MAT 3007 – Optimization

Modeling and Linear Optimization

Lecture 02 *June 9th*

Andre Milzarek

iDDA / CUHK-SZ

Repetition

Recap: Introduction to Optimization

Three main components in optimization problems:

- Decision.
- Objective.
- Constraints.

General form of optimization problem:

$$\begin{aligned} \text{minimize}_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \le 0, \quad \forall\ i = 1, \ldots, m, \\ & h_j(x) = 0, \quad \forall\ j = 1, \ldots, p. \end{aligned}$$

Terminologies:

- Feasible set, optimal solutions, optimal value.
- (Strict) local minimizer, (strict) global minimizer.

Recap: Classifications and Modeling

Classes of optimization problems:

- Constrained vs Unconstrained.
- Linear vs Nonlinear.
- Continuous vs Discrete.

Identify and formulate the three components:

- Decision $\rightsquigarrow$ Decision variables.
- Objective $\rightsquigarrow$ Objective function.
- Constraints $\rightsquigarrow$ Constraint functions: equalities/inequalities.

- Today we continue with modeling and formulating optimization problems. We will also have a more detailed look at our first class optimization problems: linear programs.

Modeling: Production Planning

Production Problem: Revisited

A company needs to decide the amount of each product to produce.

	Steel	Iron	Copper	Profit
Alloy 1	1	0	1	\$1
Alloy 2	0	2	1	\$2
Resources	100	200	150	

Decision variables:

- x_1: the amount of alloy 1 to produce.
- x_2: the amount of alloy 2 to produce.

Objective function:

- $f(x) = x_1 + 2x_2$.

Constraints?

Optimization Model

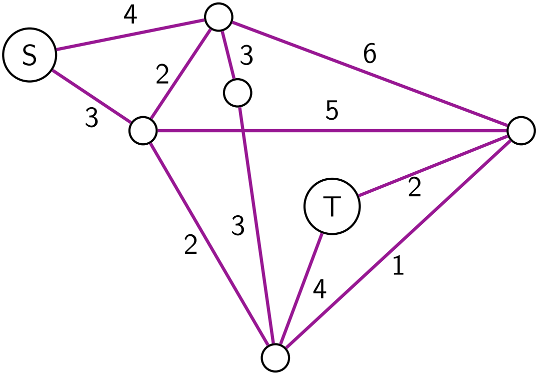
$$\begin{aligned} \text{maximize} \quad & f(x) = x_1 + 2x_2 \\ \text{subject to} \quad & x_1 \leq 100 \\ & 2x_2 \leq 200 \\ & x_1 + x_2 \leq 150 \\ & x_1, x_2 \geq 0. \end{aligned}$$

- This is a linear optimization problem.
- We call the numbers $(1, 2, 100, 150, 200, \text{etc.})$ the coefficients of the optimization problem.

Decision Variables — Objective Function — Constraints

Modeling: Shortest Paths

Shortest Path Problem



What is the shortest path from s to t? How to formulate it as an optimization problem?

- This is called the shortest path problem.

Some notations: Set of nodes: V. Set of edges: $E \subset V \times V$. The distance between node i and node j is w_{ij}.

For each edge $(i,j) \in E$, define

$$x_{ij} = \begin{cases} 1 & \text{if we use edge } (i,j), \\ 0 & \text{otherwise.} \end{cases}$$

An optimization model:

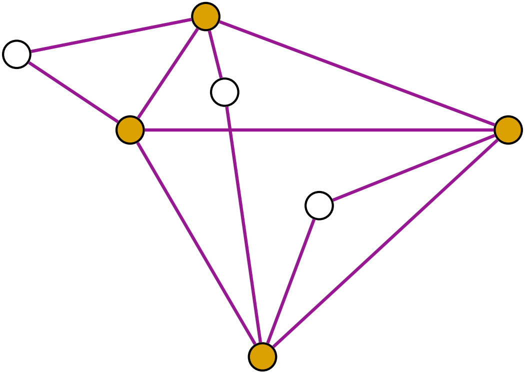
$$\begin{array}{rl} \text{minimize} & \sum_{(i,j)\in E} w_{ij} x_{ij} \\ \text{subject to} & x_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in E \\ & \sum_j x_{sj} = 1 \\ & \sum_j x_{jt} = 1 \\ & \sum_j x_{ij} = \sum_j x_{ji}, \quad \forall i \neq s,t. \end{array}$$

What category does this optimization problem belong to?

