Applications of the Simplex Method

Caleb, Gavin

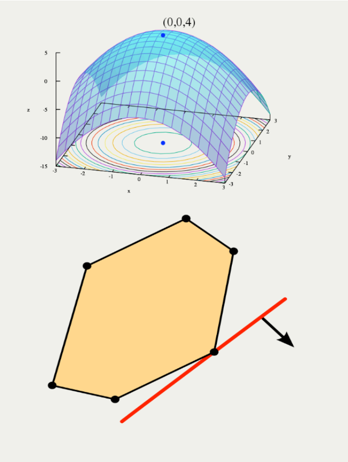
Macalester College
Saint Paul, MN

Outline

1. Introduction
1.1 Review
1.2 History
1.3 General Problem
2. Geometric Approach
3. Simplex Tableau
4. Implementation
4.1 Infeasible $\vec{0}$ Solutions
4.2 Alternative Operators
4.3 Cycling
4.4 Our Implementation
5. Applications
5.1 The Scheduling Problem
5.2 Traveling Salesman
6. Conclusion

Quick Review

- Optimization technique
- Constraints in the form of inequalities
- Feasible Region
  - Intersection of 'half spaces'
  - Convex polytope
- Solution techniques
  - Graphically
  - Simplex Method



History

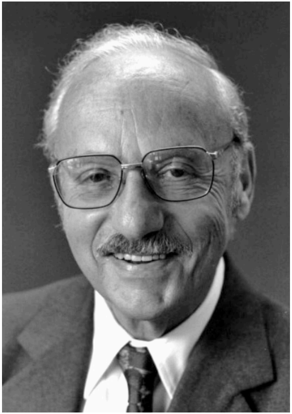


Figure: George Dantzig

Notoriously difficult to solve

- Joseph Fourier (1827): "Fourier-Motzkin" method
- Leonid Kantorovich (1939)
  - General formulation for linear programming problems
  - Expenditures and returns during WWII
- George Dantzig (1947): "Simplex Method"
  - Assigned 70 workers to 70 jobs

The General Linear Programming Problem

Objective Function:

$$z = c_1x_1 + c_2x_2 + ... + c_nx_n$$

Constraints of the form:

$$a_1x_1 + a_2x_2 + a_nx_n \begin{Bmatrix} \leq \\ = \\ \geq \end{Bmatrix} b$$

$$x_i \geq 0 \quad i \in \{1, ..., n\}$$

Definition

Feasible Solution: Any combination of variables such that all of the constraints are satisfied.

Full Standard Form

Maximize:

$$z = c_1 x_1 + c_2 x_2 + ... + c_n x_n$$

Subject to:

$$a_{11} x_{11} + a_{12} x_{12} + ... a_{1n} x_{1n} \leq b_1$$

$$\vdots$$

$$a_{m1} x_1 + a_{m2} x_2 + ... + a_{mn} x_n \leq b_n$$

$$x_i \geq 0 \text{ for } i \in 1, 2, ..., n$$

$$b_j \geq 0 \text{ for } j \in 1, 2, ..., m$$

Definition

Feasible Region: The set of all n-tuples that are feasible solutions

2-D Optimization

Maximize:

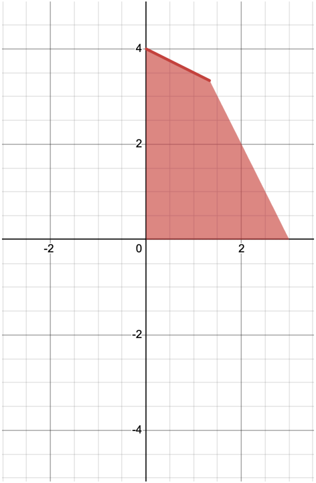
$$z = 3x_1 + 4x_2$$

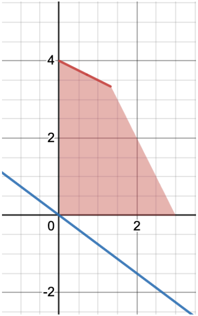
Subject to:

