Using the Foundations and Trends® LaTeX Class

Alet Heezemans
now publishers, Inc.
alet.heezemans@nowpublishers.com

Mike Casey
now publishers, Inc.
mike.casey@nowpublishers.com

February 13, 2019

Abstract

This document describes how to prepare a Foundations and Trends® article in LaTeX . The accompanying LaTeX source file FnTarticle.tex (that produces this output) is an example of such a file.

1 Introduction to Linear Programming

1.1 Preliminaries

Standard Form We usually consider the standard linear programming (LP) model:

$$\begin{array}{ll} \max & \sum_{j=1}^{n} c_j x_j \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, j = 1, \ldots, n \end{array} \tag{1.1}$$

Or more generally, the constraints with equalities:

$$\begin{array}{ll} \min & \sum_{j=1}^{n} c_j x_j \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i \in I \\ & \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i \in E \\ & x_j \geq 0, j = 1, \ldots, n \end{array}$$

It's often convenient to write the LP (1.1) into the compact matrix form:

$$\begin{array}{ll} \max & \boldsymbol{c}^{\mathrm{T}} \boldsymbol{x} \\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{array} \tag{1.2}$$

where $\boldsymbol{c} \in \mathbb{R}^n, \boldsymbol{A} \in \mathbb{R}^{m \times n}, \boldsymbol{b} \in \mathbb{R}^m$.

We also write $\boldsymbol{A}$ as the column form:

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{a}_1, & \cdots & \boldsymbol{a}_n \end{pmatrix}$$

where $\boldsymbol{a}_i$ is the i-th column of $\boldsymbol{A}$. We also express the submatrix of $\boldsymbol{A}$, i.e., $\boldsymbol{A}_I \subset A$ as:

$$\boldsymbol{A}_I := [a_i \mid i \in I],$$

where I is a subset of $\{1, 2, \ldots, n\}$.

Dictionaries of an LP We can introduce slack variables to transform (1.1) into LP with equalities:

$$x_{n+i} := b_i - \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1, \ldots, m$$

Let $z = \sum_{j=1}^{n} c_j x_j$ be the objective function, and therefore we obtain a *dictionary* for the LP (1.1):

$$\left.\begin{aligned} x_{n+i} &= b_i - \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1, \ldots, m \\ z &= \sum_{j=1}^{n} c_j x_j \end{aligned}\right\} \quad \text{Dictionary} \tag{1.3}$$

Assume that $b_i \geq 0$ for $i = 1, \ldots, m$. Therefore we obtain a *feasible solution* associated with the dictionary, say *dictionary solution*:

$$x_j = 0, \text{ for } j = 1, \ldots, n \quad x_{n+i} = b_i \text{ for } i = 1, \ldots, m$$

It's clear how to improve the current dictionary solution:

- If $c_j \leq 0, \forall j$, then we cannot possibly improve the dictionary solution
- If $c_j > 0$ for some $1 \leq j \leq n$, we increase the value for x_j from 0 into maximal value, while fixing $x_j = 0$ for $1 \leq k (\neq j) \leq n$. Keep implementing until $c_j \leq 0, \forall j$.

Example 1.1.1. Consider the optimization problem

$$\begin{array}{ll} \max & 5x_1 + 4x_2 + 3x_3 \\ \text{s.t.} & 2x_1 + 3x_2 + x_3 \leq 5 \\ & 4x_1 + x_2 + 2x_3 \leq 11 \\ & 3x_1 + 4x_2 + 2x_3 \leq 8 \\ & x_1 \geq 0, \ x_2 \geq 0, \ x_3 \geq 0 \end{array} \tag{1.4a}$$

We can find its dictionary:

$$\begin{aligned} x_4 &= 5 \boldsymbol{-} \mathbf{2} x_1 - 3x_2 - x_3 \\ x_5 &= 11 - 4x_1 - x_2 - 2x_3 \\ x_6 &= 8 - 3x_1 - 4x_2 - 2x_3 \\ z &= 0 \boldsymbol{+} \mathbf{5} x_1 + 4x_2 + 3x_3 \end{aligned} \tag{1.4b}$$

Since $c_1 > 0$, increasing value for x_1 suffices to consider the dictionary below instead:

$$
\begin{aligned}
x_1 &= \frac{5}{2} - \frac{3}{2}x_2 - \frac{1}{2}x_3 - \frac{1}{2}x_4 \\
x_5 &= 11 - 4x_1 - x_2 - 2x_3 \\
x_6 &= 8 - 3x_1 - 4x_2 - 2x_3 \\
z &= 0 + 5x_1 + 4x_2 + 3x_3
\end{aligned}
\quad \Longleftrightarrow \quad
\begin{aligned}
x_1 &= \frac{5}{2} - \frac{3}{2}x_2 - \frac{1}{2}x_3 - \frac{1}{2}x_4 \\
x_5 &= 1 + 5x_2 + 0x_3 + 2x_4 \\
x_6 &= \frac{1}{2} + \frac{1}{2}x_2 - \frac{1}{2}x_3 + \frac{3}{2}x_4 \\
z &= \frac{25}{2} - \frac{7}{2}x_2 + \frac{1}{2}x_3 - \frac{5}{2}x_4
\end{aligned}
\tag{1.4c}
$$

Also, since $c_3 > 0$, increasing value for x_3 suffices to consider the dictionary below instead:

$$
\begin{aligned}
x_1 &= \frac{5}{2} - \frac{3}{2}x_2 - \frac{1}{2}x_3 - \frac{1}{2}x_4 \\
x_5 &= 1 + 5x_2 + 0x_3 + 2x_4 \\
x_3 &= 1 + x_2 + 3x_4 - 2x_6 \\
z &= \frac{25}{2} - \frac{7}{2}x_2 + \frac{1}{2}x_3 - \frac{5}{2}x_4
\end{aligned}
\quad \Longleftrightarrow \quad
\begin{aligned}
x_3 &= 1 + x_2 + 3x_4 - 2x_6 \\
x_1 &= 2 - x_2 - 2x_4 + x_6 \\
x_5 &= 1 + 5x_2 + 2x_4 + 2x_6 \\
z &= 13 - 3x_2 - x_4 - x_6
\end{aligned}
\tag{1.4d}
$$

1.2 Simplex Method

Notations The general dictionary for the problem (1.1) can be expressed as:

$$
\begin{aligned}
x_i &= \bar{b}_i - \sum_{j \in N} \bar{a}_{ij} x_j, \quad i \in B \\
z &= \zeta - \sum_{j \in N} \bar{c}_j x_j
\end{aligned}
\tag{1.5}
$$

where

1. the set B is called a *basis*, with $|B| = m$
2. the set N is called a *non-basis*, with $|N| = n - m$. Moreover, $B \cup N = \{1, \ldots, n\}$.
3. the basis B is said to be *primal feasible* if $\bar{\boldsymbol{b}} \geq 0$, since in this case we can choose a primal feasible solution by setting non-basis variables to be zero and basis variables x_i to be $\bar{b}_i$.
4. the non-basis N is said to be *dual feasible* if $\bar{\boldsymbol{c}} \leq 0$, since in this case we can choose a dual feasible solution by setting non-basis variables to be $\bar{c}_i$ and other variables to be 0.

One can verify that in (1.5),

$$\begin{aligned}
\bar{\boldsymbol{b}} &= \boldsymbol{A}_B^{-1}\boldsymbol{b} \\
\bar{\boldsymbol{c}}_N^{\mathrm{T}} &= \boldsymbol{c}_N^{\mathrm{T}} - \boldsymbol{c}_B^{\mathrm{T}}\boldsymbol{A}_B^{-1}\boldsymbol{A}_N \\
\bar{\boldsymbol{A}}_N &= \boldsymbol{A}_B^{-1}\boldsymbol{A}_N \\
\zeta &= \boldsymbol{c}_B^{\mathrm{T}}\boldsymbol{A}_B^{-1}\boldsymbol{b}
\end{aligned}$$

One can verify that $(\boldsymbol{x}_B, \boldsymbol{x}_N) = (\boldsymbol{A}_B^{-1}\boldsymbol{b}, \boldsymbol{0})$ is a *basic solution*.

Simplex Method Algorithm The assumption for the working of simplex method is that we are given a primal feasible basic solution, i.e., $\bar{b} \geq 0$. The framework for obtaining an improved solution is summarized in (1).

Algorithm 1 Framework for the one step of the Simplex Method

Input:

Primal feasible basic solution;

Output:

Improved feasible basic solution;

1: Find Entering Basis Variable j

- Search for $j \in N$ such that $\bar{c}_j > 0$
- If none exists then the current basic solution is optimal; otherwise choose one of such j.

2: Find Leaving Basis Variable i

- Search for $i \in B$ such that $\bar{a}_{ij} > 0$
- If none exists then the problem is unbounded; otherwise choose

$$i \in \arg\min \left\{ \frac{\bar{b}_i}{\bar{a}_{ij}} : \bar{a}_{ij} > 0,\ i \in B \right\}$$

3: Basis Update: $B \leftarrow B \cup \{j\} \setminus \{i\}$, and then form the corresponding basic solution.

Remark 1.2.1. The *one-step* of the simplex method is also called a *pivot step*, i.e., choose one pivot variable entering the basis and one leaving the basis.

The objective value for a successful pivot is improved by $\frac{\bar{c}_j\bar{b}_i}{\bar{a}_{ij}}$. However, the simplex method may not necessarily increase the objective value at each pivot, e.g., the case $\bar{b}_i = 0$ coul happen. In this case, the basic solution is said to be *degenerate*.

Since there are no more than $\binom{n}{m}$ (finite) possible bases, the simplex method will stop on two cases: (a) declaring the problem is unbounded; (b) finding a basic optimal solution.