- Constrained, linear, integer.
- In fact, it can be transformed to a linear program.

Modeling: Vertex Cover Problem

Task: Given a graph with nodes V and edges E, find the smallest set of vertices that touch every edge of the graph.



Define the following decision variables:

$$x_i = \begin{cases} 1 & \text{if we choose vertex } i, \\ 0 & \text{otherwise.} \end{cases}$$

An optimization model can be written as:

$$\begin{array}{rll} \text{minimize} & \sum_i x_i & \\ \text{subject to} & x_i + x_j \geq 1 & \forall\ (i,j) \in E \\ & x_i \in \{0,1\} & \forall\ i \in V. \end{array}$$

- This is an integer (linear) optimization problem.
- Many graph/network problems can be modeled as optimization problems.

Modeling: Support Vector Machines

General Setup:

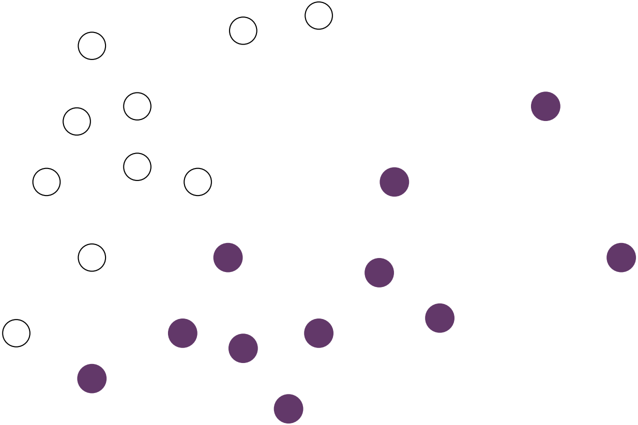
- Given: m objects represented by vectors $x_1, x_2, \ldots, x_m \in \mathbb{R}^n$ with labels $y_i \in \{-1, 1\}$.
- The two labels ± 1 indicate that the data can be separated into two classes $A \equiv +1$ and $B \equiv -1$.
- Idea: Learn a function $\ell : \mathbb{R}^n \to \{-1, 1\}$ based on the training sample $(x_1, y_1), \ldots, (x_m, y_m)$.

$\rightsquigarrow$ Predict the label of a new object z via $\ell(z)$.

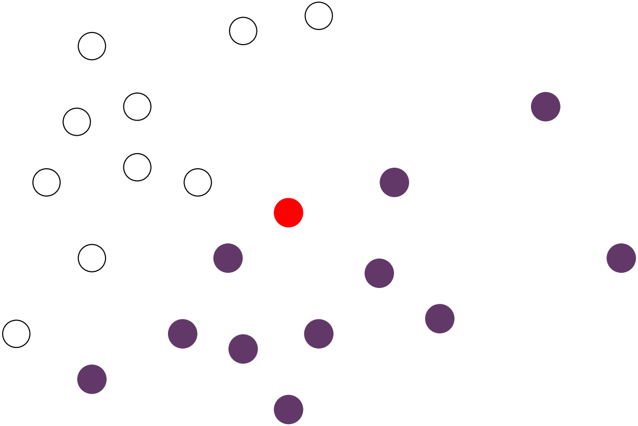
- Example: Blind taste – Does the taste determine the color?

$$x_i \equiv \begin{bmatrix} \text{juicy/fresh} \\ \text{body} \\ \text{acidity} \\ \vdots \end{bmatrix}, \quad y_i = \begin{cases} +1 & \text{white wine,} \\ -1 & \text{red wine.} \end{cases}$$

Support Vector Machines: Illustration

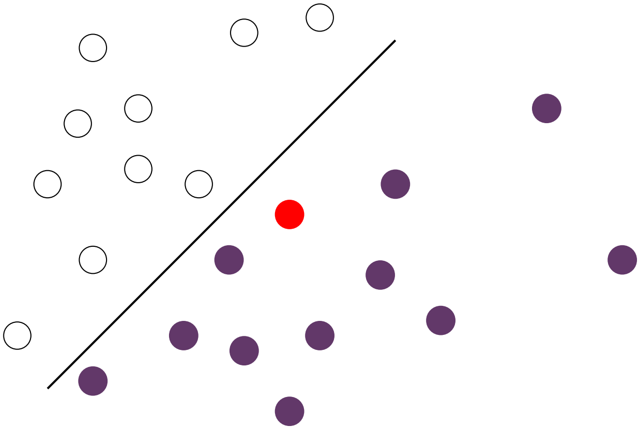


- Consider two labeled sets of points (white and purple).



