Course #1

Operations Research (1)
└ Models and Applications

Week 1
Course Overview

Motivation

- resource allocation
- "management science" synonym
- Operations Research is the methodology to allocate the available resources to the various activities in a way that is most effective for the organization as a whole

★ how to conduct & coordinate the operations within an org

Business Analytics

Business Analytics → descriptive analytics (what happened?) historical

Business Analytics → Predictive analytics (what will happen?) prediction, ML

Business Analytics → Prescriptive analytics (what should we do? decision)

this is where optimization lives!

STEPS
1. Data Analysis: collect data, understand the problem
2. OR: Allocate resources to solve the problem

example: retail store

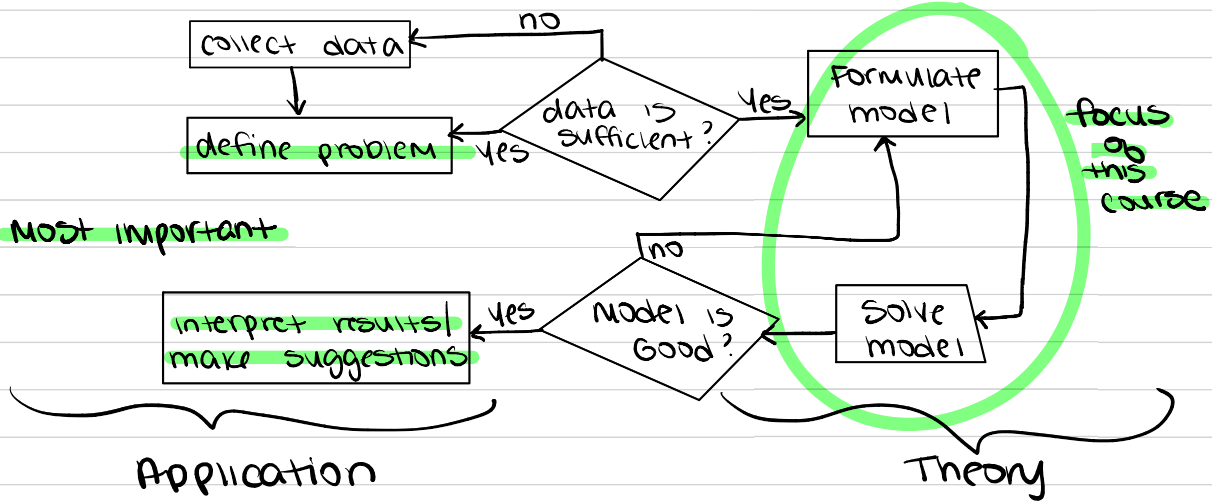
? How to set inventory levels of multiple products to maximize total expected profit?

Data Analysis → Predict expected amount of demand

OR → find inventory level to max $

consider demand substitution → if there is no Coke, what is the prob that the customer will buy a Pepsi instead? items are NOT indep.

Process of collecting an OR study



* sometimes the problem is too difficult to be solved with OR!

Mathematical Programming

- let's return to the knapsack problem:

Item	weight	value
compass	0.5	6
hatchet	1.5	5
matches	0.4	4
tarpaulin	1.0	4
telescope	1.1	3
cylinder	1.6	4
rilakkuma	0.8	1

1. weight limit - 5kg
2. item cannot be split
3. aim to max total value

* Heuristic Approach - calculate the value / weight ratio
↳ value = 22
__But__ the optimal sol'n is value = 23

let's formulate the model

decision variables

$$x_i = \begin{cases} 1 & \text{if item if chosen} \\ 0 & \text{otherwise} \end{cases}$$

abbreviated math notation

(objective function)

$$\max \ \underbrace{6x_1 + 5x_2}_{\text{value of each}} + 4x_3 + \dots + 4x_6 + x_7$$

$$\max \sum_{i=1}^{n} v_i x_i$$

value

decision variable

(constraints)

s.t.

$$0.5x_1 + 1.5x_2 + \dots + 0.8x_7 \le 5$$

weights less or = to max

integer program

$$x_i \in \{0,1\} \quad \forall i = 1, \dots, 7$$

x must be 0 or 1

s.t

$$\sum_{i=1}^{n} w_i x_i \le B$$

weight

$$x_i \in \{0,1\} \ \forall_{i=1,\dots,n}$$

"for each item"

Porfolio optimization example

w_i - max possible investment

v_i - expected return of stock i

n - # assets

B - total Budget

$$\max \sum_{i=1}^{n} v_i x_i$$

* various types of programs:

(continous) → linear, non-linear,

integer ← (discrete)

$$\text{s.t.} \ \sum_{i=1}^{n} w_i v_i \le B$$

linear program

$$0 \le x_i \le 1 \quad \forall i = 1, \dots, n$$

x_i is now the % of the total buget to put in that stock

only difference from knapsack, this example is NOT all or nothing

History

- George Dantzig - known as the father of OR
  - he found that many problems are linear optimization problems
  - no one was able to systematically solve large-scale optimization problems
  - he invented the simplex method → the 1st effective sol'n for linear programming (still used!)

Week 2
Linear Programming (LPs)

- examples: product mix, production, inventory, personnel scheduling, } resource allocation

elements of a mathematical program

- Basic elements →
  - objective function min/max
  - decision variables column vectors → $(x_1, x_2, \dots x_n) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$
    $\forall i = 1, \dots, n$ $\quad x_j \in \mathbb{R}$ ← real #'s
  - constraints $\forall j = 1, \dots, m$
- Transformations
  1. $\max f(x) \longleftrightarrow \min -f(x)$
  2. $g_i(x) \geq b_i \longleftrightarrow -g_i(x) \leq -b_i$
  3. $g_i(x) = b_i \longleftrightarrow g_i(x) \leq b_i$ AND $g_i(x) \geq b_i$

example

$$\max \; x_1 - x_2 \qquad \min \; -x_1 + x_2$$

$$\text{s.t.} \; -2x_1 + x_2 \geq -3 \longleftrightarrow \text{s.t.} \; 2x_1 - x_2 \leq +3$$

$$x_1 + 4x_2 = 5 \qquad \begin{cases} x_1 + 4x_2 \leq 5 \\ -x_1 - 4x_2 \leq -5 \end{cases}$$

- Two types of constraints
  1. sign constraints – $x_i \geq 0$ or $x_i \leq 0$ (⊕ non-negative, ⊖ non-positive)
  2. functional constraints – all others

an unrestricted in sign (urs) or free variable can be ⊕ or ⊖

- feasible solution vs infeasible solution
  - satisfies all constraints (feasible)
  - violates at least one constraint (infeasible)

- feasible region (set) of feasible solutions
  - goal is to find the optimal sol'n within this region
    - there may be multiple, or none!

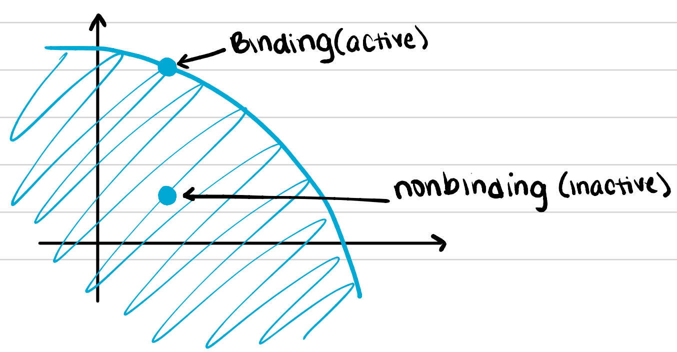
- Binding constraints (also called active/inactive)

DEFINITION

Let $g(\cdot) \le b$ be an inequality constraint and $\bar{x}$ will be a solution.