Pivot Rules The *simplex method* specializes into a *simplex algorithm* if one specifies a *pivot rule* to determine which one variable to enter the basis and which one to leave, when there is a choice to make. Note that there exists some pivot rules that will make the problem face into cycling circumstance (see the example below), but here we list some examples of pivot rules that will be shown to definitely avoid cycling circumstance:

- Dantzig's pivot rule: *choose the largest positive coefficient to enter the basis.*

- The maximum improvement rule: *try all the combinations and pick the pivot pair with the largest improvement.*

- Bland's rule: *Among the candidates always pick the one with the smallest index.*

Example 1.2.1. This example shows that some pivot rules may let the problem face into cycling circumstance, i.e., the algorithm solves the problem in a loop and fails to go out:

$$\begin{aligned}
\boldsymbol{x_5} &= -0.5x_1 + 5.5x_2 + 2.5x_3 - 9x_4 \\
x_6 &= -0.5x_1 + 1.5x_2 + 0.5x_3 - x_4 \\
x_7 &= 1 - x_1 \\
z &= \boldsymbol{10x_1} - 57x_2 - 9x_3 - 24x_4
\end{aligned}$$

Choosing x_1 to enter the basis and x_5 to leave gives:

$$\begin{aligned}
x_1 &= -2x_5 + 11x_2 + 5x_3 - 18x_4 \\
\boldsymbol{x_6} &= x_5 - 4x_2 - 2x_3 + 8x_4 \\
x_7 &= 1 + 2x_5 - 11x_2 - 5x_3 + 18x_4 \\
z &= -20x_5 + \boldsymbol{53x_2} + 41x_3 - 204x_4
\end{aligned}$$

Choosing x_2 to enter the basis and x_6 to leave gives:

$$\begin{aligned}
\boldsymbol{x_1} &= 0.75x_5 - 2.75x_6 - 0.5x_3 + 4x_4 \\
x_2 &= 0.25x_5 - 0.25x_6 - 0.5x_3 + 2x_4 \\
x_7 &= 1 - 0.75x_5 - 13.25x_6 + 0.5x_3 - 4x_4 \\
z &= -6.75x_5 - 13.25x_6 + \boldsymbol{14.5x_3} - 98x_4
\end{aligned}$$

Choosing x_3 to enter the basis and x_1 to leave gives:

$$\begin{aligned}
x_3 &= 1.5x_5 - 5.5x_5 - 2x_1 + 8x_4 \\
\boldsymbol{x_2} &= -0.5x_5 + 2.5x_5 + x_1 - 2x_4 \\
x_7 &= 1 - x_1 \\
z &= 15x_5 - 93x_5 - 29x_1 + \mathbf{18}\boldsymbol{x_4}
\end{aligned}$$

Choosing x_4 to enter the basis and x_2 to leave gives:

$$\begin{aligned}
\boldsymbol{x_3} &= -0.5x_5 + 4.5x_5 + 2x_1 - 4x_2 \\
x_4 &= -0.25x_5 + 1.25x_5 + 0.5x_1 - 0.5x_2 \\
x_7 &= 1 - x_1 \\
z &= \mathbf{10.5}\boldsymbol{x_5} - 70.5x_5 - 20x_1 - 9x_2
\end{aligned}$$

Choosing x_5 to enter the basis and x_3 to leave gives:

$$\begin{aligned}
x_5 &= 9x_6 + 4x_1 - 8x_2 - 2x_3 \\
\boldsymbol{x_4} &= -x_6 - 0.5x_1 + 1.5x_2 + 0.5x_3 \\
x_7 &= 1 - x_1 \\
z &= \mathbf{24}\boldsymbol{x_6} + 22x_1 - 93x_2 - 21x_3
\end{aligned}$$

Choosing x_6 to enter the basis and x_4 to leave gives the *same dictionary as we started*:

$$\begin{aligned}
\boldsymbol{x_5} &= -0.5x_1 + 5.5x_2 + 2.5x_3 - 9x_4 \\
x_6 &= -0.5x_1 + 1.5x_2 + 0.5x_3 - x_4 \\
x_7 &= 1 - x_1 \\
z &= \mathbf{10}\boldsymbol{x_1} - 57x_2 - 9x_3 - 24x_4
\end{aligned}$$

Theorem 1.2.1. Bland's pivot rule would aviod cycling.

Proof. We show this claim by contradiction. If Bland's pivot rule produces cycling, let's study one cycle. For a sequence of dictionaries that form a cycle, let's delete all the variables that neither leave nor enter the basis, then it will remain a cycle.

In all these dictionaries, all $\bar{b}_i$ will be zero, since otherwise the objective value will be strictly increased.

Let's study the tablau of dictionaries. It's a matrix that stores all the coefficients of a dictionary:

$$\begin{array}{cc|c}
\hline
\boldsymbol{I}_B & \boldsymbol{A}_B^{-1}\boldsymbol{A}_N & \bar{\boldsymbol{b}} \\
\hline
\mathbf{0}^{\mathrm{T}} & \bar{\boldsymbol{c}}_N^{\mathrm{T}} & \boldsymbol{c}_B^{\mathrm{T}}\boldsymbol{A}_B^{-1}\boldsymbol{b} \\
\hline
\end{array}$$

Two vectors are of special interest. The last row of the tablau in left part can be written as

$$\bar{\boldsymbol{c}}^{\mathrm{T}} = \boldsymbol{c} - \boldsymbol{c}_B^{\mathrm{T}} \boldsymbol{A}_B^{-1} \boldsymbol{A}$$

For the chosen $j \in N$, the direction

$$d_i^{(j)} = \begin{cases} -\bar{a}_{ij}, & i \in B \\ 0, & i \neq j \\ 1, & i = j \end{cases}$$

It's clear that $\bar{\boldsymbol{c}}^{\mathrm{T}} \boldsymbol{d}^{(j)} = \bar{c}_j$.

Suppose that ℓ is the largest index of all variables that are involved in the cycle. Let (B, N) be the pivot where ℓ was about to enter the basis, $\boldsymbol{v} = \bar{\boldsymbol{c}}$ be the last row for that tableau at that point; let (B', N') be the pivot where ℓ was about to leave the basis, and k was to enter the basis at that point, $\boldsymbol{d}^{(k)}$ be the corresponding direction vector, $\boldsymbol{u}$ the last row of that tableau.

It's clear that

- $\boldsymbol{v}$ is everywhere non-positive except for one position $v_\ell > 0$
- $\boldsymbol{d}^{(k)}$ is everywhere non-negative except for one position $d_\ell^{(k)} < 0$

Moreover, $\boldsymbol{v} - \boldsymbol{u} \in \mathcal{R}(\boldsymbol{A}^{\mathrm{T}})$ and $\boldsymbol{d}^{(k)} \in \mathcal{N}(\boldsymbol{A})$, which implies

$$0 = (\boldsymbol{v} - \boldsymbol{u})^{\mathrm{T}} \boldsymbol{d}^{(k)} = \boldsymbol{v}^{\mathrm{T}} \boldsymbol{d}^{(k)} - \boldsymbol{u}_k < 0,$$

which is a contradiction. □

Lemma 1.2.2. Given the condition that the LP (1.2) has one basic feasible solution, then the LP (1.2) with perturbations, i.e.,

$$\begin{aligned} \max \quad & \boldsymbol{c}^{\mathrm{T}} \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} + \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_m \end{pmatrix} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{aligned} \tag{1.6}$$

will face no degeneracy for $\forall \boldsymbol{\varepsilon} \in (\boldsymbol{0}, \boldsymbol{\varepsilon}_1)$ for some $\boldsymbol{\varepsilon}_1 > 0$.

Proof. For any basis B, the feasible solution for LP (1.6) is $\boldsymbol{A}_B^{-1}(\bar{\boldsymbol{b}} + \boldsymbol{\varepsilon})$. Suppose its i-th component is zero, i.e., $0 + 0\varepsilon_1 + \cdots + 0\varepsilon_m$.

However, its i-th component is $\boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{A}_B^{-1}(\bar{\boldsymbol{b}} + \boldsymbol{\varepsilon})$, which implies $\boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{A}_B^{-1} = \boldsymbol{0}$, which is a contradiction. □

Question: what's the conclusion for page 19 in slides 1?

Two-Phase Simplex Method Given a dictionary

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1, \dots, m,$$

with some $b_i < 0$, the question is how to choose an initial basic feasible solution? The *two-phase simplex method* proceeds as follows:

1. Introduce a new variable x_0

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij} x_j + x_0, \quad i = 1, \dots, m,$$

   and an objective $-x_0$ to maximize

2. Suppose that $b_i < 0$ is the smallest valu. Perform a pivot on x_0 and thus x_{n+i} will turn the dictionary into a feasible one.

3. This *non-cycling pivots* will lead to either (a) an optimal basis where x_0 is within the basis, and we conclude this problem is infeasible; (b) or we have x_0 out of the basis, and we just delete x_0 and plug back the original objective, and go from there.

Example 1.2.2. Given the dictionary

$$\begin{aligned} x_4 &= 4 - 2x_1 + x_2 - 2x_3 \\ x_5 &= -5 - 2x_1 + 3x_2 - x_3 \\ x_6 &= -1 + x_1 - x_2 + 2x_3 \end{aligned}$$

We first add the new variable x_0 and an objective $-x_0$:

$$\begin{aligned} x_4 &= 4 - 2x_1 + x_2 - 2x_3 + x_0 \\ \boldsymbol{x_5} &= -5 - 2x_1 + 3x_2 - x_3 + x_0 \\ x_6 &= -1 + x_1 - x_2 + 2x_3 + \boldsymbol{x_0} \\ z &= -x_0 \end{aligned}$$

Choosing x_0 entering the basis and x_5 leaving the basis, we obtain:

$$\begin{aligned} x_4 &= 9 - x_2 + x_3 + x_5 \\ x_0 &= 5 + 2x_1 - 3x_2 - x_3 + x_5 \\ x_6 &= 4 + 3x_1 - 4x_2 + 3x_3 + x_5 \\ w &= -5 - 2x_1 + 3x_2 + x_3 - x_5 \end{aligned}$$

and our feasible solution is $(x_1, x_2, x_3, x_4, x_5, x_6) = (0, 0, 0, 9, 0, 4)$.

1.3 Duality Results

Theorem 1.3.1. A linear programming problem can only be (i) *feasible*; or (ii) *infeasible*. In case (i), then *there exists a basic feasible solution*, and further with two possibilities: (i.a) *an optimal solution exists, in that case a basic optimal solution exists* (i.b) *the problem is unbounded.*

Duality problem is the best possible upper bounding problem Consider the primal problem

$$(P) \qquad \begin{array}{ll} \max & \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b} \\ & \boldsymbol{x} \ge 0 \end{array}$$

Take any $\boldsymbol{y} \ge 0$ such that $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{A} \ge \boldsymbol{c}^{\mathrm{T}}$, and thus $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{b}$ becomes an upper bound for the optimal value. Therefore the best possible upper bounding problem becomes:

$$(D) \qquad \begin{array}{ll} \max & \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} \\ \text{s.t.} & \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \ge \boldsymbol{c} \\ & \boldsymbol{y} \ge 0 \end{array}$$

which is known as the *dual* problem.

The proceed above can be summarized as the weak duality theorem:

Theorem 1.3.2 (Weak Duality)**.** Let $\boldsymbol{x}, \boldsymbol{y}$ be the primal feasible, and dual feasible solution to (P) and (D), respectively, then we always have $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} \ge \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$.

Theorem 1.3.3 (Strong Duality)**.** If (P) has an optimal solution, then (D) has an optimal solution. Moreover, the optimal values coincide.

Proof. Let B be an optimal basis for (P), then we have

$$\boldsymbol{A}_B^{-1}\boldsymbol{b} \ge 0, \quad \begin{bmatrix}\boldsymbol{c}^{\mathrm{T}} & \boldsymbol{0}_m^{\mathrm{T}}\end{bmatrix} - \boldsymbol{c}_B^{\mathrm{T}}\boldsymbol{A}_B^{-1}\begin{bmatrix}\boldsymbol{A} & \boldsymbol{I}\end{bmatrix} \le 0$$

Therefore we construct the dual feasible solution $\boldsymbol{y} := \boldsymbol{A}_B^{-1}\boldsymbol{c}_B$, which implies $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{c}_B^{\mathrm{T}}\boldsymbol{A}_B^{-1}\boldsymbol{b}$. Therefore $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}$ should be the optimal solution for (D). □

Complentarity Slackness

Theorem 1.3.4 (Complentarity Condition)**.** Consider the primal and dual problem

$$(P) \quad \begin{array}{ll} \max & \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{s} = \boldsymbol{b} \\ & \boldsymbol{x} \ge 0, \boldsymbol{s} \ge 0 \end{array}, \qquad (D) \quad \begin{array}{ll} \min & \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} \\ \text{s.t.} & \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} - \boldsymbol{w} = \boldsymbol{c} \\ & \boldsymbol{y} \ge 0, \boldsymbol{w} \ge 0 \end{array}$$

If (P) has an optimal solution $(\boldsymbol{x}, \boldsymbol{s})$ and (D) has an optimal solution $(\boldsymbol{y}, \boldsymbol{w})$, then

$$\begin{aligned}\boldsymbol{s} \circ \boldsymbol{y} &= \mathbf{0} \\ \boldsymbol{w} \circ \boldsymbol{x} &= \mathbf{0}\end{aligned}$$

Remark 1.3.1. 1. If (P) is feasible and unbounded, then (D) must be infeasible.