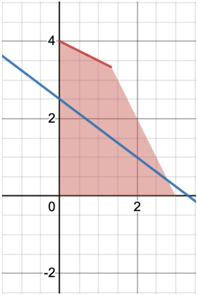
$$x_1 + 2x_2 \leq 8$$

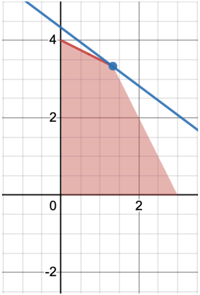
$$2x_1 + x_2 \leq 6$$

$$x_{1,2} \geq 0$$




4
2
0
2
-2


4
2
0
2
-2


4
2
0
2
-2

Simplex Tableau vs Algebraic Method

Step 1:

basis	z	x_1	x_2	s_1	s_2	b
s_1	0	1	2	1	0	8
s_2	0	2	1	0	1	6
	1	-3	-4	0	0	0

Step 2:

basis	z	x_1	x_2	s_1	s_2	b
x_2	0	$\frac{1}{2}$	1	$\frac{1}{2}$	0	4
s_2	0	$\frac{3}{2}$	0	$-\frac{1}{2}$	1	2
	1	-1	0	2	0	16

Step 1: $x_1 = x_2 = 0$

$$s_1 = 8 - x_1 - 2x_2$$

$$s_2 = 6 - 2x_1 - x_2$$

$$z = 0$$

Step 2: Switch x_2 and s_1

$$x_2 = 4 - \frac{1}{2}x_1 - \frac{1}{2}s_1$$

$$s_2 = 2 - \frac{3}{2}x_1 + \frac{1}{2}s_1$$

$$z = 4(x_2) + 3(x_1) = 16 + x_1 - 2s_1$$

Solution

With one more iteration...

basis	z	x_1	x_2	s_1	s_2	b
x_2	0	0	1	$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{10}{3}$
x_1	0	1	0	$-\frac{1}{3}$	$\frac{2}{3}$	$\frac{4}{3}$
	1	0	0	$\frac{5}{3}$	$\frac{2}{3}$	$17\frac{1}{3}$

$$x_1 = \frac{4}{3}$$

$$x_2 = \frac{10}{3}$$

$$z = 17\frac{1}{3}$$

Duality

What is the 'Dual' of L.P. in general form?

	Primal	Dual
variables	$x_1 \dots x_n$	$y_1 \dots y_m$
matrix	A	A^T
RHS	b	c
objective	max $c^T x$	min $b^T y$

The relationship between constraints is easier to picture in an example...

Example

Primal

Maximize:

$$z = 3x_1 + 4x_2$$

Subject to:

$$x_1 + 2x_2 \leq 8$$

$$5x_1 + 3x_2 \leq 6$$

$$4x_1 + 6x_2 \leq 3$$

$$x_{1,2} \geq 0$$

Dua1

Minimize:

$$z* = 8y_1 + 6y_2 + 3y_3$$

Subject to:

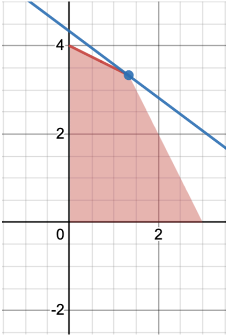
$$y_1 + 5y_2 + 4y_3 \geq 3$$

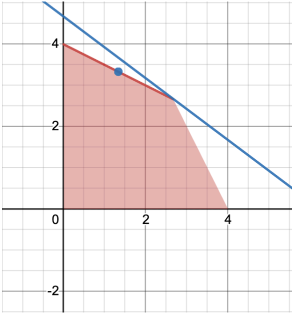
$$2y_1 + 3x_2 + 6y_3 \geq 4$$

$$y_{1,2} \geq 0$$

Why?

