Introduction to Optimization and Geoscience Applications

Jitkomut Songsiri

Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University

CUEE

June 3, 2026

Outline

How to read this handout

1. topics starting with **B** refer to **basic** lecture focusing on problem description and applications
2. topics starting with **A** refer to **advance** lecture that contain mathematical details of the optimization
3. the examples are presented according to application focus, not its mathematical problem type

B1: Fundamental of optimization

Outline

What is optimization?

Anatomy of optimization

definition: choosing the **best** element from a set of available alternatives. It involves finding the input values that result in the minimum or maximum value of a specific objective function while satisfying a given set of constraints.

essential elements of an optimization:

1. **objective:** a quantitative measure of success (e.g., profit, energy efficiency, or error rate).
2. **variables:** the 'knobs' you can turn or decisions/mechanism you can make
3. **constraints:** real-world limits—resources, physics, or legal requirements—that define the "feasible region."
4. **problem parameters:** parameters required to be set in the problem statement

Example: Diet problem

description:

- given n meals; each has different price and nutrient details
- design a diet plan to meet the nutrient requirements with the lowest cost

1. **variables:** the quantity of each meal to consume
2. **objective:** total cost of meals
3. **constraints:**
   - quantities of meals must be non-negative
   - consumed carb/protein/vitamin/fat must be within a recommended threhold
4. **problem parameters:** price and nutrients of each food, the required quantitiy of each nutrient

Diet problem: formulation

variables: x_1, x_2, x_3 quantities of milk, beans, meat

problem parameters

food	cost ($/unit)	protein (g/unit)	calorie (kcal/unit)
milk	2.0	8	120
beans	1.5	12	200
meat	5.0	25	250

requirements (constraints)

- protein at least 60g
- calories at least 1800 kcal

diet optimization problem:

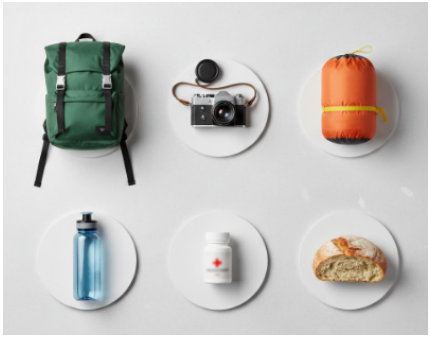
$$
\begin{array}{ll}
\text{minimize} & 2x_1 + 1.5x_2 + 5x_3 \\
\text{subject to} & x_1, x_2, x_3 \geq 0, \\
& 8x_1 + 12x_2 + 25x_3 \geq 60, \\
& 120x_1 + 200x_2 + 250x_3 \geq 1800
\end{array}
$$

the objective and contraint functions are linear; called a linear program (LP)

Structure of diet problem

- each item can be decomposed to contain several entities (each food contain a set of nutrients)
- the role of cost and nutrient can be swapped
- one can maxmize the total nutrient while the cost is within a limit
- if we only care about cost, we eat 'cheap' food; if we care about health, we each 'clean' food

Example: Knapsack problem



description:

- given a backpack that can only hold up to 15 kg
- each item has a specific weight and a specific importance value
- decide which items to take to maximize total value without breaking the bag

1. **variables:** binary decision of each item (1 if taken, 0 if not taken); you cannot take 'half' of the camera
2. **objective:** maximize sum of total **value** of items you pack
3. **constraints:** the sum of the weights of selected items must be ≤ 15 kg
4. **problem parameters:** backpack weight limit, value of each item

Knapsack: formulation

zero-one knapsack problem

$$\text{maximize } c^T x \text{ subject to } w^T x \leq K, \quad x_j \in \{0,1\}, \quad j = 1,2,\ldots,n$$

setting:

- given n items; each has weight w_j and value c_j
- given a bound K on the total weight that can be carried in a knapsack

goal: decide to select items to maximize the total value

Introductory examples: LP/MILP/Model estimation

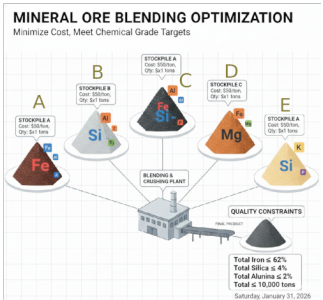
Energy economic dispatch

description:

- given power plants (solar, wind, natural gas, coal); each has different cost and CO_2 emission value
- decide how much power to draw from each plant to meet the city's electricity demand with minimum operating cost

1. **variables:** generation's power outputs
2. **objective:** minimize the total cost
3. **constraints:**
   - total power must meet the demand
   - power from a plant has its physical limit
   - CO2 emission must be below a limit
4. **problem parameters:** ✎

Mineral ore blending



description:

- given multiple stockpiles of ore; each has different concentrations of minerals (iron, silica, alumina)
- final blending must meet the strict chemical grade requirement
- decide how to mix these stockpiles to minimize the cost of extraction and transport

1. **variables:** the quantity taken from each ore stockpile
2. **objective:** minimize the production cost
3. **constraints:**
   - total iron must be $\geq 62\%$; total silica must be $\leq 4\%$ and alumina $\leq 2\%$
   - sum of quantities must equal total shipment size
   - we cannot take more ore from a stockpile than is physical limit
4. **problem parameters:** ✎

NASA Example: bring equipments to a satellite

description:

- given a backpack having a limit of 50 kg
- each item has a specific weight and a specific importance value
- decide which items to take to maximize total value within the weight limit

1. **variables:** ✎
2. **objective:** ✎
3. **constraints:** ✎
4. **problem parameters:** ✎

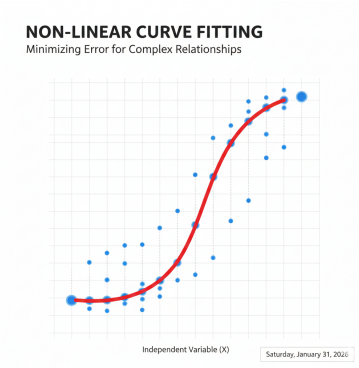
Project management: select softwares

description:

- given a total monthly budget of 100 USD
- each software has a value and running cost
- decide which softwares to launch that maximize its importance within the budget

1. **variables:** ✎
2. **objective:** ✎
3. **constraints:** ✎
4. **problem parameters:** ✎

Curve fitting

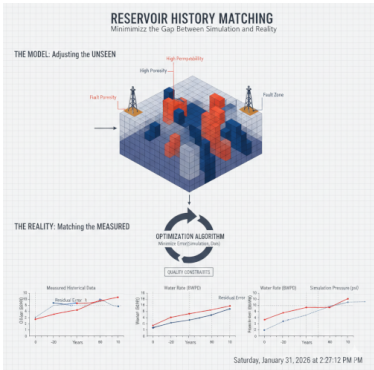


description:

- given N data points $\{(t_i, x_i)\}_{i=1}^N$, and a function description to fit (here, sigmoid function)
- find the best function parameters that minimize the error

1. **variables:** function parameter θ: $f(x; a, b, c) = a/(1 + e^{-c(x-b)})$
2. **objective:** minimize the sum of squared errors
3. **constraints:** -
4. **problem parameters:** data points $\{(t_i, x_i)\}_{i=1}^N$

Reservoir history matching



description:

- a reservoir simulation model has thousands of uncertain parameters, such as local permeability and porosity
- adjust these parameters until the model's predicted oil, water, and gas production rates match the actual historical data from the wells

1. **variables:** model parameters
2. **objective:** minimize the sum of squared errors
3. **constraints:** depends on the model
4. **problem parameters:** data points

Reservoir history matching: formulation

let $g(t, x)$ be the **reservior simulator** at time t for a given parameter x

- x contains permeability and porosity
- g usually is complex PDEs (Darcy's law and mass balance)

variable: model parameter x

problem: minimize error between the model prediction and observed data, while regularizing the estimated parameter

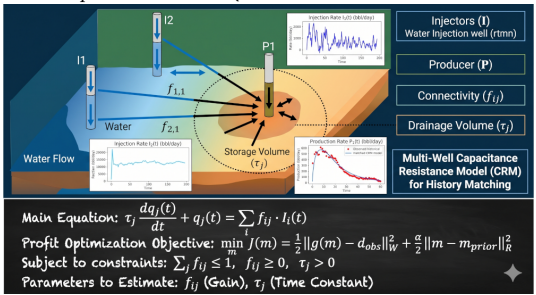
$$\underset{x}{\text{minimize}}\, f(x) := \sum_{t=1}^{N} \|g(t_i, x) - y(t)\|^2 + \gamma \|x - x_{\text{prior}}\|_2^2 \;\text{ subject to }\; x_{\min} \preceq x \preceq x_{\max}$$

proxy model: a radial flow model based on pressure data

$$g(t, x) = \frac{150(\log(t) + s)}{k}, \quad s = \text{skin factor}, \;\; k = \text{permeability}$$

Reservoir matching: CRM (capacitance resistance model)

a reservior field with N_i injectors (wells where we pump water in to maintain pressure) and N_p producers (wells where we extract oil/water)



- $f_{ij} \geq 0$ is a flow from injector i to reach producer j (connectivity)
- $\sum_{j=1}^{N_p} f_{ij} \leq 1$ (sum of all gains from an injector i must be less than 1)
- q_j is total fluid production rate (oil+water) from producer j
- I_j as the injection rate (input)

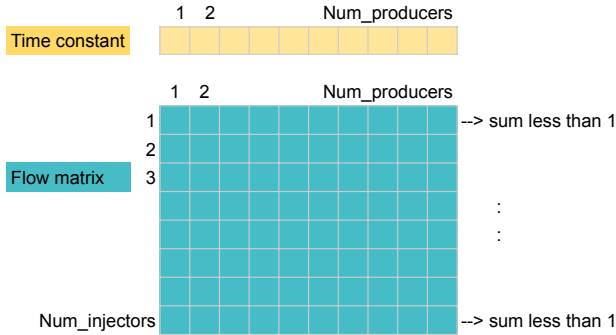
CRM model: $$\tau_j \frac{dq_j(t)}{dt} + q_j(t) = \sum_{i=1}^{N_i} f_{ij} I_i(t), \quad j = 1, 2, \ldots, N_p$$

τ_j is time constant, q_j is a response of first-order differential equation to injectors

CRM model estimation

given observed production rate q,

estimate time constant τ and flow matrix $F = [f_{ij}]$ in CRM model to match q



- both τ and F are non-negative
- each row of F sums up to one

CRM parameter optimization

problem: given signals q_j^{obs} for $j \in [N_p]$ and I_i for $i \in [N_i]$ we aim to estimate τ_j and f_{ij} for all i, j

variables: $x = (\tau_j, f_{ij})$ for $j \in [N_p]$ and $i \in [N_i]$

optimization: minimize error between observed value and CRM model output

$$\underset{\tau_j, f_{ij}}{\text{minimize}} \quad \sum_{j=1}^{N_p} \sum_{t=1}^{T} \left(q_j^{\text{obs}}(t) - q_j(t, f_{ij}, \tau_j) \right)^2 \quad \text{subject to} \quad f_{ij} \geq 0, \sum_{j=1}^{N_p} f_{ij} \leq 1, \tau_j \geq 0$$

- CRM model can be discretized using Euler method to simulate q_j

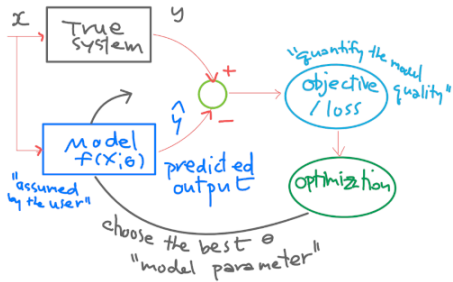
$$\tau_j \frac{(q_j[t] - q_j[t-1])}{\Delta t} = \sum_i f_{ij} I_i[t] - q_j[t-1]$$

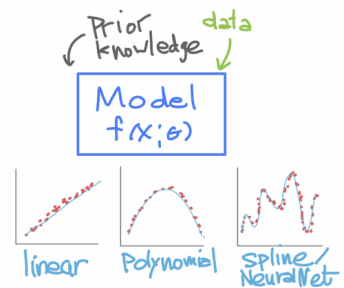
CRM model estimation

observations on the optimization

- with the constraint $\sum_j f_{ij} \leq 1$, we optimize f_{ij} for all i, j (without it, the optimization is separable in j) – so f is a matrix variable, and τ is a vector
- treating CRM model output as nonlinear function of τ_j, f_{ij}, the optimization problem is nonlinear least-squares
- using modeling techniques can cast the discretized model as an autoregressive with exogenous input (ARX) and the linear least-squaure technique can be applied

Structure of model estimation problem





1. **variables:** model parameters
2. **objective:** the concept of loss function; depend of model architecture/statistical assumption of estimation problem
3. **constraints:** depend on model assumptions
4. **problem parameters:** data points (x, y)

Problem setting

Problem setting

(mathematical) optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p \end{array} \tag{P1}$$

- $x = (x_1, \ldots, x_n)$: optimization variable
- $f_0 : \mathbf{R}^n \to \mathbf{R}$: objective function
- $f_i : \mathbf{R}^n \to \mathbf{R}, i = 1, \ldots, m$: inequality constraint functions
- $h_i : \mathbf{R}^n \to \mathbf{R}, i = 1, \ldots, p$: equality constraint functions

constraint set: $\mathcal{C} = \{x \in \mathbf{R}^n \mid f_i(x) \leq 0, i = 1, \ldots, m, h_i(x) = 0, i = 1, \ldots, p\}$

domain of the problem: $\mathcal{D} = \bigcap_{i=0}^{m} \mathbf{dom}\, f_i \cap \bigcap_{i=1}^{p} \mathbf{dom}\, h_i$