2. The dual of the dual problem is the primal problem

3. There is possibility that both (P) and (D) are infeasible. Consider the self-dual problem for example:

$$\begin{array}{ll} \max & x_1 - x_2 \\ \text{s.t.} & \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \le \begin{pmatrix} -1 \\ 1 \end{pmatrix} \\ & x_1 \ge 0, x_2 \ge 0 \end{array}$$

4. Therefore, the relationship for primal and dual problems can be summarized in the table below:

	Feasible	Unbounded	Infeasible
Feasible	Y	N	N
Unbounded	N	N	Y
Infeasible	N	Y	Y

2 Geometry and Duality for Linear Programming

2.1 The polyhedral geometry

The constraint of a LP forms a polyhedron. One example for a LP with tenary variables is shown in the Fig (2.1)

Let's introduce some terminologies formally:

Definition 2.1.1. • *Hyperplane* is the set $\{\boldsymbol{x} \mid \boldsymbol{a}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{b}\}$

- *Half-space* is the set $\{\boldsymbol{x} \mid \boldsymbol{a}^{\mathrm{T}}\boldsymbol{x} \le \boldsymbol{b}\}$

- The *polyhedron* P is the intersection of *finite* number of half-spaces:

$$P = \left\{\boldsymbol{x} \middle| \boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{x} \le \boldsymbol{b}_i,\ i = 1, \dots, m\right\}$$

- The *dimension of a polyhedron* is defined as the *lowest* dimension *affine space* containing P

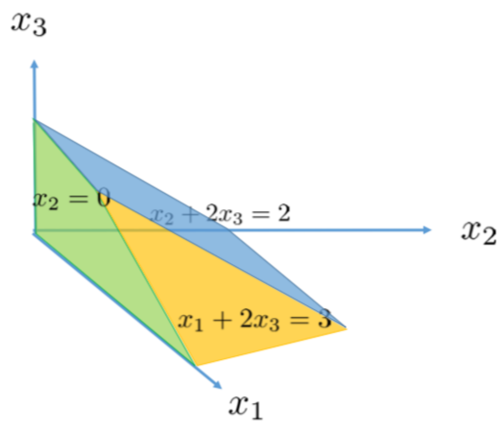

Figure 2.1: Illustration for polyhedral geometry

- The *face* of a polyhedron is defined as
$$\{\boldsymbol{x} \mid \boldsymbol{a}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{b}\} \cap P,$$
where $P \subseteq \{\boldsymbol{x} \mid \boldsymbol{a}^{\mathrm{T}}\boldsymbol{x} \leq \boldsymbol{b}\}$.

- Note that the face of a polyhedron is also a polyhedron. Therefore we define *facet* is the face of P that is one dimensional lower than that of P; the *vertex* of P is the face of P that has dimension 0.

Remark 2.1.1. In space $\mathbb{R}^n$, normally n hyperplanes intersect at one point

If P is full dimensional (i.e., with diemension n), then a vertex of P is an intersection of n facets.

However, sometimes there is a case that more than n hyperplanes intersect at one point, say a vertex, which creates *degeneracy* (show in Fig (2.4)). In such case, adding regularization, i.e., perturbation dimishes over-determination.

Definition 2.1.2. Given a full dimensional P, we say two distinct vertices of P are *adjacent* if they are in the same $n-1$ hyperplane.

Remark 2.1.2. Every update for simplex pivots move from a vertice to one of its adjacent position.

Definition 2.1.3. A *polyhedral cone* is defined as an intersection of a finite number of half-spaces, i.e., $K = \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{0}\}$, where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$. In geometry shown in Fig (2.6), a polyhedral cone is a cone where its boundaries are polyhedral

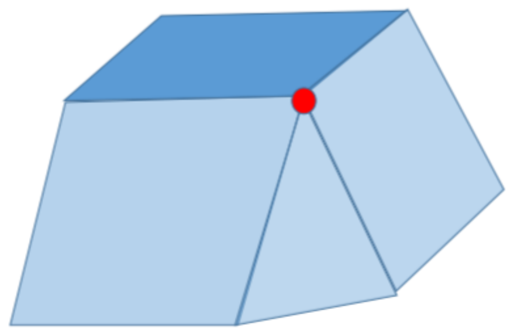

Figure 2.2: Over-determination results in Degeneracy

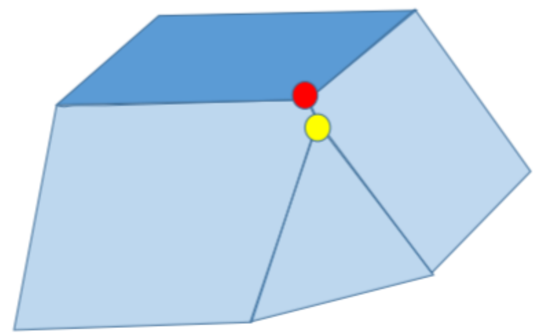

Figure 2.3: Perturbation diminishes over-determination

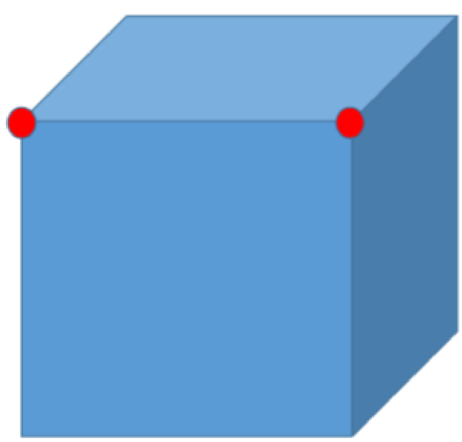

Figure 2.4: Illustration for *adjacent* vertices

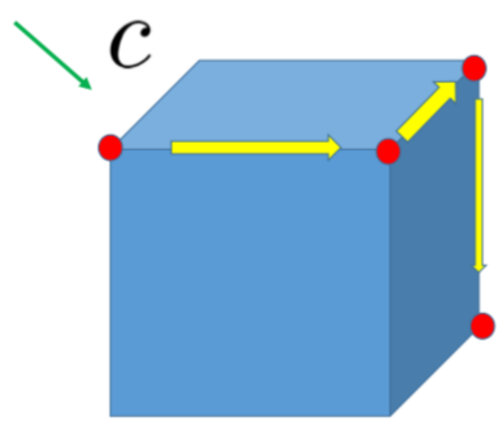

Figure 2.5: Path for simplex pivots

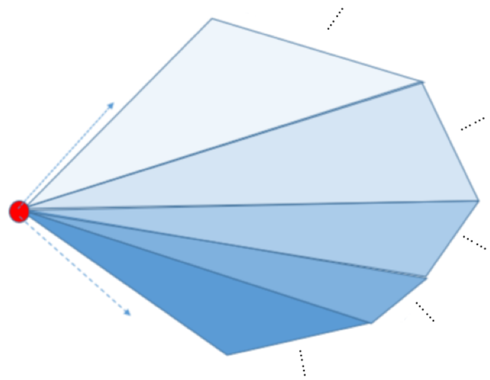

Figure 2.6: Illustration for a polyhedral cone

Definition 2.1.4. Given a set of points S, we can define its convex hull as

$$\text{conv hull}(S) := \left\{ \sum_{i=1}^{m} \lambda_i \boldsymbol{y}_i \middle| \ \forall \boldsymbol{y}_i \in S, \lambda_i \geq 0, \sum_{i=1}^{m} \lambda_i = 1 \right\}$$

In other words, a convex hull of S is a set, in which each point is a convex combination of points in S.

2.2 Fundamental Theorems for Linear Programming

Theorem 2.2.1 (Caratheodory's Theorem)**.** Let $P \subseteq \mathbb{R}^n$ be a set of points. For each $\boldsymbol{x} \in \text{conv}(P)$, there exists a set $P' \subseteq P$ of cardinality at most $n+1$, such that $\boldsymbol{x} \in \text{conv}(P')$.

Definition 2.2.1. The point $\boldsymbol{x}$ in a convex set S is an *extremal point* if

1. $x \in S$

2. x is a convex combination of $y, z \in S$ implies $y = z = x$.

Sometimes it is difficult to define the extremal point, if the convex set covers infinite area. Therefore, we define the points in the boundary of vertices instead:

Definition 2.2.2. A ray d in a convex cone $\mathcal{K}$ is an *extremal ray* if

1. $d \in \mathcal{K}$

2. If d can be written as non-negative summation of two rays $\xi_1, \xi_2 \in \mathcal{K}$, then $d = \xi_1 = \xi_2$

Definition 2.2.3. 1. A *polytope* is a convex hull of a *finite* number of points

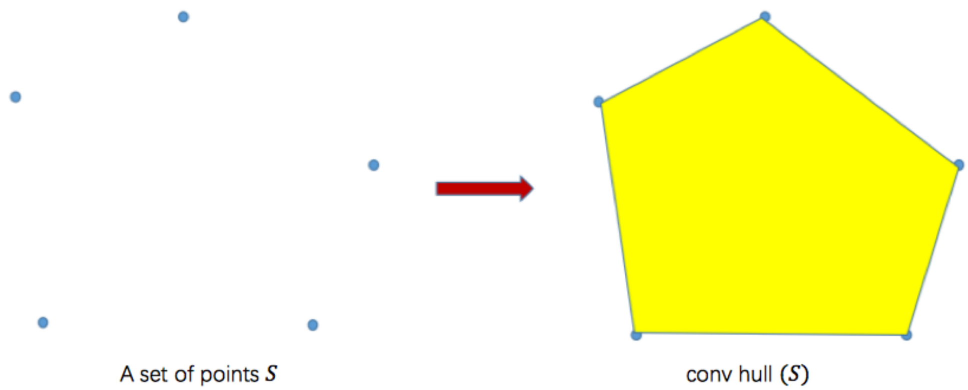

Figure 2.7: Illustration for a convex hull

2. A *polytope* is also a *polyhedron*

3. In general, a polyhedron can be written as
$$\left\{\sum_{i=1}^{m}\lambda_i p_i+\sum_{j=1}^{\ell}\mu_j d_j \,\middle|\, \lambda_i\ge 0,\sum_{i=1}^{m}\lambda_i=1,\mu_j\ge 0\right\}$$
In other words,
$$H=P+C,$$
where H,P,C denote collections of polyhedrons,polytopes,polyhedral cones, respectively.