1. Sensitivity analysis
2. Establishes lower bound for Primal problem





*does not depict the previous slide

Problems

When we went to implement our own version of the simplex method, we ran into four problems we had to solve

1. Infeasible $\vec{0}$ Solution
2. Alternate Relational Operators ($=$, $\geq$)
3. Cycling

Infeasible $\vec{0}$ Solution

Guessing $\vec{0}$ as the basic feasible solution doesn't always work
Instead, we use the Two-Phase Simplex Method to find a basic feasible solution, then optimize the objective

Two-Phase Simplex Method

1. Add "Additional Variables" a_i to infeasible constraints
2. Ignoring the objective, minimize

$$\sum a_i$$

   - If $\sum_i a_i \neq 0$, the LP is infeasible
   - Otherwise, you've found a basic feasible solution
3. Remove the additional variables and solve the LP

Example

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 \\ \text{subject to} \quad & x_1 + 2x_2 \leq 8 \\ & 2x_1 + x_2 \leq 6 \\ & x_1 + x_2 \geq 2 \\ & x_{1,2} \geq 0 \end{aligned}$$

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 \\ \text{subject to} \quad & x_1 + 2x_2 + s_1 \leq 8 \\ & 2x_1 + x_2 + s_2 \leq 6 \\ & x_1 + x_2 - s_3 + a_1 \geq 2 \\ & x_{1,2} \geq 0 \end{aligned}$$

Example Cont.

Step 1: Make a Tableau

Step 2: Minimize

$$\sum a_i$$

Step 3: Run Simplex Iterations

basis	z	x_1	x_2	s_1	s_2	s_3	a_1	b
s_1	0	1	2	1	0	0	0	8
s_2	0	2	1	0	1	0	0	6
a_1	0	1	1	0	0	-1	1	2
z	1	-3	-4	0	0	0	0	0

basis	z	x_1	x_2	s_1	s_2	s_3	a_1	b
s_1	0	1	2	1	0	0	0	8
s_2	0	2	1	0	1	0	0	6
a_1	0	1	1	0	0	-1	1	2
z	1	0	0	0	0	0	1	0

basis	z	x_1	x_2	s_1	s_2	s_3	a_1	b
s_1	0	1	2	1	0	0	0	8
s_2	0	2	1	0	1	0	0	6
a_1	0	1	1	0	0	-1	1	2
z	1	-1	-1	0	0	1	0	-2

Example Cont.

Step 3 Cont.

basis	z	x_1	x_2	s_1	s_2	s_3	a_1	b
s_1	0	0	1	1	0	1	-1	6
s_2	0	0	-1	0	1	2	-2	2
x_1	0	1	1	0	0	-1	1	2
z	1	0	0	0	0	0	1	0

Step 4: Remove additional variables and plug in objective

basis	z	x_1	x_2	s_1	s_2	s_3	b
s_1	0	0	1	1	0	1	6
s_2	0	0	-1	0	1	2	2
x_1	0	1	1	0	0	-1	2
z	1	-3	-4	0	0	0	0

Step 5: Run Normal Simplex Iterations

basis	z	x_1	x_2	s_1	s_2	s_3	b
s_1	0	0	0	$\frac{1}{3}$	$\frac{1}{3}$	1	$\frac{8}{3}$
s_2	0	1	0	$-\frac{1}{3}$	$\frac{2}{3}$	0	$\frac{4}{3}$
a_1	0	0	1	$\frac{2}{3}$	$-\frac{1}{3}$	0	$\frac{10}{3}$
z	1	0	0	$\frac{5}{3}$	$\frac{2}{3}$	0	$\frac{52}{3}$

Optimal Solution: $x_1 = \frac{4}{3}$, $x_2 = \frac{10}{3}$

Alternate Operators

How can we deal with different relational operators?

- $(\leq)$ If $\sum_i a_{ij}x_i \leq b_j$, it becomes $\sum_i a_{ij}x_i + s_j = b_j$
- $(=)$ If $\sum_i a_{ij}x_i = b_j$, it becomes $\sum_i a_{ij}x_i + a_j = b_j$
- $(\geq)$ If $\sum_i a_{ij}x_i \geq b_j$, it becomes $\sum_i a_{ij}x_i - s_j + a_j = b_j$

Cycling

- Cycling is when you get to a previous tableau after performing simplex operations

Ex: If you pick the most negative value in the objective

basis	z	x_1	x_2	x_3	x_4	s_1	s_2	b
s_1	0	$\frac{1}{4}$	$-\frac{1}{8}$	12	10	1	0	0
s_2	0	$\frac{1}{10}$	$\frac{1}{20}$	$\frac{1}{20}$	$\frac{1}{5}$	0	1	0
z	1	-5	-4	20	2	0	0	0

$\Rightarrow$ After 6 Iterations

basis	z	x_1	x_2	x_3	x_4	s_1	s_2	b
s_1	0	$\frac{1}{4}$	$-\frac{1}{8}$	12	10	1	0	0
s_2	0	$\frac{1}{10}$	$\frac{1}{20}$	$\frac{1}{20}$	$\frac{1}{5}$	0	1	0
z	1	-5	-4	20	2	0	0	0

Bland's Rule

To stop cycling, we use Bland's Rule
Bland's Rule says we pick the first pivot of the possible choices in the objective

basis	z	x_1	x_2	x_3	x_4	s_1	s_2	b
s_1	0	$\frac{1}{4}$	$-\frac{1}{8}$	12	10	1	0	0
s_2	0	$\frac{1}{10}$	$\frac{1}{20}$	$\frac{1}{20}$	$\frac{1}{5}$	0	1	0
z	1	-5	-4	20	2	0	0	0

$\Rightarrow$

basis	z	x_1	x_2	x_3	x_4	s_1	s_2	b
x_2	0	2	1	1	14	20	0	0
s_2	0	$\frac{1}{2}$	0	$\frac{49}{4}$	$\frac{21}{1}$	$\frac{5}{2}$	1	0
z	1	3	0	24	18	80	0	0

Implementation

Using all of these, we made a Simplex Method implementation in Python/Numpy that uses Bland's Rule to stop cycling and can take any relation operation in its input

Scheduling Problem

- Three different types of employees
  - Managers ($25 per hour, 1.0 labor)
  - Regular workers ($18 per hour, 0.9 labor)
  - Trainees ($15 per hour, 0.5 labor)
- $b = \{3, 4, 5, 5, 4, 3, 3, 2\}$ Labor demand at hours (7am, ... 3pm)
- Constraints
  - Must have at least the minimum number of people necessary to cover demand
  - At least one manager working at all times
  - At least one trainee per day
  - At least as many regular workers as trainees
  - Each worker must work a 4 hour shift
- Minimize the cost of employees!

```
[[1.  0.9 0.5 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [1.  0.9 0.5 1.  0.9 0.5 0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [1.  0.9 0.5 1.  0.9 0.5 1.  0.9 0.5 0.  0.  0.  0.  0.  0. ]
 [1.  0.9 0.5 1.  0.9 0.5 1.  0.9 0.5 1.  0.9 0.5 0.  0.  0. ]
 [0.  0.  0.  1.  0.9 0.5 1.  0.9 0.5 1.  0.9 0.5 1.  0.9 0.5]
 [0.  0.  0.  0.  0.  0.  1.  0.9 0.5 1.  0.9 0.5 1.  0.9 0.5]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.9 0.5 1.  0.9 0.5]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.9 0.5]
 [1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [1.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [1.  0.  0.  1.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [1.  0.  0.  1.  0.  0.  1.  0.  0.  1.  0.  0.  0.  0.  0. ]
 [0.  0.  0.  1.  0.  0.  1.  0.  0.  1.  0.  0.  1.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  1.  0.  0.  1.  0.  0.  1.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0.  1.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0. ]
 [0.  0.  1.  0.  0.  1.  0.  0.  1.  0.  0.  1.  0.  0.  1. ]]
```

Solution

7am: 1 manager, 2 regular workers, 1 trainee

8am: 2 regular employees

9am: 2 regular employee

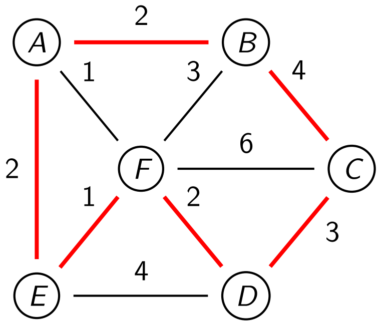
10am: nobody

11am: 1 manager, 2 regular employees

Cost: $\$200 + \$576 + \$60 = \$836$

Traveling Salesman

Given a graph with edge costs, find the least expensive cycle that visits every vertex



Naive Formulation

Let

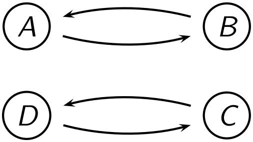
$$x_{ij} = \begin{cases} 1 \text{ if the edge from Node } i \text{ to Node } j \text{ is included} \\ 0 \text{ otherwise} \end{cases}$$

Then we want to

$$\begin{aligned} \text{minimize} \quad & \sum_{i=0}^{n} \sum_{j=0, j\neq i}^{n} c_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{i=0, i\neq j}^{n} x_{ij} = 1 \qquad \forall j \quad \text{Outflow from each node is 1} \\ & \sum_{j=0, j\neq i}^{n} x_{ij} = 1 \qquad \forall i \quad \text{Inflow to each ndoe is 1} \\ & x_{ij} \in \{0, 1\} \quad \forall i, j \end{aligned}$$

Naive Formulation Cont.

This can get disconnected cycles, which doesn't solve the TSP



MTZ Formulation

To address this, we use the Miller–Tucker–Zemlin Formulation
For every $i \geq 2$, add a u_i to the Decision Variables with the constraint that $u_j > u_i$ if $x_{ij} = 1$.
In the program, this means

$$u_i - u_j + nx_{ij} \leq n - 1$$

Therefore, the actual LP Formulation should be

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i=0}^{n} \sum_{j=0, j\neq i}^{n} c_{ij} x_{ij} & & \\
\text{subject to} \quad & \sum_{i=0, i\neq j}^{n} x_{ij} = 1 & \forall j & \quad \text{Outflow from each node is 1} \\
& \sum_{j=0, j\neq i}^{n} x_{ij} = 1 & \forall i & \quad \text{Inflow to each ndoe is 1} \\
& u_i - u_j + n x_{ij} \leq n - 1 & \forall i, j \geq 2 & \quad \text{MTZ Formulation} \\
& x_{ij} \in \{0, 1\} & \forall i, j &
\end{aligned}
$$

Example

For example, for a graph with cost matrix

	A	B	C	D
A		8	6	2
B	9		14	32
C	49	69		81
D	19	18	17	

The column is the origin and the row is the destination

Gets the cycle

$$A \to C \to B \to D \to A$$

Minimize:

```
[ 9 49 19  8 69 18  6 14 17  2 32 81  0  0  0]
```

$\vec{x}$

Subject To:

```
[[0. 0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0.]]
```

$\vec{x} =$

```
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
```

```
[[ 0.  0.  0.  0.  4.  0.  0.  0.  0.  0.  0.  0.  1. -1.  0.]
 [ 0.  0.  0.  0.  0.  4.  0.  0.  0.  0.  0.  0.  1.  0. -1.]
 [ 0.  0.  0.  0.  0.  0.  0.  4.  0.  0.  0.  0. -1.  1.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  4.  0.  0.  0.  0.  1. -1.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  4.  0. -1.  0.  1.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  4.  0. -1.  1.]]
```

$\vec{x} \leq$

```
3
3
3
3
3
3
```

Gets:

```
[0. 1. 0. 0. 0. 1. 0. 1. 0. 1. 0. 0. 2. 0. 3.]
```

Summary

What we did:

- Build a robust simplex implementation
- Look at two applications
  - Scheduling Problem
  - TSP

What we're working on:

- More examples
- Write up