DM545 – Linear and Integer Programming

Answers to the Take-home Assignment, Winter 2025

In this assignmnent I will be using Tableau Prinitng function provided on github by Marco always including it as:

```
from util import tableau
```

I will also be using *numpy*, and *fractions* for matrix operations:

```
import numpy as np
from fractions import Fraction
```

Task 1

Subtask 1.a

In the following problem:

$$\begin{aligned} \max \ & 4x_1 + 5x_2 - 7x_3 \\ & -x_1 - x_2 + x_3 \leq 2, \\ & -5x_1 + 10x_3 \leq 10, \\ & x_1 \in [0, 5], \\ & x_2 \in [-1, 1], \\ & x_3 \in [-2, 2]. \end{aligned}$$

By inspection of the objective function:

$$\begin{aligned} 4x_1 &\Rightarrow 4 \text{ is positive } &\Rightarrow x_1 \text{ as large as possible,} \\ 5x_2 &\Rightarrow 5 \text{ is positive } &\Rightarrow x_2 \text{ as large as possible,} \\ -7x_3 &\Rightarrow -7 \text{ is negative } &\Rightarrow x_3 \text{ as small as possible.} \end{aligned}$$

Check potential solution formed by limits of interval bounds $[x_1, x_2, x_3] = [5, 1, -2]$:

$$\begin{aligned} \text{1st constraint: } & -x_1 - x_2 + x_3 \leq 2 \\ & -5 - 1 + (-2) \leq 2 \\ & -8 \leq 2 \quad \text{is feasible} \\ \text{2nd constraint: } & -5x_1 + 10x_3 \leq 10 \\ & -25 + 10(-2) \leq 10 \\ & -45 \leq 10 \quad \text{is feasible} \end{aligned}$$

$$\begin{aligned} \text{Objecive function value:} \\ 4x_1 + 5x_2 - 7x_3 = \\ 20 + 5 + 14 = \\ 39 \end{aligned}$$

We conclude the optimal solution is $[x_1, x_2, x_3] = [5, 1, -2]$

Subtask 1.b

$$\begin{aligned}\max\ & x_1 + x_2 \\ & sx_1 + tx_2 \leq 1 \\ & x_1, x_2 \geq 0\end{aligned}$$

We can choose (s) and (t) to get different cases:

I) Single Optimal Solution

$$s = 2, \quad t = 1$$

- The slope of the only constraint is different from the slope of the objective function.
- Geometrically, the feasible region intersects the objective function at a single vertex.
- The vertex is our optimal solution

II) Infinite Optimal Solutions

$$s = 1, \quad t = 1$$

- The constraint line is parallel to the objective function.
- Objective function “placed” on the face gives us infinitely many optimal solutions.

III/IV) Infeasible / Unbounded

- If either s or t (or both) are negative, the problem becomes unbounded as each variable balances the other in the constraint allowing the objective function to grow indefinitely.

$$s = -1, \quad t = 1 \text{ is an example of unbounded}$$

- For any $s, t \geq 0$, setting $x_1 = \frac{1}{s}$, $x_2 = \frac{1}{t}$ gives a feasible solution (except if $s = 0$ or $t = 0$, in which case the value of x_1 or x_2 does not matter as it's multiplied by 0).

It is **impossible** to set s, t s.t. the problem becomes *infeasible*, we considered all posssible cases

Task 2

Subtask 2.a

First, we transform the problem into the standard form:

- Change a minimization into a maximization:
  $\min(c^T x)$ into $-\max -(c^T x)$.
- Replace equalities with two inequalities.

- Convert all constraints to "$\leq$".

The result is:

$$\begin{aligned} \max -3x_1 - 2x_2 - 7x_3 \\ -x_1 + x_2 \leq 10, \\ x_1 - x_2 \leq -10, \\ -2x_1 + x_2 - x_3 \leq -10. \end{aligned}$$

Then the tableau looks like:

```
SLACK_COUNT = 3

A = np.array([[-1, -1, 0],
              [1,  -1, 0],
              [-2, 1, -1]], dtype=object)
C = np.array([-3, -2, -7], dtype=object)
B = np.array([10, -10,-10], dtype=object)

A = np.vectorize(Fraction)(A)
C = np.vectorize(Fraction)(C)
B = np.vectorize(Fraction)(B)

I = np.array([[Fraction(int(i == j)) for j in range(SLACK_COUNT)] \
for i in range(SLACK_COUNT)], dtype=object)
Z = np.array([Fraction(0) for _ in range(SLACK_COUNT)], dtype=object)

T = np.concatenate([A.T, I], axis=1)
T = np.column_stack((T, Z))
T = np.column_stack((T, B))
T = np.vstack((T, np.concatenate([C, np.full(SLACK_COUNT, Fraction(0), \
dtype=object), [Fraction(1), Fraction(0)]])))

print("\n \n") # for pdf formating

tableau(T)
```

x1	x2	x3	x4	x5	x6	-z	b
-1	1	-2	1	0	0	0	10
-1	-1	1	0	1	0	0	-10
0	0	-1	0	0	1	0	-10
-3	-2	-7	0	0	0	1	0