Definition 2.2.4. If $\mathcal{K}$ is a convex cone, then its *dual cone* is defined as
$$\mathcal{K}^*=\{\boldsymbol{s}\mid\langle\boldsymbol{s},\boldsymbol{x}\rangle\ge 0,\ \forall\boldsymbol{x}\in\mathcal{K}\}.$$

The Fig (2.8) represents one illustration for convex cone and its dual cone, but note that the dual cone may not necessarily larger than the convex cone itself. (Counter-example: Lorentz cone)

A polyhedral cone can be represented in two ways:
$$\{\boldsymbol{A}\boldsymbol{x}\mid\boldsymbol{x}\ge 0\}\quad\text{and}\quad\{\boldsymbol{x}\mid\boldsymbol{B}^{\mathrm{T}}\boldsymbol{x}\ge 0\}$$

Here we consider two cones $\{\boldsymbol{A}\boldsymbol{x}\mid\boldsymbol{x}\ge 0\}$ and $\{\boldsymbol{y}\mid\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}\ge 0\}$:

Theorem 2.2.2. Two cones $\mathcal{K}_1=\{\boldsymbol{A}\boldsymbol{x}\mid\boldsymbol{x}\ge 0\}$ and $\mathcal{K}_2=\{\boldsymbol{y}\mid\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}\ge 0\}$ are actually *duality pairs*.

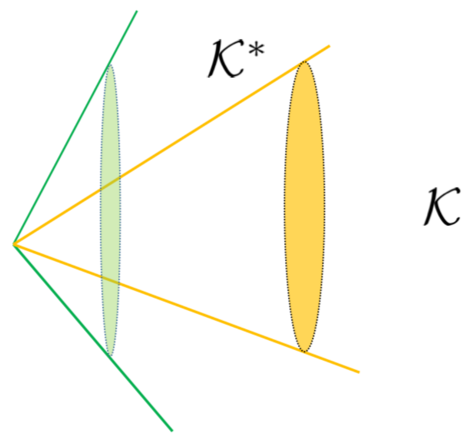

Figure 2.8: Illustration for a convex cone and its dual cone

Proof. It's obvious that $\{\boldsymbol{A}\boldsymbol{x} \mid \boldsymbol{x} \geq 0\} \subseteq \{\boldsymbol{y} \mid \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \geq 0\}^*$.

Given that $\boldsymbol{b} \notin \{\boldsymbol{A}\boldsymbol{x} \mid \boldsymbol{x} \geq 0\}$, there exists $\boldsymbol{y}$ such that $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \geq 0$ and $\langle b, y\rangle < 0$ (by Farkas Lemma), i.e., $\boldsymbol{b}$ cannot be in $\{\boldsymbol{y} \mid \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \geq 0\}^*$. $\square$

Theorem 2.2.3 (Farkas Lemma)**.** If $\boldsymbol{b} \notin \{\boldsymbol{A}\boldsymbol{x} \mid \boldsymbol{x} \geq 0\}$, then there exists $\boldsymbol{y}$ such that $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \geq 0$ and $\langle \boldsymbol{b}, \boldsymbol{y}\rangle < 0$.

An equivalent form of Farkas Lemma is as follows (known as the *theorem of alternatives*):

> Either $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0$ has a solution, or $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \geq 0, \langle \boldsymbol{b}, \boldsymbol{y}\rangle = -1$ has a solution, but not neither, nor both.

Proof. Consider the primal LP

$$(P) \qquad \begin{array}{ll} \max & -\boldsymbol{s} \\ \text{such that} & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{s}\boldsymbol{b} = \boldsymbol{b}, \\ & \boldsymbol{x} \geq 0, \boldsymbol{s} \geq 0 \end{array}$$

The dual LP is

$$(D) \qquad \begin{array}{ll} \min & \langle \boldsymbol{b}, \boldsymbol{y}\rangle \\ \text{such that} & \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \geq 0 \\ & \langle \boldsymbol{b}, \boldsymbol{y}\rangle \geq -1 \end{array}$$

Since the primal problem is feasible and bounded, and so an optimal solution exists. By strong duality theorem, $\{\boldsymbol{x} \mid \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0\} = \emptyset$ iff the (D) admits negative optimal value. $\square$

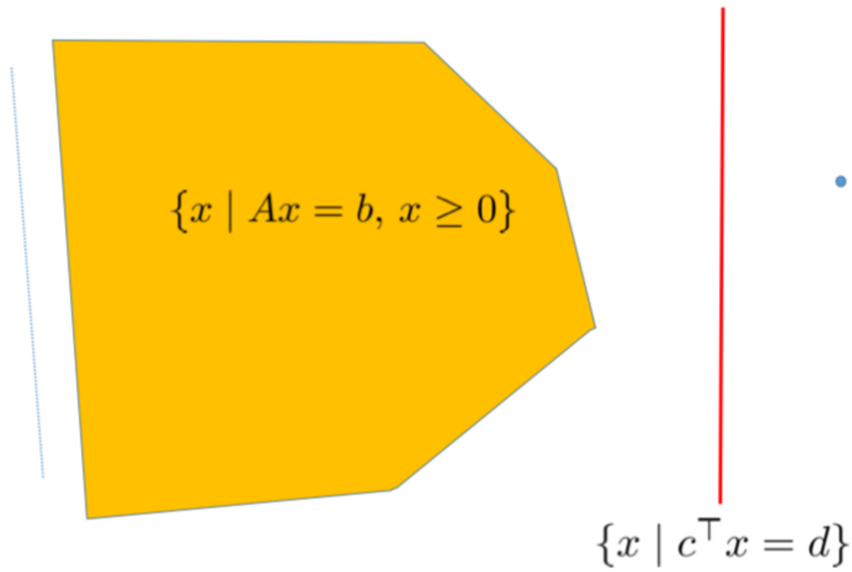

Figure 2.9: Illustration for the separation of $\mathcal{K}_1$ and $\boldsymbol{y}$

Remark 2.2.1. The Farkas Lemma essentially claims that given a polyhedron $\mathcal{K}_1 = \{\boldsymbol{x} \mid \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq 0\}$ and a point $\boldsymbol{y}$ outside $\mathcal{K}_1$, we can always find an affine that separates $\mathcal{K}_1$ and $\boldsymbol{y}$ (see Fig (2.9))

Actually, we can generalize this separation into general polyhedron.

Theorem 2.2.4 (Separation Theorem)**.** Let $H \subseteq \mathbb{R}^n$ be a general *polyhedron*. Suppose that $\mathbb{R}^n \ni \boldsymbol{p} \notin H$, then there exists an affine $f(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{d}$ satisfying

$$f(\boldsymbol{p}) < 0, \quad f(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{d} > 0, \forall \boldsymbol{x} \in H$$

Proof. Consider a general polyhedron H with vertices $\{\boldsymbol{p}_i \mid i = 1, \ldots, m\}$ and extreme rays $\{\boldsymbol{d}_j \mid j = 1, \ldots, \ell\}$. Since $\boldsymbol{p} \notin H$, the system (2.1) does not have a solution:

$$\begin{aligned}
\boldsymbol{p} &= \sum_{i=1}^{m} \lambda_i \boldsymbol{p}_i + \sum_{j=1}^{\ell} \mu_j \boldsymbol{d}_j \\
1 &= \sum_{i=1}^{m} \lambda_i \\
0 &\leq \lambda_i, \quad i = 1, \ldots, m \\
0 &\leq \mu_j, \quad j = 1, \ldots, \ell
\end{aligned} \tag{2.1}$$

By Farkas Lemma, there exists $\boldsymbol{y} \in \mathbb{R}^n$ and $\boldsymbol{s} \in \mathbb{R}$ such that

$$\begin{aligned} \boldsymbol{s} + \boldsymbol{p}_i^{\mathrm{T}} \boldsymbol{y} &\geq 0, \quad i = 1, \ldots, m \\ \boldsymbol{d}_j^{\mathrm{T}} \boldsymbol{y} &\geq 0, \quad j = 1, \ldots, \ell \\ \boldsymbol{s} + \boldsymbol{p}^{\mathrm{T}} \boldsymbol{y} &< 0 \end{aligned}$$

□

Remark 2.2.2. It's also easy to derive the Farkas Lemma from the Separation Theorem: Suppose $\boldsymbol{b} \notin \{\boldsymbol{A}\boldsymbol{x} \mid \boldsymbol{x} \geq 0\}$, then there is a separating hyperplane $\boldsymbol{c}^{\mathrm{T}} \boldsymbol{z} = d$, such that

$$\boldsymbol{c}^{\mathrm{T}} \boldsymbol{b} < d, \quad \boldsymbol{c}^{\mathrm{T}} \boldsymbol{z} > d, \forall \boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}, \text{ with } \boldsymbol{x} \geq 0$$

which implies that $\boldsymbol{A}^{\mathrm{T}} \boldsymbol{c} \geq 0, d < 0, \boldsymbol{c}^{\mathrm{T}} \boldsymbol{b} < d$.

There are different ways to show the same result. Our previous proof of Theorem (2.2.3) is based on the *finiteness of the simplex method*, i.e., the duality of LP. Terence Tao applied a different and more direct method to show the Farkas Lemma, let's study it in detail.

First we aim to show the Farkas Lemma in the dual way:

Theorem 2.2.5 (Rephrased Form of Farkas Lemma)**.** Let $P_i(x) = \sum_{j=1}^n p_{ij} x_j - r_i, i = 1, \ldots, m$. If the system $P_i(x) \geq 0$ for $i = 1, \ldots, m$ does not have a solution, then there exists $y_i \geq 0$ such that $\sum_{i=1}^m y_i P_i(x) = -1$.

Proof. We can show this result by induction on n:

- When $n = 1$, we can scale $P_i(x) \geq 0$ possibly into three cases:

  $$x_1 - a_i \geq 0, \ i \in I_+; \quad -x_1 + b_j \geq 0, \ j \in I_-; \quad c_k \geq 0, \ k \in I_0$$

  If there exists $k \in I_0$ with $c_k < 0$, we let $y_k = -1/c_k$ and $y_\ell = 0, \forall \ell \neq k$; otherwise there exists $i \in I_+$ and $j \in I_-$ with $a_i > b_j$, we let $y_i = y_j = 1/(a_i - b_j)$ and $y_\ell = 0, \forall \ell \neq i, j$. Therefore, we obtain $\sum_{i=1}^m y_i P_i(x) = -1$.

- Now suppose that this theorem is shown for dimension no more than n. Consider the case where we have $n+1$ variables: $\bar{\boldsymbol{x}} = (\boldsymbol{x}, x_{n+1})$, $\boldsymbol{x} \in \mathbb{R}^n$. We can scale the original inequalities over $P_i(\bar{\boldsymbol{x}})$ according to the coefficients of x_{n+1} to obtain the following equivalent system of inequalities:

  $$\begin{cases} P_i(\bar{\boldsymbol{x}}) := x_{n+1} - Q_i(\boldsymbol{x}) \geq 0, \quad i \in I_+ \\ P_j(\bar{\boldsymbol{x}}) := -x_{n+1} - Q_j(\boldsymbol{x}) \geq 0, \quad j \in I_- \\ P_k(\bar{\boldsymbol{x}}) := Q_k(\boldsymbol{x}) \geq 0, \quad k \in I_0 \end{cases} \tag{2.2}$$

Note that the system

$$\begin{cases} -Q_i(x) + Q_j(x) \geq 0, & i \in I_+, j \in I_- \\ \qquad\qquad Q_k(x) \geq 0, & k \in I_0 \end{cases} \tag{2.3}$$

has a solution implies that the original system would have a solution too. Due to the hypothesis of the lemma, (2.3) does not have a solution.

