NEW CS 473: Theory II, Fall 2015

Linear Programming

Lecture 18
October 27, 2015

A Factory Example

Problem

Suppose a factory produces two products I and II. Each requires three resources A, B, C.

1. Producing one unit of Product I requires 1 unit each of resources A and C.
2. One unit of Product II requires 1 unit of resource B and 1 units of resource C.
3. We have 200 units of A, 300 units of B, and 400 units of C.
4. Product I can be sold for **$1** and product II for **$6**.

How many units of product I and product II should the factory manufacture to maximize profit?

Solution: Formulate as a linear program.

A Factory Example

Problem

Suppose a factory produces two products I and II. Each requires three resources A, B, C.

1. Producing unit I: Req. 1 unit of A, C.
2. Producing unit II: Requ. 1 unit of B, C.
3. Have A: 200, B: 300 , and C: 400.
4. Price I: **$1**, and II: **$6**.

How many units of I and II to manufacture to max profit?

A Factory Example

Problem

Suppose a factory produces two products I and II. Each requires three resources A, B, C.

1. Producing unit I: Req. 1 unit of A, C.
2. Producing unit II: Requ. 1 unit of B, C.
3. Have A: 200, B: 300 , and C: 400.
4. Price I: **$1**, and II: **$6**.

How many units of I and II to manufacture to max profit?

$$\begin{array}{lll} \textbf{max} & x_I + 6x_{II} & \\ \text{s.t.} & x_I \leq 200 & (A) \\ & x_{II} \leq 300 & (B) \\ & x_I + x_{II} \leq 400 & (C) \\ & x_I \geq 0 & \\ & x_{II} \geq 0 & \end{array}$$

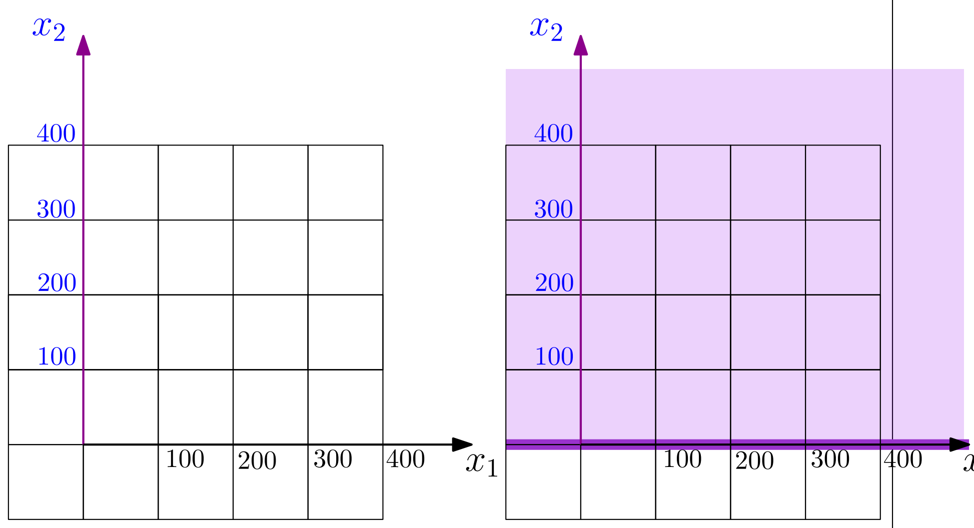
Linear Programming Formulation

Let us produce x_1 units of product I and x_2 units of product II. Our profit can be computed by solving

$$\begin{aligned} \text{maximize} \quad & x_1 + 6x_2 \\ \text{subject to} \quad & x_1 \leq 200 \\ & x_2 \leq 300 \\ & x_1 + x_2 \leq 400 \\ & x_1, x_2 \geq 0 \end{aligned}$$

What is the solution?

Graphical interpretation of LP



Economic planning

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

1. Penguina: a country.
2. Ruler need to decide how to allocate resources.
3. Maximize benefit.
4. 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.
5. Penguina need to prove a certain level of security:
   $$x_{gun} + 1000 * x_{nuclear-bomb} \geq 1000,$$
   where x_{guns}: # guns $x_{nuclear-bomb}$: # nuclear-bombs constructed.
6. $100 * x_{gun} + 1000000 * x_{nuclear-bomb} \leq x_{security}$
   $x_{security}$: 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):

1. $x_1, \ldots, x_n$: variables.
2. For $j = 1, \ldots, m$: $a_{j1}x_1 + \ldots + a_{jn}x_n \leq b_j$: linear inequality.
3. i.e., **constraint**.
4. Q: ∃ assignment of values to $x_1, \ldots, x_n$ such that all inequalities are satisfied?
5. Many possible solutions... Want solution that maximizes some linear quantity.
6. **objective function**: linear inequality being maximized.

