp = np.zeros(len(array1));
        i = 0;
        for x in array2:
            if( x <= 0 ):
                temp[i] = math.inf;
            else:
                temp[i] = array1[i]/array2[i];
            i += 1;
        return temp;

    def maximize(self, matrix):
        while(min(matrix[-1]) < 0):
            print("Starting a new iteration with the following matrix: ");
            Simplex.printArray(matrix);
            entryColumn = np.argmin(matrix[-1]);
            temp = Simplex.divideArrays(matrix[0:-1, -1] , matrix[0:-1,␣
↪entryColumn]);
            entryRow = np.argmin(temp);
            print("Pivot point is at: [" + str(entryRow) + ";" +␣
↪str(entryColumn) + "]");
            matrix = Simplex.performElimiation(matrix, entryRow, entryColumn);
            print();
        return matrix;

    @staticmethod
    def printArray(matrix):
        for row in matrix:
            line = "   [";
            for el in row:
                line = line + "%7.1f " % (el);
            line = line + "]";
            print(line);
        print();

    @staticmethod
    def readFile(filepath):
        with open(filepath) as file:
            reader = csv.reader(file, delimiter=",");
            matrix = [];
            obj_fun = [];
            objective = None;
            directives = [];

            for row in reader:
                line = [];
                for el in row[1:]:
                    line.append(float(el));
                if(row[0] == "min" or row[0] == "max"):
                    obj_fun = line;
                    objective = row[0];
                else:
                    directives.append(row[0]);
                    matrix.append(line);

            matrix.append(obj_fun);
            directives.append(objective);
            matrix = np.asarray(matrix);

            return directives, matrix;

    @staticmethod
    def performElimiation(matrix, entryRow, entryColumn):
        matrix[entryRow] = matrix[entryRow]/matrix[entryRow][entryColumn];
        i = 0;
        while i < len(matrix):
            if(i != entryRow):
                matrix[i] = matrix[i] - matrix[i][entryColumn]*matrix[entryRow]
            i += 1;
        return matrix;
```

[3]:
```
v15min = Simplex("15min.txt");
v15max = Simplex("15max.txt");
```

```
Init. Filename = 15min.txt
['geq', 'leq', 'leq', 'min']
  [    1.0     1.0     3.0 ]
  [    1.0    -1.0     3.0 ]
  [    1.0     5.0    15.0 ]
  [    1.0     3.0     0.0 ]

Init. Filename = 15max.txt
['geq', 'leq', 'leq', 'max']
  [    1.0     1.0     3.0 ]
  [    1.0    -1.0     3.0 ]
  [    1.0     5.0    15.0 ]
  [    1.0     3.0     0.0 ]
```

[4]:
```
v15min.solve();
```

```
Solving using simplex...
Starting a new iteration with the following matrix:
  [    1.0    -1.0    -1.0     1.0     0.0     1.0 ]
  [    1.0     1.0    -5.0     0.0     1.0     3.0 ]
  [   -3.0     3.0    15.0     0.0     0.0    -0.0 ]

Pivot point is at: [0;0]

=========================Final Result=========================

  [    1.0    -1.0    -1.0     1.0     0.0     1.0 ]
  [    0.0     2.0    -4.0    -1.0     1.0     2.0 ]
  [    0.0     0.0    12.0     3.0     0.0     3.0 ]

==============================================================
```

[5]:
```
v15max.solve();
```

```
Solving using simplex...
Starting a new iteration with the following matrix:
  [    1.0     1.0    -1.0     0.0     0.0     3.0 ]
  [    1.0    -1.0     0.0     1.0     0.0     3.0 ]
  [    1.0     5.0     0.0     0.0     1.0    15.0 ]
  [   -1.0    -3.0     0.0     0.0     0.0    -0.0 ]

Pivot point is at: [0;1]

Starting a new iteration with the following matrix:
  [    1.0     1.0    -1.0     0.0     0.0     3.0 ]
  [    2.0     0.0    -1.0     1.0     0.0     6.0 ]
  [   -4.0     0.0     5.0     0.0     1.0     0.0 ]
  [    2.0     0.0    -3.0     0.0     0.0     9.0 ]

Pivot point is at: [2;2]

Starting a new iteration with the following matrix:
  [    0.2     1.0     0.0     0.0     0.2     3.0 ]
  [    1.2     0.0     0.0     1.0     0.2     6.0 ]
  [   -0.8     0.0     1.0     0.0     0.2     0.0 ]
  [   -0.4     0.0     0.0     0.0     0.6     9.0 ]

Pivot point is at: [1;0]

=========================Final Result=========================

  [    0.0     1.0     0.0    -0.2     0.2     2.0 ]
  [    1.0     0.0     0.0     0.8     0.2     5.0 ]
  [    0.0     0.0     1.0     0.7     0.3     4.0 ]
  [    0.0     0.0     0.0     0.3     0.7    11.0 ]

==============================================================
```