Lecture 4: Linear Optimization and CVX

Zizhuo Wang

Institute of Data and Decision Analytics (iDDA)
Chinese University of Hong Kong, Shenzhen

Sep 14, 2018

Logistics

Homework 1 posted on Blackboard

- Due on Wednesday, Sep 19th, 12pm. Submit electronically

Recap: Linear Optimization

A *linear optimization* problem, or a *linear program* (LP) is an optimization problem in which the objective function and all constraint functions are linear (in the decision variables).

General form of linear program:

$$
\begin{aligned}
\text{minimize/maximize}_{\mathbf{x}} \quad & \mathbf{c}^T\mathbf{x} \\
\text{subject to} \quad & A_1\mathbf{x} \geq \mathbf{b}_1 \\
& A_2\mathbf{x} \leq \mathbf{b}_2 \\
& A_3\mathbf{x} = \mathbf{b}_3 \\
& x_i \geq 0 \qquad \forall i \in N_1 \\
& x_i \leq 0 \qquad \forall i \in N_2 \\
& x_i \text{ free} \qquad \forall i \in N_3
\end{aligned}
$$

Recap: Standard Form of Linear Optimization

An LP is said to be of *standard form* if it is of the form:

$$\begin{aligned} \text{minimize}_{\mathbf{x}} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^n$, A is an $m \times n$ matrix $(m < n)$ and $\mathbf{b} \in \mathbb{R}^m$.

Recap: Transform to Standard Form

If the objective was maximization

- Use $-\mathbf{c}$ instead of $\mathbf{c}$ and change it to minimization

Eliminating inequality constraints $A\mathbf{x} \le \mathbf{b}$ or $A\mathbf{x} \ge \mathbf{b}$

- Write it as $A\mathbf{x} + \mathbf{s} = \mathbf{b}, \mathbf{s} \ge 0$, or $A\mathbf{x} - \mathbf{s} = \mathbf{b}, \mathbf{s} \ge 0$
- We call $\mathbf{s}$ the slack variables

If one has $x_i \le 0$

- Define $y_i = -x_i$

Eliminating "free" variables x_i (no constraint on x_i)

- Define $x_i = x_i^+ - x_i^-$, with $x_i^+ \ge 0$, $x_i^- \ge 0$

We showed another example of the nurse staffing problem

Air Traffic Control Problem

An air traffic controller needs to control the landing time of n aircrafts

- Flights must land in the order $1, \ldots, n$
- Flight j must land in time interval $[a_j, b_j]$
- The objective is to maximize the minimum *separation time*, which is the interval between two landings

An Optimization Formulation

Decision variable

- Let t_j be the landing time of flight j

Optimization problem:

$$\begin{array}{lccr} \max & \min_{j=1,\ldots,n-1}\{t_{j+1}-t_j\} & & \\ \text{s.t.} & a_j \le t_j \le b_j, & & j=1,\ldots,n \\ & t_j \le t_{j+1}, & & j=1,\ldots,n-1 \end{array}$$

The objective function is not a linear function. We call it a maximin objective.

LP Formulation

Define

$$\Delta = \min_{j=1,...,n-1} \{t_{j+1} - t_j\}$$

Therefore, $t_{j+1} - t_j \geq \Delta$, $\forall j$.

Write an LP:

$$\begin{array}{rll} \text{maximize} & \Delta & \\ \text{subject to} & t_{j+1} - t_j - \Delta \geq 0, & j = 1, ..., n-1 \\ & a_j \leq t_j \leq b_j, & j = 1, ..., n \\ & t_j \leq t_{j+1}, & j = 1, ..., n-1 \end{array}$$

At optimal, Δ must equal the minimal separation.

This is called a maximin problem, next we will talk about more generally about minimax/maximin objective and absolute values.

Minimax Objective

Similar to the air traffic control problem, sometimes we are interested in a minimax objective:

$$\begin{array}{ll} \text{minimize}_{\mathbf{x}} & \max_{i=1,\ldots,n}\{\mathbf{c}_i^T\mathbf{x} + d_i\} \\ \text{subject to} & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array}$$

We can deal it in a similar manner

- Define $y = \max_{i=1,\ldots,n}\{\mathbf{c}_i^T\mathbf{x} + d_i\}$

$$\begin{array}{ll} \text{minimize}_{\mathbf{x},y} & y \\ \text{subject to} & y \geq \mathbf{c}_i^T\mathbf{x} + d_i \quad \forall i \\ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array}$$

Dealing with Absolute Values

Problems with absolute values might be handled as well by LP.

$$\begin{array}{rl} \text{minimize} & \sum_{i=1}^n |x_i| \\ \text{s.t.} & A\mathbf{x} = \mathbf{b} \end{array}$$

This can be equivalently written as

$$\begin{array}{rl} \text{minimize} & \sum_{i=1}^n y_i \\ \text{s.t.} & y_i \geq x_i \\ & y_i \geq -x_i \\ & A\mathbf{x} = \mathbf{b} \end{array}$$

Similar idea can be applied when there are constraints like $|\mathbf{a}^T\mathbf{x} + b| \leq c$.