As we can see the tableau is **optimal** (no positive reduced costs) but **infeasible** (two of the $b_i \leq 0$), we could apply Dual Simplex to work towards feasibility

Subtask 2.b

Tableau is given:

```
T = np.array([[0,0,0,1,1,0,0,0],
             [0,1,1,2,0,-1,0,30],
             [1,0,1,1,0,-1,0,20],
             [0,0,2,-7,0,5,1,-120]],dtype=object)

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |     b |
|-------+-------+-------+-------+-------+-------+-------+-------+
|     0 |     0 |     0 |     1 |     1 |     0 |     0 |     0 |
|     0 |     1 |     1 |     2 |     0 |    -1 |     0 |    30 |
|     1 |     0 |     1 |     1 |     0 |    -1 |     0 |    20 |
|-------+-------+-------+-------+-------+-------+-------+-------+
|     0 |     0 |     2 |    -7 |     0 |     5 |     1 |  -120 |
|-------+-------+-------+-------+-------+-------+-------+-------+
```

It is worth noting that this tableau is *unbounded* as all coefficients of x_6 in A are negative, but x_6 has a positive reduced cost, however we can still technically perform a change of basis and see the results.

By largest coefficient x_6 would have to enter (and x_3 leave as it's constraint is tighter) and that would make the problem infeasible, and unoptimal

```
# III * -1
T[2] = T[2] * Fraction(-1, 1)

# II + III
T[1] = T[1] + T[2]

# IV - 5*III
T[3] = T[3] - 5*T[2]

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |     b |
|-------+-------+-------+-------+-------+-------+-------+-------+
|     0 |     0 |     0 |     1 |     1 |     0 |     0 |     0 |
|    -1 |     1 |     0 |     1 |     0 |     0 |     0 |    10 |
|    -1 |     0 |    -1 |    -1 |     0 |     1 |     0 |   -20 |
|-------+-------+-------+-------+-------+-------+-------+-------+
|     5 |     0 |     7 |    -2 |     0 |     0 |     1 |   -20 |
|-------+-------+-------+-------+-------+-------+-------+-------+
```

By ratio test x_3 enters (as x_6 has only negative coefficients in A) and x_1 leaves as $20/1 < 30/1$ (we ignore first line $0/0 = ?$).

Coincidentally by Bland's Rule x_3 enters and x_1 leaves (we take the lowest index)

We can follow with both of them:

```
T = np.array([[0,0,0,1,1,0,0,0],
             [0,1,1,2,0,-1,0,30],
             [1,0,1,1,0,-1,0,20],
             [0,0,2,-7,0,5,1,-120]],dtype=object)


# II - III
T[1] = T[1] - T[2]

# IV - 2*III
T[3] = T[3] - 2*T[2]

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |     b |
|-------+-------+-------+-------+-------+-------+-------+-------+
|     0 |     0 |     0 |     1 |     1 |     0 |     0 |     0 |
|    -1 |     1 |     0 |     1 |     0 |     0 |     0 |    10 |
|     1 |     0 |     1 |     1 |     0 |    -1 |     0 |    20 |
|-------+-------+-------+-------+-------+-------+-------+-------+
|    -2 |     0 |     0 |    -9 |     0 |     7 |     1 |  -160 |
|-------+-------+-------+-------+-------+-------+-------+-------+
```

As we can see the tableau is still unbounded (by the case of x_6)

Task 3

Subtask 3.a

Tableau given:

```
T = np.array([[1,0,1,-1,0,5],
              [0,1,-2,3,0,15],
              [0,0,-2,-2,1,-110]],dtype=object)

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    -z |     b |
|-------+-------+-------+-------+-------+-------+
|     1 |     0 |     1 |    -1 |     0 |     5 |
|     0 |     1 |    -2 |     3 |     0 |    15 |
|-------+-------+-------+-------+-------+-------+
|     0 |     0 |    -2 |    -2 |     1 |  -110 |
|-------+-------+-------+-------+-------+-------+
```

From the tableau, we observe the following:

- The solution is $[x_1, x_2, x_3, x_4] = [5, 15, 0, 0]$ with objective value 110.
- The reduced costs are $-2, -2$
- The values of dual variables are $2, 2$ (negative reduced costs).
- The shadow prices are the same as the dual variables: $2, 2$
- There is no over-capacity, as all the constraints are tight (no slacks in the basis).