$g(\cdot) \le b$ is binding at $\bar{x}$ if $g(\bar{x}) = b$

- an equality constraint is always binding at any feasible solution (since it MUST be =)
- examples:
  1. $x_1 + x_2 \le 10$ BINDING at $(x_1, x_2) = (4, 6)$
  2. $2x_1 + x_2 \ge 6$ NONBINDING at $(x_1, x_2) = (2, 8)$
  3. $x_1 + 3x_2 = 9$ BINDING at $(x_1, x_2) = (6, 1)$



- Strict vs weak inequalities

↳ the 2 sides cannot be equal
↳ $x_1 + x_2 > 5$

↳ two sides can be equal
↳ $x_1 + x_2 \geq 5$

* In OR (in practical applications) inequalities are ALWAYS WEAK! - strict inequalities may make optimal sol'n unattainable

example: want to spend 500 →
→ cannot spend > 500, but you CAN spend 500

Linear Programs

- a MP is an LP if all functions are linear functions

$$a_1x_1 + \dots + a_nx_n = \sum_{j=1}^{n} a_jx_j = a^Tx \qquad [a_1, \dots a_n]\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

general form for LPs

objective coefficients

$$\min \sum_{j=1}^{n} c_jx_j$$

constraint coefficients

$$\text{s.t.} \sum_{j=1}^{n} A_{ij}x_j \leq b_i \quad \forall i = 1, \dots, m$$

right-hand-side values (RHS)

As vectors

$$\min \ c^Tx$$
$$\text{s.t.} \ a_i^Tx \leq b_i \ \forall i = 1, \dots, m$$

→ $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$, $c \in \mathbb{R}^n$
→ $x \in \mathbb{R}^n$

As matrices

$$\min \ c^Tx$$
$$\text{s.t.} \ Ax \leq b$$

columns rows

→ $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^{m}$

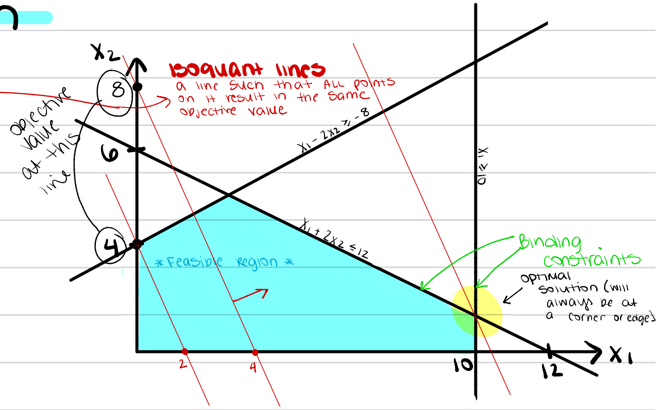
$$\begin{bmatrix} A_{11} & A_{12} & \dots & A_{mn} \\ A_{21} & A_{22} & \dots & A_{mn} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

The Graphical Approach

example

$$\max \quad 2x_1 + x_2$$

$$\begin{aligned} \text{s.t.} \quad x_1 \quad\quad &\le 10 \\ x_1 + 2x_2 &\le 12 \\ x_1 - 2x_2 &\ge -8 \\ x_1 \quad\quad &\ge 0 \\ x_2 &\ge 0 \end{aligned}$$



Step 6: Set the binding constraints to equalities — solve the linear system to find optimal sol'n

$$x_1 = 10$$

$$x_1 + 2x_2 = 12 \rightarrow 10 + 2x_2 = 12 \rightarrow 2x_2 = 2 \rightarrow x_2 = 1$$

optimal sol'n $(x_1^*, x_2^*) = (10, 1)$

↳ objective value $= 2(10) + (1) = 21$

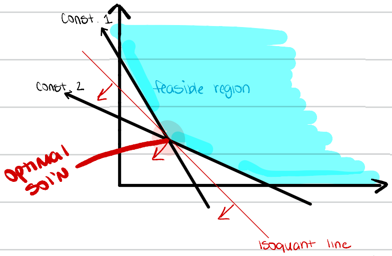
* can use gaussian elimination to solve *

$$\left[\begin{array}{cc|c} 1 & 0 & 10 \\ 1 & 2 & 12 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 1 & 0 & 10 \\ 0 & 2 & 2 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 1 & 0 & 10 \\ 0 & 1 & 1 \end{array}\right]$$

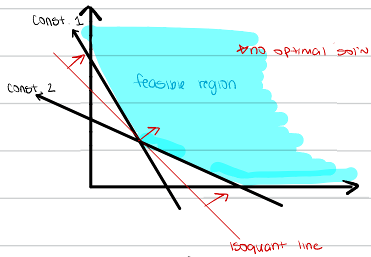
3 Types of LP's

1. infeasible (feasible region is empty)
2. unbounded
   ↳ for any feasible sol'n, we can find another that is better
   ↳ unbounded feasible region does NOT imply unbounded LP

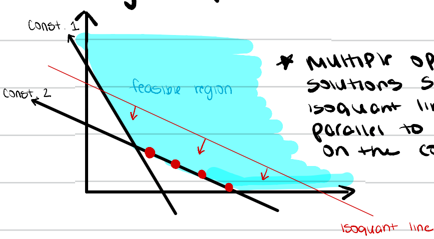
unbounded F.R.



unbounded F.R. & LP



3. finitely optimal (has at least 1 optimal sol'n)



* multiple optimal solutions since isoquant line is parallel to one on the constraints

* Summary *

Feasible — Infeasible

Feasible → Finitely optimal, unbounded

Finitely optimal → one optimal sol'n, multiple optimal sol'ns

↳ we don't really care if there are multiple

Modeling examples (LP Formulation)

#1. Product Mix - max total sales w/ available resources

Problem description →

desks	tables
• 3 units wood	• 5 units wood
• 1 hour labor	• 2 hours labor
• 50 min machine time	• 20 min machine time

each day we have

- 200 workers, 8 hours each
- 50 machines, 16 hours each
- supply of 3600 units of wood
- desks sell for \$700/unit
- tables sell for \$900/unit

LP Formulation

d = # desks produced each day
t = # tables produced each day

always clearly define your decision variables first!

$$
\begin{aligned}
\max\quad & 700d + 900t && \text{sales}\\
\text{s.t.}\quad & 3d + 5t \le 3600 && \text{wood}\\
& 1d + 2t \le 1600 && \text{labor}\\
& 50d + 20t \le 48000 && \text{machine}\\
& d \ge 0\\
& t \ge 0
\end{aligned}
$$

→ add comments!

↗ If we forced d and t to be whole → integer program NOT linear program!

optimal solution is 884.21 desks & 189.47 tables
↳ can choose to round or not dep on use case

"All models are wrong, but some are useful"

#2. Product & store inventory

- When we make decisions we need to consider what will happen in the future! → Multi period problems
- Products produced today may be stored

production decision must be jointly considered with inventory

- maybe daily capacity is not enough
- maybe production is cheaper today
- maybe the price is higher in the future