Absolute Values

Consider a similar problem

$$\begin{array}{rl} \text{maximize} & \sum_{i=1}^n |x_i| \\ \text{s.t.} & A\mathbf{x} = \mathbf{b} \end{array} \tag{1}$$

Can we use the similar idea and transform it into:

$$\begin{array}{rl} \text{maximize} & \sum_{i=1}^n y_i \\ \text{s.t.} & y_i \geq x_i \\ & y_i \geq -x_i \\ & A\mathbf{x} = \mathbf{b} \end{array}$$

Answer: No. There is some intrinsic property of problem (1) that prevents us from formulating it as an LP (non-convexity). We will talk about it later in this course.

Use Software to Solve LP

In this course, we mainly use MATLAB:

- Use a package called CVX
- Read the instruction documents
- `http://cvxr.com/cvx/`

You may also use Python (cvxpy) or Julia (cvx.jl).

Example: Production Planning Problem

$$
\begin{array}{lrrl}
\text{maximize} & x_1 & +2x_2 & \\
\text{subject to} & x_1 & & \leq 100 \\
 & & 2x_2 & \leq 200 \\
 & x_1 & +x_2 & \leq 150 \\
 & x_1, & x_2 & \geq 0
\end{array}
$$

Example: Nurse Scheduling Problem

Let $d = [14, 15, 15, 16, 12, 6, 7]$, solve

$$
\begin{array}{lcccccccccc}
\min & x_1 & +x_2 & +x_3 & +x_4 & +x_5 & +x_6 & +x_7 & & \\
\text{s.t.} & x_1 & & & +x_4 & +x_5 & +x_6 & +x_7 & \geq & d_1 \\
 & x_1 & +x_2 & & & +x_5 & +x_6 & +x_7 & \geq & d_2 \\
 & x_1 & +x_2 & +x_3 & & & +x_6 & +x_7 & \geq & d_3 \\
 & x_1 & +x_2 & +x_3 & +x_4 & & & +x_7 & \geq & d_4 \\
 & x_1 & +x_2 & +x_3 & +x_4 & +x_5 & & & \geq & d_5 \\
 & & x_2 & +x_3 & +x_4 & +x_5 & +x_6 & & \geq & d_6 \\
 & & & x_3 & +x_4 & +x_5 & +x_6 & +x_7 & \geq & d_7 \\
 & x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7 & \geq & 0
\end{array}
$$

Example: Support Vector Machine Problem

$$
\begin{array}{rll}
\text{minimize}_{\mathbf{a},b,\delta,\sigma} & \sum_i \delta_i + \sum_j \sigma_j & \\
\text{subject to} & \mathbf{x}_i^T \mathbf{a} + b + \delta_i \geq 1, & \forall i \\
& \mathbf{y}_j^T \mathbf{a} + b - \sigma_j \leq -1, & \forall j \\
& \delta_i \geq 0, \ \sigma_j \geq 0, & \forall i, j
\end{array}
$$

Example: Shortest Path Problem

Suppose we have a graph $G = (V, E)$, where $V = \{1, ..., n\}$ is the set of nodes and E is the set of edges.

- We denote the source node by 1, the terminal node by n
- We use w_{ij} to denote the distance from i to j. In general, w_{ij} does not necessarily equal w_{ji} (it is a directed graph)
- We assume E contains all pairs of (directed) nodes: If there was no edge for (i, j), we can just set w_{ij} to be extremely large (larger than n times the maximum of the rest of w_{ij})

We want to write a general shortest path solver using LP:

- Input: A matrix $W = \{w_{ij}\}_{i,j=1,...,n}$
- Output: The shortest path from 1 to n and its distance

Example: Shortest Path Problem

Originally, we had the following optimization formulation:

$$\begin{array}{rll} \text{minimize} & \sum_{(i,j)\in E} w_{ij}x_{ij} & \\ \text{subject to} & \sum_j x_{sj} = 1 & \\ & \sum_j x_{jt} = 1 & \\ & \sum_j x_{ij} = \sum_j x_{ji}, & \forall i \neq s, t \\ & x_{ij} \in \{0, 1\}, & \forall (i, j) \in E \end{array}$$

We modify it to the following (now $s = 1$ and $t = n$):

$$\begin{array}{rll} \text{minimize} & \sum_{i,j} w_{ij}x_{ij} & \\ \text{subject to} & \sum_{j\neq 1} x_{1j} - \sum_{j\neq 1} x_{j1} = 1 & \\ & \sum_{j\neq n} x_{jn} - \sum_{j\neq n} x_{nj} = 1 & \\ & \sum_{j\neq i} x_{ij} - \sum_{j\neq i} x_{ji} = 0, & \forall i \neq 1, n \\ & x_{ij} \in \{0, 1\}, & \forall i, j \end{array}$$

Example: Shortest Path Problem

For simplicity of implementation, we further include x_{ii} as decision variables and set w_{ii} to be very large.

Decision variable: A matrix $X = \{x_{ij}\}_{i,j=1,...,n}$

Objective function: $\sum_{i,j} w_{ij}x_{ij}$

- MATLAB representation: sum(sum(W .* X))

Constraints:

- sum($X(1,:)$) - sum($X(:,1)$) $= 1$
- sum($X(:,n)$) - sum($X(:,n)$) $= 1$
- sum($X(i,:)$) - sum($X(:,i)$) $= 0$ for $i \neq 1, n$

Integer constraints:

- We relax $x_{ij} \in \{0,1\}$ to $0 \leq x_{ij} \leq 1$. For this problem, this will not change the solution (will learn it later).

Example: Shortest Path Problem

After having the general solver, we can solve a specific problem: