Simplex Algorithm Overview

- Simplex algorithm is an algorithm to solve linear programming problems.

- The linear programming problems we are to solve have a special constraint on the decision varibles requiring them to be non-negative.

Example

$$maximize\ 3x_1 + 2x_2 + 5x_3$$

$$\begin{aligned} & 2x_1 + 3x_2 + 6x_3 \geqslant 5 \\ s.t.\ & 7x_1 + 5x_2 + 2x_3 \leqslant 1 \\ & 3x_1 + 4x_2 = 2 \end{aligned}$$

$$\forall i \in \{1, 2, 3\}, x_i \geqslant 0$$

Transformation

$$maximize\ 3x_1 + 2x_2 + 5x_3$$

$$\begin{aligned} & 2x_1 + 3x_2 + 6x_3 - x_4 + a_1 = 5 \\ s.t.\ & 7x_1 + 5x_2 + 2x_3 + x_5 = 1 \\ & 3x_1 + 4x_2 + a_2 = 2 \end{aligned}$$

The added x variables are called slack variables.

The added a variables are called auxiliary/artificial variables.

Notations

- $\vec{c_b^T}$: Coefficient vector of Basic Variables (Variables that have non zero value) in the Objective Function

- $\vec{c_n^T}$: Coefficient vector of Non-Basic Variables (Variables are zero) in the Objective Function

- B: Coefficient matrix of Basic Variables in the constraints

- N: Coefficient matrix of Non-Basic Variables in the constraints

- $\vec{b}$: Vector of the RHS values in the constraints

- $\vec{x_b}$: Vector of Basic Variables

- $\vec{x_n}$: Vector of Non-Basic Variables

Matrix Representation

$$maximize\ \vec{c_b^T} \cdot \vec{x_b} + \vec{c_n^T} \cdot \vec{x_n}$$

$$s.t.\ B\vec{x_b} + N\vec{x_n} = \vec{b}$$

The Objective function can also be written as:

$$c_b^T B^{-1}\vec{b} + (\vec{c_n^T} - \vec{c_b^T} B^{-1} N)\vec{x_n}$$

The goal of the simplex algorithm is to set all dimensions of the vector $\vec{c_n^T} - \vec{c_b^T} B^{-1} N$ to be negative by swapping Basic and Non-Basic Variables (pivot).

The objective value is $c_b^T B^{-1}\vec{b}$

Each pivot will “improve” the objective value

Two Phases

- If $\exists$ Artificial Variable(s),"Phase One" is required
- The objective of "Phase One" is to maximize $-\sum_i a_i$
- If the optimal value of "Phase One" $\neq 0$, the linear program is infeasible; otherwise, it is feasible

Implementation

- Single phase solver – Finished

- “Phase One” builder – Work in Process

```
struct Model {
    int is_max;
    int stat;
    int num_non_basic;
    int num_basic;
    int num_art;
    double** b_matrix;
    double** n_matrix;
    double* cb_vector;
    double* cn_vector;
```

```
    int* xb_index_vector;
    int* xn_index_vector;
    double* b_vector;
    double** art_matrix;
    int* art_ind_vector;
};
```

```
int pivot(Model* model, int in, int out);
int check(Model* model, int* in, int* out);
int ratio_check(Model* model, int in_ind,
                int* ind);
```

Sample I/O

Input

```
3 3
1 10 3 5
2 4 -4 0 4
1 -20 -5 0 7
4 2 3 0 10
```

$$
\begin{aligned}
& maximize\ 10x_1 + 3x_2 + 5x_3 \\
& \\
& \begin{aligned}
2x_1 + 4x_2 - 4x_3 &\leqslant 5 \\
s.t.\ 1x_1 - 20x_2 - 5x_3 &\leqslant 1 \\
4x_1 + 2x_2 + 3x_3 &\leqslant 2
\end{aligned}
\end{aligned}
$$

Output

```
x_1 = 2.363636

x_3 = 0.181818

x_2 = 0.000000

opt_val = 24.545455
```

Output from Xpress IVE

```
2

Optimum found

x_1 = 2.36364

x_2 = 0

x_3 = 0.181818

24.5455
```

Input

```
3 3
1 10 3 5
2 4 -4 0 4
1 -20 -5 0 7
4 2 -3 0 10
```

$$maximize\ 10x_1 + 3x_2 + 5x_3$$

$$\begin{aligned} 2x_1 + 4x_2 - 4x_3 &\leqslant 5 \\ s.t.\ 1x_1 - 20x_2 - 5x_3 &\leqslant 1 \\ 4x_1 + 2x_2 - 3x_3 &\leqslant 2 \end{aligned}$$

Output

```
Unbounded
```

Output from Xpress IVE

```
8

Unbounded

x_1 = 0

x_2 = 0

x_3 = 0

0
```