Linear programming – example

$$
\begin{aligned}
&a_{11}x_1 + \ldots + a_{1n}x_n \le b_1\\
&a_{21}x_1 + \ldots + a_{2n}x_n \le b_2\\
&\ldots\\
&a_{m1}x_1 + \ldots + a_{mn}x_n \le b_m\\
&\textbf{max} \quad c_1x_1 + \ldots + c_nx_n.
\end{aligned}
$$

Linear Programming: A History

1. First formalized applied to problems in economics by Leonid Kantorovich in the 1930s
   1.1 However, work was ignored behind the Iron Curtain and unknown in the West
2. Rediscovered by Tjalling Koopmans in the 1940s, along with applications to economics
3. First algorithm (Simplex) to solve linear programs by George Dantzig in 1947
4. Kantorovich and Koopmans receive Nobel Prize for economics in 1975 ; Dantzig, however, was ignored
   4.1 Koopmans contemplated refusing the Nobel Prize to protest Dantzig's exclusion, but Kantorovich saw it as a vindication for using mathematics in economics, which had been written off as "a means for apologists of capitalism"

Network flow via linear programming

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

$$
\begin{aligned}
&\forall (u, v) \in E && 0 \le x_{u\to v}\\
& && x_{u\to v} \le \mathrm{c}(u \to v)\\
&\forall v \in V \setminus \{\mathrm{s}, \mathrm{t}\} && \sum_{(u,v)\in E} x_{u\to v} - \sum_{(v,w)\in E} x_{v\to w} \le 0\\
& && \sum_{(u,v)\in E} x_{u\to v} - \sum_{(v,w)\in E} x_{v\to w} \ge 0\\
&\text{maximizing} && \textstyle\sum_{(\mathrm{s},u)\in E} x_{\mathrm{s}\to u}
\end{aligned}
$$

Maximum weight matching

Input: $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ and weight $w(\cdot)$ on the edges. Compute max matching in $\mathbf{G}$.

$$
\begin{aligned}
&\forall uv \in \mathbf{E} && 0 \le x_{uv}\\
& && x_{uv} \le 1\\
&\forall v \in V && \sum_{uv\in \mathrm{E}} x_{uv} \le 1\\
&\textbf{max} && \textstyle\sum_{uv\in \mathrm{E}} w(uv)x_{uv}
\end{aligned}
$$

Rewriting an LP

$$\max \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \le b_i \quad \text{for } i = 1, 2, \ldots, m$$

1. Rewrite: so every variable is non-negative.
2. Replace variable x_i by x_i' and x_i'', where new constraints are: $x_i = x_i' - x_i''$, $x_i' \ge 0$ and $x_i'' \ge 0$.
3. Example: The (silly) LP $2x + y \ge 5$ rewritten:
   $2x' - 2x'' + y' - y'' \ge 5$,
   $x' \ge 0$, $y' \ge 0$,
   $x'' \ge 0$, and
   $y'' \ge 0$.

Rewriting an 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**

Standard form of

A linear program in standard form.

$$\max \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \le b_i \quad \text{for } i = 1, 2, \ldots, m$$

$$x_j \ge 0 \quad \text{for } j = 1, \ldots, n.$$

Standard form of

Because everything is clearer when you use matrices. Not.

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1(n-1)} & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2(n-1)} & a_{2n} \\ \vdots & \ldots & \ldots & \ldots & \vdots \\ a_{(m-1)1} & a_{(m-1)2} & \ldots & a_{(m-1)(n-1)} & a_{(m-1)n} \\ a_{m1} & a_{m2} & \ldots & a_{m(n-1)} & a_{mn} \end{pmatrix},$$

c, b and A: prespecified. x is vector of unknowns. Solve LP for x.

LP in standard form.