Optimal value

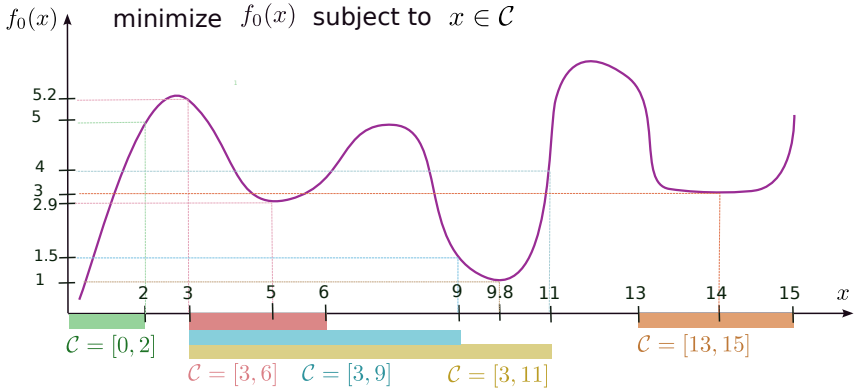
$$p^\star = \inf \{ f_0(x) \mid f_i(x) \leq 0,\ i = 1, \ldots, m,\ h_i(x) = 0,\ , i = 1, \ldots, p \}$$

- we say x is **feasible** if $x \in \mathbf{dom}\, f_0(x)$ and $x \in \mathcal{C}$
- $p^\star = \infty$ if the problem is **infeasible**
- $p^\star = -\infty$ if the problem is unbounded below
- a feasible x is called **optimal** if $f_0(x) = p^\star$; there can be many
- x is **locally optimal** if $\exists \epsilon > 0$ such that x is optimal for

$$\begin{array}{ll} \text{minimize} & f_0(z) \\ \text{subject to} & z \in \mathcal{C}, \quad \|z - x\|_2 \leq \epsilon \end{array}$$

in other words, a locally optimal point is the best solution in a neighborhood

Example



find achievable objective values, $p^\star$ and $x^\star$ for each $\mathcal{C}$

Optimality of unconstrained problems

assumption: f is twice continuously differentiable (smooth objective)

- **1st-order necessary condition:**

$$\text{if } x^\star \text{ is a local minimizer of } f \text{ then } \nabla f(x^\star) = 0$$

- **2nd-order necessary condition:** if $x^\star$ is a local minimizer of f then $\nabla f(x^\star) = 0$ and $\nabla^2 f(x^\star) \succeq 0$ (positive semidefinite)
- **2nd-order sufficient condition:** if $\nabla f(x^\star) = 0$ and $\nabla^2 f(x^\star) \succ 0$ (pdf)

$$\text{then } x^\star \text{ is a strict local minimizer of } f$$

local minimizers can be distinguished from other stationary points by examining positive definiteness of $\nabla^2 f$

example: $f(x) = x^4$ has $x^\star = 0$ as a local minimizer; $\nabla^2 f(x^\star) = 0$ (hence, 2nd-order sufficient condition fails)

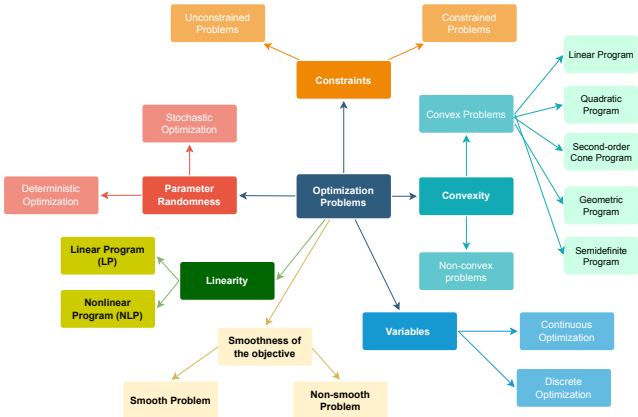
Problem types

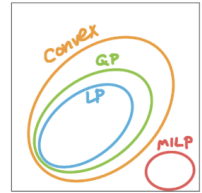
Problem types

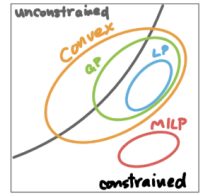
we can categorize optimization problems by

- **constraints**
  - unconstrained problem
  - constrained problems
- **variable types**
  - continuous optimization
  - discrete optimization
- **linearity of objective and constraints**
  - linear program
  - nonlinear program
- **convexity of objective and constraint set**
  - convex problem
  - non-convex problem
- **smoothness of the objective**
  - smooth problem
  - non-smooth problem
- **parameter randomness**
  - stochastic optimization
  - deterministic optimization

a numerical method generally is tailored to a specific problem type







other specific problem types are integer programming, vector optimization

Numerical tools

Numerical tools

- software tiers:
  - modeling languages/wrapper (describe the problem and connect with solvers): AMPL, cvxpy, Google OR-tools
  - solvers (computational engine): Gurobi, CPLEX
  - library/toolbox (bundle both modeling and solvers): MATLAB, Scipy
- commercial vs open source
- domain specificity: e.g. cvxpy is for convex problems

Modeling language and wrappers

- these tools allow us to describe what the problem is (objective and constraint) without worrying about how the computer solves it
  - **declarative modeling:** AMPL and CVXPY: we can write $Ax \leq b$ and the softwares builds the mathematical tree
  - **procedural modeling:** Google OR tools: it builds the model row-by-row (one constraint at a time); it is more of an **API** than a language
- acts as a middle man to translate our Python/math code to a format the computational engine can read

Solvers

this is the computational engine that runs the actual algorithms (simplex, interior point, branch and bound, etc.)

- **commerial (high performance):** Gurobi, CPLEX, and MOSEK; generally the fastest but require licenses
- **open source:**
  - GLOP/PDLP/CP-SAT: inside OR-tools
  - HiGHS: high-performance softwares to solve large-scale LP, MIP and QP
  - OSQP: solve QP problem using splitting method
  - SCS: solve large-scale convex cone problems
  - ECOS: solve second-order cone programming
  - SCIP: constraint integer programs (MIP and MINLP)

and more

Library and toolbox

provide a complete environment that includes both a way to model problems and built-in solvers

- **MATLAB:** uses optimization toolbox (`linprog, quadprog`) and provides a self-contained ecosystem
- **SciPy:** the `scipy.optimize` module includes basic solvers (like HiGHS for LP) and a functional interface
- **PyTorch/TensorFlow:** while primarily for AI, they are essentially *optimization libraries* for unconstrained problems using stochastic gradient descent.

CVXPY

- cvxpy is a modeling language that follows the rules of **Disciplined Convex Programming (DCP)**
- declarative modeling: we describe *what* the problem is using standard matrix algebra
- problem transformation: transform high-level math into a standard form that specialized solvers can understand
- solver independence: we can swap between different solver backends

Google OR tools

- solver interface (API): provides a bridge between high-level languages (Python, Java, C#) and low-level, high-performance C++ solvers like GLOP or CP-SAT
- multi-solver framework: a **wrapper** that can talk to many different types of engines. It has its own internal solvers but can also *plug in* to commercial ones like Gurobi or CPLEX
- procedural model building: unlike *declarative* modeling (where you describe what the math looks like), OR-Tools uses *procedural* building, where you tell the computer how to add each variable and constraint to the memory
- highly specialized for combinatorial and discrete optimization

Scope of problems solved by OR tools

specialized for combinatorial and discrete optimization

1. constraint programming (CP): handled by **CP-SAT** solver; problems with discrete variables and constraints can be non-linear or logical
2. linear programming (LP): solve by its internal **GLOP** engine or can interface with commerial solvers like Gurobi or CPLEX
3. mixed-integer programming (MIP)
4. bin packing and knapsack problems: packing items into containers with fixed capacities
5. vehicle routing (VRP): finding optimal routes for vehicles delivering to specified locations
6. graph and network flow: movement through a network

what is outside the scope of OR tools

- genenral nonlinear programming
- general convex programming

Optimization workflow and modeling

Workflow

1. get problem statement from an application of interest
2. explore all elements: variables, constraint, objective
3. acquire all problem parameters
4. **formulate** the problem (require prior knowledge about the application)
5. use modeling language to **solve** the problem with example of problem data
6. **interpret** the solution; if not reasonable, check the root causes:
   - formulation structure
   - problem parameter values
7. reformulate the problem if needed
8. **implement** to solve the optimization ; select appropriate solver or softwares

Importance of problem formulation

the main reason why we need optimization techniques because

- the objective and constraints give non-trivial info to find optimal solutions

$$\text{P1: min } 2x_1 + 3x_2 \text{ s.t. } x \succeq 0 \quad \text{P2: min } 2x_1 + 3x_2 \text{ s.t. } -5 \leq x_1 + x_2 \leq 5$$

- the objective has coupled terms in x

$$\text{min } (x_1 - 1)^2 + (x_2 - 3)^2 \quad \text{versus} \quad \text{min } 4x_1^2 - 2x_1x_2 + 6x_2^2$$

- the constraints have coupled terms in x

$$\text{min } (x_1 - 1)^2 + (x_2 - 3)^2 \quad \text{s.t. } |x_1 + x_2| \leq 3$$

(versus constraint: $0 \leq x_1 \leq 2$ and $0 \leq x_2 \leq 2$)

conclusion: the resulting formulation should be non-trival (avoid unbounded problem)

Importance of feasible set

when deriving constraints to be included in the problem, ensure that

- the feasible set is not empty (no point of optimizing in the first place)
- avoid redundant constraint functions

$$\text{C1: } |x| \leq 5 \quad \text{C2: } x^2 \leq 28$$

- equivalent constraint function is used when re-formulating

$$x_1^2 + x_2^2 \leq 3 \text{ is not equivalent to } |x_1|, |x_2| \leq \sqrt{3}$$

- if we have an equivalent feasible set from different constraint functions, choose a representation that is simple or suitable for the target problem type:

$$\text{C1: } |x| \leq 5 \text{ is equivalent to C2: } x^2 \leq 25$$

C1 can be framed as linear constraint (simpler), while C2 is quadratic

B2: Introductory examples in logistics optimization

Outline

Introduction to logistics optimization

Description

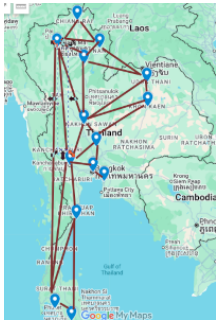
focus on the efficient movement of goods, the strategic placement of assets, and the minimization of operational costs

characteristics of problems:

1. combinatorial optimization: finding the best possible solution from a finite, but often overwhelming, set of options; in logistics, this usually means determining the best **ordering** or **sequencing** of stops to minimize distance or time
2. strategic or tactical planning: this group covers different layers of business decision-making

Traveling saleman problem (TSP)

TSP: problem statement



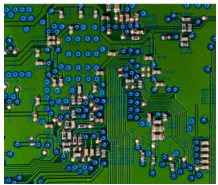
description:

- given n cities and the cost of every possible path from city j to i, $i = 1, 2, \ldots, n$
- visit n cities exactly once and return to the starting city
- pick the **shortest/cheapest** possible route

1. **variables:** $x_{ij} \in \{0, 1\}$ the status of path from city j to i
2. **objective:** minimize the total distance or cost associated with the path
3. **constraints:** each city has one coming edge and outgoing edge
4. **problem parameters:** cost of all connections

challenge: exact algorithm requires all possible combinations $(n!)$; impractical for $n \geq 20$

TSP replica: Printed Circuit Board (PCB) Drilling

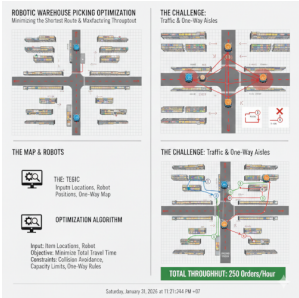


description:

- robotic drill must create hundreds of tiny holes on a circuit board at specific coordinates
- minimize the total distance the drill head travels to reduce manufacturing time and mechanical wear

1. **variables:** $x_{ij} \in \{0, 1\}$ the status of path from hole j to i
2. **objective:** minimize the total distance
3. **constraints:** each hole is drilled once; the drill starts and ends at home coordinate
4. **problem parameters:** coordinates of locations; distance of each pair of holes

TSP replica: Robotic warehouse picking

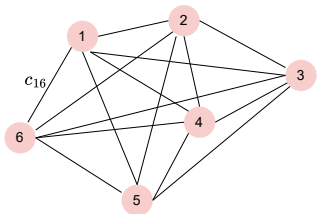


description:

- autonomous mobile robot (AMR) needs to collect items from n shelves to fulfill an order
- find the shortest loop (minimize time) to visit all shelves and return to the packing station
- unlike TSP, we can have many robots at the same time

1. **variables:** $x_{ij} \in \{0, 1\}$ the status of path from shelf j to i
2. **objective:** minimize the travel distance
3. **constraints:** for any location, only one robot can pick up and avoid collision; one-way aisle (robot only go forward) and many more constraints in real system
4. **problem parameters:** graph structure of shelves and distance

Structure of TSP



c_{ij} the cost from node j to i

problem parameters:

- a graph of n nodes
- a path from node j to i has a cost of c_{ij} (given as an adjacency matrix)
- graph can be directed (cost $c_{ij} \neq c_{ji}$) or undirected (symmetric)

- optimization solution is to determine $x_{ij} \in \{0, 1\}$ for all $1 \leq i \leq n$, where 1 if there is a path from j to i
- the cost c_{ij} is not limited to distance; can reflect the fuel, terrain nature, or the difficulty of that path

TSP replica

1. **power grid restoration:** a repair crew has 10 damaged substations or transformers that need to be inspected and fixed; minimize the total time the crew is on the road to restore powe
2. **seismic surveying:** companies need to collect data from thousands of geographic points (nodes) across difficult terrain; to predetermine *shot-points* where they either set off a small underground charge or use a *vibroseis* truck to send sound waves into the earth
   - the cost is a combination of fuel, time, terrain difficulty (steep hill and downhill have different cost)
   - find the shortest and most fuel-efficient sequence to visit all shot-point

TSP replica

1. **mineral exploration:** in remote areas (like the Australian Outback or the Canadian Shield), helicopters or off-road vehicles are used to move between sites; optimizing the flight path to drop off and pick up sampling teams at various outcrops to ensure maximum coverage before the sun sets or fuel runs out
2. **drone survey:** drones have limited battery life; cannot simply fly a standard lawnmower pattern; TSP: calculating a path that visits all high-priority survey points in a single battery charge

TSP: Mixed-integer linear program

Miller-Tucker-Zemlin (MTZ) formulation

variable: x_{ij} binary status of a path from node j to i

objective function:

$$f(x) = \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} c_{ij} x_{ij}$$

constraints:

- each city must be left and entered exactly once:

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1, \quad i \in [n]$$

- each city must be entered exactly once:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \quad j \in [n]$$

TSP: Mixed-integer linear program

- the above two constraints are not enough as it could allow subtours like $1 \to 2 \to 1$ and $3 \to 4 \to 3$ separately
- subtour elimination constraint: create a time-stamp at each node

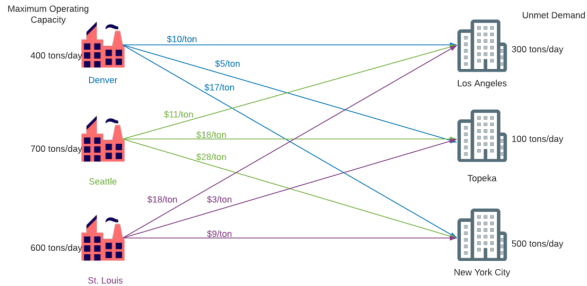
$$u_i - u_j + nx_{ij} \leq n - 1, \quad 2 \leq i, j \leq n, i \neq j$$

  (if $x_{ij} = 1$ the constraint is $u_j \geq u_i + 1$; this forces the sequence to be strictly increasing; impossible in a closed loop)
- variable domain: $u_i \geq 0$ for $i = 2, \ldots, n$

the problem can be hard to solve when n is large

Facility location problem

Basic description

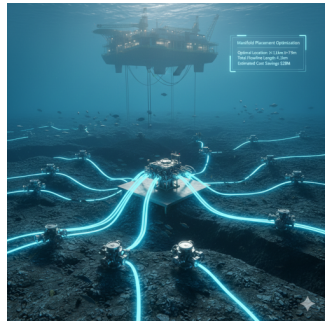


credit: `https://optimization.cbe.cornell.edu/`

given a set of potential facility locations and a set of demand points

- which facilities to open (incorporating a fixed **opening cost**)
- which demand points to assign to each open facility to minimize the **transportation cost**

Petroleum engineering: Manifold Placement



- in subsea oil field development, multiple wells are connected to a single **manifold** or gathering center
- find the geographic coordinates of the manifold that minimize the weighted sum of distances to all connected wellheads

Renewable energy: substation siting

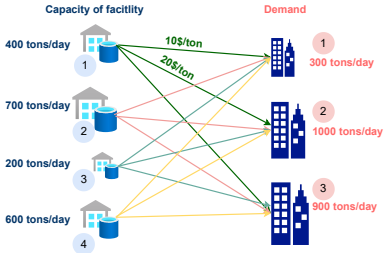
- in a large-scale solar or wind farm, power from individual turbines/panels must be collected at a substation before being sent to the main grid
- placing the substation at the **geometric center** of the turbines to minimize electrical transmission losses (I^2R losses)

Logistics: disaster response

- placing emergency supply depots (food, water, medicine) in a region prone to natural disasters
- pick the locations that minimize the maximum distance any citizen has to travel to reach the depot after an event

(Capacitated) Facility location problem

given n facility locations indexed by i and m clients to be serviced indexed by j



- each client j has a demand d_j to be filled by selected facilities
- u_i is the capacity of facility i
- f_i is a fixed cost of opening facility i, while there is a cost c_{ij} of serving client j from facility i (\$/demand unit)

1. **goal:** select a set of facility locations and assign each client to facilities that minimizes the total cost while meeting the targeted demand
2. **variables:** $x_i \in \{0, 1\}$ for $i = 1, 2, \ldots, m$ where $x_i = 1$ if facility i is open ; $y_{ij} \in \mathbf{R}$ represents the **fraction** of the demand d_j filled by facility i

(Capacitated) Facility location problem: formulation

minimize	$\sum_{i=1}^n f_j x_i + \sum_{i=1}^n \sum_{j=1}^m c_{ij} d_j y_{ij}$	
subject to	$y_{ij} \geq 0, \quad i = 1, \ldots, n, j = 1, \ldots, m$	fractions are non-negative
	$x_i \in \{0, 1\}, \quad i = 1, 2, \ldots, n$	facility i is either open or closed
	$\sum_{i=1}^n y_{ij} = 1, \quad j = 1, 2, \ldots, m$	fractions of facilities to serve j sum to 1
	$\sum_{j=1}^m d_j y_{ij} \leq u_i x_i, \quad i = 1, 2, \ldots, n$	max capacity of facility i
		force y_{ij} to be positive when $x_i = 1$

in matrix form: $C, Y \in \mathbf{R}^{n \times m}$ and $x, d, u \in \mathbf{R}^n$

- cost: $f^T x + \mathbf{1}^T (C \odot Y) d$
- constraints:

$$Y \geq 0, \quad x \in \{0, 1\}^n, \quad \mathbf{1}_n^T Y = \mathbf{1}_m, \quad Yd \leq u \odot x$$

(Uncapacitated) Facility location problem

when all facilities has ∞ capacity,

- capacitated: optimization solely selects the facility with the lowest cost
- change: modify $y_{ij} \in \{0,1\}$ where $y_{ij} = 1$ if customer j is served by facility i

minimize	$\sum_{i=1}^{n} f_j x_i + \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} d_j y_{ij}$	
subject to	$y_{ij} \in \{0,1\}, \;\; i = 1,\ldots,n, j = 1,\ldots,m$	binary value
	$x_i \in \{0,1\}, \;\; i = 1,2,\ldots,n$	facility i is either open or closed
	$\sum_{i=1}^{n} y_{ij} = 1, \;\; j = 1,2,\ldots,m$	to serve j only pick one facility i
	$y_{ij} \le x_i, \;\; i = 1,2,\ldots,n, \; j = 1,2,\ldots,m$	use facility i only if it's open
		force y_{ij} to be positive when $x_i = 1$

in matrix form: $C, Y \in \mathbf{R}^{n \times m}$ and $x, d \in \mathbf{R}^n$

- cost: $f^T x + \mathbf{1}^T (C \odot Y) d$
- constraints:

$$Y \in \{0,1\}^{n \times m}, \quad x \in \{0,1\}^n, \quad \mathbf{1}_n^T Y = \mathbf{1}_m, \quad Y \le [x \; x \; \cdots \; x]$$

Property of facility location problem

- a mixed integer linear programming (MILP)
- no of binary variable grows with number of clients
- fraction of demand filled by each facility grows with number of clients multiplied with number of locations
- can be solved by `cvxpy` or `OR tools`

Facility location: Petroleum engineering

an oil company identifies n sites for Gathering Centers (GC) and has m active oil fields

```
# 1. PROBLEM DATA
# Fixed costs for building 4 potential sites (Millions of USD)
f = np.array([20.0, 25.0, 18.0, 30.0])

# Capacities of the 4 sites (kbpd) kilo barrel per day
u = np.array([50.0, 60.0, 45.0, 80.0])

# Production Demands of 6 Oil Fields (kbpd)
d = np.array([15.0, 22.0, 18.0, 12.0, 25.0, 10.0])

# Transport Costs ($/bbl) from 4 Sites to 6 Fields
C = np.array([
    [1.20, 2.50, 3.10, 4.00, 5.20, 6.00],
    [3.50, 1.10, 2.00, 3.20, 4.10, 5.00],
    [5.00, 4.20, 1.50, 1.00, 2.80, 3.50],
    [6.50, 5.50, 4.00, 2.10, 1.20, 0.90]])
```

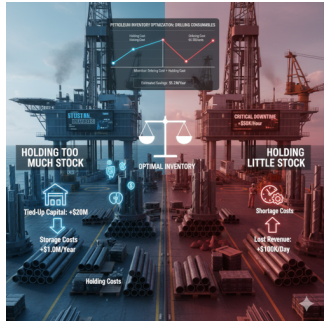
- minimize the total cost (fixed setup + transport costs) while serving the production demand
- find which site to be open and fraction allocation from each site to each oil field

Inventory optimization

Basic description

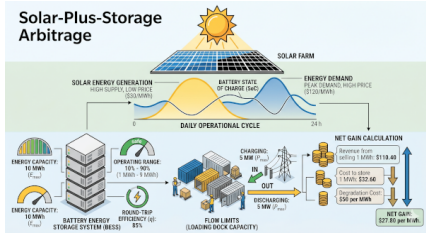
- the process of determining the right amount of stock to hold to meet demand while minimizing the total costs associated with ordering, holding, and shortages
- it depends on accurate demand forecasting, or predicting the quantity of goods that customers will purchase in the future

Petroleum engineering: drilling consumables



- a drilling rig in a remote location (offshore or desert) requires a constant supply of drill bits and casing pipes
- balancing the high cost of shipping equipment to a remote site (ordering cost) against the cost of storing bulky steel pipes on a space-constrained offshore platform (holding cost)

Renewable energy: battery storage



- in a wind or solar farm, energy (MWh) is an intangible inventory
- optimize the **state of charge** to ensure they have enough **stock power** to meet demand when the sun goes down, without building an unnecessarily large and expensive battery array
- MWh in battery are products on the shelf
- renewable generation is the supplier while grid/load is the customer

Costs involved in inventory models

1. **ordering and setup cost:** paperwork, billing, setting up/shutting down a machine
2. **unit purchasing cost:** variable cost associated with purchasing a single unit
3. **holding cost:** cost of carrying one unit of inventory for a time period ($/unit/year)
4. **stockout or shortage cost:** demands can by back-ordered (customer accept delay delivery) or lost sales; both cases incur a cost

assumptions on the classsic **Economic Order Quantity (EOQ) model**

1. repetitive ordering: ordering decision is repeated regularly
2. constant demand: demand occurs at a known, constant rate
3. constant lead time L: when an order is placed to the order arrives
4. continuous ordering: an order may be placed at any given time

Basic EOQ model

assumptions:

1. D: annual demand (units/year) occurs at a constant rate
2. K: setup cost (\$/order)
3. h: holding cost per unit per year (\$/unit/year)
4. q: order quantity (the decision variable)
5. p: per-unit purchasing cost (\$/unit)
6. I: inventory level (no. of units)
7. the lead time for each order is zero
8. no shortages are allowed

observations:

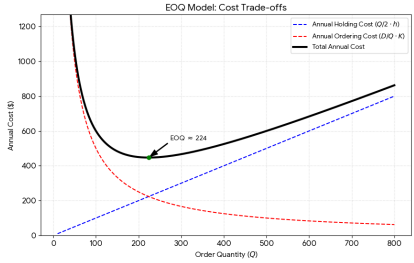
- we should never place an order when $I > 0$ (as it incurs a holding cost)
- we must place an order when $I = 0$ (to prevent a shortage from occuring)
- determine q units are ordered each time that $I = 0$ that minimizes the annual cost

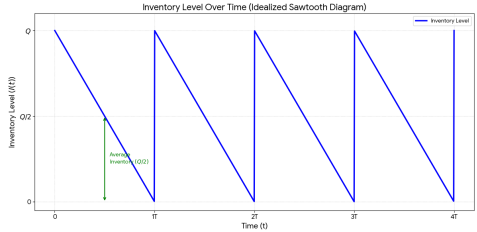
Objective function and optimal quantity

total annual cost: sum of ordering cost, purchasing cost and holding cost

$$\mathrm{TC}(q) = \frac{D}{q}K + pD + \frac{q}{2}h$$

(I drops linearly from q to 0, the average inventory is $q/2$)





Optimal order quantity

- TC is convex so the stationary point minimizes TC

$$\frac{d\text{TC}}{dq} = -\frac{DK}{q^2} + \frac{h}{2} = 0$$

- solving q gives the EOQ formula:

$$q^\star = \sqrt{\frac{2DK}{h}} \quad \text{(units)}, \quad \text{total of } \frac{D}{q^\star} \text{ orders must be placed each year}$$