By the induction hypothesis, there exists $y_{ij}, y_k \geq 0$ such that

$$\sum_{i \in I_+, j \in I_-} y_{ij}(-Q_i(x) + Q_j(x)) + \sum_{k \in I_-} y_k Q_k(x) = -1$$

Therefore, we imply

$$\sum_{i \in I_+} \left(\sum_{j \in I_-} y_{ij} \right) (x_{n+1} - Q_i(x)) + \sum_{j \in I_-} \left(\sum_{i \in I_+} y_{ij} \right) (-x_{n+1} + Q_j(x)) \\ + \sum_{k \in I_0} y_k Q_k(x) = -1$$

The Farkas Lemma is shown by this induction argument. □

Then we are ready to show the following form of separation theorem:

Theorem 2.2.6 (Repharsed Form of Separation Theorem)**.** If polytopes P_1, P_2 do not intersect, then there is an affine $f(\boldsymbol{x}) = \langle \boldsymbol{y}, \boldsymbol{x} \rangle + y_0$ such that

$$f(\boldsymbol{x}) \geq 1,\ \forall \boldsymbol{x} \in P_1 \quad f(\boldsymbol{x}) \leq 1,\ \forall \boldsymbol{x} \in P_2$$

Proof. Suppose the extreme points of P_1, P_2 are $\{\boldsymbol{p}_1, \dots, \boldsymbol{p}_s\}$ and $\{\boldsymbol{q}_1, \dots, \boldsymbol{q}_t\}$, respectively. Therefore, it suffices to show that

$$\langle \boldsymbol{y}, \boldsymbol{p}_i \rangle + y_0 \geq 1,\ i = 1, \dots, s; \quad \langle \boldsymbol{y}, \boldsymbol{q}_j \rangle + y_0 \leq -1,\ j = 1, \dots, t \tag{2.4}$$

Suppose on the contrary that (2.4) does not have a solution $(\boldsymbol{y}, \boldsymbol{y}_0)$. Then the Farkas Lemma asserts that there exists $u_i \geq 0, i = 1, \dots, s$ and $v_j \geq 0, j = 1, \dots, t$ such that

$$\sum_{i=1}^{s} u_i(\langle \boldsymbol{y}, \boldsymbol{p}_i \rangle + y_0 - 1) + \sum_{j=1}^{t} v_j(-\langle \boldsymbol{y}, \boldsymbol{q}_j \rangle - y_0 - 1) = -1$$

which implies

$$\sum_{i=1}^{s} u_i \boldsymbol{p}_i - \sum_{j=1}^{t} v_j \boldsymbol{q}_j = 0$$
$$\sum_{i=1}^{s} u_i - \sum_{j=1}^{t} v_j = 0$$
$$\sum_{i=1}^{s} u_i + \sum_{j=1}^{t} v_j = 1$$

Therefore, $\sum_{i=1}^{s} u_i = \sum_{j=1}^{t} v_j = 1/2$, and thus

$$P_1 \ni 2\sum_{i=1}^{s} u_i \boldsymbol{p}_i = 2\sum_{j=1}^{t} v_j \boldsymbol{q}_j \in P_2$$

which contradicts to the assumption that $P_1 \cap P_2 = \emptyset$. $\square$

The same argument applies if *polytopes* is replaced by *polyhedron*.

2.3 More Theorems of Alternatives: the case for polyhedrons

Some more refined forms of the theorems of alternatives exist. Here we list some examples.

Notations Let $\boldsymbol{x}, \boldsymbol{y}$ be two vectors, we denote $\boldsymbol{x} \gneqq \boldsymbol{y}$ to be "$\boldsymbol{x} \geq \boldsymbol{y}$ and $\boldsymbol{x} \neq \boldsymbol{y}$", and similarly for $\boldsymbol{x} \lneqq \boldsymbol{y}$.

Theorem 2.3.1 (Gordan)**.** Either $\boldsymbol{A}\boldsymbol{x} > 0$ has a solution, or $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} = 0, \boldsymbol{y} \gneqq 0$ has a solution.

Theorem 2.3.2 (Stiemke)**.** Either $\boldsymbol{A}\boldsymbol{x} \gneqq 0$ has a solution, or $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} = 0, \boldsymbol{y} > 0$ has a solution.

Theorem 2.3.3 (Gale)**.** Assuming $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$ is feasible, then either $\boldsymbol{A}\boldsymbol{x} \lneqq \boldsymbol{b}$ has a solution, or $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} = 0, \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0, \boldsymbol{y} > 0$ has a solution.

Theorem 2.3.4 (Tucker)**.** Suppose that $\boldsymbol{A} \neq \boldsymbol{0}$. Either $\boldsymbol{A}\boldsymbol{x} \gneqq 0, \boldsymbol{B}\boldsymbol{x} \geq 0, \boldsymbol{C}\boldsymbol{x} = 0$ has a solution, or $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{u} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{v} + \boldsymbol{C}^{\mathrm{T}}\boldsymbol{w} = 0, \boldsymbol{u} > 0, \boldsymbol{v} \geq 0$ has a solution.

Theorem 2.3.5 (Motzkin)**.** Suppose that $\boldsymbol{A} \neq \boldsymbol{0}$. Either

$$\boldsymbol{A}\boldsymbol{x} > 0, \boldsymbol{B}\boldsymbol{x} \geq 0, \boldsymbol{C}\boldsymbol{x} = 0$$

has a solution, or

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{u} + \boldsymbol{B}^{\mathrm{T}}\boldsymbol{v} + \boldsymbol{C}^{\mathrm{T}}\boldsymbol{w} = 0, \boldsymbol{u} \gneqq 0, \boldsymbol{v} \geq 0$$

has a solution.

3 Computational Complexity for Linear Programming

3.1 A case study

Suppose we aim to find a shortest path on a directed network, from the source node s to the sink node t. The network is supposed to be $\mathcal{N} = (V, E; w)$, where $V = \{v_1, \dots, v_n\}$ is the set of nodes, E is the set of edges, and w is the vectors of weights, i.e., w_{ij} denotes the weight on the edge (v_i, v_j).

We need to input this instance into the computer. One way is to use the *node-arc* incidence matrix:

$$a_{ij} = \begin{cases} -1, & \text{if } v_i \text{ is the head of arc } e_j \\ 0, & \text{if } v_i \text{ is not related to arc } e_j \\ +1, & \text{if } v_i \text{ is the tail of arc } e_j \end{cases}$$

Input size of a problem

1. Suppose $|V| = n$ and $|E| = m$, then inputting the matrix A requires to touch the keyboard $mn + 2m$ times.

2. To tell the vector w into computer, we need w to be *integer-valued* (since otherwise the problem will be much more difficult. question: do w need to be positive?). Typing w_{ij} requires at most $\lfloor \log_2(|w_{ij}| + 1) \rfloor + 1$ bits. Therefore typing the vector w requires totally
$$\sum_{(v_i, v_j) \in E} \lfloor \log_2(|w_{ij}| + 1) \rfloor + |E| \text{ bits}$$

3. Therefore, the input-length (size) of this problem is
$$L = mn + 3m + \sum_{(v_i, v_j) \in E} \lfloor \log_2(|w_{ij}| + 1) \rfloor.$$

The Running Time Complexity It's logical that the total amount of perations required by an algorithm to solve this problem is *dependent* on L.

Consider applying the *Dijkstra* algorithm to find the shortest path from the source node s to the sink node t. Define a relevant working set S that contains all the nodes with *known shortest distance* to t. Initially $S = \{t\}$. At each iteration, by *dynamic programming principle*, one node is identified to be in S after at most $\mathcal{O}(|V|)$ comparisions. Therefore, the overall computational complexity is $\mathcal{O}(|V|^2)$.

3.2 Computational Complexity

Polynomial Time Algorithm If an algorithm solves a combinatorial problem with input size L with total number of operations no more than a polynomial of L, then it is called a *polynomial-time* algorithm.

The Dijkstra algorithm requires no more than $\mathcal{O}(L^2)$ (question: $\mathcal{O}(L)$ or $\mathcal{O}(L^2)$?) operations to terminate, and therefore it is a polynomial-time algorithm.

However, there are many combinatorial problems for which no polynomial time algorithms are known, e.g., the longest path problem, the Hamitonian circle problem, the maximum cut, and many more.

3.2.1 P & NP

Definition 3.2.1 (NP problem)**.** If a combinatorial decision problem with input size L is such that:

> if the answer to the problem is yes, then a yes-certificate exists and the length (size) of which is polynomial in L.

then we call the problem is *NP*.

Definition 3.2.2 (co-NP problem)**.** If a combinatorial decision problem with input size L is such that:

> if the answer to the problem is no, then a no-certificate exists and the length (size) of which is polynomial in L.

then we call the problem is *co-NP*.

Definition 3.2.3 (P problem)**.** A *polynomially solvable* problem is called to be *P*.

It's clear that $P \subseteq NP$ and $P \subseteq$ *co-NP*.

Definition 3.2.4 (NP-Complete problem)**.** The problem in NP such that any other problem in NP can be reduced to it is called to be *NP-Complete*.

It's *strongly believed (haven't shown)* that NP-Complete problems are not in P.

3.2.2 Dimensions and Parameters

A decision problem typically involves *dimension* and the *parameters* of the problem, though thery are mixed in the definition of input-length L.

- In the case study, $n = |V|$ and $m = |E|$ are known to be the problem dimensions, and the weight w is known as the problem parameter.

- For the linear programming with integer-valued parameters

$$\begin{array}{ll} \max & \sum_{j=1}^{n} c_j x_j \\ \text{such that} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \quad j = 1, \ldots, n \end{array}$$

  the problem dimension would be m and n, and the input-length is

$$L = mn + m + n + \sum_{j=1}^{n} \lfloor \log_2(|c_j| + 1) \rfloor + \sum_{i=1}^{m} \lfloor \log_2(|b_i| + 1) \rfloor + \sum_{i=1}^{m} \sum_{j=1}^{n} \lfloor \log_2(|a_{ij} + 1|) \rfloor.$$

3.2.3 Other terminologies

Definition 3.2.5 (Weak & Strong Polynomial Time Algorithms)**.** If a *polynomial-time* algorithm requires number of operations to be *polynomial* in dimensions, then the algorithm is called the *strongly polynomial*, otherwise it is only *weakly polynomial*

Remark 3.2.1. Note that the input size of a problem defines the polynomial time, and the dimension specializes the strong & weak polynomial time. The number of operations is different from the number of *bit* operations. The Dijkstra algorithm is strongly polynomial.