(Matrix notation.)

$$\max \quad c^T x$$
$$\text{s.t.} \quad Ax \le b.$$
$$x \ge 0.$$

$$c = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}, b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}, x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}.$$

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.

$$\begin{array}{ll} \max & c^T x \\ \text{subject to} & Ax = b. \\ & x \geq 0. \end{array}$$

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

$$x_{n+1} = b - \sum_{i=1}^{n} a_i x_i$$
$$x_{n+1} \geq 0.$$

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

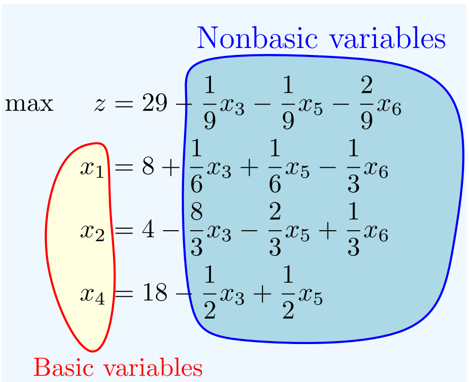
Slack form...

1. LP now made of equalities of the form: $x_{n+1} = b - \sum_{i=1}^{n} a_i x_i$
2. Variables on left: **basic variables**.
3. Variables on right: **nonbasic variables**.
4. LP in this form is in **slack form**.

Linear program in slack form.

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

Basic/nonbasic



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).

- B - Set of indices of basic variables
- N - Set of indices of nonbasic variables
- $n = |N|$ - number of original variables
- b, c - two vectors of constants
- $m = |B|$ - number of basic variables (i.e., number of inequalities)
- $A = \{a_{ij}\}$ - The matrix of coefficients
- $N \cup B = \{1, \ldots, n+m\}$
- v - objective function constant.

Slack form formally

Final form

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

Example

Consider the following LP which is in slack form.

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

Example

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

$B = \{1, 2, 4\}$, $N = \{3, 5, 6\}$

$$A = \begin{pmatrix} a_{13} & a_{15} & a_{16} \\ a_{23} & a_{25} & a_{26} \\ a_{43} & a_{45} & a_{46} \end{pmatrix} = \begin{pmatrix} -1/6 & -1/6 & 1/3 \\ 8/3 & 2/3 & -1/3 \\ 1/2 & -1/2 & 0 \end{pmatrix}$$

$$b = \begin{pmatrix} b_1 \\ b_2 \\ b_4 \end{pmatrix} = \begin{pmatrix} 8 \\ 4 \\ 18 \end{pmatrix} \qquad c = \begin{pmatrix} c_3 \\ c_5 \\ c_6 \end{pmatrix} = \begin{pmatrix} -1/9 \\ -1/9 \\ -2/9 \end{pmatrix}$$

$v = 29$.

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.

Another example...

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

Transform into slack form...

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

The Simplex algorithm by example

$$\begin{array}{ll} \max & 5x_1 + 4x_2 + 3x_3 \\ s.t. & 2x_1 + 3x_2 + x_3 \leq 5 \\ & 4x_1 + x_2 + 2x_3 \leq 11 \\ & 3x_1 + 4x_2 + 2x_3 \leq 8 \\ & x_1, x_2, x_3 \geq 0 \end{array}$$

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

$$\Rightarrow \begin{array}{lll} \max & z = & 5x_1 + 4x_2 + 3x_3 \\ s.t. & w_1 = & 5 - 2x_1 - 3x_2 - x_3 \\ & w_2 = & 11 - 4x_1 - x_2 - 2x_3 \\ & w_3 = & 8 - 3x_1 - 4x_2 - 2x_3 \\ & & x_1, x_2, x_3, w_1, w_2, w_3 \geq 0 \end{array}$$

Example continued I...

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

1. w_1, w_2, w_3: slack variables. (Also currently basic variables).
2. Consider the slack representation trivial solution... all non-basic variables assigned zero: $x_1 = x_2 = x_3 = 0$.

1. $\implies w_1 = 5$, $w_2 = 11$ and $w_3 = 8$.
2. Feasible!
3. Objection function value: $z = 0$.
4. Further improve the value of objective function (i.e., z). While keeping feasibility.

Example continued II...

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

1. $x_1 = x_2 = x_3 = 0$ $\implies w_1 = 5$, $w_2 = 11$ and $w_3 = 8$.
2. All w_i positive – change x_i a bit does not change feasibility.

1. $z = 5x_1 + 4x_2 + 3x_3$: want to increase values of x_1s... since z increases (since $5 > 0$).
2. How much to increase x_1???
3. Careful! Might break feasibility.
4. Increase x_1 as much as possible without breaking feasibility!

Example continued III...

Set $x_2 = x_3 = 0$

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

$$\begin{aligned} w_1 &= 5 - 2x_1 - 3x_2 - x_3 \\ &= 5 - 2x_1 \\ w_2 &= 11 - 4x_1 - x_2 - 2x_3 \\ &= 11 - 4x_1 \\ w_3 &= 8 - 3x_1 - 4x_2 - 2x_3 \\ &= 8 - 3x_1. \end{aligned}$$

1. Want to increase x_1 as much as possible, as long as:

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

Example continued IV...