- observe effects of K and h on $q^\star$
  - setup cost K increases then $q^\star$ increases
  - holding cost h increases then $q^\star$ reduces
  - if both K and h are doubled, $q^\star$ remains unchanged
- can be shown that if EOQ is ordered then holding cost $=$ ordering cost (per year)

Example: Two-item inventory with storage constraint

setting:

- decide the quantities of drill bits (q_1) and casing pipes (q_2) to order
- in offshore drilling, we have the physical deck space: total area of $A = 5,000$ sq.ft

parameter	symbol	drill bits	casing pipes
annual demand	D_i	120 units	2,400 joints
ordering cost (per shipment)	K_i	\$5,000	\$15,000
holding cost (per unit/year)	h_i	\$300	\$1,200
storage area	a_i	2 sq.ft	40 sq. ft

optimization problem:

$$\begin{aligned} \text{minimize} \quad & \text{TC(q)} = \textstyle\sum_{i=1}^{2} \left(\frac{D_i}{q_i} K_i + \frac{q_i}{2} h_i \right) \\ \text{subject to} \quad & a_1 q_1 + a_2 q_2 \leq A \end{aligned}$$

try EOQ solution $q_i = \sqrt{\frac{2D_iK_i}{h_i}}$ and find that it is infeasible; let's solve numerically

Example: Two-item inventory with storage constraint

- the problem is nonlinear programming (NLP) with two variables
- the problem is convex ($1/q$ is convex when $q \succ 0$)
- a constrained problem; define Lagrangian and multiplier

$$L(q_1, q_2, \lambda) = \sum_{i=1}^{2} \left(\frac{D_i}{q_i} K_i + \frac{q_i}{2} h_i \right) + \lambda \left(\sum_{i=1}^{2} a_i q_i - A \right)$$

- optimality conditions: zero gradient of L and complementary slackness

$$q_i(\lambda) = \sqrt{\frac{2 D_i K_i}{h_i + 2\lambda a_i}}, \qquad \lambda \cdot \left(A - \sum_{i=1}^{2} a_i q_i \right) = 0$$

- solved by `scipy.optimize` or `cvxpy`

Storage arbitrage as inventory optimization

maximize revenue by storing energy during the day (low price/high supply) and selling it during the evening peak (high price/low supply), while managing the holding cost (battery degradation)

inventory	BESS equivalent	parameter
warehouse capacity	energy capacity	$E_{\max}$ (MWh)
stock level	state of charge (SoC)	S_t (MWh or %)
restock rate	charging power	$P_{\text{chg},t}$ (MW)
sale rate	discharging power	$P_{\text{dchg},t}$ (MW)
perishability	self-discharge/loss	$\eta_{\text{chg}}, \eta_{\text{dchg}}$ (efficienty)
holding cost	degradation cost	C_{deg} ($/MWh cycled)
external driver	electricity market price	Price_t ($/MWh)

- price is the signal that dictates the flow of the inventory
- the battery is the warehouse and the energy is the stock
- the price acts the external driver that determines when the warehouse should open its doors to restock (charge) or sell (discharge)

Storage arbitrage

variables: $P_{\text{chg},t}, P_{\text{dchg},t}, S_t$

objective function: maximize profit over a 24-hour horizon ($T = 24$)

$$\text{maximize} \quad \Delta t \sum_{t=1}^{T} \text{Price}_t(P_{\text{dchg},t} - P_{\text{chg},t}) - C_{\text{deg}}(P_{\text{chg},t} + P_{\text{dchg},t})$$

constraints:

1. inventory balance: SoC dynamic

$$S_{t+1} = S_t + \left(\eta_{\text{chg}} P_{\text{chg,t}} - \frac{P_{\text{dchg,t}}}{\eta_{\text{dchg}}}\right) \Delta t$$

2. storage limit (physical warehouse): battery has operating range

$$S_{\min} \leq S_t \leq S_{\max}$$

3. flow limit (loading dock capacity): inverters limit how fast we can move

$$0 \leq P_{\text{chg,t}} \leq P_{\max}, \quad 0 \leq P_{\text{dchg,t}} \leq P_{\max}$$

A1: LP-QP-MILP

Outline

Linear programming

- math standard setting
- feasibility, optimality
- common formulation techniques

Linear programming

a general linear program (LP) has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

where $G \in \mathbf{R}^{m \times n}$ and $A \in \mathbf{R}^{p \times n}$

- n optimization variables: $x = (x_1, \ldots, x_n) \in \mathbf{R}^n$
- the objective function: $c^T x = \sum_{i=1}^n c_i x_i$
- the inequality constraint: $\sum_{j=1}^n g_{ij} x_j \leq h_i$ for $i = 1, 2, \ldots, m$
- the equality constraint: $\sum_{j=1}^n a_{ij} x_j = b_i$ for $i = 1, 2, \ldots, p$
- the objective function and constraint functions are *linear* in x

called **linear program** (LP) or **linear optimization problem**

Another standard form

LP can also be represented in another form

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

using the facts that

- any $x \in \mathbf{R}$ can be written $x = x^+ - x^-$
- $a^T x \leq b \iff a^T x + s = b, \quad s \succeq 0$

note: we assume A is fat and has full row rank

exercise: transform into the two general forms

$$\begin{array}{ll} \text{minimize} & 2x_1 - x_2 + x_3 \\ \text{subject to} & -3x_1 + x_2 - 5x_3 \leq 3 \\ & 2x_2 + 7x_3 \geq 10 \\ & 3x_2 + 4x_3 = 2 \end{array}$$

Mixed integer programming

if $x \in \mathbf{Z}^n$ (integers) the problem on page 215 is called an **integer linear programming (ILP)**

if some components of x are integers and some are real numbers, the problem is called a **mixed integer linear programming**

examples of integer programing:

- x represents quantities, countable units (pieces)
- number of sale products
- number of persons assigned on a work schedule
- $x \in \{0, 1\}$: **binary integer programming**
- x is status of a functioning unit in factory, '1' is on, '0' is off

Geometrical interpretation

- hyperplane: solution set of a linear equation with coefficient vector $a \neq 0$

$$\{x \mid a^T x = b\}$$

- halfspace: solution set of a linear inequality with coefficient vector $a \neq 0$

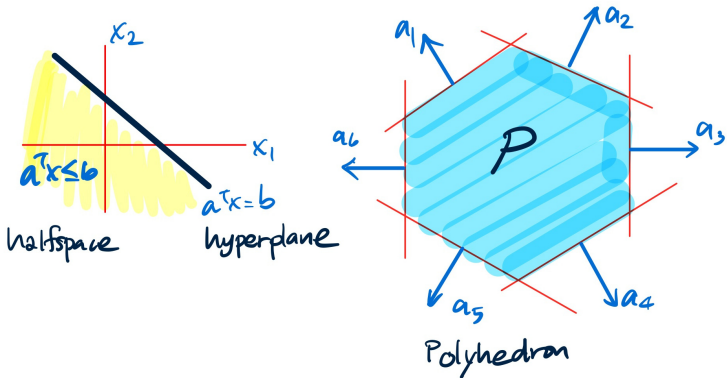
$$\{x \mid a^T x \leq b\}$$

  we say a is the **normal vector**

- polyhedron: solution set of a finite number of linear inequalities

$$\{x \mid a_1^T x \leq b_1, \;\; a_2^T x \leq b_2, \;\; \ldots, \;\; a_m^T x \leq b_m\} = \{x \mid Ax \preceq b\}$$

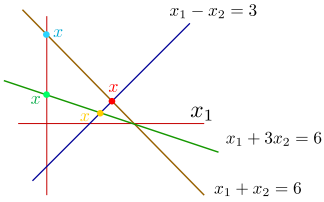
  intersection of a finite number of halfspaces



extreme point of $\mathcal{C}$

a vector $x \in \mathcal{C}$ is an extreme point (or a vertex) if we cannot find $y, z \in \mathcal{C}$ both different from x and a scalar $\alpha \in [0, 1]$ such that $x = \alpha y + (1 - \alpha)z$

Solving LPs graphically



$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & x_1 + x_2 \leq 6 \\ & x_1 - x_2 \leq 3 \\ & x_1 + 3x_2 \geq 6 \\ & x_1, x_2 \geq 0 \end{aligned}$$

- the gradient of $c^T x$ is c
- the objective increases along direction c
- the level set: $\{x | c^T x = \alpha\}$ is the line $\perp$ to c

- feasible set is a bounded polyhedron ✎
- find the level set: $c^T x = \alpha$ with smallest α that intersects with the feasible set
- find the optimal solution for a given c

$$c = (0, 1), x^\star = \quad c = (-1, 0), x^\star = \quad c = (-1, 1), x^\star = \quad c = (1, 3), x^\star =$$

- solving graphically is for conceptual understanding; for 2-dimensional problems

Simple linear programs

minimize $c^T x$ over each of these simple sets

we can derive an explicit solution of these LPs

- **box constraint:** $l \preceq x \preceq u$
- **probability simplex** (or budget allocation): $\mathbf{1}^T x = 1, x \succeq 0$
- **not all budget is used:** $\mathbf{1}^T x \leq 1, x \succeq 0$
- **halfspace:** $a^T x \leq b$

draw the constraint set and inspect the solution for a given c

Properties

refer to the standard form on page 86

- an LP may not have a solution (constraints are inconsistent or the feasible set is unbounded)
- we assume A is full row rank; if not, considering $Ax = b$
  - depending on A, the system could be inconsistent (hence, no extreme points), or
  - $Ax = b$ contains redundant equations, which can be removed
- if a standard LP has a finite optimal solution then

  a solution can always be chosen from among the vertices of the feasible set

  (called **basic feasible solutions**)
- the dual of an LP is also an LP
- solutions of some simple LPs can be analytically inspected

Common patterns in modeling

- piecewise linear functions
- problems involving absolute values
- problems involving ℓ_1 or ℓ_∞ norms

these problems can be cast as an LP

Problems involving absolute values

using the properties of $|\cdot|$,

linear inequality of absolute-valued function can be cast as linear inequality

- for $x \in \mathbf{R}$, $\quad |x| \le u \iff -u \le x \le u$
- for $x \in \mathbf{R}^n$, $\quad \sum_i^n |x_i| \le t \iff \exists u \in \mathbf{R}^n,\ -u \preceq x \preceq u,\ \mathbf{1}^T u \le t$
- ℓ_1**-norm:** $\|Ax-b\|_1 \le t \iff \exists u \in \mathbf{R}^n,\ -u \preceq Ax-b \preceq u,\ \mathbf{1}^T u \le t$

minimization of a function involving $|\cdot|$ can be cast as an LP (in variables x, u)

$$\text{minimize}_x \quad \|Ax-b\|_1 \iff \begin{array}{ll} \text{minimize}_{x,u} & \mathbf{1}^T u \\ \text{subject to} & -u \preceq Ax-b \preceq u \end{array}$$

Problems involving max of linear functions

recall the fact that

$$\max(f_1(x), f_2(x)) \leq t \quad \Longleftrightarrow \quad f_1(x) \leq t, \;\; f_2(x) \leq t$$

max of linear functions can be reformulated as linear inequalities

- for $x \in \mathbf{R}^n$, $\quad \max(0, x) \leq t \quad \Longleftrightarrow \quad 0 \preceq t, \;\; x \preceq t\mathbf{1}$
- **piece-wise linear function:**

$$\max(a_1^T x + b_1, a_2^T x + b_2, a_3^T x + b_3) \leq t \quad \Leftrightarrow \quad a_i^T x + b_i \leq t, \;\; i = 1, 2, 3$$

- ℓ_∞**-norm:**

$$\|x\|_\infty \leq t \;\; \Leftrightarrow \;\; \max_i |x_i| \leq t \;\; \Leftrightarrow \;\; |x_i| \leq t, \; \forall i, \;\; \Leftrightarrow \;\; -t\mathbf{1} \preceq x \preceq t\mathbf{1}$$

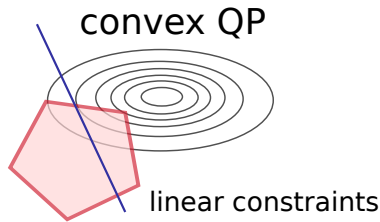
Quadratic Programming

Quadratic Programming

a **quadratic program (QP)** is in the form

$$\begin{array}{ll} \text{minimize} & (1/2)x^T Px + q^T x \\ \text{subject to} & Gx \preceq h \\ & Ax = b, \end{array}$$

where $P \in \mathbf{S}^n, G \in \mathbf{R}^{m\times n}$ and $A \in \mathbf{R}^{p\times n}$



example: constrained least-squares

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2^2 \\ \text{subject to} & l \preceq x \preceq u \end{array}$$

QP has **linear** constraints

Properties of QP

- an unconstrained QP is unbounded below if P is not positive definite
- an unconstrained QP has a unique solution: $x = -P^{-1}q$ when $P \succ 0$
- a QP is a convex problem if P is positive semidifinite definite
  - if $P \succeq 0$ then a local minimizer $x^\star$ is a global minimizer (by convexity)
  - if $P \succ 0$ then $x^\star$ is a *unique* global solution (by strictly convexity)
- the feasible set (polyhedron) may be empty (hence, the problem is infeasible)
- the feasible set can be unbounded (but if $P \succ 0$ it implies boundedness)
- solution of a QP may not be at a vertex
- the dual of a QP is also a QP