- Consider two labeled sets of points (white and purple).

⇝ Question: Can we predict the label of a newly added point?



- Separate the two data sets with a hyperplane!

Linear Classification

Decision: Find a hyperplane $\ell(x) := w^\top x + b$ defined by (w, b) separating the datapoints such that:

$$y_i = \begin{cases} +1 & \text{if } \ell(x_i) > 0, \\ -1 & \text{if } \ell(x_i) \leq 0. \end{cases}$$

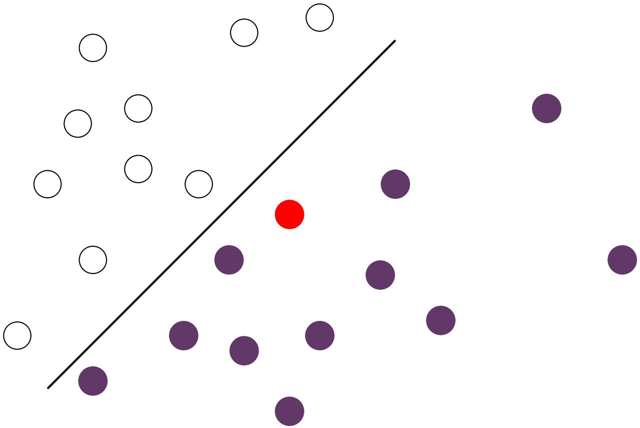
This is equivalent to choosing (w, b) such that:

$$y_i = \begin{cases} +1 & \text{if } \ell(x_i) \geq +1, \\ -1 & \text{if } \ell(x_i) \leq -1, \end{cases} \quad \forall\ i = 1, ..., m.$$

The associated optimization problem is given by:

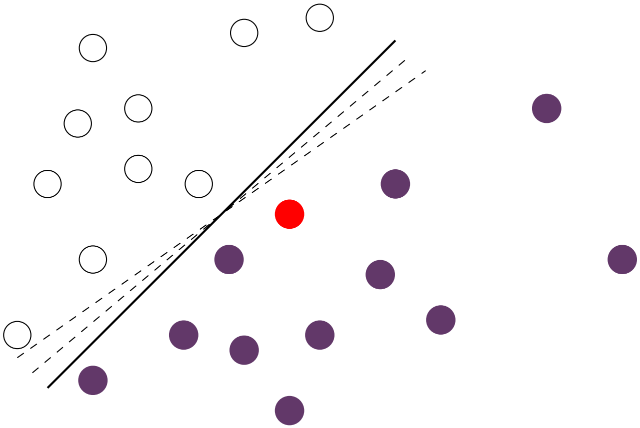
$$\text{minimize}_{w,b}\ 0 \quad \text{s.t.} \quad y_i(x_i^\top w + b) \geq 1, \quad \forall\ i.$$

- This problem is a feasibility problem: feasibility problems are a special kind of optimization problem.
- SVMs are used in pattern recognition, machine learning, etc.



- Is this a good optimization problem or formulation?

Support Vector Machines: Illustration



- Is this a good optimization problem or formulation?
- Problem: The separating hyperplane might not be unique!

- Select the "best" hyperplane to separate the two groups.
- Distance between the hyperplanes $\{x : w^\top x + b = 1\}$ and $\{x : w^\top x + b = -1\}$ is $2/\|w\|$ (why?).

