Economic planning

Guns/nuclear-bombs/napkins/star-wars/professors/butter/mice problem

- Penguina: a country.
- Ruler need to decide how to allocate resources.
- Maximize benefit.
- Budget allocation
  - (i) Nuclear bomb has a tremendous positive effect on security while being expensive.
  - (ii) Guns, on the other hand, have a weaker effect.
- Penguina need to prove a certain level of security:
  \[ x_{\text{gun}} + 1000 \ast x_{\text{nuclear-bomb}} \geq 1000, \]
  where \( x_{\text{gun}} \): \# guns, \( x_{\text{nuclear-bomb}} \): \# nuclear-bombs constructed.
- \( 100 \ast x_{\text{gun}} + 1000000 \ast x_{\text{nuclear-bomb}} \leq x_{\text{security}} \)
  \( x_{\text{security}} \): total amount spent on security.
  \( 100/1,000,000 \): price of producing a single gun/nuclear bomb.

Linear programming

An instance of linear programming (LP):

- \( x_1, \ldots, x_n \): variables.
- For \( j = 1, \ldots, m \): \( a_{j1}x_1 + \ldots + a_{jn}x_n \leq b_j \): linear inequality.
- i.e., constraint.
- Q: \( \exists \) s an assignment of values to \( x_1, \ldots, x_n \) such that all inequalities are satisfied.
- Many possible solutions... Want solution that maximizes some linear quantity.
- **objective function**: linear inequality being maximized.
Linear programming – example

\[
\begin{align*}
\sum_{i=1}^{n} a_{i1} x_1 + \ldots + a_{in} x_n & \leq b_1 \\
\sum_{i=1}^{n} a_{i2} x_1 + \ldots + a_{in} x_n & \leq b_2 \\
& \vdots \\
\sum_{i=1}^{n} a_{im} x_1 + \ldots + a_{in} x_n & \leq b_m \\
\max \sum_{i=1}^{n} c_i x_i + \ldots + c_n x_n.
\end{align*}
\]

History

- 1939: L. V. Kantorovich noticed the importance of certain type of Linear Programming problems for resource allocation.
- 1947: Dantzig invented the simplex method for solving LP problems for the US Air force planning problems.
- 1947: T. C. Koopmans showed LP provide the right model for the analysis of classical economic theories.
- 1975: Koopmans and Kantorovich got the Nobel prize of economics.
- Kantorovich the only the Russian economist that got the Nobel prize.

Network flow via linear programming

Input: \(G = (V, E)\) with source \(s\) and sink \(t\), and capacities \(c(\cdot)\) on the edges. Compute max flow in \(G\).

\[
\begin{align*}
\forall (u \to v) \in E & \quad 0 \leq x_{u \to v} \\
\ & \quad x_{u \to v} \leq c(u \to v) \\
\forall v \in V \setminus \{s, t\} & \quad \sum_{(u \to v) \in E} x_{u \to v} - \sum_{(v \to w) \in E} x_{v \to w} \leq 0 \\
\ & \quad \sum_{(u \to v) \in E} x_{u \to v} - \sum_{(v \to w) \in E} x_{v \to w} \geq 0 \\
\text{maximizing} & \quad \sum_{(s \to u) \in E} x_{s \to u}
\end{align*}
\]

Part II

The Simplex Algorithm
Rewriting an LP

\[
\text{max} \quad \sum_{j=1}^{n} c_j x_j \\
\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, \ldots, m \\
x_j \geq 0 \quad \text{for } j = 1, \ldots, n.
\]

- Rewrite: so every variable is non-negative.
- Replace variable \( x_i \) by \( x'_i \) and \( x''_i \), where new constraints are: 
  \( x_i = x'_i - x''_i \), \( x'_i \geq 0 \) and \( x''_i \geq 0 \).
- Example: The (silly) LP \( 2x + y \geq 5 \) rewritten:
  \[2x' - 2x'' + y' - y'' \geq 5, \quad x' \geq 0, \quad y' \geq 0, \quad x'' \geq 0, \text{ and } y'' \geq 0.\]

Standard form of LP

A linear program in standard form.

\[
\text{max} \quad \sum_{j=1}^{n} c_j x_j \\
\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, \ldots, m \\
x_j \geq 0 \quad \text{for } j = 1, \ldots, n.
\]

Standard form of LP into standard form

**Lemma**

Given an instance \( I \) of LP, one can rewrite it into an equivalent LP, such that all the variables must be non-negative. This takes linear time in the size of \( I \).