Mixed integer programming

Mixed integer linear programming

a general mixed integer linear programming (MILP) has two general forms

$$\begin{array}{ll} \text{minimize} & c^T x + d^T y \\ \text{subject to} & Ax + By = b \\ & x, y \succeq 0, \\ & x \text{ is integer} \end{array} \qquad \begin{array}{ll} \text{minimize} & c^T x + d^T y \\ \text{subject to} & Gx + Fy \preceq h \\ & Ax + By = b \\ & x \text{ is integer} \end{array}$$

- $x = (x_1, \ldots, x_n)$ where $x_i \in \mathbf{Z}$ (integer variables)
- $y = (y_1, \ldots, y_p) \in \mathbf{R}^p$ (real-valued variables)
- the objective function: $c^T x + d^T y = \sum_{i=1}^n c_i x_i + \sum_{i=1}^p d_i x_i$
- the objective function and constraint functions are *linear* in x and y

Common patterns in modeling

- binary choice (zero-one knapsack)
- forcing constraints (facility location problem)
- relation between variables
- disjunctive constraints
- restricted range of values
- arbitrary piecewise linear functions (set packing, set covering)
- sequencing problem with setup times

Binary choice

scenario: when to encode a choice between two alternativers, use binary variables

zero-one knapsack problem

$$\text{maximize } c^T x \text{ subject to } w^T x \leq K, \quad x_j \in \{0,1\}, \quad j = 1, 2, \ldots, n$$

setting:

- given n items; each has weight w_j and value c_j
- given a bound K on the total weight that can be carried in a knapsack

goal: decide to select items to maximize the total value

Forcing constraints

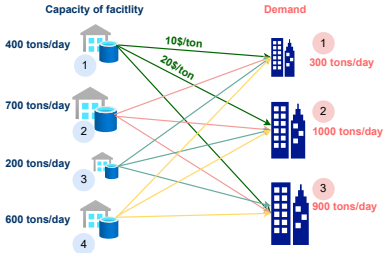
scenario: some variables are dependent; decision A can be made only if decision B has been made

modeling:

- introduce binary variables x, y corresponding to decision A and B, respectively
- set x to equal to 1 if A is chosen and 0 otherwise
- the dependence of the two decisions can be modeled using the constraint: $x \leq y$
- if $y = 0$ (decision B is not made), then $x = 0$ (decision A is not made)

(Capacitated) Facility location problem

given n facility locations indexed by i and m clients to be serviced indexed by j



- each client j has a demand d_j to be filled by selected facilities
- u_i is the capacity of facility i
- f_i is a fixed cost of opening facility i, while there is a cost c_{ij} of serving client j from facility i (\$/demand unit)

1. **goal:** select a set of facility locations and assign each client to facilities that minimizes the total cost while meeting the targeted demand
2. **variables:** $x_i \in \{0, 1\}$ for $i = 1, 2, \ldots, m$ where $x_i = 1$ if facility i is open ; $y_{ij} \in \mathbf{R}$ represents the **fraction** of the demand d_j filled by facility i

(Capacitated) Facility location problem: formulation

minimize	$\sum_{i=1}^n f_j x_i + \sum_{i=1}^n \sum_{j=1}^m c_{ij} d_j y_{ij}$	
subject to	$y_{ij} \geq 0, \quad i = 1, \ldots, n, j = 1, \ldots, m$	fractions are non-negative
	$x_i \in \{0, 1\}, \quad i = 1, 2, \ldots, n$	facility i is either open or closed
	$\sum_{i=1}^n y_{ij} = 1, \quad j = 1, 2, \ldots, m$	fractions of facilities to serve j sum to 1
	$\sum_{j=1}^m d_j y_{ij} \leq u_i x_i, \quad i = 1, 2, \ldots, n$	max capacity of facility i
		force y_{ij} to be positive when $x_i = 1$

in matrix form: $C, Y \in \mathbf{R}^{n \times m}$ and $x, d, u \in \mathbf{R}^n$

- cost: $f^T x + \mathbf{1}^T (C \odot Y) d$
- constraints:

$$Y \geq 0, \quad x \in \{0, 1\}^n, \quad \mathbf{1}_n^T Y = \mathbf{1}_m, \quad Yd \leq u \odot x$$

Either-Or

we want at least one of the two constraints is satisfied

$$f(x_1, \ldots, x_n) \leq 0, \quad g(x_1, \ldots, x_n) \leq 0$$

we can introduce $y \in \{0, 1\}$ and a big number M and force

$$f(x_1, \ldots, x_n) \leq My, \quad g(x_1, \ldots, x_n) \leq M(1 - y)$$

If-Then constraints

we want to ensure that

$$f(x) > 0 \quad \text{implies} \quad g(x) \geq 0$$

- introduce $y \in \{0, 1\}$ and a big number M_f

$$f(x) \leq M_f \cdot y \quad \text{if } f(x) > 0 \text{ then } y \text{ must be 1}$$

(if $f(x) \leq 0$ then y can be 0 or 1)

- link the 'then' part: $y = 1 \Rightarrow g(x) \geq 0$

$$g(x) \geq -M_g(1 - y) \quad \text{if } y = 1 \text{ then } g(x) \geq 0$$

(if $y = 0$ this constraint does nothing as $-M_g$ is very small)

in conclusion, we formulate

$$f(x) \leq M_f y, \quad g(x) \geq -M_g(1 - y)$$

Fixed-charge problem

scenario: there is a cost associated with performing an activity at a nonzero level that does not depend on the level of activity

a fixed cost of operating machine A when there is an order of product A (but the cost does not depend on the units of A)

modeling:

- introduce binary variable y and let $x \in \mathbf{R}$ be the level/number of units of such activity
- goal: set y to 1 whenever x is nonzero
- use the constraint: $x \leq My$ where M should be set equal to the maximum of x that can attain

Relations between variables

scenario: when at most one of binary variables can be 1

modeling:

- at most one of the variables x_j can be one

$$\mathbf{1}^T x \leq 1$$

- exactly one of the variables x_j should be one

$$\mathbf{1}^T x = 1$$

Disjunctive constraints

scenario: given two constraints $a^T x \geq b$ and $c^T x \geq d$ where $a, c \succeq 0$, we would like to have at least one of the two constraints are satisfied

modeling: define a binary y and impose

$$a^T x \geq yb, \quad c^T x \geq (1-y)d, \quad y \in \{0,1\}$$

general scenario: given m constraints: $a_i^T x \geq b_i$, for $i = 1, \ldots, m$ where $a_i \succeq 0$, we require at least K of them are satisfied

modeling:

$$a_i^T x \geq b_i y_i, \;\; i = 1, 2, \ldots, m, \quad \mathbf{1}^T y \geq K, \;\; y_i \in \{0,1\}$$

Restricted range of values

scenario: aim to restrict x to take values in a set $\{a_1, a_2, \ldots, a_m\}$

modeling: define $a = (a_1, a_2, \ldots, a_m)$ and a binary variable $y \in \mathbf{R}^m$ such that

$$x = a^T y, \quad \mathbf{1}^T y = 1, \quad y_j \in \{0, 1\}$$

Constraint Programming

Constraint programming

constraint programming (CP) is based on identifying feasible solutions, rather than minimizing some objective

CP may or may not have an objective function

examples:

- employee scheduling: create work schedules for employees over a horizon
- job shop problem: design schedule of multiple jobs processed on several machines

CP problems can be solved by CP-SAT solver in Google OR-Tools

Nurse scheduling problem

a problem of finding an optimal way to assign employees to shifts with a set of hard constraints

employee scheduling problems are defined by three key elements:

1. **resources:** the *entities* being scheduled; here, the nurses/employees
2. **tasks/slots:** time intervals needed to be filled; day and night shifts over an N-day horizon
3. **constraints:** rules of scheduling
   - capacity constraint: (3 people per shift)
   - temporal constraint: (no double-shifts, 12-hour recovery after nights)
   - distribution constraint: (min 3/ max 5 shifts per week)

scheduling is often treated as a **constraint satisfaction problem (CSP)**

unlike trying to minimize an objective, the primary challenge is simply finding *any* valid schedule that does not violate constraints

solver CP-SAT in Google OR-Tools can be used to solve this problem

Numerical tools

Solver overview (1)

Solver	LP	QP	MILP	Convex	w/ cvxpy	w/ OR-Tools	Notes
Gurobi	Yes	Yes	Yes	Yes	Yes	Yes	Leading commercial solver; offers the highest performance across all problem types
CPLEX	Yes	Yes	Yes	Yes	Yes	Yes	IBM's high-performance commercial solver; a standard in industrial-scale optimization
HiGHS	Yes	Yes	Yes	Partial	Yes	Yes/installed	A high-performance, modern open-source software for sparse linear optimization; support convex QP, not SOCP; currently, not implemented with solving QP within MathOpt interface in OR-Tools
SCIP	Yes	Yes	Yes	Yes	Yes	Yes/installed	A powerful global optimizer that handles MINLP and non-convex problems; fast non-commercial solvers available

Solver overview (2)

Solver	LP	QP	MILP	Convex	w/ cvxpy	w/ OR-Tools	Notes
ECOS	Yes	Yes	No	Yes	Yes/installed	No	A lightweight high precision interior-point solver specialized for LP, QP, SOCP
SCS	Yes	Yes	No	Yes	Yes/installed	No	ADMM (splitting methods); solve very large problems to moderate accuracy; for all conic problems
OSQP	Yes	Yes	No	Yes	Yes/installed	Yes	the **operator splitting QP** solver; currently the default and fastest open-source choice for convex QP
Clarabel	Yes	Yes	No	Yes	Yes/installed	No	all conic types; robust, modern implementation with strong numerical stability

Solver overview (3)

Solver	LP	QP	MILP	Convex	w/ cvxpy	w/ OR-Tools	Notes
PDLP	Yes	Yes	No	Yes	No	Yes/installed	**Google's primal-dual hybrid gradient** solver; built for massive-scale LP and convex diagonal QP
GLOP	Yes	No	No	No	No	Yes/installed	**Google's primary first-party LP solver**; highly optimized for speed in pure LP tasks
CP-SAT	Yes	No	Yes	No	No	Yes/installed	**Google's flagship solver of OR-tools**; a hybrid solver using Lazy Clause Generation that combines Constraint Programming, SAT, and LP; world-class for combinatorial problems

Solver overview (4)

Solver	LP	QP	MILP	Convex	w/ cvxpy	w/ OR-Tools	Notes
CBC	Yes	No	Yes	No	Yes	Yes	open-source MILP solver from the COIN-OR project; robust for discrete linear problems
GLPK	Yes	No	No	No	Yes	Yes	part of the GNU project; a reliable, widely-used engine for pure LP
GLPK_MI	Yes	No	Yes	No	Yes	No	MILP interface for GLPK
CVXOPT	Yes	Yes	No	Yes	Yes	No	legacy Python-based library for convex optimization that uses interior-point methods

A2: Geoscience Applications

Outline

Examples on Geoscience

Example list

- ℓ_1- and ℓ_∞-norm approximation
- capturing geologic fold
- gas production allocation
- asset portfolio risk management
- rig scheduling
- pipeline network design
- fuel distribution

some compact notations: we always deal with vectors $x \in \mathbf{R}^n$

- for an integer n, $[n] = \{1, 2, \ldots, n\}$
- $c \odot x$ (Hadamard product) is a vector of $(c_1 x_1, \ldots, c_n x_n)$
- $c^T x$ is a scalar returning $\sum_{i=1}^n c_i x_i$
- $\mathbf{1}^T x$ is a scalar returning $\sum_{i=1}^n x_i$

ℓ_1-norm and ℓ_∞-norm approximations

given $A \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^m$

- ℓ_1-norm approximation: minimize $\|Ax - b\|_1$

  equivalent LP:

  $$\begin{array}{ll} \text{minimize} & \mathbf{1}^T u \\ \text{subject to} & -u \preceq Ax - b \preceq u \end{array}$$

  with variable x and auxiliary variable u

- ℓ_∞-norm (or Chebyshev) approximation: minimize $\|Ax - b\|_\infty$

  equivalent LP:

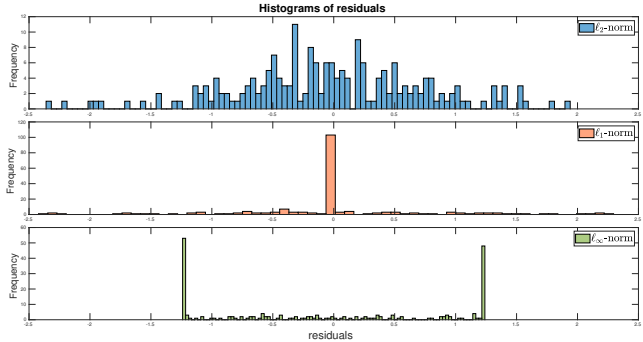
  $$\begin{array}{ll} \text{minimize} & t \\ \text{subjec to} & -t\mathbf{1} \preceq Ax - b \preceq t\mathbf{1} \end{array}$$

  with variable x and auxiliary variable t

ℓ_1- and ℓ_∞-norm approximation results

compare histograms of residuals $Ax - b$ for

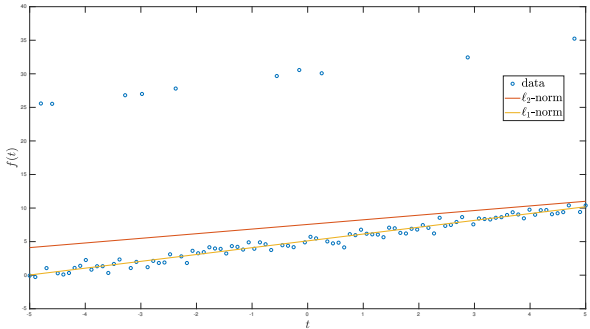
$$x_{\text{ls}} = \operatorname{argmin} \|Ax - b\|_2, \quad x_1 = \operatorname{argmin} \|Ax - b\|_1, \quad x_\infty = \operatorname{argmin} \|Ax - b\|_\infty$$



example of $A \in \mathbf{R}^{200\times 100}$: residuals of 1-norm approximation is concentrated at zero