Problem description →

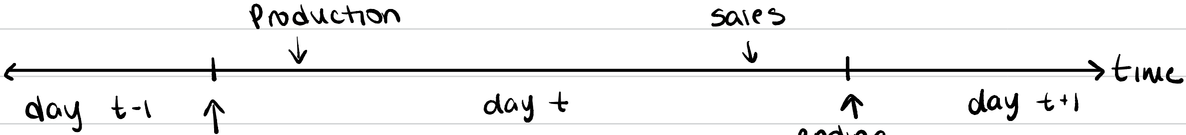
- Product Demand

day	1	2	3	4
demand	100	150	200	170

- Production costs

day	1	2	3	4
$	9	12	10	12

- prices are fixed, so max profit = min cost
- can store a product & sell it later
- inventory cost of $1 / unit / day



beginning inventory + production − sales = ending inventory

used to calculate inventory costs

LP Formulation

let x_t = production quantity of day t

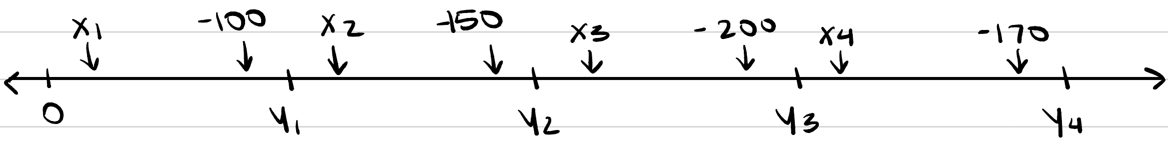
y_t = ending inventory of day t

$t = [1, 2, 3, 4]$

\# units produced each day   inventory cost per item per day

$$\text{Min} \quad 9x_1 + 12x_2 + 10x_3 + 12x_4 + 1y_1 + 1y_2 + 1y_3 + 1y_4$$

production cost each day   \# units at end of day



$$\text{Min} \quad 9x_1 + 12x_2 + 10x_3 + 12x_4 + 1y_1 + 1y_2 + 1y_3 + 1y_4$$

s.t.

Day 1 $\quad x_1 - 100 = y_1$

Day 2 $\quad y_1 + x_2 - 150 = y_2$

Day 3 $\quad y_2 + x_3 - 200 = y_3$

Day 4 $\quad y_3 + x_4 - 170 = y_4$

Inventory balancing

$x_1 \geq 100$

$y_1 + x_2 \geq 150$

$y_2 + x_3 \geq 200$

$y_3 + x_4 \geq 170$

Demand Fulfillment

$x_t, y_t \geq 0 \quad \forall t = 1, \ldots, 4$ non-negativity

Simplify it!

$$\text{Min} \quad 9x_1 + 12x_2 + 10x_3 + 12x_4 + 1y_1 + 1y_2 + 1y_3 + 1y_4$$

$$\begin{aligned}
\text{s.t.} \quad & x_1 - 100 = y_1 \\
& y_1 + x_2 - 150 = y_2 \\
& y_2 + x_3 - 200 = y_3 \\
& y_3 + x_4 - 170 = y_4
\end{aligned} \Bigg] \text{ Inventory balancing}$$

$$\cancel{x_1 \geq 100}$$
$$\cancel{y_1 + x_2 \geq 150}$$
$$\cancel{y_2 + x_3 \geq 200}$$
$$\cancel{y_3 + x_4 \geq 170}$$

$$x_t, y_t \geq 0 \quad \forall t = 1, \ldots, 4 \Big] \text{ non-negativity}$$

Inventory balancing & nonnegativity imply → demand fulfillment

ex) $x_1 - 100 = y_1$ and $y_1 \geq 0$ mean $x_1 \geq 100$

#3. Personel Scheduling

Problem description →

- each employee must work 5 consectutive days and then rest for 2 consecutive days
- # employees req each day

Mon	Tues	Wed	Thurs	Fri	Sat	Sun
110	80	150	30	70	160	120

* This means there are 7 shift schedules : Mon - Fri, Tues - Sat, ..., Sun - Thurs
- we want to min # of employees hired
* easy to find a feasible sol'n

LP Formulation

- let monday be day 1, tuesday be day 2...
- let x_i be the # of employees who start their 5 day shift on day i (assigned to shift i)

$$\min \quad x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7$$

shift that starts tuesday does **NOT** work on monday

Mon	$x_1 + \qquad\quad x_4 + x_5 + x_6 + x_7$	≥ 110
Tues	$x_1 + x_2 \qquad\quad + x_5 + x_6 + x_7$	≥ 80
Wed	$x_1 + x_2 + x_3 \qquad + x_6 + x_7$	≥ 150
Thurs	$x_1 + x_2 + x_3 + x_4 \qquad + x_7$	≥ 30
Fri	$x_1 + x_2 + x_3 + x_4 + x_5$	≥ 70
Sat	$x_2 + x_3 + x_4 + x_5 + x_6$	≥ 160
Sun	$+ x_3 + x_4 + x_5 + x_6 + x_7$	≥ 120

$$x_i \geq 0, \quad \forall i = 1, \ldots, 7$$

↳ do **NOT** say they need to be integers. integer programming is more complicated

Compact Linear Formulations

1. Indices (i, j, k)
2. Summation (Σ)
3. For all ($\forall$)

ex) production & inventory problem

$$\min \quad 9x_1 + 12x_2 + 10x_3 + 12x_4 + y_1 + y_2 + y_3 + y_4$$

↓

$$\min \sum_{t=1}^{4} (c_t x_t + y_t)$$

constraints

$$x_1 - 100 = y_1$$
$$y_1 + x_2 - 150 = y_2$$
$$y_2 + x_3 - 200 = y_3$$
$$y_3 + x_4 - 170 = y_4$$

denote Demand on day t as D_t, $t = 1, \ldots, 4$

y_t = ending inventory of day t, $t = 0, \ldots, 4$

$$\longrightarrow$$

$$y_{t-1} + x_t - D_t = y_t$$
$$y_0 = 0$$
$$\underbrace{\forall t = 1, \ldots, 4,}_{\text{Dont forget the for all statements!}} \; x_t, y_t \geq 0$$

* common convention is
- lowercase for variables (x_t) – endogenous → unknown, to be solved for
- uppercase for parameters (C_t) - exogenous → given in description

ex) Product Mix

- let n be # of products
- let m be # of resources
- let i be the index for products
- let j be the index for resources
- sales price of product j as P_j
- supply limit of resource i as R_i
- limit of resource req. for one unit of product j as A_{ij} where $i = 1, \ldots, M$, $j = 1, \ldots, M$
- let x_j be the production quantity for product j, $j = 1, \ldots, n$

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

$$\text{s.t.} \sum_{j=1}^{n} A_{ij} x_j \leq R_i \quad \forall i = 1, \ldots, M$$

$$x_j \geq 0 \quad \forall j = 1, \ldots, n$$

$$J = \{1, \ldots, n\}$$
$$I = \{1, \ldots, m\}$$

$$\max \sum_{j \in J} P_j x_j$$

$$\text{s.t.} \sum_{j \in J} A_{ij} x_j \leq R_i \quad \forall i \in I$$

$$x_j \geq 0 \quad \forall i \in J$$

can use sets instead

Problem V.S Instance

Abstract description of a task | question

$$\max \sum_{j \in J} P_j x_j$$

$$\text{s.t.} \sum_{j \in J} A_{ij} x_j \leq R_i \quad \forall i \in I$$

$$x_j \geq 0 \quad \forall j \in J$$

another reason to write like this! Can apply to multiple <u>INSTANCES</u>!

Concrete specification of a problem

$$\max \; 700x_1 + 900x_2$$

$$\text{s.t.} \quad 3x_1 + 5x_2 \leq 3600$$

$$x_1 + 2x_2 \leq 1600$$

$$50x_1 + 20x_2 \leq 48000$$

$$x_1 \geq 0, \; x_2 \geq 0$$

Week 3
Integer Programming

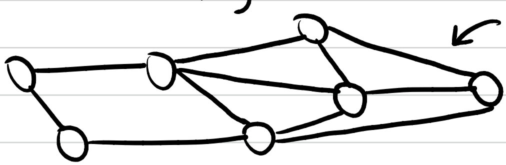
Introduction

- can restrict to discrete values (LP is cont.)
- examples|

Selection/assignment {
1. Set up cost
2. Facility location
3. Machine scheduling
4. vehicle routing

- use when variables can <u>only</u> take integer values
- An IP is typically a linear IP (if a constraint or objective is nonlinear, it is a nonlinear IP, NLIP)
- use integer programming if value is very small → rounding is innaccurate!