An LP where all variables must be non-negative is in **standard form**

**LP in standard form.**

(Matric notation.)

\[
\begin{align*}
\text{max} \quad & c^T x \\
\text{s.t.} \quad & Ax \leq b. \\
& x \geq 0.
\end{align*}
\]

\( c \), \( b \) and \( A \): prespecified. 
\( x \) is vector of unknowns. 
Solve LP for \( x \).
Slack Form

1. Next rewrite LP into **slack form**.
2. Every inequality becomes equality.
3. All variables must be positive.
4. See resulting form on the right.

**New slack variables.** Rewrite inequality: $\sum_{i=1}^{n} a_i x_i \leq b$. As:

$$
\begin{align*}
  x_{n+1} &= b - \sum_{i=1}^{n} a_i x_i \\
  x_{n+1} &\geq 0.
\end{align*}
$$

Value of slack variable $x_{n+1}$ encodes how far is the original inequality for holding with equality.

Slack form formally

Because everything is clearer when you use tuples. Not.

The slack form is defined by a tuple $(N, B, A, b, c, v)$.

<table>
<thead>
<tr>
<th>$B$</th>
<th>Set of indices of basic variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>Set of indices of nonbasic variables</td>
</tr>
<tr>
<td>$n =</td>
<td>N</td>
</tr>
<tr>
<td>$b, c$</td>
<td>two vectors of constants</td>
</tr>
<tr>
<td>$m =</td>
<td>B</td>
</tr>
<tr>
<td>(i.e., number of inequalities)</td>
<td></td>
</tr>
<tr>
<td>$A = {a_{ij}}$</td>
<td>The matrix of coefficients</td>
</tr>
<tr>
<td>$N \cup B = {1, \ldots, n + m}$</td>
<td></td>
</tr>
<tr>
<td>$v$</td>
<td>objective function constant.</td>
</tr>
</tbody>
</table>
Example

Consider the following LP which is in slack form.

$$\text{max } z = 29 - \frac{1}{9}x_3 - \frac{1}{9}x_5 - \frac{2}{9}x_6$$
$$x_1 = 8 + \frac{1}{6}x_3 + \frac{1}{3}x_5$$
$$x_2 = 4 - \frac{2}{3}x_3 - \frac{1}{3}x_5$$
$$x_4 = 18 - \frac{1}{2}x_3 + \frac{1}{2}x_5$$

Another example...

$$\text{max } 5x_1 + 4x_2 + 3x_3$$
$$s.t. \quad 2x_1 + 3x_2 + x_3 \leq 5$$
$$4x_1 + x_2 + 2x_3 \leq 11$$
$$3x_1 + 4x_2 + 2x_3 \leq 8$$
$$x_1, x_2, x_3 \geq 0$$

Transform into slack form...

$$\text{max } z = 5x_1 + 4x_2 + 3x_3$$
$$s.t. \quad w_1 = 5 - 2x_1 - 3x_2 - x_3$$
$$w_2 = 11 - 4x_1 - x_2 - 2x_3$$
$$w_3 = 8 - 3x_1 - 4x_2 - 2x_3$$
$$x_1, x_2, x_3, w_1, w_2, w_3 \geq 0$$

Example

...translated into tuple form \((N, B, A, b, c, v)\).

\[ B = \{1, 2, 4\}, N = \{3, 5, 6\} \]
\[ A = \begin{pmatrix} a_{13} & a_{15} & a_{16} \\ a_{23} & a_{25} & a_{26} \\ a_{43} & a_{45} & a_{46} \end{pmatrix} = \begin{pmatrix} -1/6 & -1/6 & 1/3 \\ 8/3 & 2/3 & -1/3 \\ 1/2 & -1/2 & 0 \end{pmatrix} \]
\[ b = \begin{pmatrix} b_1 \\ b_2 \\ b_4 \end{pmatrix} = \begin{pmatrix} 8 \\ 4 \\ 18 \end{pmatrix} \quad c = \begin{pmatrix} c_3 \\ c_5 \\ c_6 \end{pmatrix} = \begin{pmatrix} -1/9 \\ -1/9 \\ -2/9 \end{pmatrix} \]
\[ v = 29. \]

Note that indices depend on the sets \(N\) and \(B\), and also that the entries in \(A\) are negation of what they appear in the slack form.