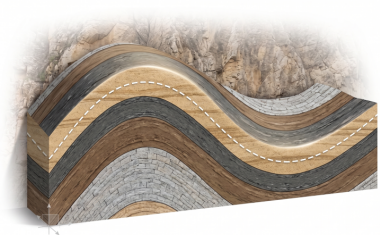
Estimation with outliers

fitting $f(t) = \alpha + \beta t$ to data containing 10% outliers



- ℓ_2-norm approximation tends to reduce large residuals occurred from outliers
- ℓ_1-norm has less penalty than ℓ_2 when residuals are large; it is more robust to outliers

Capturing geologic fold



the surface of a reservoir is rarely a flat plane

setting:

- to capture the actual curvature of rock layers, geoscientists represent the depth y as a function of geographic coordinates (easting d_e and northing d_n)

$$\hat{y} \approx x_0 + x_1 d_e + x_2 d_n + x_3 d_e^2 + x_4 d_n^2 + x_5 d_e d_n$$

- data y may contain noisy outliers (from seismic reflection and well-log data)

goal: estimate parameter x that gives $\hat{y} \approx y$ while avoiding sensitivity to outliers

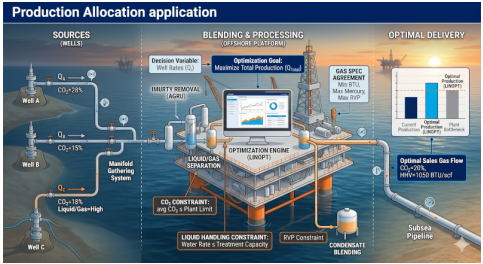
Capturing geologic fold

modeling: apply ℓ_1-norm approximation where $x = (x_0, x_1, x_2, x_3, x_2)$

$$\hat{y} = \begin{bmatrix} 1 & d_e & d_n & d_e^2 & d_n^2 & d_e d_n \end{bmatrix} x$$

- note that $\hat{y}$ is nonlinear in coordinate data but linear in x (parameter)
- frame as $\hat{y} = Ax$ and minimize $\|y - Ax\|_1$ to suppress the influence from outliers
- treat the problem as a convex, or reformulate as an LP (this lecture)

Gas production allocation



- n wells production gas with different CO_2
- gas must be blended before it enters the pipeline to meet the Sales Gas Agreement (SGA)

- **variables:** flow rate from each well, choke size (discrete), well status (on/off)
- **objective:** maximize total gas production (revenue)
- **constraints:**
  - CO_2 from all wells must be limited
  - heating value (HHV): gas must be hot enough but not so rich
  - liquid handling: limit how much gas you can draw from each well
  - daily contract quantity (DCQ): deliver a minimum amount to avoid penalty

Gas production allocation

assume no binary well status variable

- **variables:** flow rate $q \in \mathbf{R}^n$ (unit = MMscf/day)– Million standard cubic feet
- **objective:** maximize total revenue minus cost

$$\text{revenue} = \sum_{i=1}^{n} p_i q_i = p^T q, \quad p = \text{price} = \text{gas price} + \text{CGR} \cdot \text{condensate price}$$

$$\text{opex cost} = \sum_{i=1}^{n} \text{opex}_i q_i$$

$$\text{objective} = f(q) = (p - \text{opex})^T q \triangleq c^T q \qquad (\text{unit} = \$)$$

notes:

1. price coeffcients vary with wells
2. gas contains condensate (light oil) and has higher value than dry gas
3. the cost may additionally contain some penalty on impurities

Gas production allocation: Constraints

constraints:

1. $0 \preceq q \preceq q_{\max}$ (well capacity)
2. $\mathbf{1}^T q \leq$ capacity (total volume has a limit)
3. a limit for total mass of CO_2

$$\frac{\sum_{i=1}^n \mathrm{CO}_{2,i} q_i}{\sum_{i=1}^n q_i} \leq \mathrm{LCO2} \quad \Longrightarrow \quad (\mathrm{CO}_{2,i} - \mathrm{LCO2})^T q \leq 0$$

4. liquid handling: $\sum_{i=1}^n \mathrm{WGR}_i q_i \leq \mathrm{LW}$ WGR = water gas ratio
5. SGA: Heating value must be in a range

$$\mathrm{HHV}_{\min} \leq \frac{\sum_{i=1}^n \mathrm{HHV}_i q_i}{\sum_{i=1}^n q_i} \leq \mathrm{HHV}_{\max} \quad \Longrightarrow \quad \begin{array}{l} (\mathrm{HHV} - \mathrm{HHV}_{\min})^T q \geq 0 \\ (\mathrm{HHV} - \mathrm{HHV}_{\max})^T q \leq 0 \end{array}$$

Gas production allocation: Impurities

effect: increasing q makes the pressure drop near the wellbore increases and cause:

- water coning: higher rate pull water into the well, increasing WGR
- composition change: higer flow may draw gas from different layers that have higher CO_2

impurity modeling:

- concentration of impurity: % CO_2 or bbl/MMscf of water
  - each well has its own impurity model (assume linear)

$$c_i(q) = a_i q_i + b_i, \quad i = 1, 2, \ldots, n$$

  - a_i is the sensitivity of impurity to the flow rate
  - b_i is the baseline impurity level at minimum flow
- total impurity function: can be put in the objective or as constraints

$$g(q) = \sum_{i=1}^{n} c_i(q) q_i = \sum_{i=1}^{n} a_i q_i^2 + b_i q_i = q^T A q + b^T q \quad \text{(quadratic function)}$$

(unit = MMscf of CO_2 per day, or bbl/d for water)

Gas production allocation: Impurities

impurity model as **constraint:** replace the limit of CO_2 constraint with

$$g(q) = q^T Aq + b^T q \leq \text{Limit} \sum_{i=1}^{n} q_i$$

(total CO_2 mass entering must be $\leq$ a limit multiplied by the total volume of gas)

impurity model as **objective:** minimize

$$\text{cost of impurity} = \lambda \cdot g(q) = \lambda(q^T Aq + b^T q)$$

(λ is the penalty coefficient: unit = \$ per MMscf of CO_2

Gas production allocation: Optimizations

net cost = OPEX cost - revenue

default constraints = defined on page 131

1. P1: minimize net cost with default constraints **LP**
2. P2: minimize net cost + impurity cost with default constraints **QP**
3. P3: minimize net cost with default + impurity constraints **QCQP**

Asset portfolio risk management

setting:

- n wells with different risk profiles and value; some has low decline, low risk but expensive to operate, while others are volatile but cheaper
- given production rate's average and covariance calculated from historical data

goal: determine the production rate for each well to meet a total demand while minimizing the total risk of the production

modeling:

- $x \in \mathbf{R}^n$: percentage of the total demand production from n wells
- $\Sigma \in \mathbf{S}^n$: covariance of the production rate (mmCFD^2)
- $x^T \Sigma x$: the risk or volatility of the total gas supply
- μ: expected production rate (mmCFD) decline curve analysis (DCA)
- $\mu^T x$: weighted average production

Asset portfolio risk management: Covariance

- diagonals (variance): individual uncertainty of each well
- off-diagonals (covariance): Σ_{ij} is the covariance between well i and j
  - positive: if two wells are in the same reservoir layer, they likely have high positive covariance (some technical issues in that layer would cause both to decrease simultaneously)
  - low/zero: two wells are uncorrelated, providing the diversification for minimizing the overall risk

Asset portfolio risk management: Optimization

variable: x (allocation of production rate)

objective: $(1/2)x^T\Sigma x$ (the risk)

constraints:

1. allocation constraints: $0 \preceq x \preceq \mathbf{1},\ \mathbf{1}^T x = 1$
2. demand requirement: $\mu^T x \geq D$ where D is the total required demand
3. physical capacity limit: $0 \preceq x \cdot D \preceq C$ where $C \in \mathbf{R}^n$ is the maximum capacity

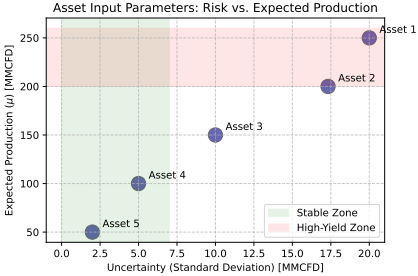
optimization: quadratic program (QP)

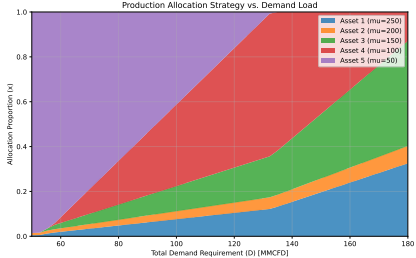
$$\begin{array}{ll} \text{minimize} & (1/2)x^T\Sigma x \\ \text{subject to} & 0 \preceq x \preceq \mathbf{1},\ \mathbf{1}^T x = 1 \\ & \mu^T x \geq D,\ \ 0 \preceq x \cdot D \preceq C \end{array}$$

with variable $x \in \mathbf{R}^n$

Asset portfolio risk management: Results

example: five wells with different uncertainty and yield





- vary D (demand) and observe the portion of allocation
- for low D, invest all in well no. 5 (low risk)
- for high D, invest distributedly in well no. 1,2,3,4

Asset portfolio risk management: as an insurance policy

the formulation on page 223 is essentially an insurance policy model

- by setting D as a hard constraint, meeting the national gas demand is non-negotiable
- by minimizing $x^T \Sigma x$, we are looking for the *quietest* way to meet that demand–the allocation that is least likely to result in a supply shock if one of the wells underperforms

Asset portfolio risk management: Variance and production trade-off

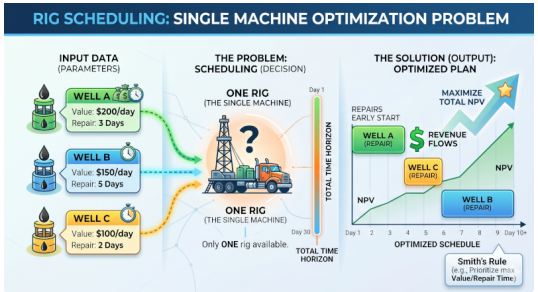
if we add the average production to the objective:

$$\begin{array}{ll} \text{minimize} & (1/2)\, x^T \Sigma x - \lambda \mu^{\mathbf{T}} \mathbf{x} \\ \text{subject to} & 0 \preceq x \preceq \mathbf{1},\ \mathbf{1}^T x = 1 \\ & \mu^T x \geq D,\ \ 0 \preceq x \cdot D \preceq C \end{array}$$

with variable $x \in \mathbf{R}^n$

- $\lambda > 0$ is **risk aversion parameter**: trade-off between *stable* and *cheap* production
- adding $\mu^T x$ to the objective seems redundant but it changes the behavior of the optimal solution within the feasible region
  - with only the constraint ($\mu^T x \geq D$): the solver finds the allocation x that minimizes risk and merely satisfies the demand D; likely to hit the boundary where $\mu^T x = D$
  - with both ($\lambda \mu^T x$ in objective + constraint): the solver minimizes risk but is also *rewarded* for exceeding the demand D; if there are assets that are both stable and high-producing, the solver may choose an allocation that results in $\mu^T x > D$ because the *benefit* of the extra production outweighs the marginal increase in risk.