- Vehicle routing



← construct network w/ nodes & edges
- binary variable for each edge

IP Formulation

- integer programming allows us to implement some selection rules
- At least | At most some items

ex) must have at least 1 item of items 2, 3, & 4

$$x_2 + x_3 + x_4 \geq 1$$

- or condition
  - Select x_2 OR $x_3 \rightarrow x_2 + x_3 \geq 1$
  - select (x_2) or $(x_3$ AND $x_4) \rightarrow 2x_2 + x_3 + x_4 \geq 2$
- If - else
  - If x_2 is selected, select $x_3 \rightarrow x_2 \leq x_3$
  - If x_1 is selected, DO NOT select x_3 & $x_4 \rightarrow$
    $2(1-x_1) \geq x_3 + x_4$

using a similar technique we may flexibly add constraints

- satisfy one of the two constraints

$g_1(x) \leq b_1$ and $g_2(x) \leq b_2$

define binary variable z:

$$z = \begin{cases} 0 & \text{if } g_1(x) \leq b_1 \\ 1 & \text{if } g_2(x) \leq b_2 \end{cases}$$

* does not mean you cant have also satisfied

very Big #

with M_i being an upper bound of each LHS, the following 2 constraints are what we need

$$g_1(x) - b_1 \leq M_1 z$$
$$g_2(x) - b_2 \leq M_2(1-z)$$

ex) if $z = 0$

$g_1(x) - b_1 \leq 0$ ← says that this must be satisfied

$g_2(x) - b_2 \leq M_2$ ← but we don't care if this is

- satisfy 2 of 3 constraints

$g_i(x) \leq b_i \quad , i = 1,2,3$

$$z_i \begin{cases} 1 & \text{if } g_i(x) \leq b_i \text{ is satisfied} \\ 0 & \text{if } g_i(x) \leq b_i \text{ may be satisfied} \end{cases}$$

very large #

with M_i being an upper Bound of each LHS, we need:

$g_i(x) - b_i \leq M_i(1-z_i) \quad \forall i = 1,2,3$

at least 2 must be satisfied → $z_1 + z_2 + z_3 \geq 2$

- Fixed-charge constraints

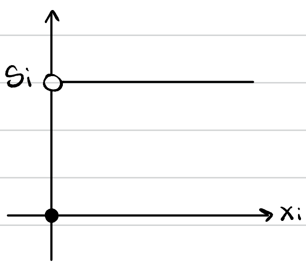
Consider the following →

- n factories, 1 market, 1 product
- K_i is capacity at factory i
- C_i is unit production at factory i
- D is demand

ex) shipping cost

* GOAL: satisfy the demand w/ minimum cost

- setup cost at factory i: S_i — one needs to pay the setup cost as long as any POSITIVE amount of product is produced



Need to link x_i to y_i!

If $x_i > 0$, $y_i = 1$ o/w $y_i = 0$

$x_i \le K_i y_i \quad \forall i = 1, \dots, n$

* this will push y_i to $= 0$ when $x_i = 0$

Fixed-charge constraint

- $X \le My$ (format)
- when x is 0, y can be any #, when x is ⊕ y = 1.
- y typically is binary
- M is upper bound of X
  ↳ sometimes this is given (K_i), sometimes need to find what works

Basic Formulation

Decision variables

x_i = prod. quantity at factory i

$$y_i = \begin{cases} 1 & \text{if } x_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

Objective Function

$$\min \underbrace{\sum_{i=1}^{n} C_i x_i}_{\text{production cost}} + \underbrace{\sum_{i=1}^{n} S_i y_i}_{\text{set-up cost}}$$

Constraints

(1) $x_i \le K_i y_i \quad \forall i = 1, \dots, n$ (capacity)

(2) $\sum_{i=1}^{n} x_i \ge D$ (demand)

(3) $x_i \ge 0,\ y_i \in \{0, 1\} \quad \forall i = 1, \dots, n$
(binary & nonneg)

Facility Location IP

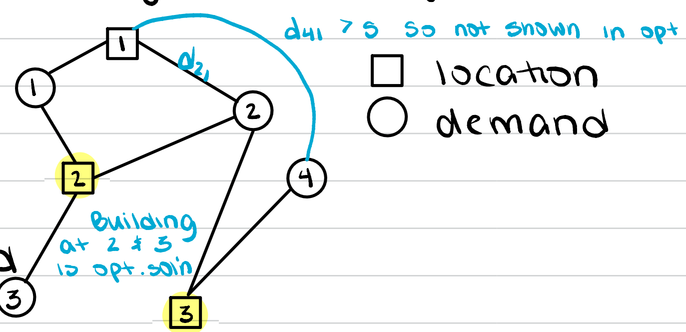
- Where should I build my facilities?
- Where should I locate a scarce resource?
- This lecture will focus on discrete facility location problems
- In general, there are <u>demand nodes</u> and <u>potential locations</u>
- This lecture focuses on <u>3 types</u>:

1. Set covering - build a minimum # of facilities to cover all demands

→ When we are <u>required</u> to take care of (almost) everyone. Fire stations, Police stations

- Consider a set of Demands (I) and a set of locations (J)
- The distance (travel time) between i and j is $d_{ij} > 0$, $i \in I$, $j \in J$
- A service level $s > 0$ is given: Demand i is said to be "covered" by location j if $d_{ij} < s$

Question:
How to allocate as few facilities as possible to cover <u>ALL</u> demand



Problem Formulation:

- let's define $a_{ij} = 1$ if $d_{ij} < S$ or 0 o/w
- let's define $x_j = 1$ if a facility is built at location j, 0 o/w

$$\min \sum_{j \in J} x_j$$ ← min total # of facilities

s.t.

of all facilities that cover each demand, at least 1 must be built

$$\sum_{j \in J} a_{ij} x_j \geq 1 \quad \forall i \in I$$ (set covering)

$$x_j \in \{0,1\} \quad \forall j \in J$$ (Binary)

Weighted version:

$$\sum_{j \in J} w_j x_j$$ w_j may be the cost of building the facility

2. Maximum covering - build a given # of facilities to cover as many demands as possible

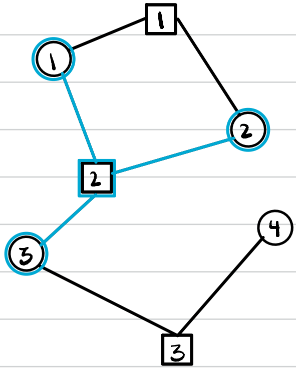
when budgets are limited, cellular data networks

- Consider a set of Demands (I) and a set of locations (J)
- The distance d_{ij}, service level s, and covering coef a_{ij} are also given
- We are restricted to build <u>AT MOST</u> $p \in \mathbb{N}$ facilities

Question:

How to allocate at most p facilities to cover as many demands as possible

↖ Max D!