Maximize the possible margin (distance between the datasets):

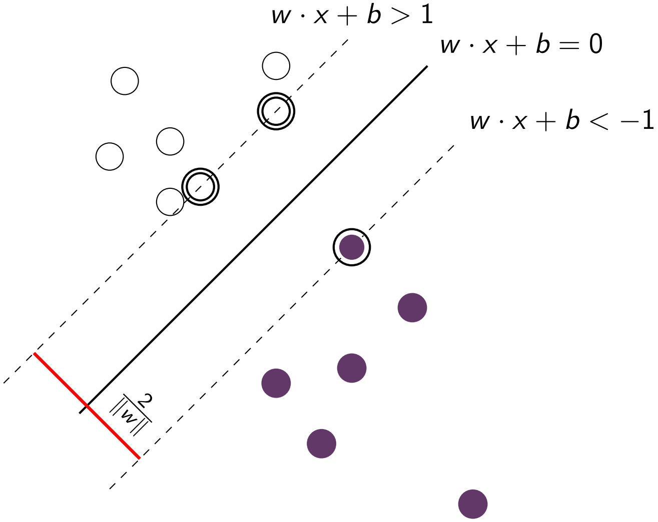
$$\text{maximize}_{w,b} \quad \frac{2}{\|w\|} \quad \text{s.t.} \quad y_i(x_i^\top w + b) \geq 1, \quad \forall\ i.$$

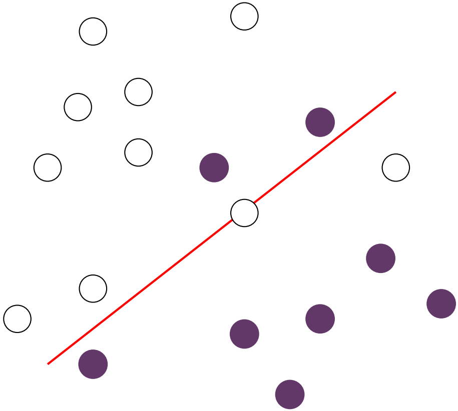
Compact and equivalent formulation:

$$\text{minimize}_{w,b} \quad \frac{1}{2}\|w\|^2 \quad \text{s.t.} \quad y_i(x_i^\top w + b) \geq 1, \quad \forall\ i.$$

- We prefer to use $\|w\|^2$ instead of $\|w\|$ because $w \mapsto \|w\|$ is not differentiable at 0. ($\rightsquigarrow$ Later!).
- Nonlinear (quadratic objective), constrained, continuous.

Support Vector Machines: Illustration





- Question: What to do if the training set can not be perfectly separated by a hyperplane?

Strategy and the Full SVM Problem:

$\rightsquigarrow$ Try to minimize the total or misclassification error:

$$\sum\nolimits_{i=1}^{m} \max\{0, 1 - y_i \ell(x_i)\} \quad \text{(Hinge-Loss Function)}.$$

- SVM combines large margin and small misclassification cond.:

$$\min_{w,b} \quad \frac{\lambda}{2}\|w\|^2 + \sum\nolimits_{i=1}^{m} \max\{0, 1 - y_i(x_i^\top w + b)\}, \quad \lambda > 0.$$

- λ is chosen to balance margin and misclassification. (Typically: $\lambda = \frac{1}{m} \rightsquigarrow$ fine-tuning ...).
- This is an unconstrained, nonlinear, continuous problem.
- We now show how to equivalently rewrite it as a linear optimization problem in the case $\lambda = 0$.

Define $t_i = \max\{0, 1 - y_i(x_i^\top w + b)\} =: (1 - y_i(x_i^\top w + b))^+$.

We can first rewrite the SVM problem as follows:

$$\begin{array}{rl} \text{minimize}_{w,b,t} & \sum_i t_i \\ \text{subject to} & t_i = (1 - y_i(x_i^\top w + b))^+, \quad \forall\, i. \end{array}$$

We claim that we can relax "$=$" to "$\geq$" (why?):

$$\begin{array}{rl} \text{minimize}_{w,b,t} & \sum_i t_i \\ \text{subject to} & t_i \geq (1 - y_i(x_i^\top w + b))^+, \quad \forall\, i. \end{array}$$

Furthermore, $t_i \geq (1 - y_i(x_i^\top w + b))^+$ is equivalent to:

$$t_i \geq 1 - y_i(x_i^\top w + b), \qquad t_i \geq 0.$$

Therefore, the optimization problem can be reformulated as:

$$\begin{array}{rcl} \text{minimize}_{w,b,t} & \sum_i t_i & \\ \text{subject to} & y_i(x_i^\top w + b) + t_i \geq 1, & \forall\ i \\ & t_i \geq 0 & \forall\ i. \end{array}$$

- This is a linear optimization problem with decision variables $w \in \mathbb{R}^n$, $b \in \mathbb{R}$, and $t \in \mathbb{R}^n$.