Rig scheduling: single machine scheduling



setting:

- n wells require repair time and offers different production gain
- a workover rig can repair one well at a time
- a total available repair time is given

goal: find a schedule that maximize the total cumulative value after recovery

- fix the most productive wells as early as possible
- maximize the total value recovered presented as NPV

Rig scheduling

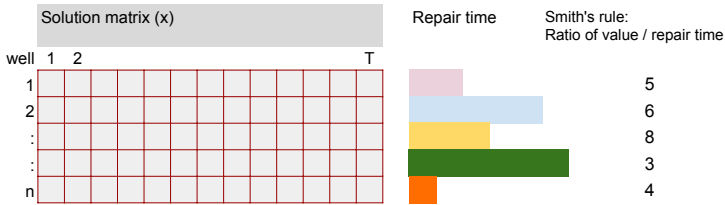
suppose there are n wells to be repaired

parameters:

- $v \in \mathbf{R}^n$: daily revenue or value of well once it is fixed
- $d \in \mathbf{Z}^n$: number of days required to repair well i
- T: total time horizon allowed for the rig to work
- r: the discount rate to calculate Net Present Value (NPV)

variables:

- $x \in \{0,1\}^{n \times T}$: $x_{it} = 1$ if well i is finished on day t



Rig scheduling: NPV

we prioritize fixing high-value wells as early as possible

- a well only begins generating revenue after the repair is complete
- if the repair of well i is finished at day c_i, the well will produce revenue for the remaining days $(T - c_i)$
- for well i, NPV (net present value) is the sum of discounted daily revenue:

$$\text{NPV}_i = \sum_{t=c_i+1}^{T} \frac{v_i}{(1+r)^t}, \quad r \text{ is the discount rate}$$

note: c_i is a function of x_{it}, and NPV appears to be a function of x

Rig scheduling: NPV

as $x_{it} = 1$ at day t and 0 otherwise, we can arrange a NPV matrix $(n \times T)$ to be the sum starting from t to T:

$$\text{NPV}_{it} = \sum_{i=1}^{n} \left(\sum_{k=t}^{T} \frac{v_i}{(1+r)^k} \right)$$

the objective is to maximize the NPV that only counts after finishing the repair

$$\underset{x}{\text{maximize}} \quad \sum_{i=1}^{n} \sum_{t=1}^{T} x_{it} \cdot \text{NPV}_{it}$$

which has a linear objective

Rig scheduling: minimum lead time constraint

the sum of all *finish* possibilities for days earlier than the duration must be zero

$$\sum_{t=1}^{d_i-1} x_{i,t} = 0$$

example: if well i requires 4 days of work

$$x_{i1} = x_{i2} = x_{i3} = 0$$

Rig scheduling: non-overlap constraints

the rig can only work on one well at a time

for any specific day τ in your schedule, the **active repair** status for well i is the sum of all x_{it} that would force the rig to be at that well on day τ:

$$\text{active}_{i,\tau} = \sum_{t=\tau}^{\min(\tau+d_i-1,T)} x_{it}$$

the window $[\tau, \tau + d_i - 1]$ represents all possible finish dates that would mean the well is currently being worked on

Rig scheduling: non-overlap constraints

example: two wells where well 1 takes 2 days, and well 2 takes 3 days to finish

1. if the rig is working on well 1 on day $\tau = 4$ then $\text{active}_{1,4} = x_{1,4} + x_{1,5}$
   - it could finish at the end of day 4 (started day 3)
   - it could finish at the end of day 5 (started day 4)
2. if the rig is working on well 2 on $\tau = 4$ then $\text{active}_{2,4} = x_{2,4} + x_{2,5} + x_{2,6}$
   - it could finish at the end of day 4 (started day 2)
   - it could finish at the end of day 5 (started day 3)
   - it could finish at the end of day 6 (started day 4)
3. the constraint on day $\tau = 4$ says that the sum across all wells must be ≤ 1

$$(x_{1,4} + x_{1,5}) + (x_{2,4} + x_{2,5} + x_{2,6}) \leq 1$$

example: $x_{1,4}$ and $x_{2,4}$ cannot be 1 at the same time; that would mean the rig is active on well 1 during day 3,4 while being active on well 2 during 2,3,4, creating an impossible overlap

Rig scheduling optimization

we can now present the optimization problem as IP:

$$\begin{array}{ll} \text{maximize}_x & \text{NPV} \\ \text{subject to} & \text{minimum lead time constraint} \\ & \text{non-overlap constraint} \end{array}$$

with variable $x \in \{0,1\}^{n \times T}$

explain the solution outcomes under these considerations:

- when T is less than the total repair time, not all wells can be fixed
- what if the value does not decrease over time (not using NPV assumption), would the scheduling plan change?

Rig scheduling: the effect of using NPV

effect on a scenario where a well with a high value also has a long repair duration
setup: $T = 10$, two wells: $v = (500, 200)$ and $d = (5, 2)$

case 1: $r = 0$

- option 1: fix well A before well B, finished on day 5 and then 7

$$\text{revenue} = A + B = (10 - 5) \times 500 + (10 - 7) \times 200 = 3100\$$$

- option 2: fix well B before well A, finished on day 2 and then 7

$$\text{revenue} = B + A = (10 - 2) \times 200 + (10 - 7) \times 500 = 3100\$$$

areas under the revenue curve are the same: the solver is indifferent

the sequence of choosing well to be fixed is not relevant

Rig scheduling: the effect of using NPV

setup: $T = 10$, two wells: $v = (500, 200)$ and $d = (5, 2)$

case 2: $r = 0.001$ a dollar on day 1 is worth more than on day 10

- option 1: fix well A before well B (high value first)
  - well A starts generating revenue on day 6 ($500/day), which is more valuable than well B ($200/day)
  - choose well A first if r is relatively low (discount penalty is low as compared to value generating)
- option 2: fix well B before well A (short duration first)
  - well B finishes fast (day 2) and generate revenue immediately
  - if the discount rate r is high enough, the solver might prefer starting the smaller $200 on day 3, rather than waiting until day 6 for $500 option

when $r > 0$, its value and (v, d) affect the sequence of scheduling

the solver feels *pressure* to finish high-value wells early to avoid losing value over time

Rig scheduling: connection to other problems

without NPV: the actual value is $v_i \times$ remaining days

the problem share similarities to knapsack and bin packing problems

- knapsack goal: choose subset of wells that fits into the time slot (as capacity) to maximize the total value from the subset you pick
- bin packing goal: focus on space utilization (arrange wells so there is no idle time for the rig)

Rig scheduling: as a single machine scheduling

in general, a single machine scheduling has several objectives: *e.g.*,

- minimize the sum of completion times
- minimize the cost of lateness
- minimize the profit of earliness

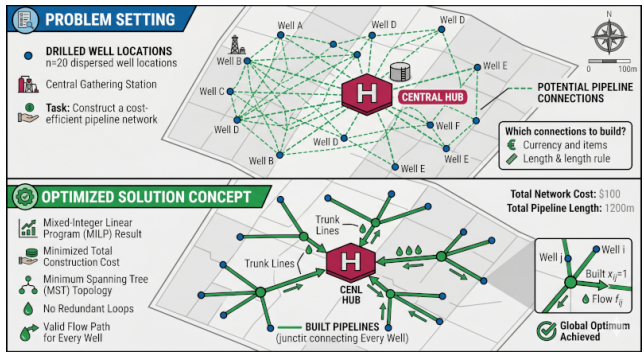
the smith's rule (weighted shortest processing time, WSPT) is optimal algorithm for **infinite horizon**

with NPV or a fixed T, smith's rule is a heuristic (provide a good starting point) but not optimal

Pipeline network design

setting:

- n newly drilled wells across a field and one central station
- aim to connect every well to the station using minimum cost
- ensure that every well has a valid flow path to the destination



Pipeline network design: Variables and Objective

variables:

- $x_{ij} \in \{0,1\}^{n \times n}$: $x_{ij} = 1$ if a pipeline is constructed from node j to node i
- $f_{ij} \geq 0$: the amount of fluid flowing from node j to node i

objective:

$$\text{minimize} \quad \sum_{1 \leq i,j \leq n} c_{ij} x_{ij}$$

example of the cost coefficient: c_{ij} can be the distance between location i and j

once x and f are solved, we can represent the network using **graph** concept

- a set of nodes indexed by $1, 2, \ldots, n$
- a set of edges connected between node i and j; here, using x as the adjacency matrix to determine the edges

  an edge connected between from node j to node i if $x_{ij} = 1$

Pipeline network design: Constraints

- **flow balance:** for each node i, the net flow (total flow out - total flow in) must equal the production

$$\sum_k f_{ki} - \sum_j f_{ij} = b_i \quad \text{(production)}$$

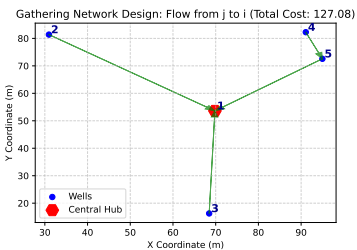
(f_{ki} is flow out of i to k, and f_{ij} is flow from j into i)
  - if $b_i > 0$, the node i is a generator or a source
  - if $b_i < 0$, the node i is a sink (or load), *e.g.*, the station
- **capacity link:**

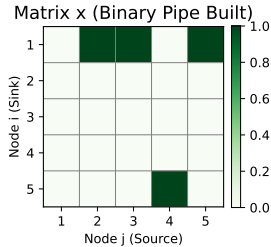
$$f_{ij} \leq M \cdot x_{ij}, \quad M \text{ is a large number}$$

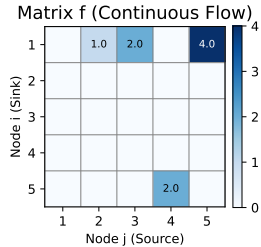
this ensures flow can only exist from j to i if the pipe x_{ij} is built

Pipeline network design: Results

the optimization is MILP







- node 1 is the hub: row 1 has nonzero entries on column 2,3,5 (from nodes 2,3,5)
- link from node 4 to node 5: row 5 has a nonzero entry on column 4
- sum of row 1: sum of flows to the hub

Pipeline network design: extension

some relevant constraint in a more realistic setting

1. pressure drop and nonlinear flow

$$f_{ij}|f_{ij}| = C_{ij}^2\sqrt{P_j^2 - P_i^2} \quad (C_{ij} \text{ is a pipeline constant})$$

the flow is proportional to the square root of the difference of squared pressures

(this is nonlinear; require some more math technique to re-formulate)

2. define an auxilliary variable: $\pi_i = P_i^2$ and use a **relaxation**

$$f_{ij}^2 \leq C_{ij}^2(\pi_j - \pi_i) \quad \text{(relax to inequality)}$$

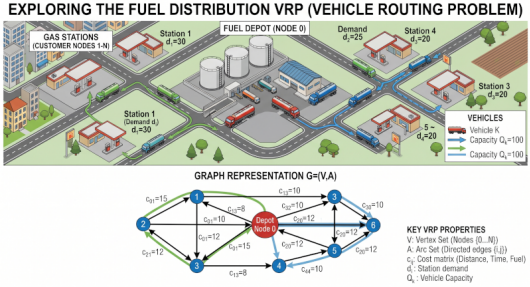
3. this can be rewritten as a **second-order cone** form:

$$2\left(\frac{f_{ij}}{C_{ij}}\right)^2 + 2\pi_i \leq 2\pi_j \quad \Longrightarrow \quad \left\| \begin{matrix} \sqrt{2}\frac{f_{ij}}{C_{ij}} \\ \pi_j - \pi_i - 1 \end{matrix} \right\|_2 \leq \pi_j - \pi_i + 1$$

the problem is a mixed-integer second-order cone program (MISOCP)

Fuel distribution

a central terminal (depot) must supply a set of gas stations using a fleet of trucks



network: a directed graph $G = (V, A)$

- $V = \{0, 1, \ldots, n\}$: set of nodes (node 0 is the depot and nodes $1, \ldots, n$ are delivery points
- $A = \{(i, j) \mid i, j \in V, \;\; i \neq j\}$: an ordered pair (i, j) implying a direction from node i to j

- **depot:** a single starting and ending point for all vehicles
- **customers:** n gas stations, each with a known fuel demand d_i
- **fleet:** K identical tanker trucks, each with a maximum fuel capacity Q
- **travel cost:** c_{ij} (distance or time) is the cost from location i to j

Fuel distribution: Variables

- $x \in \{0,1\}^{(n+1)\times(n+1)\times K}$: travel status

$$x_{ijk} = 1 \text{ if vehicle } k \text{ travels directly from node } i \text{ to node } j$$

- $u \in \mathbf{R}^{(n+1)\times K}$: cumulative load variable

  u_{ik} represents the total load carried by vehicle k after visiting node i (to track capacity and prevent sub-tours)

Fuel distribution: Objective

the goal is to minimize the total operational cost of the fleet.

$$\text{minimize} \quad \sum_{k=1}^{K} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} \cdot x_{ijk}$$

this minimizes the sum of all segments traveled by all active tanker trucks

indices i, j starts at **0** referring to the depot index and $1, \ldots, n$ for customers

Fuel distribution: Contraints

notation: $V = \{0, 1, \ldots, n\}$ and $[n] = \{1, \ldots, n\}$

every station is serviced without exceeding truck limits

1. **service guarantee:** each gas station must be visited exactly once by one truck

$$\sum_{k \in [K]} \sum_{i \in V} x_{ijk} = 1, \quad \forall j \in [n]$$

(this is assign constraint to gas station, so j runs from 1, not 0)

2. **flow conservation:** if a truck enters a gas station (node i), it must also leave that station to go to the next node or return to the depot

$$\sum_{i \in V} x_{ipk} - \sum_{j \in V} x_{pjk} = 0, \quad \forall p \in V, \forall k \in [K]$$

(go from i to p and then go from p to j)

Fuel distribution: Contraints

3. **depot continuity:** every truck used must start its route at the depot and return to the depot after its final delivery

$$\sum_{j\in[n]} x_{0jk} = 1, \quad \forall k \in [K], \qquad \sum_{i\in[n]} x_{i0k} = 1, \quad \forall k \in [K]$$

4. **capacity constraint:** for any vehicle k, the sum of demands d_j for all stations on its route cannot exceed the tanker's capacity Q_k

$$\sum_{i\in V} \sum_{j\in[n]} d_i \cdot x_{ijk} \leq Q_k \quad \forall k \in [K]$$

5. **sub-tour elimination:** ensure the solution is one continuous loop per truck rather than several disconnected small cycles