The Simplex algorithm by example

$$\text{max } 5x_1 + 4x_2 + 3x_3$$
$$s.t. \quad 2x_1 + 3x_2 + x_3 \leq 5$$
$$4x_1 + x_2 + 2x_3 \leq 11$$
$$3x_1 + 4x_2 + 2x_3 \leq 8$$
$$x_1, x_2, x_3 \geq 0$$

Next, we introduce slack variables, for example, rewriting
\[ 2x_1 + 3x_2 + x_3 \leq 5 \]

as the constraints: \(w_1 \geq 0\) and \(w_1 = 5 - 2x_1 - 3x_2 - x_3\). The resulting LP in slack form is

$$\text{max } z = 5x_1 + 4x_2 + 3x_3$$
$$s.t. \quad w_1 = 5 - 2x_1 - 3x_2 - x_3$$
$$w_2 = 11 - 4x_1 - x_2 - 2x_3$$
$$w_3 = 8 - 3x_1 - 4x_2 - 2x_3$$
$$x_1, x_2, x_3, w_1, w_2, w_3 \geq 0$$
Example continued...

**Consider the slack**

**Want to increase**

\[
\begin{align*}
\text{max } z &= 5x_1 + 4x_2 + 3x_3 \\
\text{s.t. } w_1 &= 5 - 2x_1 - 3x_2 - x_3 \\
w_2 &= 11 - 4x_1 - x_2 - 2x_3 \\
w_3 &= 8 - 3x_1 - 4x_2 - 2x_3 \\
x_1, x_2, x_3, w_1, w_2, w_3 &\geq 0
\end{align*}
\]

\[\implies w_1 = 5, w_2 = 11 \text{ and } w_3 = 8.\]

Feasible!

**Objection function value:** \(z = 0\).

**Further improve t value of objective function (i.e., z).**

While keeping feasibility.

---

Example continued...

**Set \(x_2 = x_3 = 0\)**

\[
\begin{align*}
\text{max } z &= 5x_1 + 4x_2 + 3x_3 \\
\text{s.t. } w_1 &= 5 - 2x_1 - 3x_2 - x_3 \\
w_2 &= 11 - 4x_1 - x_2 - 2x_3 \\
w_3 &= 8 - 3x_1 - 4x_2 - 2x_3 \\
x_1, x_2, x_3, w_1, w_2, w_3 &\geq 0
\end{align*}
\]

Want to increase \(x_1\) as much as possible, as long as:

\[
\begin{align*}
w_1 &= 5 - 2x_1 \geq 0, \\
w_2 &= 11 - 4x_1 \geq 0, \\
\text{and } w_3 &= 8 - 3x_1 \geq 0.
\end{align*}
\]

---

Example continued...

**Constraints:**

\[
\begin{align*}
\text{max } z &= 5x_1 + 4x_2 + 3x_3 \\
\text{s.t. } w_1 &= 5 - 2x_1 - 3x_2 - x_3 \\
w_2 &= 11 - 4x_1 - x_2 - 2x_3 \\
w_3 &= 8 - 3x_1 - 4x_2 - 2x_3 \\
x_1, x_2, x_3, w_1, w_2, w_3 &\geq 0
\end{align*}
\]

Maximum we can increase \(x_1\) is 2.5.

\[x_1 = 2.5, \; x_2 = 0, \; x_3 = 0, \; w_1 = 0, \; w_2 = 1, \; w_3 = 0.5\]

\[\implies z = 5x_1 + 4x_2 + 3x_3 = 12.5.\]

Improved target!

A nonbasic variable \(x_1\) is now non-zero. One basic variable \((w_1)\) became zero.
Example continued...

$$\text{max } z = 5x_1 + 4x_2 + 3x_3$$
$$\text{s.t. } w_1 = 5 - 2x_1 - 3x_2 - x_3$$
$$w_2 = 11 - 4x_1 - x_2 - 2x_3$$
$$w_3 = 8 - 3x_1 - 4x_2 - 2x_3$$
$$x_1, x_2, x_3, w_1, w_2, w_3 \geq 0$$

- $x_1 = 2.5$, $x_2 = 0$, $x_3 = 0$, $w_1 = 0$, $w_2 = 1$, $w_3 = 0.5$
- A nonbasic variable $x_1$ is now non-zero. One basic variable ($w_1$) became zero.

Want to keep invariant: All non-basic variables in current solution are zero...

Idea: Exchange $x_1$ and $w_1$!

Consider equality LP with $w_1$ and $x_1$.
$$w_1 = 5 - 2x_1 - 3x_2 - x_3.$$  Rewrite as: $x_1 = 2.5 - 0.5w_1 - 1.5x_2 - 0.5x_3.$

Example continued...