□ location

○ demand

ex/ if $p=1$

Problem Formulation:

- let $x_j = 1$ if facility is built at location j
- let $y_i = 1$ if demand i is covered by any facilities

$$\max \sum_{i \in I} y_i$$

weighted version, ex/ # people at location y_i

$$\sum_{i \in I} w_i y_i$$

$$\text{s.t.} \quad \sum_{j \in J} a_{ij} x_j \geq y_i \quad \forall i \in I \qquad \text{(covering)}$$

$$\sum_{j \in J} x_j \leq p \qquad \text{(\# facilities)}$$

$$x_j \in \{0,1\} \quad \forall j \in J$$

$$y_i \in \{0,1\} \quad \forall i \in I$$

3. Fixed charge location - finding balance between benefit of covering demands and cost of building facilities

When service cost depends on distance, distribution centers

- Consider set of demands, I and a set of locations, J
- at demand i, the demand size is $h_i > 0$
- the unit shipping cost from location j to
- demand j is $d_{ij} > 0$

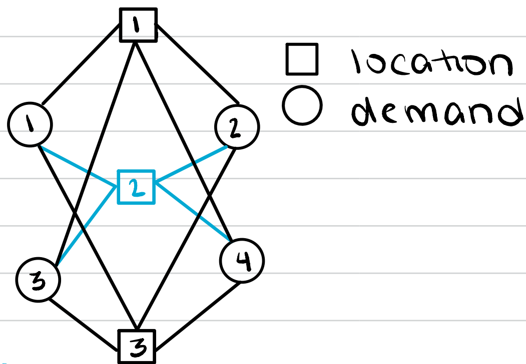
the fixed construction cost at j is $f_j > 0$

Question:

How to allocate some facilities to minimize the total shipping & construction costs

need to make 2 decisions

1. Allocate Facilities
2. Assign them to customers



* The tradeoff is higher shipping cost with less facilities or higher construction costs

$$\min \underbrace{\sum_{i\in I}\sum_{j\in J} \overbrace{h_i}^{\text{\# times traveled to } x_j} \overbrace{d_{ij}}^{\text{cost to travel}} \overbrace{y_{ij}}^{\text{if we travel}}}_{\text{shipping cost}} + \underbrace{\sum f_j x_j}_{\text{fixed con. cost}}$$

connects demand to facility

$$\text{s.t.} \quad y_{ij} \le x_j \quad \forall i \in I, j \in J$$

must be ONE fac. serving each demand

$$\sum_{j\in J} y_{ij} = 1 \quad \forall i \in I$$

$$x_j \in \{0,1\} \quad \forall j \in J$$

$$y_{ij} \equiv \{0,1\} \quad \forall i \in I$$

- $x_j = 1$ if facility is built at location j, o/w $x_j = 0$
- $y_{ij} = 1$ if demand i is served by facility at location j, o/w $y_{ij} = 0$

need to tell each customer (demand) who will be in charge of your service

(UFL)

- Previous example was the <u>uncapacitated</u> version
  - ↳ a facility can serve any amount of demand
- The capacitated version →

  capacitated facility location problem (CFL)

  if facility j has limited capacity $k_j > 0$, we can add the capacity constraint:

$$\sum_{i \in I} h_i y_{ij} \leq k_j \quad \forall j \in J$$

Machine Scheduling IP

↳ jobs | tasks assigned to machines / agents

- ex) Factory producing one product for n customers
  - one job processed at a time
  - each job has a due date
  - how to schedule n jobs to min the # of delayed jobs?

in this example, scheduling is <u>sequencing</u>

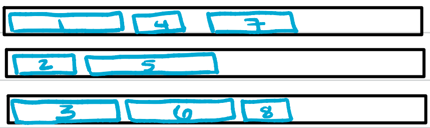
- splitting jobs is not helpful
- there are $n!$ ways to sequence

job 1   job 2   ...   job n → time

* let's use an IP!

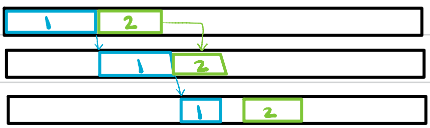
↗ single machine serial production

- if one machine, only need to determine order of jobs
- if > one machine
  - multiple parallel machines



- assign 8 jobs to machines
- each job has only one stage

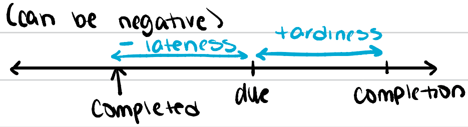
  - flow shop problems



- job has multiple stages
- each product needs to go through multiple machines

  - job shop problems
    - each job has different orders that have different requirements for machines

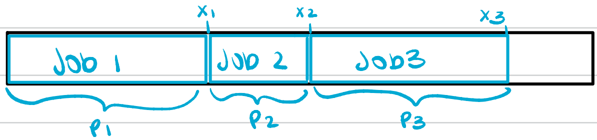
- Job Splitting
  - Preemptive: can stop & start tasks
  - nonpreemptive: CANNOT stop & start tasks
- Performance measurements
  - Makespan: time for all jobs to complete
  - (weighted) total (job) completion time
  - (weighted) # delayed jobs
  - (weighted) total tardiness (only relevant if after due date)
  - (weighted) total lateness (can be negative)
  - ...



example # 1

Machine Scheduling - completion time minimization

- 1 machine, n jobs, 1 job at a time
- job $j \in J = \{1, 2 \dots n\}$, processing time p_j
- completion time of job j is x_j ← dec. var.



so $p_2 = x_1 + x_2$, $x_2 - x_2 \geq p_2$

* splitting jobs is not helpful because it won't help us to FINISH JOBS FASTER GOAL!

* if the machine can only start job 2 after completion of job 1 $\longrightarrow$ $x_2 \geq x_1 + p_2$, o/w :

↳ in a feasible schedule, job i is either before or after job j for all $j \neq i$

↳ need to satisfy one of the two:

$$x_j \geq x_i + p_j \quad \text{and} \quad x_i \geq x_j + p_i$$

let $z_{ij} = 1$ if job j is BEFORE job i or 0 o/w

$i \in J$, $j \in J$, $i < j$

s.t. $x_i + p_j - x_j \leq M z_{ij}$ ← Big M constraints

$x_j + p_i - x_i \leq M(1 - z_{ij})$ — M must be $\geq \underbrace{\sum_{j \in J} p_j + p_j}_{\text{upper bound}}$

if $z_{ij} = 1$, then $= 0$, then

if $z_{ij} = 0$, then $x_i + p_j \leq x_j$, then

IP Formulation:

$$\min \sum_{j \in J} x_j$$ (total completion time)

$$
\begin{aligned}
\text{s.t.}\quad & x_i + p_j - x_j \le M z_{ij} && \forall i \in J, j \in J, i<j \\
& x_j + p_i - x_i \le M(1 - z_{ij}) && \forall i \in J, j \in J, i<j \\
& x_j \ge p_j && \text{(completion time} \ge \text{processing time)} \\
& x_j \ge 0 && \text{(non-neg)} \\
& z_{ij} \in \{0,1\} && \forall i \in J, j \in J, i<j \text{ (binary)}
\end{aligned}
$$

ensure jobs do not conflict w/ one another

prevents the sol'n from setting $x_i = 0$

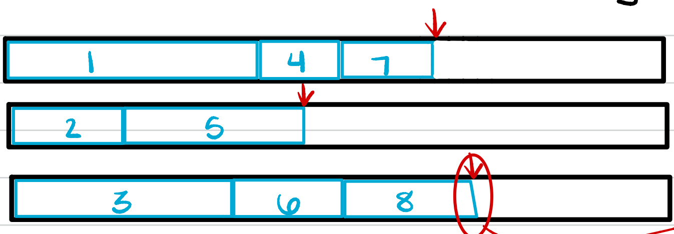
* This problem becomes harder if the job may be processed only after it's released

↳ let R_j be the release time of job j

example #2

Minimizing makespan on parallel machines

- Schedule n jobs on m parallel machines
- Job $j \in J = \{1, 2, \ldots, n\}$ has processing time p_j
- a job can be processed at any machine. However it can be processed ONLY by ONE machine



- different schedules give machines different completion times
- How can we minimize makespan?