Definition 3.2.6 (Weak & Strong NP-Completeness)**.** If an *NP-complete* is such that *even when it is restricted to be the case where all the parameters are constants*, it still remains to be *NP-complete*, then it is called *strongly NP-Complete*, otherwise it is called *weakly NP-complete.*

Example: the 2-partition problem is weakly NP-complete; the Hamiltonian circle problem is strongly NP-complete.

Definition 3.2.7. If a decision version of the problem is *NP-complete* or *co-NP-complete*, then the problem is called *NP-Hard*.

For example, the travelling salesman (TRS) problem is NP-Hard.

3.3 Complexity of Linear Programming

3.3.1 LP is both NP and Co-NP

The decision version of LP is: does there exists $\boldsymbol{x}$ satisfying

$$\boldsymbol{c}^{\mathrm{T}} \boldsymbol{x} \geq \boldsymbol{v}, \quad \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}, \quad \boldsymbol{x} \geq 0?$$

1. If the answer is yes, then a certificate is such $\boldsymbol{x}$, whose size can be bounded by a polynomial of the input-length.

2. If the answer is no, then the above system is infeasible. By Farkas Lamma, there exists $y_0 \geq 0$ and $\boldsymbol{y} \geq 0$ such that

$$-y_0 \boldsymbol{c}^{\mathrm{T}} + \boldsymbol{y}^{\mathrm{T}} \boldsymbol{A} \geq 0, \quad -y_0 \boldsymbol{v} + \boldsymbol{y}^{\mathrm{T}} \boldsymbol{b} < 0.$$

   In this case, the size of $(y_0, \boldsymbol{y})$ can be bounded by a polynomial of the input-length.

3.3.2 Is LP in P?

In the complexity theory, we believe that $P \subsetneq NP$, i.e., P $\neq$ NP-Complete and NP $\cap$ Co-NP = P. Therefore, a natural question arises: Is linear programming in P? The answer is no, but the simplex method is *not* a polynomial time algorithm. Let's study an example first.

The Klee-Minty Example (n=3) Consider solving a LP using the Largest Coefficient Rule:

$$\begin{array}{ll} \max & 100x_1 + 10x_2 + x_3 \\ \text{such that} & x_1 \leq 1 \\ & 20x_1 + x_2 \leq 100 \\ & 200x_1 + 20x_2 + x_3 \leq 10000 \\ & x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{array} \tag{3.1a}$$

We can find its dictionary:

$$\begin{aligned} \boldsymbol{x_4} &= 1 - x_1 \\ x_5 &= 100 - 20x_1 - x_2 \\ x_6 &= 10000 - 200x_1 - 20x_2 - x_3 \\ z &= \mathbf{100}\boldsymbol{x_1} + 10x_2 + x_3 \end{aligned} \tag{3.1b}$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_1, we obtain:

$$\begin{aligned} x_1 &= 1 - x_4 \\ \boldsymbol{x_5} &= 80 + 2 - x_4 - x_2 \\ x_6 &= 9800 + 200x_4 - 20x_2 - x_3 \\ z &= 100 - 100x_4 + \mathbf{10}\boldsymbol{x_2} + x_3 \end{aligned} \tag{3.1c}$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_2, we obtain:

$$\begin{aligned} \boldsymbol{x_1} &= 1 - x_4 \\ x_2 &= 80 + 20x_4 - x_5 \\ x_6 &= 8200 - 200x_4 + 20x_5 - x_3 \\ z &= 900 + \mathbf{100}\boldsymbol{x_4} - 10x_5 + x_3 \end{aligned} \tag{3.1d}$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_4, we obtain:

$$
\begin{aligned}
x_4 &= 1 - x_1 \\
x_2 &= 100 - 20x_1 - x_5 \\
\boldsymbol{x_6} &= 8000 + 200x_1 + 20x_5 - x_3 \\
z &= 1000 - 100x_1 - 10x_5 + \boldsymbol{x_3}
\end{aligned}
\tag{3.1e}
$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_3, we obtain:

$$
\begin{aligned}
\boldsymbol{x_4} &= 1 - x_1 \\
x_2 &= 100 - 20x_1 - x_5 \\
x_3 &= 8000 + 200x_1 + 20x_5 - x_6 \\
z &= 9000 + \mathbf{100}\boldsymbol{x_1} + 10x_5 - x_6
\end{aligned}
\tag{3.1f}
$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_1, we obtain:

$$
\begin{aligned}
x_1 &= 1 - x_4 \\
\boldsymbol{x_2} &= 80 + 20x_4 - x_5 \\
x_3 &= 8200 - 200x_4 + 20x_5 - x_6 \\
z &= 9100 - 100x_4 + \mathbf{10}\boldsymbol{x_5} - x_6
\end{aligned}
\tag{3.1g}
$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_5, we obtain:

$$
\begin{aligned}
\boldsymbol{x_1} &= 1 - x_4 \\
x_5 &= 80 + 20x_4 - x_2 \\
x_3 &= 9800 + 200x_4 - 20x_2 - x_6 \\
z &= 9900 + \mathbf{100}\boldsymbol{x_4} - 10x_2 - x_6
\end{aligned}
\tag{3.1h}
$$

If we choose the variable with largest coefficient to enter the basis, i.e., x_4, we obtain:

$$
\begin{aligned}
x_4 &= 1 - x_1 \\
x_5 &= 100 - 20x_1 - x_2 \\
x_3 &= 10000 - 200x_1 - 20x_2 - x_6 \\
z &= 10000 - 100x_1 - 10x_2 - x_6
\end{aligned}
\tag{3.1i}
$$

After total 7 simplex pivot steps, we get the optimal solution. This process is mazy, while it only contains 3 variables.

General Case Consider the general Klee-Minty example:

$$
\begin{array}{ll}
\text{max} & \sum_{j=1}^{n} 10^{n-j} x_j \\
\text{such that} & 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \leq 100^{i-1}, \quad i = 1, \ldots, n \\
& x_j \geq 0, \quad j = 1, \ldots, n
\end{array}
$$

Its input-length (in digits) should be:

$$
\begin{aligned}
&\sum_{j=1}^{n}(n-j-1)+\sum_{i=1}^{n}(2(i-1)+1)+\sum_{i=1}^{n}\left(\sum_{j=1}^{i-1}(i-j-1)+1\right)\\
&=\frac{n^3+12n^2+5n}{6}\\
&=\mathcal{O}(n^3)
\end{aligned}
$$

Let the slack variables be:

$$
s_i := 100^{i-1}-2\sum_{j=1}^{i-1}10^{i-j}x_j - x_i, \quad i=1,\dots,n
$$

One can show that in every feasible basis, either x_i or s_i must be in the basis, $i=1,\dots,n$, which implies that there are 2^n feasible basis in total.

If applying the largest coefficient pivot rule, then after $2^{n-1}-1$ iterations, the last row reads

$$
z = 10\left(100^{n-2}-\sum_{j=1}^{n-2}10^{n-1-j}x_j - s_{n-1}\right)+x_n
$$

After further 2^{n-1} iterations, the last row reads

$$
z = 90\cdot 100^{n-2}+10\left(\sum_{j=1}^{n-2}10^{n-1-j}x_j + s_{n-1}\right)-s_n
$$

After further 2^{n-1} iterations, the last row reads

$$
z = 100^{n-1}-\sum_{j=1}^{n-1}10^{n-j}x_j - s_n,
$$

which corresponds to an optimal basic solution.

Remark 3.3.1. Since the total number of pivot steps for the Klee-Minty example is $\mathcal{O}(2^n-1)$ for a problem whose input-length is $\mathcal{O}(n^3)$, it shows that in the worst case the simplex method with the largest coefficient pivot rule is exponential.

Most other conceivable simplex pivot rules all admit similar exponential examples. However, it remains open that whether we can find a special simplex pivot rule that is polynomial even in the worst case.

The following figure shows the Geometric process for the Klee-Minty example with $n=3$, and here $(s_1,s_2,s_3)=(x_1,100x_2,10000x_3)$:

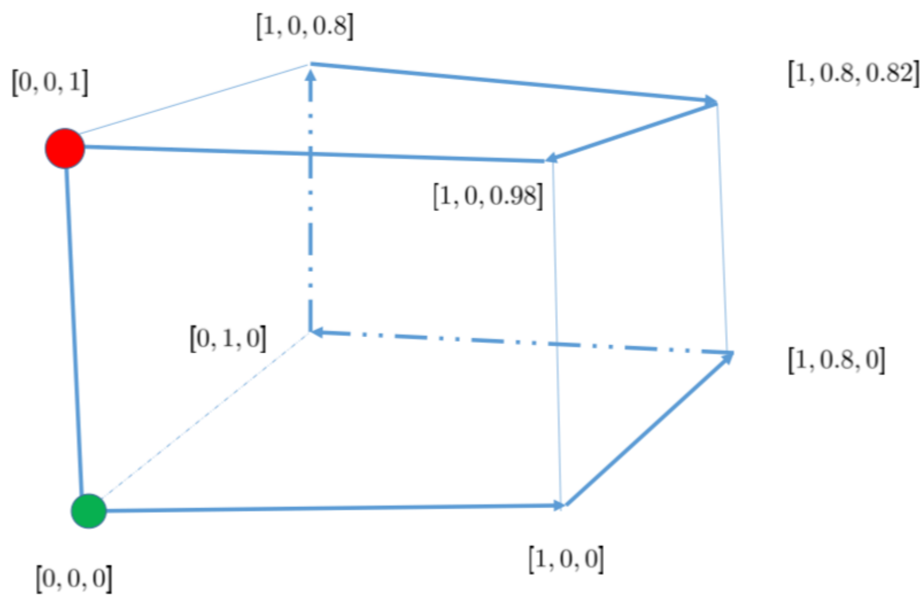

Figure 3.1: A Geometric Picture for the Klee-Minty example with $n = 3$

4 The KKT Condition for Nonlinear Programming

4.1 unconstrained Optimality Condition

Consider the unconstrained optimization

$$\min \quad f(x) \tag{4.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is an m-th order *continuously differentiable* function. We aim to find the optimality condition for this problem.

Theorem 4.1.1 (First Order Necessary Condition)**.** Given the condition that $f \in \mathcal{C}^1$, if x^* is a local minimum point, then $\nabla f(x^*) = 0$.

Theorem 4.1.2 (Second Order Necessary Condition)**.** Given the condition that $f \in \mathcal{C}^2$, if x^* is a local minimum point, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succeq 0$.

Theorem 4.1.3 (Second Order Sufficient Condition)**.** Given the condition that $f \in \mathcal{C}^2$, if $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$, then x^* is a local minimum point.

Theorem 4.1.4 (First Order Necessary and Sufficient Condition)**.** If f is convex, then x^* is a local minimum point if and only if $0 \ni \partial f(x^*)$.

Proof. All theorems above relies on the Taylor expansion and

$$f(x^* + \Delta x) \geq f(x^*), \forall \Delta x \Longleftrightarrow \text{the point } x^* \text{ is minimum}$$

$\square$

All the conditions above only involve the explicit quantities related to x. Now we turn into the constrained optimization case.

4.2 Constrained Optimality Condition

Consider the special constrained optimization

$$\begin{array}{ll} \min & f(x) \\ \text{such that} & x \in \mathcal{X} \end{array} \tag{4.2}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ is usually pre-assumed to be a closed convex set.