Task 4

Subtask 4.a

We denote original problem as **P** and relaxed original problem as **PR** Let's start by considering the original problem with marked constraints by α, β, γ

$$
\begin{aligned}
\min \; & \sum_{i=1}^{n} c_i y_i \\
& \sum_{j=1}^{m} a_{ij} x_{ij} \leq b_i y_i, \quad i = 1, \dots, n \quad (\alpha) \\
& \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, \dots, m \quad (\beta) \\
& y_i \leq 1, \quad i = 1, \dots, n \quad (\gamma) \\
\\
& y_i \geq 0, \quad x_{ij} \geq 0.
\end{aligned}
$$

We denote the potential dual variables:

$$
\begin{aligned}
\alpha &= [\alpha_1, \dots, \alpha_n] \\
\beta &= [\beta_1, \dots, \beta_m] \\
\gamma &= [\gamma_1, \dots, \gamma_n]
\end{aligned}
$$

Then measure the violation of constraints (by putting everything to one side and multiplying by corresponding dual variable):

$$
\begin{matrix}
\alpha_1 \left(0 + b_1 y_1 - \sum_{j=1}^{m} a_{1j} x_{1j}\right) & & \beta_1 \left(1 - \sum_{i=1}^{n} x_{i1}\right) & & \gamma_1 (1 - y_1) \\
\alpha_2 \left(0 + b_2 y_2 - \sum_{j=1}^{m} a_{2j} x_{2j}\right) & || & \beta_2 \left(1 - \sum_{i=1}^{n} x_{i2}\right) & || & \gamma_2 (1 - y_2) \\
\vdots & & \vdots & & \vdots \\
\alpha_n \left(0 + b_n y_n - \sum_{j=1}^{m} a_{nj} x_{nj}\right) & & \beta_m \left(1 - \sum_{i=1}^{n} x_{im}\right) & & \gamma_n (1 - y_n)
\end{matrix}
$$

Then we denote **PR** relaxed problem:

$$\mathrm{PR}(\alpha,\beta,\gamma) = \min_{\text{by all } y,x\geq 0} \left\{ \begin{aligned} c_1 y_1 + \dots + c_n y_n + \\ \alpha_1\,(b_1 y_1 - \sum_{j=1}^{m} a_{1j} x_{1j}) + \\ \alpha_2\,(b_2 y_2 - \sum_{j=1}^{m} a_{2j} x_{2j}) + \\ \vdots \\ \alpha_n\,(b_n y_n - \sum_{j=1}^{m} a_{nj} x_{nj}) + \\ \beta_1\,(1 - \sum_{i=1}^{n} x_{i1}) + \\ \beta_2\,(1 - \sum_{i=1}^{n} x_{i2}) + \\ \vdots \\ \beta_m\,(1 - \sum_{i=1}^{n} x_{im}) + \\ \gamma_1\,(1 - y_1) + \\ \gamma_2\,(1 - y_2) + \\ \vdots \\ \gamma_n\,(1 - y_n) \end{aligned} \right\} \Rightarrow \min_{\text{by all } y,x\geq 0} \left\{ \begin{aligned} y_1\,(c_1 + \alpha_1 b - \gamma_1) + \\ y_2\,(c_2 + \alpha_2 b - \gamma_2) + \\ \vdots \\ y_1\,(c_n + \alpha_n b - \gamma_n) + \\ x_{1,1}\,(0 - \alpha_n a_1 - \beta_1) + \\ x_{2,1}\,(0 - \alpha_2 a_1 - \beta_2) + \\ \vdots \\ x_{n,1}\,(0 - \alpha_n a_1 - \beta_n) + \\ x_{1,2}\,(0 - \alpha_1 a_2 - \beta_1) + \\ x_{2,2}\,(0 - \alpha_2 a_2 - \beta_2) + \\ \vdots \\ x_{n,m}\,(0 - \alpha_n a_m - \beta_n) + \\ \sum_{j=1}^{m} \beta_j + \\ \sum_{i=1}^{n} \gamma_i + \end{aligned} \right\}.$$