1. Constraints:

$$w_1 = 5 - 2x_1 \geq 0,$$
$$w_2 = 11 - 4x_1 \geq 0,$$
and $w_3 = 8 - 3x_1 \geq 0.$

2. $x_1 \leq 2.5$, $x_1 \leq 11/4 = 2.75$ and $x_1 \leq 8/3 = 2.66$

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

1. Maximum we can increase x_1 is 2.5.
2. $x_1 = 2.5,\ x_2 = 0,\ x_3 = 0,\ w_1 = 0,\ w_2 = 1,\ w_3 = 0.5$
   $$\Rightarrow \quad z = 5x_1 + 4x_2 + 3x_3 = 12.5.$$
3. Improved target!
4. A nonbasic variable x_1 is now non-zero. One basic variable (w_1) became zero.

Example continued V...

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

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

1. Want to keep invariant: All non-basic variables in current solution are zero...
2. Idea: Exchange x_1 and w_1!
3. Consider equality LP with w_1 and x_1. $w_1 = 5 - 2x_1 - 3x_2 - x_3$.
4. Rewrite as: $x_1 = 2.5 - 0.5w_1 - 1.5x_2 - 0.5\,x_3$.

Example continued VI...

Substituting $x_1 = 5 - 2x_1 - 3x_2 - x_3$, the new

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

1. nonbasic variables: $\{w_1, x_2, x_3\}$
   basic variables: $\{x_1, w_2, w_3\}$.
2. Trivial solution: all nonbasic variables $= 0$ is feasible.
3. $w_1 = x_2 = x_3 = 0$. Value: $z = 12.5$.

Example continued VII...

1. Rewriting stop done is called **pivoting**.
2. pivoted on x_1.
3. Continue pivoting till reach optimal solution.

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

4. Can not pivot on w_1, since if w_1 increase, then z decreases. Bad.
5. Can not pivot on x_2 (coefficient in objective function is -3.5).
6. Can only pivot on x_3 since its coefficient ub objective $\mathbf{0.5}$. Positive number.

Example continued VIII...

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

1. Can only pivot on x_3...
2. x_1 can only be increased to $\mathbf{1}$ before $w_3 = 0$.
3. Rewriting the equality for w_3 in LP:
   $w_3 = 0.5 + 1.5w_1 + 0.5x_2 - 0.5x_3$,
4. ...for x_3: $x_3 = 1 + 3w_1 + x_2 - 2w_3$.
5. Substituting into LP, we get the following LP.

$$\begin{aligned} \max \quad z &= 13 - w_1 - 3x_2 - w_3 \\ s.t. \quad x_1 &= 2 - 2w_1 - 2x_2 + w_3 \\ w_2 &= 1 + 2w_1 + 5x_2 \\ x_3 &= 1 + 3w_1 + x_2 - 2w_3 \end{aligned}$$

Example continued – can this be further improved?

$$\begin{aligned} \max \quad z &= 13 - w_1 - 3x_2 - w_3 \\ s.t. \quad x_1 &= 2 - 2w_1 - 2x_2 + w_3 \\ w_2 &= 1 + 2w_1 + 5x_2 \\ x_3 &= 1 + 3w_1 + x_2 - 2w_3 \end{aligned}$$

1. NO!
2. All coefficients in objective negative (or zero).
3. 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

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

Starting somewhere...

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

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

Idea: Add a variable x_0, and minimize it!

$$\begin{aligned} \min \quad & x_0 \\ \text{s.t.} \quad & x_i = x_0 + b_i - \sum_{j \in N} a_{ij} x_j \quad \text{for } i \in B, \\ & x_i \geq 0, \quad \forall i = 1, \ldots, n+m. \end{aligned}$$

Finding a feasible solution...

1. $L' = \textbf{Feasible}(L)$ (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. $\textbf{LPStartSolution}(L')$: Solution of **Simplex** to L'.
5. If $x_0 = 0$ in solution then L feasible. Have valid basic solution.
6. If $x_0 > 0$ then LP not feasible. Done.

Lemma...

Lemma

LP L *is feasible* $\iff$ *optimal objective value of* LP L' *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 $\textbf{LPStartSolution}(L)$..
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 $\textbf{Simplex}(L')$.
4. First try to decrease x_0 as much as possible.