Definition 4.2.1 (Tangent Cone)**.** Let $\hat{x} \in \mathcal{X}$. The *tangent cone* of $\mathcal{X}$ at point $\hat{x}$ is defined as

$$\mathcal{T}(\hat{x}) := \left\{ y \neq 0 \middle| \exists x^k \in \mathcal{X} : \ x^k \to \hat{x} \ \& \ \frac{x^k - \hat{x}}{\|x^k - \hat{x}\|} \to \frac{y}{\|y\|} \right\} \cup \{0\}$$

It's clear that $\mathcal{T}(\hat{x})$ denotes all feasible directions from $\hat{x}$ within $\mathcal{X}$

Therefore we obtain a necessary optimality condition for constrained optimization, which is beautiful in theory but not that useful in practice:

Theorem 4.2.1 (First Order Necessary Condition)**.** If x^* is a local minimum point for the constraint (neither necessarily convex nor closed) set $\mathcal{X}$, then

$$\langle \nabla f(x^*), y \rangle \geq 0, \ \forall y \in \mathcal{T}(x^*),$$

i.e., $\nabla f(x^*) \in (\mathcal{T}(x^*))^*$.

Theorem 4.2.2. The above condition becomes necessary and sufficient if f is convex. In that case, we may weaken the condition into

$$\partial f(x^*) \cap (\mathcal{T}(x^*))^* \neq \emptyset.$$

Proof. The proof relies on the fact that f is convex iff

$$f(y) \geq f(x) + \langle d, (y - x) \rangle, \forall x, y \in \mathcal{X},$$

where $d \in \partial f(x^*)$. $\square$

Theorem 4.2.3. Given further condition that $\mathcal{X}$ is a convex set, if x^* is a local minimum point for the constraint set $\mathcal{X}$, then

$$\langle \nabla f(x^*), y - x^* \rangle \geq 0, \ \forall y \in \mathcal{X}.$$

This condition becomes sufficient if f is convex.

4.2.1 Characteration of Tangent Cone

1. Consider the polyhedral constraint set

$$\mathcal{X} = \{x \mid a_i^{\mathrm{T}} x \le b_i,\ i = 1, \ldots, m\}$$

   Let $\hat{x} \in \mathcal{X}$, and $I(\hat{x}) = \{i \mid a_i^{\mathrm{T}} \hat{x} = b_i\}$. Then we have

$$\mathcal{T}(\hat{x}) = \{d \mid a_i^{\mathrm{T}} d \le 0,\ \forall i \in I(\hat{x})\}.$$

2. Consider another more general constraint set

$$\mathcal{X} = \left\{ x \middle| \begin{matrix} h_i(x) = 0,\ i = 1, \ldots, m; \\ g_j(x) \le 0,\ j = 1, \ldots, r \end{matrix} \right\}.$$

   For $\hat{x} \in \mathcal{X}$, define $I(\hat{x}) = \{j \mid g_j(\hat{x}) = 0\}$ likewise. Let's introduce a easily computable cone

$$\mathcal{C}(\hat{x}) = \left\{ d \middle| \begin{matrix} \langle \nabla h_i(\hat{x}), d \rangle = 0,\ i = 1, \ldots, m; \\ \langle \nabla g_j(\hat{x}), d \rangle \le 0,\ \forall j \in I(\hat{x}) \end{matrix} \right\}.$$

   It's clear that $\mathcal{T}(\hat{x}) \subseteq \mathcal{C}(\hat{x})$. However, in general $\mathcal{T}(\hat{x}) \ne \mathcal{C}(\hat{x})$. (Consider $\mathcal{X} = \{(x_1, x_2) \mid x_1 \ge 0, x_2^2 \le 0\}$).

   A popular condition to ensure the equality is the *Linear Independence Constraint Qualification* (LICQ):

   > The vectors $\nabla h_i(x),\ i = 1, \ldots, m; \nabla g_j(x),\ j \in I(x)$ are always linearly independent for $\forall x \in \mathcal{X}$.

   •

5 The Distribution and Installation

5.1 Pre-requisites

You will need a working LaTeX installation. We recomend using pdflatex to process the files. You will also need biber.exe installed. This is distributed as part of the latest versions of LiveTex and MikTex. If you have problems, please let us know.

5.2 The Distribution

The distribution contains 2 folders: `nowfnt` and `nowfnttexmf`.

5.2.1 Folder `nowfnt`

This folder contains the following files using a flat stucture required to compile a FnT issue:

- essence_logo.eps
- essence_logo.pdf
- now_logo.eps
- now_logo.pdf
- nowfnt.cls
- nowfnt-biblatex.sty
- NOWFnT-data.tex

It also contains the following folders:

journaldata A set of data files containing the journal-specific data for each journal. There are three files per journal:
<jrnlcode>-editorialboard.tex
<jrnlcode>-journaldata.tex
<jrnlcode>-seriespage.tex
<jrnlcode> is the code given in Appendix A. You will need these three files to compile your article.

SampleArticle This folder contains this document as an example of an article typeset in our class file. The document is called `FnT`article.tex. It also contains this PDF file and the .bib file.

5.2.2 Folder `nowfnttexmf`

This folder contains all the files required in a texmf structure for easy installation.

5.3 Installation

If your LaTeX installation uses a localtexmf folder, you can copy the `n`owtexmf folder to the localtexmf folder and make it known to your TeX installation. You can now proceed to use the class file as normal.

If you prefer to use the flat files, you will need to copy all the required files each time into the folder in which you are compiling the article. Do not forget to copy the three data files for the specific journal from the folder `journaldata`.

You may need to configure your TEX editor to be able to run the programs. If you have problems installing these files in your own system, please contact us. We use Computer Modern fonts for some of the journals. You will need to make sure that these fonts are installed. Refer to your system documentation on how to do this.

6 Quick Start

The now-journal class file is designed is such a way that you should be able to use any commands you normally would. However, do **not** modify any class or style files included in our distribution. If you do so, we will reject your files.

The preamble contains a number of commands for use when making the final versions of your manuscript once it has been accepted and you have been instructed by our production team.

6.1 \documentclass

The options to this command enable you to choose the journal for which you producing content and to indicate the use of biber.

```
\documentclass[<jrnlcode>,biber]{nowfnt}.
```

<jrnlcode> is the pre-defined code identifying each journal. See Appendix A for the appropriate <jrnlcode>.

6.2 \issuesetup

These commands are only used in the final published version. Leave these as the default until our production team instructs you to change them.

6.3 \maintitleauthorlist

This is the authors list for the cover page. Use the name, affilliation and email address. Separate each line in the address by `\\`.

Separate authors by \and. Do not use verbatim or problematic symbols. _ (underscore) in email address should be entered as `\_`. Pay attention to long email addresses.

If your author list is too long to fit on a single page you can use double column. In this case, precede the \maintitleauthorlist command with the following:

```
\booltrue{authortwocolumn}
```

6.4 \author and \affil

These commands are used to typeset the authors and the afilliations on the abstract page of the article and in the bibliographic data.

\author uses an optional number to match the author with the affiliation. The author name is written <surname>, <firstname>.

\affil uses an optional number to match the author with the author name. The content is <affililiation>; <email address>.

6.5 \addbibresource

Use this to identify the name of the bib file to be used.

7 Style Guidelines and LaTeX Conventions

In this section, we outline guidelines for typesetting and using LaTeX that you should follow when preparing your document

7.1 Abstract

Ensure that the abstract is contained within the

```
\begin{abstract}
```

environment.

7.2 Acknowledgements

Ensure that the acknowledgements are contained within the

```
\begin{acknowledgements}
```

environment.

7.3 References

now publishers uses two main reference styles. One is numeric and the other is author/year. The style for this is pre-defined in the LaTeX distribution and must not be altered. The style used for each journal is given in the table in Appendix A. Consult the sample-now.bib file for an example of different reference types.

The References section is generated by placing the following commands at the end of the file.

```
\backmatter
\printbibliography
```

7.4 Citations

Use standard \cite, \citep and \citet commands to generate citations.

Run biber on your file after compiling the article. This will automatically create the correct style and format for the References.

7.4.1 Example citations

This section cites some sample references for your convenience. These are in author/year format and the output is shown in the References at the end of this document.

Example output when using `citet`: **arvolumenumber** is a citation of reference 1 and **ber1995** is a citation of reference 2.

Example output when using `citep`: (**beditorvolumenumber**) is a citation of reference 3 and (**inproceedings**) is a citation of reference 4.

7.5 Preface and Other Special Chapters

If you want to include a preface, it should be defined as follows:

```
\chapter*{Preface}
\markboth{\sffamily\slshape Preface}
  {\sffamily\slshape Preface}
```

This ensures that the preface appears correctly in the running headings.

You can follow a similar procedure if you want to include additional unnumbered chapters (*e.g.*, a chapter on notation used in the paper), though all such chapters should precede Chapter 1.

Unnumbered chapters should not include numbered sections. If you want to break your preface into sections, use the starred versions of `section`, `subsection`, *etc.*

7.6 Long Chapter and Section Names

If you have a very long chapter or section name, it may not appear nicely in the table of contents, running heading, document body, or some subset of these. It is possible to have different text appear in all three places if needed using the following code:

```
\chapter[Table of Contents Name]{Body Text Name}
\chaptermark{Running Heading Name}
```

Sections can be handled similarly using the `sectionmark` command instead of `chaptermark`.

For example, the full name should always appear in the table of contents, but may need a manual line break to look good. For the running heading, an abbreviated version of the title should be provided. The appearance of the long title in the body may look fine with LaTeX's default line breaking method or may need a manual line break somewhere, possibly in a different place from the contents listing.

Long titles for the article itself should be left as is, with no manual line breaks introduced. The article title is used automatically in a number of different places by the class file and manual line breaks will interfere with the output. If you have questions about how the title appears in the front matter, please contact us.

7.7 Internet Addresses

The class file includes the `url` package, so you should wrap email and web addresses with `\url{}`. This will also make these links clickable in the PDF.

8 Compiling Your FnT Article

During the first run using the class file, a number of new files will be created that are used to create the book and ebook versions during the final production stage. You can ignore these until preparing the final versions as described in Section 8.3. A complete list of the files produced are given in Appendix B.

8.1 Compiling Your Article Prior to Submission

To compile an article prior to submission proceed as follows:

Step 1: Compile the LaTeX file using pdflatex.

Step 2: Run biber on your file.

Step 3: Compile again using pdfLaTeX. Repeat this step.

Step 4: Inspect the PDF for bad typesetting. The output PDF should be similar to FnTarticle.pdf. Work from the first page when making adjustments to resolve bad line breaks and bad page breaks. Re-run pdfLaTeX on the file to check the output after each change.

8.2 Preparing the Final Versions

If you choose the option to compile the final versions of your PDF for publication, you will receive a set of data from our production team upon final acceptance. With the exception of "lastpage", enter the data into the `\issuesetup` command in the preamble.

lastpage= This is the last page number in the sequential numbering of the journal volume. You will need to enter this once you have compiled the article once (see below).

8.3 Compiling The Final Versions

The final versions should be created once you received all the bibliographic data from our Production Team and you've entered it into the preamble. You will be creating a final online journal version pdf; a printed book version pdf; and an ebook version pdf.

