Simplex Algorithm

(I) Pivot Algorithm

Pivot (N, B, A, b, c, v, l, e)

Compute the coefficients of the equation for new basic variable x_e

1 $\widehat{b_e} \leftarrow {}^{b_l}/_{b_{le}}$

2 **For each** $j \in (N - \{e\})$

3 $\quad \widehat{a_{ej}} \leftarrow {}^{a_{lj}}/_{a_{le}}$

4 $\widehat{a_{el}} \leftarrow {}^{1}/_{a_{le}}$

Compute the coefficients of the remaining constraints

5 **For each** $i \in (B - \{l\})$

6 $\quad \widehat{b_i} \leftarrow b_i - a_{ie}\widehat{b_e}$

7 $\quad$ **For each** $j \in (N - \{e\})$

8 $\quad\quad \widehat{a_{ij}} \leftarrow a_{ij} - a_{ie}\widehat{a_{ej}}$

9 $\quad \widehat{a_{il}} \leftarrow (-a_{ie}\widehat{a_{el}})$

Compute the objective function

10 $\hat{v} \leftarrow v + c_e\widehat{b_e}$

11 **For each** $j \in (N - \{e\})$

12 $\quad \widehat{c_j} \leftarrow c_j - c_e\widehat{a_{ej}}$

13 $c_l \leftarrow (-c_e\widehat{a_{el}})$

Compute new sets of basic and non − basic variables

14 $\hat{N} \leftarrow (N - \{e\}) \cup \{l\}$

15 $\hat{B} \leftarrow (B - \{l\}) \cup \{e\}$

16 **Return** $(\hat{N}, \hat{B}, \hat{A}, \hat{b}, \hat{c}, \hat{v})$

e : *Index of the Entering variable*
l : *Index of the Leaving variable*
x_e : *Entering variable*
x_l : *Leaving variable*
N : *Set of Non − basic variables*
B : *Set of Basic variables*

**Green* colour text is code comments.

(2) Simplex Algorithm

```
Simplex(A, b, c)
1   (N, B, A, b, c, v) ← INITIALIZE_SIMPLEX(A, b, c)
2   While some index j ∈ N has c_j > 0
3       Choose an index e ∈ N for which c_e > 0
4       For each i ∈ B
5           If a_ie > 0
6               Δ_i ← b_i / a_ie
7           else
8               Δ_i ← ∞
9       Choose and index l ∈ B that minimizes Δ_i
10      If Δ_l = ∞
11          Return "unbounded"
12      else
13          (N, B, A, b, c, v) ← PIVOT(N, B, A, b, c, v, l, e)
14  For i ← 1 to n
15      If i ∈ B
16          x̂_i ← b_i
17      else
18          x̂_i ← 0
19  Return (x̂_1, x̂_2, ..., x̂_n)
```

Increase x_e without violating constraints

INITIALIZE_SIMPLEX :

A function that takes Standard Form of a LPP as input and, returns the Slack Form of the LPP, if the LPP is Feasible returns a message and terminates, if the LPP is Infeasible

Δ_i: *The value for* x_e *which makes the* i^{th} *constraint* ***Tight*** *and increases* Z

n : *Number of elements in* $(B \cup N)$