↳ latest completion time

* balance the loading

∘ Job assignment not sequencing (order doesn't matter)

IP Formulation:

let $x_{ij} = 1$ if job $j \in J$ is assigned to machine $i \in I$
or 0 o/w

on machine i, the last job is completed at

completion time for machine i $\left\{ \sum_{j \in J} p_j x_{ij} \right\}$ sum of all processing times ~~for~~ jobs on this machine

the makespan w is the max completion time among ALL machines

for each machine i

$$w \geq \sum_{j \in J} p_j x_{ij} \quad \forall i \in I$$

but since we're minimizing it won't go over the real value

min w

technically the w could be larger than the actual w given these constraints...

$$\text{s.t.} \quad w \geq \sum_{j \in J} p_j x_{ij} \quad \forall i \in I \qquad (1)$$

$$\sum_{i \in I} x_{ij} = 1 \quad \forall i \in I \qquad (2)$$

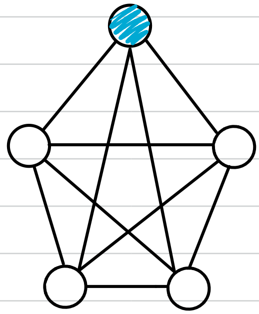
$$x_{ij} \in \{0, 1\} \quad \forall i \in I, j \in J \qquad (3)$$

Vehicle Routing IP

- deliver / collect items in most efficient way
- find the shortest path between any pair of 2 addresses

a special case of vehicle routing

- Traveling Salesman Problem: choose a route starting from the post office, passing each address exactly once, then returning to office



- This is a sequencing problem; in total there are $4! = 24$ feasible routes
- which route minimizes total distance (time)?

- CAPACITY CONSTRAINT: sometimes the vehicle has limited capacity and must return to "home base" once it is reached to unload → multiple sequential routes
  - ↳ assume none for this example

Formulate TSP into Integer Program:

- let's consider a directed complete network: $G = (V, E)$ (network; nodes; directed edges)
  - n nodes, $n(n-1)$ arcs
  - arc weight for arc (i,j) is $d_{ij} > 0$
- select a few arcs in E to form a tour (complete visit passing all nodes)
  - select n arcs → should form a cycle passing all nodes

- let $x_{ij} = 1$ if arc $(i,j) \in E$ is selected, 0 o/w
  - objective

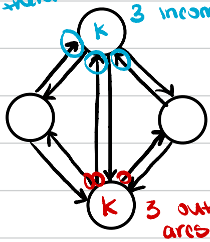
Min (total distance traveled)

all arcs → $$\sum_{(i,j) \in E} d_{ij} x_{ij}$$

s.t.

for node $k \in V$: $$\sum_{i \in V, i \neq k} x_{ik} = 1$$

this means you can only choose one of them



$$\sum_{j \in V, j \neq k} x_{kj} = 1$$

this means you can only choose 1 of them

* However, this is NOT enough to prevent subtours

Subtour elimination

Alternative #1: for each subset of nodes with at least two nodes, we limit the max # of arcs selected:

For 3 nodes, max # of arcs you may select is 2

For all subsets

$$\sum_{i \in S, j \in S, i \neq j} x_{ij} \leq |S| - 1 \quad \forall S \subsetneq V, \; |S| \geq 2$$

|S|: cardinality of S (magn) man

proper subset

set of all nodes

only care about S with more than 1 node

- This way leads to LOTS of constraints...
  - 2^n ways to choose a subset
  - n ways to choose a subset of 1 node
  - 2 ways to choose a subset of ∅ or n nodes

→ this is why we look at the next altern.

Alternative #2

- Let u_is rep. the order of passing nodes. $u_i = k$ if node i is the k^{th} node passed in a tour
- Add the following constraints

$$u_1 = 1$$

$$2 \le u_i \le n \qquad \forall i \in V \setminus \{1\}$$

$$u_i - u_j + 1 \le (n-1)(1 - x_{ij}) \qquad \forall (i,j) \in E,\ \underline{i \ne 1,\ j \ne 1}$$

don't impose on initial node

* If $x_{ij} = 0$, there is no constraint for u_i and u_j, o/w u_j must be larger than u_i by at least 1

* If tour does not contain node 1, the last constraint pushes those u_is to ∞ and violates constraint 2

* only node 1 is not restricted by these const.'s!

much fewer const.'s than Alt #1:

When we have n nodes, we have n additional variables

$n + (n-1)(n-2)$

Complete TSP Formulation

So is Alt 1 or 2 better?
↳ Alt 1 solves faster
↳ Alt 2 has less cont.

$$\min \sum_{(i,j) \in E} d_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{i \in V, i \ne k} x_{ik} = 1 \qquad \forall k \in V$$

$$\sum_{j \in V, j \ne k} x_{kj} = 1 \qquad \forall k \in V$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in E$$

Critical !

* Whether to set a variable to be integer, binary or continuous depends on how critical that integer constraint is

Quantity → continuous
Assignment → Binary

* Very few to set to integer

Week 4
Nonlinear Programming

- examples such as: $\overbrace{\text{Pricing, inventory, Porfolio opt.}}^{\text{Balancing}}$

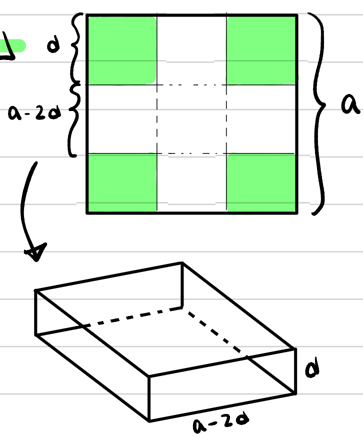
ex) Pricing a single good

- A retailer buys 1 product at unit cost c
- it chooses retail price p
- demand is a function of price, p: $D(p) = a - bp$
  - ↳ How to formulate the problem of finding the profit-maximizing price?
    - Parameters $a > 0$, $b > 0$, $c > 0$
    - decision variable: p
    - constraint $p \geq 0$

Formulation:

$$\left.\begin{aligned} &\max_{p} \ \overbrace{\underbrace{(p-c)}_{\text{margin}}\underbrace{(a-bp)}_{\text{demand}}}^{\text{non-linear}} \\ &\text{s.t.} \quad p \geq 0 \end{aligned}\right\} \quad \text{OR} \quad \max_{p \geq 0} (p-c)(a-bp)$$

ex)

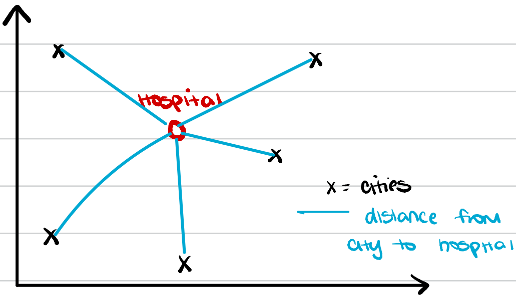


- given an axa piece of paper
- how to choose d to max. the volume of the container?

$$\max_{d \in [0, \frac{a}{2}]} (a-2d)^2 d$$

ex) locating a hospital

- in a country with n cities, each at location (x_i, y_i)
- we want to locate a hospital at location (x, y) to min the avg. Euclidean distance from the cities to the hospital



$$\min_{x,y} \sum_{i=1}^{n} \underbrace{\sqrt{(x-x_i)^2+(y-y_i)^2}}_{\text{distance hospital} \leftrightarrow \text{city}}$$

In General, a Nonlinear program (NLP) can be formulated as

there are n decision variables

$$\min_{x \in \mathbb{R}^n} f(x)$$

$$\text{s.t. } g_i(x) \le b_i \;\; \forall i = 1, \ldots, m$$

There are m constraints

* LP if all f and g_is are linear in x
NLP if at least one of f and g_is is nonlinear in x

Formulation of NLP is easy → solving is hard

The Economic Order Quantity (EOQ) Problem

ex)

- Airline uses 500 taillights per year, \$500 each
- taillights are consumed at a constant rate
- ordering cost of \$5, regardless of order size
- holding cost of \$0.02 per taillight per month
- GOAL: minimize total cost

How much to order? When to order?