Linear Optimization

Definition: Linear Program

Linear Problems

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

Example (Production Planning Problem):

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

- Linear optimization is a subclass of continuous optimization.

A linear optimization problem can be generally written as:

$$\begin{array}{rll} \text{minimize/maximize}_{x\in\mathbb{R}^n} & c^\top x & \\ \text{subject to} & a_i^\top x \geq b_i & \forall\ i \in M_1 \\ & a_i^\top x \leq d_i & \forall\ i \in M_2 \\ & a_i^\top x = e_i & \forall\ i \in M_3 \\ & x_i \geq 0 & \forall\ i \in N_1 \\ & x_i \leq 0 & \forall\ i \in N_2 \\ & x_i \text{ free} & \forall\ i \in N_3 \end{array}$$

where M_1, M_2, M_3 are subsets of $\{1, ..., m\}$, N_1, N_2, N_3 are subsets of $\{1, ..., n\}$.

We can write LPs in a more compact way:

$$\begin{aligned} \text{minimize/maximize}_{x \in \mathbb{R}^n} \quad & c^\top x \\ \text{subject to} \quad & A_1 x \geq b \\ & A_2 x \leq d \\ & A_3 x = e \\ & x_i \geq 0 \quad \forall\, i \in N_1 \\ & x_i \leq 0 \quad \forall\, i \in N_2 \\ & x_i \text{ free} \quad \forall\, i \in N_3 \end{aligned}$$

- Here, A_1, A_2, and A_3 are matrices (with dimensions $m_1 \times n$, $m_2 \times n$ and $m_3 \times n$).
- b, d, and e are vectors (with dim. $m_1 \times 1$, $m_2 \times 1$ and $m_3 \times 1$).
- The variable x is an n dimensional column vector.

Standard Form of LPs

In order to study LPs more systematically, we want to have a standard (and even more compact) form for LPs.

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

$$\begin{aligned} \text{minimize}_{x \in \mathbb{R}^n} \quad & c^\top x \\ \text{subject to} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{1}$$

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

In fact, any LP can be written in the standard form, using "tricks".

Remark:

- The definition of "standard form" may differ from book to book. We use (1) as the standard form in this course (cons. with Bertsimas and Tsitsiklis's book).

If the objective was maximization:

- Use $-c$ instead of c and change it to minimization.

Eliminating inequality constraints $Ax \leq b$ or $Ax \geq b$:

- Write it as $Ax + s = b, s \geq 0$ or $Ax - s = b, s \geq 0$.
- We call s the slack variables.

If one has $x_i \leq 0$:

- Define $y_i = -x_i$.

Eliminating "free" variables x_i (no constraints on x_i):

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

Example: Production Planning

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

Standard form

$$\begin{array}{rrrrrrl}
\text{minimize} & -x_1 & -2x_2 & & & & \\
\text{subject to} & x_1 & & +s_1 & & & = 100 \\
& & 2x_2 & & +s_2 & & = 200 \\
& x_1 & +x_2 & & & +s_3 & = 150 \\
& x_1, & x_2, & s_1, & s_2, & s_3 & \geq 0
\end{array}$$

Example: Support Vector Machines

$$\begin{array}{rll} \text{minimize}_{w,b,t} & \sum_i t_i & \\ \text{subject to} & y_i(x_i^\top w + b) + t_i \geq 1, & \forall\ i \\ & t_i \geq 0 & \forall\ i. \end{array}$$

- Define $w = w^+ - w^-$, $b = b^+ - b^-$, with w^+, w^-, b^+,$b^- \geq 0$.
- Add slack variables to eliminate inequality constraints.

Standard Form for SVMs

$$\begin{aligned}
\min_{w^+,w^-,b^+,b^-,t,s} \quad & \textstyle\sum_i t_i \\
\text{subject to} \quad & y_i(x_i^\top w^+ - x_i^\top w^- + b^+ - b^-) + t_i - s_i = 1 \quad \forall i \\
& w^+, w^-, b^+, b^- \geq 0 \\
& t_i, s_i \geq 0 \quad \forall\ i.
\end{aligned}$$

- Standard form is mainly used for analysis purposes. We do not need to write a problem in standard form unless necessary.
- Usually we just represent the problem in a way that makes it easy to understand.

- Transforming an LP into the standard form is an important skill. It is helpful for analyzing LP problems as well as when using software to solve it.

Questions?