All bounds have to be usefull, so to get to the dual constraints:

$$\begin{aligned} (c_1 + \alpha_1 b - \gamma_1) &\geq 0 \\ (c_2 + \alpha_2 b - \gamma_2) &\geq 0 \\ &\vdots \\ (c_n + \alpha_n b - \gamma_n) &\geq 0 \\ (0 - \alpha_n a_1 - \beta_1) &\geq 0 \\ (0 - \alpha_2 a_1 - \beta_2) &\geq 0 \\ &\vdots \\ (0 - \alpha_n a_1 - \beta_n) &\geq 0 \\ (0 - \alpha_1 a_2 - \beta_1) &\geq 0 \\ (0 - \alpha_2 a_2 - \beta_2) &\geq 0 \\ &\vdots \\ (0 - \alpha_n a_m - \beta_n) &\geq 0 \end{aligned} \quad \Rightarrow \quad \begin{aligned} \alpha_1 b - \gamma_1 &\geq -c_1 \\ \alpha_2 b - \gamma_2 &\geq -c_2 \\ &\vdots \\ -\alpha_n b - \gamma_n &\geq 0 \\ -\alpha_n a_1 - \beta_1 &\geq 0 \\ -\alpha_2 a_1 - \beta_2 &\geq 0 \\ &\vdots \\ -\alpha_n a_1 - \beta_n &\geq 0 \\ -\alpha_1 a_2 - \beta_1 &\geq 0 \\ -\alpha_2 a_2 - \beta_2 &\geq 0 \\ &\vdots \\ -\alpha_n a_m - \beta_n &\geq 0 \end{aligned}$$

To get new Objective function we take the sums uncorrelated with y, x in front of the $\min_{\text{by all } y,x} \{\vdots\}$. Now we want to:

$$\max_{\alpha,\beta,\gamma} \left\{ \mathrm{PR}(\alpha,\beta,\gamma) = \sum_{j=1}^{m} \beta_j + \sum_{i=1}^{n} \gamma_i + \min_{\text{by all } y,x} \{\vdots\} \right\}.$$

However to keep the:

$$\mathrm{opt}(\mathrm{PR}(\alpha,\beta,\gamma)) \;\leq\; \mathrm{opt}(P)$$

We need to penalize breaking the constraints, i.e. each dual variable must be chosen so that **violating a original constraint increases the value of the objective**.

For the constraints of type α: $\sum_{j=1}^{m} a_{ij}x_{ij} \leq b_i y_i$

- The violation measure is
  $b_i y_i - \sum_{j=1}^{m} a_{ij}x_{ij}$.
- When **broken**, this becomes **negative**.
- To penalize the objective it requires

$$\alpha_i \leq 0.$$

For the constraints of type γ: $y_i \leq 1$:

- The violation measure is
  $1 - y_i$.
- When **breaking** this becomes **negative**.
- To ensure the penalty, we need

$$\gamma_i \geq 0.$$

For the constraints of type β $\sum_{i=1}^{n} x_{ij} = 1$:

- Violations can happen **in both ways**.
- Therefore we set:

$$\beta_j \in \mathbb{R}.$$

This ensures that always:

$$\text{opt}(\text{PR}(\alpha, \beta, \gamma)) \leq \text{opt}(P)$$

Combinig everything togheter we are left with:

$$\begin{aligned}
\max\ & \sum_{j=1}^{m} \beta_j + \sum_{i=1}^{n} \gamma_i \\
& \alpha_1 b - \gamma_1 \geq -c_1 \\
& \alpha_2 b - \gamma_2 \geq -c_2 \\
& \vdots \\
& -\alpha_n b - \gamma_n \geq 0 \\
& -\alpha_n a_1 - \beta_1 \geq 0 \\
& -\alpha_2 a_1 - \beta_2 \geq 0 \\
& \vdots \\
& -\alpha_n a_1 - \beta_n \geq 0 \\
& -\alpha_1 a_2 - \beta_1 \geq 0 \\
& -\alpha_2 a_2 - \beta_2 \geq 0 \\
& \vdots \\
& -\alpha_n a_m - \beta_n \geq 0 \\
\\
& \alpha_i \leq 0. \quad i = 1, \dots, n \\
& \beta_j \in \mathbb{R}. \quad j = 1, \dots, m \\
& \gamma_i \leq 0. \quad i = 1, \dots, n
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\max\ & \sum_{j=1}^{m} \beta_j + \sum_{i=1}^{n} \gamma_i \\
\\
& -b\alpha_i + \gamma_i \leq c_i \quad i = 1, \dots, n \\
& \alpha_i a_j + \beta_j \leq 0 \quad i = 1, \dots, n \qquad j = 1, \dots, m \\
\\
& \alpha_i \leq 0. \quad i = 1, \dots, n \\
& \beta_j \in \mathbb{R}. \quad j = 1, \dots, m \\
& \gamma_i \leq 0. \quad i = 1, \dots, n
\end{aligned}$$

Task 5

Subtask 5.a

Subtask 5.b

Subtask 5.c

Task 6

Subtask 6.a

Task 7

Subtask 7.a

Subtask 7.b

Task 8

Subtask 8.a

Subtask 8.b

Subtask 8.c

Subtask 8.d

Subtask 8.e

Subtask 8.f