* Balance ordering cost & holding cost

* Formulate NLP whose optimal sol'n is the optimal order quantity

Assumptions for basic EOQ model

1. Demand is deterministic & occurs at constant rate
2. regardless the order quantity - fixed ordering cost is incurred
3. No shortage is allowed
4. ordering lead time is zero
5. inventory holding cost is constant

Parameters

D = annual demand (units)

K = unit ordering cost (\$)

h = unit holding cost per year (\$)

p = unit purchasing cost

Decision variable

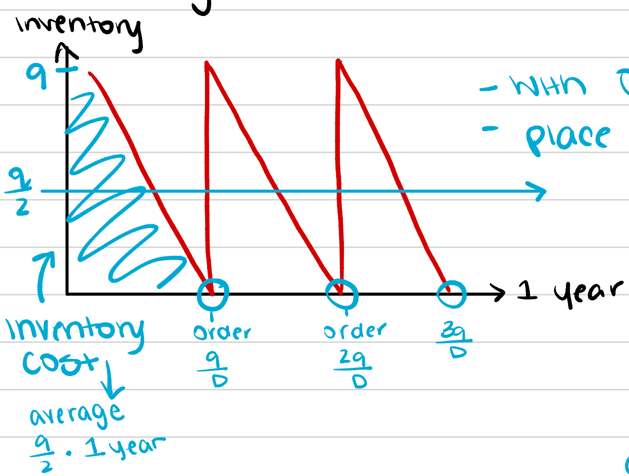
q = order quantity per order (units)

Objective

Min annual total cost

* We will use 1 year as our time unit

Inventory Level



- with 0 ordering lead time
- place order when inventory = 0

Annual costs

avg inventory cost

- annual holding cost $= h \times \frac{q}{2} = \frac{hq}{2}$
- annual purchasing cost $= pD$
- annual ordering cost $= K \times \frac{D}{q} = \frac{KD}{q}$

orders per year

NLP Formulation

$$\min_{q \geq 0} \frac{KD}{q} + pD + \frac{hq}{2}$$

Since pD is just a constant, a more relevant obj is

$$TC(q) = \frac{KD}{q} + \frac{hq}{2}$$

Portfolio optimization

ex) invest $100,000 in 3 stocks

stock	current price	expected price
1	$50	$55
2	$40	$50
3	$25	$20

how to allocate budget to max exp. profit?

let x_i be the share of stock i purchased

$$\max \quad 55x_1 + 50x_2 + 20x_3$$
$$\text{s.t.} \quad 50x_1 + 40x_2 + 25x_3 \le 100000$$
$$x_1 \ge 0, \; x_2 \ge 0, \; x_3 \ge 0$$

easy to find optimal → 2,500 shares of stock 2

* need to consider RISK
  ↳ the larger the variance of the total revenue → the higher the risk

so this:

min the total variance while ensuring a certain expected revenue

Variance

random variable

$$\mathrm{Var}(X) = \sum_{i=1}^{n} \underbrace{\Pr(X = x_i)}_{\text{prob for } x \text{ to occur}} (x_i - \mu)^2$$

μ: expected value

- lets assume future price for stock 1 may be \$65 or \$45, each w/ prob 50%.
  - if we buy 1 share → variance is
    $$\frac{1}{2}(65-55)^2 + \frac{1}{2}(45-55)^2 = 100$$
  - var of buying x_1 shares is $100x_1^2$
    - in general, $var(bX) = b^2 var(X)$

The more you buy, the higher the risk!

Minimizing the risk

- let variances for stocks be 100, 1600, 100
- var of total revenue $= 100x_1^2 + 1600x_2^2 + 100x_3^2$

$$\min \quad 100x_1^2 + 1600x_2^2 + 100x_3^2$$

$$\text{s.t.} \quad 50x_1 + 40x_2 + 25x_3 \leq 100{,}000$$

$$55x_1 + 50x_2 + 20x_3 \geq R$$

$$x_1 \geq 0, \; x_2 \geq 0, \; x_3 \geq 0$$

expected revenue

The higher the expected profit → the higher the risk!

Compact Formulation

- invest B in n stocks. min expected return is R.
- for stock i, current price is p_i, expected price is μ_i, variance of buying 1 share is σ_i^2
- let x_i be the # of shares we buy of stock i

$$\min \sum_{i=1}^{n} \sigma_i^2 x_i^2$$

s.t.

$$\sum_{i=1}^{n} p_i x_i \leq B$$

$$\sum_{i=1}^{n} \mu_i x_i \geq R$$

$$x_i \geq 0 \quad \forall i = 1, \ldots, n$$

Considering correlation among stocks

- let σ_{ij} be the covariance b/w stock i and j
- extended formulation is:

$$\min \sum_{i=1}^{n} \sigma_i^2 x_i^2 + \underbrace{2\sum_{i=1}^{n} \sum_{j=i+1}^{n} \sigma_{ij} x_i x_j}_{\text{added covariance term}}$$

s.t.

$$\sum_{i=1}^{n} p_i x_i \leq B$$

$$\sum_{i=1}^{n} \mu_i x_i \geq R$$

$$x_i \geq 0 \quad \forall i = 1, \ldots, n$$

* Why linearize?
  - makes solving much easier!

Linearizing max/min functions

Absolute value function

ex) Fair allocation

- want to allocate \$1000 to 2 people fairly
- the smaller the difference, the fairer
- obviously, \$500 to each

abs value is non-linear

$$\min \ |x_2 - x_1|$$
$$\text{s.t.} \quad x_1 + x_2 = 1000$$
$$x_i \geq 0 \quad \forall i = 1, 2$$

w/out abs function, opt value is −1000

how can we linearize it?

$$w = |x_2 - x_1|$$
$$\min \ w$$
$$\text{s.t.} \quad x_1 + x_2 = 1000$$
$$w \geq |x_2 - x_1|$$
$$x_i \geq 0 \quad \forall i = 1, 2$$

arguement that in this problem we can swap = for ≥

now notice that $|x_2 - x_1| = \max\{x_2 - x_1, x_1 - x_2\}$ and

$$w \geq \max\{x_2 - x_1, x_1 - x_2\} \longleftrightarrow w \geq x_2 - x_1 \ \& \ w \geq x_1 - x_2$$

therefore the LP that we want is:

$$\min \ w$$
$$\text{s.t.} \quad x_1 + x_2 = 1000$$
$$w \geq x_2 - x_1$$
$$w \geq x_1 - x_2$$
$$x_i \geq 0 \quad \forall i = 1, 2$$

Linearizing constraints

* Works for min function at the larger side or max function at the smaller side *

$y \ge \max\{x_1, x_2\} \longleftrightarrow y \ge x_1$ and $y \ge x_2$

* y, x_1 and x_2 can be variables, parameters, or a function

$y + x_1 + 3 \ge \max\{x_1 - x_3, 2x_2 + 4\} \longleftrightarrow y + x_1 + 3 \ge x_1 - x_3$ AND
$y + x_1 + 3 \ge 2x_2 + 4$

* might be multiple terms in max function

$y \le \min\{x_i\} \longleftrightarrow y \le x_i \ \forall x_i = 1, \ldots, n$

$y \ge \max_{i=1,\ldots,n}\{x_i\} \longleftrightarrow y \ge x_i \ \forall x_i = 1, \ldots, n$

* This tecnique does <u>NOT</u> apply to

use OR
1. $y \le \max\{x_1, x_2\} \ne y \le x_1$ and $y \le x_2$
2. $y \ge \min\{x_1, x_2\} \ne y \ge x_1$ and $y \ge x_2$
3. max or min function in an equality