Step 1: Compile the LaTeX file using pdfLaTeX.**arvolumeonly**

Step 2: Run biber on your file.

Step 3: Compile again using pdfLaTeX. Repeat this step.

Step 4: Inspect the PDF for bad typesetting. The output PDF should be similar to FnTarticle.pdf. Work from the first page when making adjustments to resolve bad line breaks and bad page breaks. Re-run pdfLaTeX on the file to check the output after each change.

Step 5: When you are happy with the output make a note of the last page number and enter this in `\issuesetup`.

Step 6: Compile the article again.

Step 7: Open the file <YourFilename>-nowbook.tex. This will generate the printed book version pdf.

Step 8: Compile the LaTeX file using pdflatex.

Step 9: Run biber on your file.

Step 10: Compile again using pdfLaTeX. Repeat this step.

Step 11: Repeat steps 7-10 on the file: <YourFilename>-nowebook.tex. This will generate the ebook version pdf.

Step 12: Repeat steps 7-10 on the file: <YourFilename>-nowplain.tex. This will generate a plain version pdf of your article. If you intend to post your article in an online repository, please use this version.

8.4 Wednesday

Proposition 8.4.1. G is hamiltionian implies that for any nonempty $S \subseteq V$, $G - S$ has at most $|S|$ components.

Theorem 8.4.1. G is Hamiltionian if for any vetices v, w such that $(v, w) \notin E$,

$$\deg(v) + \deg(w) \geq n$$

Knapsacle Problem

$$\begin{array}{ll} \max & \sum_{i=1}^n v_i u_i \\ \text{such that} & \sum_{i=1}^n w_i u_i \leq W \\ & u_i \in \{0,1\},\ i = 1, \dots, n \end{array}$$

This is the binary integer programming problem, which is NP-complete. We cannot find exact solution to this problem. There are 2^n possible solutions, which requires exponential time.

We can find some “pseudo”-polynomial algorithm. Assumption:

1. W is an integer.

State: remaining capacity, define as X_k.
Stage: iterm $1, \dots, n$.

$$J_N(x_N) = \begin{cases} v_N & \text{if } N \leq X_N \\ 0, & \text{if } w_N > X_N \end{cases}$$

$$J_N(x_{N-1}) = \begin{cases} 0 + J_N(X_N), & \text{if } w_{N-1} > X_{N-1}, \text{Here } X_N = X_{N-1} \\ \max\{v_{N-1} + J_N(x_N), 0 + J_N(x_N)\}, & \text{if } w_{N-1} \leq X_{N-1} \end{cases}$$

Here why the DP has "pesduo"-polynomial time, and the w_i should be integer?

Let's list states $\{x_1, \ldots, x_N\}$. For each stage k, there would be $W + 1$.

Each iteration at most 2 computation, and therefore we face $2(W + 1) \cdot N$

8.4.1 Label Correcting Methods

Shortest Path. Back up some information.

The label refers to the intermediate information.

A^*-algorithm; Bellman-Ford Algorithm

Benchmark: MILP, CPLEX, CVX.

8.4.2 DP problems with perfect state information

Linear Quadratic System Let $x_k \in \mathbb{R}^1$ be the state; $u_k \in \mathbb{R}^n$ be the control; $\omega_k \in \mathbb{R}^n$ be the disturbance.

System Dynamics.

$$x_{k+1} = A_k x_k + B_k u_k + \omega_k$$

Stage cost:

$$x_k' Q_k x_k + u_k' R_k u_k$$

Here Q_k is required to be positive-semi-definite symmetric matrix; R_k to be positive-definite symmetric matrix.

$$J_{n-1}(x_{n-1}) = \min_{u_{n-1}} \mathbb{E}_\omega \left\{x_{n-1}' Q_{n-1} x_{n-1} + u_{n-1}' R_{n-1} u_{n-1} + J_n(x_n)\right\}$$

where

$$J_n(x_n) = (A_{n-1}x_{n-1} + Bu_{n-1} + \omega)' Q_n (A_{n-1}x_{n-1} + Bu_{n-1} + \omega)$$

8.4.3 Linear Quadratic

Dynamics:

$$x_{k+1} = A_k x_k + B_k u_k + \omega_k$$

Stage cost:

$$x_k' Q_k x_k + u_k' R_k u_k, \ Q_k \succeq 0, R_k \succ 0$$

Cost to go function:

$$J_k(x_k) = \min_{u_k} \mathbb{E}_{\omega_k}[x_k' Q_k x_k + u_k' R_k u_k + J_{k+1}(x_{k+1})]$$

The final cost is

$$J_N(x_N) = x'_N Q_N x_N$$

At $N-1$ stage

$$\begin{aligned}&x'_{N-1}Q_{N-1}x_N + u'_{N-1}R_{N-1}u_{N-1}\\&+\mathbb{E}[(A_{N-1}x_{N-1}+B_{N-1}u_{N-1}+\omega_{N-1})'Q_N(A_{N-1}x_{N-1}+B_{N-1}u_{N-1}+\omega_{N-1})]\end{aligned}$$

Optimization model:

$$\frac{\partial}{\partial u}(u'Hu + 2r'u + c) = 2Hu + 2r \implies Hu = -r \implies u^* = -H^{-1}r$$

For $N-1$ stage, we have

$$\begin{aligned}H &= R_{N-1} + B'_{N-1}Q_N B_{N-1}\\ r' &= x'_{N-1}A'_{N-1}Q_N B_N\\ c &= x'_{N-1}(Q_{N-1} + A'_{N-1}Q_N A_{N-1})x_{N-1} + \mathbb{E}_\omega(\omega'_{N-1}Q_N\omega_{N-1})\end{aligned}$$

Therefore,

$$u^*_{N-1} = -(R_{N-1} + B'_{N-1}Q_N B_{N-1})^{-1}B_N Q_N A_{N-1}x_{N-1},$$

which is linaer in terms of x_{N-1}, which is so called linear controller.

Therefore,

$$J_{N-1}(x_{N-1}) = x'_{N-1}K_{N-1}x_{N-1} + \mathbb{E}(\omega'_{N-1}Q_N\omega_{N-1})$$

The linear controller will be tested during mid-term or final.

$$J_0(x_0) = x'_0 K_0 x_0 + \sum_{k=0}^{N-1}\mathbb{E}_\omega\{\omega'_k K_{k+1}\omega_k\}$$

$$K_{k-1} = f(K_k) \implies K_{k-1} = K_k, \quad \text{for large } k.$$

Riccati Equation for stationary ststem, i.e., $A_k = A, B_k = B, Q_k = Q, R_k = R$.

$$K = A'(K - KB(B'KB + R)^{-1}B'K)A + Q$$

Therefore as time goes long enough, the cost to go function $J_k(x_k)$ will be a constant plus over stages. such a constant K is called the *optimal stationary controller.*

Stability. For $u^* = Lx$, we imply

$$x_{k+1} = Ax_k + Bu_k + \omega_k = (A + BL)x_k + \omega_k$$

Care about $\lim_{k\to\infty}(A + BL)^k = 0$.

Here

$$L = -(B'KB + R)^{-1}B'KA$$

It suffices to solve

$$P_{k+1} = A^2 \left(P_k - \frac{B^2 P_k^2}{B^2 P_k + R} \right) + Q$$

Then consider the case that A_k, B_k are all random matrices, i.e., independent. $Q_k \succeq 0, R_k \succ 0$.

$$L_k = -\left(R_k + \mathbb{E}(B_k' K_{k+1} B_k)\right)^{-1} \mathbb{E}(B_j' K_{k+1} A_k)$$

Note that

$$P_\infty = \frac{\mathbb{E}A^2 RP}{R + EB^2 P} + \frac{\mathbb{E}A^2 \mathbb{E}B^2 - (\mathbb{E}A)^2 (\mathbb{E}B)^2}{R + \mathbb{E}B^2 P}$$

Certainty Equivalence
State: x_k, inventory level
Control: $u_k \geq 0$, number of orders placed
Disturbance: ω_k: damand
Dynamics:

$$x_{k+1} = (x_k + u_k - \omega_k)$$

Stage cost:

- Ordering cost: $c \cdot u_k$
- Maintaining cost: full backlog.$\mathbb{E}_{\omega_k}(r(x_k + u_k - \omega_k))$.

  where $r(z) = hz^+ + pz^-$ is a convex function.

  We imply the maintaining cost is $H(x_k + u_k)$, which is convex.

$$J_k(x_k) = \min_{u_k \geq 0} \{cu_k + H(x_k + u_k) + \mathbb{E}[J_{k+1}(x_k + u_k - \omega_k)]\}$$

Define $Y = x_k + u_k$, and therefore

$$G(Y) = cY + H(Y) + \mathbb{E}[J(Y - \omega_k)]$$

Therefore, we can always set $Y = x_k + u_k = S_k$, where S_k minimizes $G(Y)$.

Acknowledgements

The authors are grateful to Ulrike Fischer, who designed the style files, and Neal Parikh, who laid the groundwork for these style files.

A Journal Codes

The table below shows the journal codes to be used in `\documentclass`.
For Example: `\documentclass[ACC,biber]{nowfnt}`

Journal	<jrnlcode>	Ref. Style
Annals of Corporate Governance	ACG	Author/Year
Annals of Science and Technology Policy	ASTP	Author/Year
FnT Accounting	ACC	Author/Year
FnT Comm. and Information Theory	CIT	Numeric
FnT Databases	DBS	Author/Year
FnT Econometrics	ECO	Author/Year
FnT Electronic Design Automation	EDA	Author/Year
FnT Electric Energy Systems	EES	Author/Year
FnT Entrepreneurship	ENT	Author/Year
FnT Finance	FIN	Author/Year
FnT Human-Computer Interaction	HCI	Author/Year
FnT Information Retrieval	INR	Author/Year
FnT Information Systems	ISY	Author/Year
FnT Machine Learning	MAL	Author/Year
FnT Management	MGT	Author/Year
FnT Marketing	MKT	Author/Year
FnT Networking	NET	Numeric
FnT Optimization	OPT	Numeric
FnT Programming Languages	PGL	Author/Year
FnT Robotics	ROB	Author/Year
FnT Privacy and Security	SEC	Author/Year
FnT Signal Processing	SIG	Numeric
FnT Systems and Control	SYS	Author/Year
FnT Theoretical Computer Science	TCS	Numeric
FnT Technology, Information and OM	TOM	Author/Year
FnT Web Science	WEB	Author/Year

Continues

B Files Produced During Compilation

The files that are created during compilation are listed below. The additional *.tex files are used during the final production process only. See Section 8.3.

```
<YourFilename>-nowbook.tex
<YourFilename>-nowchapter.tex
<YourFilename>-nowebook.tex
<YourFilename>-nowechapter.tex
<YourFilename>-nowsample.tex
<YourFilename>-nowplain.tex
<YourFilename>.aux
<YourFilename>.bbl
<YourFilename>.bcf
<YourFilename>.blg
<YourFilename>.log
<YourFilename>.out
<YourFilename>.pdf
```

<YourFilename>.run.xml
<YourFilename>.synctex.gz
<YourFilename>.tex
<YourFilename>.toc