- Rewriting stop done is called **pivoting**.
- pivoted on $x_1$.
- Continue pivoting till reach optimal solution.

$$\text{max } z = 12.5 - 2.5w_1 - 3.5x_2 + 0.5x_3$$
$$x_1 = 2.5 - 0.5w_1 - 1.5x_2 - 0.5x_3$$
$$w_2 = 1 + 2w_1 + 5x_2$$
$$w_3 = 0.5 + 1.5w_1 + 0.5x_2 - 0.5x_3.$$

Can only pivot on $x_3$...

- $x_1$ can only be increased to 1 before $w_3 = 0$.
- Rewriting the equality for $w_3$ in LP:
  $$w_3 = 0.5 + 1.5w_1 + 0.5x_2 - 0.5x_3.$$  ...for $x_3$: $x_3 = 1 + 3w_1 + x_2 - 2w_3$.
- Substituting into LP, we get the following LP.

$$\text{max } z = 13 - w_1 - 3x_2 - w_3$$
$$\text{s.t. } x_1 = 2 - 2w_1 - 2x_2 + w_3$$
$$w_2 = 1 + 2w_1 + 5x_2$$
$$x_3 = 1 + 3w_1 + x_2 - 2w_3.$$
Example continued – can this be further improved?

\[
\begin{align*}
\text{max } z &= 13 - w_1 - 3x_2 - w_3 \\
\text{s.t. } &x_1 = 2 - 2w_1 - 2x_2 + w_3 \\
&w_2 = 1 + 2w_1 + 5x_2 \\
&x_3 = 1 + 3w_1 + x_2 - 2w_3
\end{align*}
\]

- NO!
- All coefficients in objective negative (or zero).
- Trivial solution (all nonbasic variables zero) is maximal.

Pivoting changes nothing

Observation

Every pivoting step just rewrites the LP into EQUIVALENT LP. When LP objective can no longer be improved because of rewrite, it implies that the original LP objective function can not be increased any further.

Simplex algorithm – summary

- This was an informal description of the simplex algorithm.
- At each step pivot on a nonbasic variable that improves objective function.
- Till reach optimal solution.
- Problem: Assumed that the starting (trivial) solution (all zero nonbasic vars) is feasible.

Starting somewhere...

\[
\begin{align*}
\text{max } z &= v + \sum_{j \in N} c_j x_j, \\
\text{s.t. } &x_i = b_i - \sum_{j \in N} a_{ij} x_j \quad \text{for } i \in B, \\
&x_i \geq 0, \quad \forall i = 1, \ldots, n + m.
\end{align*}
\]

- **L**: Transformed LP to slack form.
- **Simplex** starts from feasible solution and walks around till reaches opt.
- **L** might not be feasible at all.
- Example on left, trivial sol is not feasible, if \( \exists b_i < 0 \).

Idea: Add a variable \( x_0 \), and minimize it!

\[
\begin{align*}
\text{min } x_0 \\
\text{s.t. } &x_i = x_0 + b_i - \sum_{j \in N} a_{ij} x_j \quad \text{for } i \in B, \\
&x_i \geq 0, \quad \forall i = 1, \ldots, n + m.
\end{align*}
\]
Finding a feasible solution...

1. \( L' = \text{Feasible}(L) \) (see previous slide).
2. Add new variable \( x_0 \) and make it large enough.
3. \( x_0 = \max(-\min_i b_i, 0), \forall i > 0, x_i = 0 \): feasible!
4. \( \text{LPStartSolution}(L') \): Solution of Simplex to \( L' \).
5. If \( x_0 \) is solution then \( L \) feasible, and we found a valid basic solution.
6. If \( x_0 > 0 \) then LP not feasible. Done.

Lemma...

**Lemma**

\( \text{LP } L \text{ is feasible} \iff \text{optimal objective value of LP } L' \text{ is zero.} \)

**Proof.**

A feasible solution to \( L \) is immediately an optimal solution to \( L' \) with \( x_0 = 0 \), and vice versa. Namely, given a solution to \( L' \) with \( x_0 = 0 \) we can transform it to a feasible solution to \( L \) by removing \( x_0 \).

Technicalities, technicalities everywhere

1. Starting solution for \( L' \), generated by \( \text{LPStartSolution}(L) \).
2. .. not legal in slack form as non-basic variable \( x_0 \) assigned non-zero value.
3. Trick: Immediately pivoting on \( x_0 \) when running \( \text{Simplex}(L') \).
4. First try to decrease \( x_0 \) as much as possible.