Linearizing Objective Function

Min a max function

$$\min \max\{x_1, x_2\} \longrightarrow \begin{aligned} &\min \quad w \\ &\text{s.t.} \quad w \geq x_1 \\ &\qquad\ \ w \geq x_2 \end{aligned}$$

max a min function

$$\begin{aligned} &\max \quad \min\{x_1, x_2, 2x_3 + 5\} + x_4 \\ &\text{s.t.} \quad 2x_1 + x_2 - x_4 \leq x_3 \end{aligned} \longleftrightarrow \begin{aligned} &\max \quad w + x_4 \\ &\text{s.t.} \quad w \leq x_1 \\ &\qquad\ \ w \leq x_2 \\ &\qquad\ \ w \leq 2x_3 + 5 \\ &\qquad\ \ 2x_1 + x_2 - x_4 \leq x_3 \end{aligned}$$

* doesn't apply to max a max function or min a min function
* Absolute function is just a max function

$$|x| = \max\{x, -x\}$$

Another example

$$\min_{x,y} \sum_{i=1}^{n} (|x - x_i| + |y - y_i|)$$

$\downarrow$ linearized

$$\min \sum_{i=1}^{n} (u_i + v_i)$$

$$\begin{aligned} \text{s.t.} \quad & u_i \geq x - x_i, \quad u_i \geq x_i - x \quad \forall i = 1, \dots n \\ & v_i \geq y - y_i, \quad v_i \geq y_i - y \quad \forall i = 1, \dots n \end{aligned}$$

Linearize Products of decision variables

- can be done if 1 or both are binary (not if both are contin.)

↳ x^2 , xy

Scenario 1A

- 2 products, limited resources, setup costs, reduction in setup cost if you make both

$$\begin{aligned}
\max \quad & 10x_1 + 12x_2 - 20x_1 - 25x_2 \cancel{+10z_1z_2} + 10w \\
\text{s.t.} \quad & 2x_1 + x_2 \le 6 \\
& x_1 + 2x_2 \le 8 \\
& x_1 \le 3z_1 \\
& x_2 \le 4z_2 \\
& x_1, x_2 \ge 0 \\
& w \le z_1 \\
& w \le z_2 \\
& z_1, z_2 \in \{0,1\} \\
& w \in \{0,1\}
\end{aligned}$$

resource constraints

→ can we linearize $z_1 z_2$?

Scenario 1B

- 2 product, limited resources, setup costs, ADDITIONAL setup cost if you make both

$$\begin{aligned}
\max \quad & 10x_1 + 12x_2 - 20x_1 - 25x_2 \cancel{-10z_1z_2} \quad -10w \\
\text{s.t.} \quad & 2x_1 + x_2 \le 6 \\
& x_1 + 2x_2 \le 8 \\
& x_1 \le 3z_1 \\
& x_2 \le 4z_2 \\
& w \ge z_1 + z_2 - 1 \\
& x_1, x_2 \ge 0 \\
& z_1, z_2 \in \{0,1\} \\
& w \in \{0,1\}
\end{aligned}$$

w now appears in a minimization obj. function so $w \le z_1$ & $w \le z_2$ do not = $w = z_1 z_2$

→ if z_1 & $z_2 = 1$ → requires $w \ge 1$ → $w = 1$

Scenario 1C (constraints)

"larger" side

↳ linearized as if it appears in a max function

max ...

s.t. $x \le 5 z_1 z_2$

$x \ge 0$

$z_1, z_2 \in \{0,1\}$

↓

max ...

s.t. $x \le 5w$

$x \ge 0$

$z_1, z_2 \in \{0,1\}$

$w \le z_1 \quad , w \le z_2$

$w \in \{0,1\}$

*Product term should be 1 only if both terms are 1

"smaller" side

↳ linearized as if it appears in a min function

max ...

s.t. $x \ge 5 z_1 z_2$

$x \ge 0$

$z_1, z_2 \in \{0,1\}$

↓

max ...

s.t. $x \ge 5w$

$x \ge 0$

$z_1, z_2 \in \{0,1\}$

$w \ge z_1 + z_2 + 1$

$w \in \{0,1\}$

* Product term cannot be 1 if either term is 0

Scenario 2A

- 2 products with 2 limited resources, fixed payment required to do the business

~~max $(10x_1 + 12x_2)z - 15z$~~

max $10w_1 + 12w_2 - 15z$

s.t. $2x_1 + x_2 \le 6$

$x_1 + 2x_2 \le 8$

$x_1, x_2 \ge 0$

$z \in \{0, 1\}$

$w_1 \le x_1, \; w_1 \le 3z$

$w_2 \le x_2, \; w_2 \le 4z$

why 3? comes from $2x_1 + x_2 \le 6$. x_1 CANNOT be larger than 3 to hold this!

can we linearize $\underbrace{\cancel{x_1 z}}_{w_1}$ & $\underbrace{\cancel{x_2 z}}_{w_2}$?

* cannot impose $w_1 \le x_1$ & $w_1 \le z$ b/c $w_1 \le z$ is too tight

↓

we should "remove" the constraint when $z = 1 \rightarrow$ AKA the RHS should contain a value that is an upper bound of x_1

Scenario 2B

- A company may run 2 prod. processes to fulfill demands for 2 products if it accepts an order

max ~~$50z - (10x_1 + 12x_2)z$~~

max $50z - 10w_1 - 12w_2$

s.t. $2x_1 + x_2 \ge 6$

$x_1 + 2x_2 \ge 8$

$x_1, x_2 \ge 0$

$z \in \{0, 1\}$

$w_1 \ge x_1 - 8(1 - z)$

$w_2 \ge x_2 - 6(1 - z)$

$w_1, w_2 \ge 0$

x_1 is upper bounded by 8, if you don't know exact upper, use big #

* w now appears in a minimization obj. function, w should be lower bounded rather than upper

Scenario 2C

- when product term appears on the "larger" side of a constraint, can be lin. as if it appears in a max. obj. function (should be upper bounded

max ...

s.t. $\cancel{x_1 z}\ w \geq 5x_2$

$x_1 + x_2 \leq 10$

$x_1, x_2 \geq 0$

$z \in \{0,1\}$

$w \leq x_1,\ w \leq 10z$ give w an upper bound

$w \geq 0$

Scenario 2D

- when product term appears on the "smaller" side of a constraint, can be lin. as if it appears in a min. obj. function (should be lower bounded

max ...

s.t. $\cancel{x_1 z}\ w \leq 5x_2$

$x_1 + x_2 \leq 10$

$x_1, x_2 \geq 0$

$z \in \{0,1\}$

$w \geq x_1 - 10(1-z)$ lower bounded

$w \geq 0$

Week 5

case study - Personnel Scheduling

Real Problem → conceptual model → math model → computer model

- CORE decision variable vs Derived decision variable
- soft constraints → add to objective as penalties since they don't NEED to be satisfied
- make sure to have constraints to link core decision variables ⟶ derived decision variables
- make sure to include "For all" statement ($\forall i$) and the boundaries ($\in J$)
- Performance evaluation is IMPORTANT! Find valuable KPI's to compare before & after
  ↳ monatize it!. this is better for management
- More time to solve usually means "more optimal" solution
- we want to implement better operational decisions for LONG TERM improvements
- only need OR for DIFFICULT and IMPORTANT problems

Week 6

Course Summary & Future Directions

Types of Programming Problems

Objective function & constraints	Variables: All continuous	Variables: Some integral
All Linear	Linear Programming • resource allocation • Production Planning	Integer Programming • Machine scheduling • Facility location (TSP)
Some Nonlinear	non-linear Programming • Product Pricing • inventory	non-linear integer Programming Advanced problems (Still very difficult to solve)