Fuel distribution: Sub-tour elimination

a common approach is the **Miller-Tucker-Zemlin (MTZ)** formulation:

$$\text{MTZ 1:} \quad u_{ik} - u_{jk} + Q_k x_{ijk} \leq Q_k - d_j, \quad \forall (i,j) \in A, j \neq 0, \forall k \in [K]$$
$$\text{MTZ 2:} \quad d_i \leq u_{ik} \leq Q_k, \quad \forall i \in [n], \forall k \in [K]$$

how it works: the constraint MTZ 1 is binding only when traveling from i to j

- if travel from $i \to j$ ($x_{ijk} = 1$): constraint reduces to

$$u_{jk} \geq u_{ik} + d_j \quad \Rightarrow \quad \text{force } u_{jk} \text{ to strictly increase}$$

creating a logical flow that cannot loop back on itself

example: consider a loop $1 \to 2 \to 3 \to 1$

1. for $1 \to 2$: $u_{2k} \geq u_{1k} + d_2$
2. for $2 \to 3$: $u_{3k} \geq u_{2k} + d_3$ $\Longrightarrow$
3. for $3 \to 1$: $u_{1k} \geq u_{3k} + d_1$

take the sum of three inequalities

we get a **contradiction**

$0 \geq d_1 + d_2 + d_3$

therefore, **no such loop can exist** in a feasible solution

Fuel distribution: Sub-tour elimination

- if not travel from $i \to j$ ($x_{ijk} = 0$): constraint reduces to

  $$u_{ik} - u_{jk} \leq Q_k - d_j, \quad \Rightarrow \quad \text{easily satisfied by any load value (loose constraint)}$$

  meaning u_{jk} is free to be any value that fits the rest of the route's logic

Fuel distribution: Optimization

the problem is MILP and regarded as a **capaciated vehicle routing (CVRP)** problem

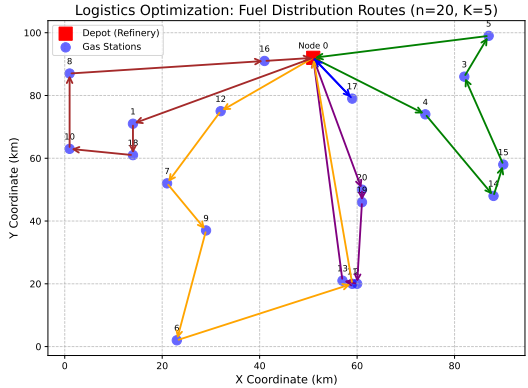
minimize	travel cost
subject to	service guarantee
	flow conservation
	depot continuity
	capacity constraint
	sub-tour elimination

with variables $x \in \{0,1\}^{(n+1)\times(n+1)\times K}$ (travel status) and $u \in \mathbf{R}^{(n+1)\times K}$ (cumulative load variable)

can be solved by `routing` library in Google OR tools

Fuel distribution: Results

example: $n = 20, K = 5, Q = (25, 25, 25, 25, 25)$, random coordinates and demand, cost is the distance



the solver naturally groups gas stations that are geographically close to each other

CVRP

essentially a collection of TSP with added constraints

- for one vehicle with infinite capacity, CVRP simplifies to a standard TSP
- since TSP is NP-hard[1], CVRP must also be NP-hard
- MILP formulations can be used for small-to-medium problems but for large-scale logistics, we often rely on specialized heuristics
- OR-tools use metaheuristics (guided local search, GLS) in routing problem
- CVRP introduces elements of bin packing problem as we must decide which gas stations (demand) fit into which truck (bins of capacity)
- CVRP is a VRP with a capacity constraint; if we set infinite capacity ($Q = \infty$), it becomes multi-vehicle VRP

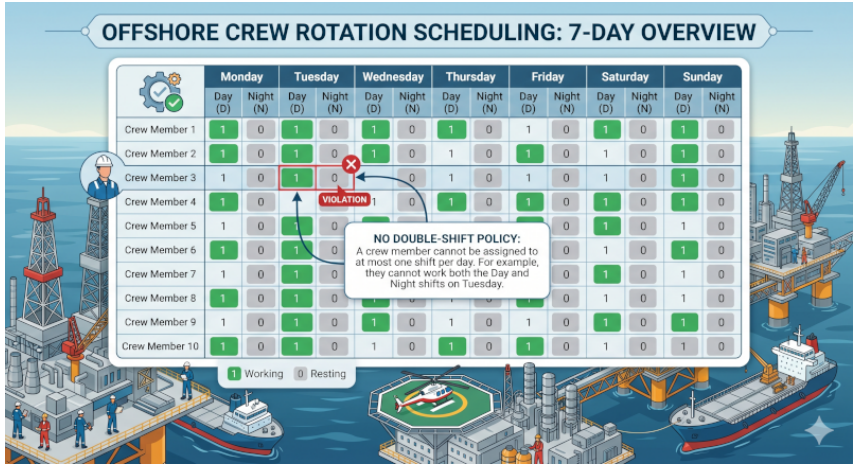
[1](non-deterministic polynomial-time hard; it is at least as difficult as the hardest problems in NP)

Shift assignment

an offshore gas platform operates two 12-hour shifts (Day and Night) per day. You must assign a pool of 10 crew members to these shifts over a 7-day rotation period while satisfying strict safety and labor regulations.

1. **minimum skill coverage:** each shift (day and night) must have exactly 3 crew members assigned; no more,no less
2. **no double-shift rule:** a crew member can be assigned to at most one shift per day; they cannot work both the day and night shifts
3. **recovery period:** if a crew works a night shift, they cannot work the day shift immediatly following it; need at least 12 hours of rest
4. **fair work-life balance:** over 7-day period, every member must work at least 3 shifts but no more than 5 shifts total

Shift assignment: Example of results



Project scheduling

goal: to schedule a set of n maintenance activities during a planned facility shutdown

- maintenance list: valve replacement, sensor calibration, pressure vessel inspection
- the platform has limited specialized personnel (divers, welders) and equipment (heavy-lift cranes)
- each task has a known duration, but some tasks cannot start until others are finished

variables:

- start time: S_i an integer showing when task i begins for $i \in [n]$
- end time: E_i an integer showing when task i finishes for $i \in [n]$
- presence: P_i a boolean (if a task is optional, though in shutdowns, most are mandatory)
- interval: I_i a logical object that link start, end, and duration ($E_i = S_i +$ duration)

Project scheduling: Logic constraints

1. precedence constraints: task B cannot start until task A is complete: $S_B \geq E_A$
2. resource capacity (cumulative): the total number of welders required by all active tasks at any hour t cannot exceed the 5 welders currently offshore

$$\sum_i \text{welders}_i \leq \text{available welders}$$

3. disjunctive constraints (no-overlap): two specific tasks cannot happen at the same time because they share the same physical space or tools

$$I_A \text{ and } I_B \text{ must not overlap}$$

4. fixed windows: the flare tip replacement must be during daylight hours for safety

$$S_{\text{flare}} \geq \text{06:00} \text{ and } E_{\text{flare}} \leq \text{18:00}$$

Project scheduling: Objective

minimize the maximum E_i (the finish time of the very last task)

this translate to minimizing lost production revenue

CP-SAT uses interval variables and cumulative functions designed specifically to handle blocks of time efficiently

Project scheduling: Example

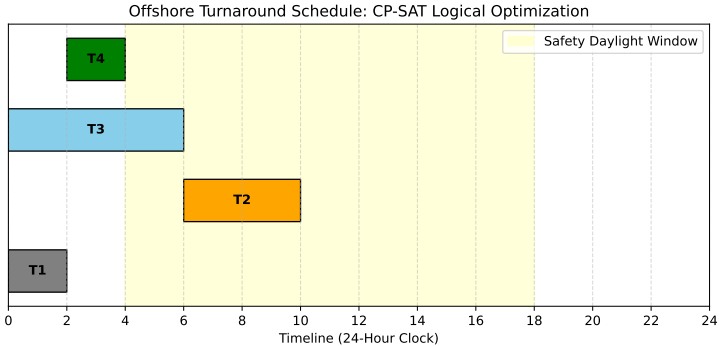
horizon: 24-hour to complete all the tasks

Task	Work	Duration (h)	Resource
T1	scaffold	2	2 workers
T2	flare removal	3	3 workers + crane
T3	pipe welding	6	2 workers
T4	crane calibration	2	1 worker + crane

logic constraints:

1. precedence: we must scaffold before removing the flare tip (T2)
2. resource capacity: T2 and T3 must require the workers to stays ≤ 4
3. no-overlap: T2 and T4 both require the single crane
4. fixed window: T2 (flare removal) must occur during 6:00-18:00

Project scheduling: Result



A3: Regression problems in geoscience

Outline

Regression problem

Regression problem

setting: suppose $\{(x_i, y_i)\}_{i=1}^N$ are input/output data

- y (output/response variable) is a function of x (input/regressor)
- ground-truth: $y = g(x;\theta) + e$ where g is the actual relationship between x and y parametrized by θ but g is unknown
- we model $\hat{y} = f(x;\theta)$ and estimate θ such that $\hat{y} \approx y$

method: to quantify that $y \approx \hat{y}$, several choices of loss functions can be applied

- ℓ_2-norm: $\ell(\theta) = (1/N)\|y - \hat{y}\|_2^2 = (1/N)\sum_{i=1}^N |y_i - f(x_i;\theta)|^2$
- ℓ_1-norm: $\ell(\theta) = (1/N)\|y - \hat{y}\|_1 = (1/N)\sum_{i=1}^N |y_i - f(x_i;\theta)|$
- ℓ_∞-norm: $\ell(\theta) = \|y - \hat{y}\|_\infty = \max_{i=1,\ldots,N} |y_i - f(x_i;\theta)|$

other robust loss functions exist in literature (huber, median absolute deviation (MAD), etc.)

Regularized regression

we provide a concept of estimation with two objectives:

$$\underset{x}{\text{minimize}} \quad f(x) := g(x) + \gamma h(x)$$

- x is model parameter
- g is a loss function that indicates **model fitting**
- h is a **regularization function** that affects solution properties (aka **penalty**)
- $\gamma > 0$ is a penalty weight controlling a balance between model quality and regularization of x

g is typically a quadratic loss or log-likelihood function

the choice of h dictates the solution behaviour

ℓ_2 regularization

given x_r as a targeted solution (defined by the user)

we add the 2-norm penalty to the objective function

$$\underset{x}{\text{minimize}} \quad g(x) + \gamma \|x - x_r\|_2^2$$

- seek for a solution that minimizes $g(x)$ but also close to x_r
- the trade-off is controlled by $\gamma > 0$ (large value heavily forces $x \approx x_r$)
- also called **Tikhonov regularized least-squares** or **ridge regression**
- when $g(x)$ is the linear least-squares loss: $g(x) = \|Ax - y\|_2^2$, the ℓ_2-regularized LS has the analytical solution for any $\gamma > 0$:

$$x^\star = (A^T A + \gamma I)^{-1} A^T y$$

(no restrictions on shape, rank of A)

- interpreted as a MAP estimation with the log-prior of the Gaussian

ℓ_1 regularization

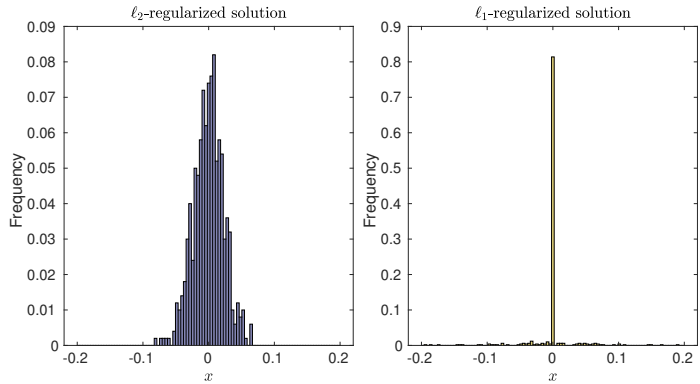
adding the ℓ_1-norm penalty to the least-square problem

$$\underset{x}{\text{minimize}} \quad g(x) + \gamma \|x\|_1$$

- a convex heuristic method for finding a sparse x that also minimizes g
- also called **Lasso** or **basis pursuit**
- a nondifferentiable problem due to $\|\cdot\|_1$ term
- no analytical solution (even when g is linear LS loss), but can be solved efficiently
- interpreted as a MAP estimation with the log-prior of the Laplacian distribution
- find many applications known as **sparse** methods: sparse- logistic regression, PCA, SVM, Gaussian graphical model, etc.

Regularization: Example

solve linaer LS with $h(x) = \|x\|_2^2$ and $h(x) = \|x\|_1$ using $\gamma = 0.2$



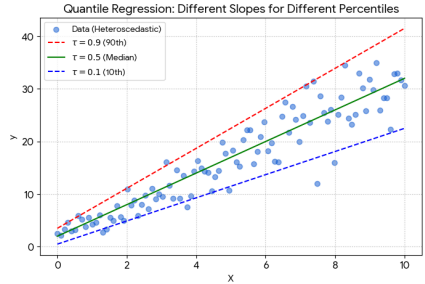
- solution of ℓ_2 regularization is more widely spread
- solution of ℓ_1 regularization is concentrated at zero

Quantile regression

suppose we model $y = f(x;\beta)$ where f can be any nonlinear mapping

goal: find β_τ that makes $\hat{y} = f(x;\beta)$ match well with τ-quantile of y

$$\underset{\beta_\tau}{\text{minimize}} \;\; (1/N)\sum_{i=1}^{N} \rho_\tau(y_i - f(x_i;\beta_\tau))$$



where ρ_τ is called the **pinball loss**

$$\rho_\tau(r) = \max(\tau r, (\tau-1)r) = \begin{cases} (\tau-1)r, & r < 0, \\ \tau r, & r \geq 0 \end{cases}$$

- can be shown that $\hat{y} = f(x;\beta_\tau)$ agrees with the sample τ-quantile of y
- if $f(x;\beta_\tau)$ is **linearly parametrized** in β_τ, the problem can be cast as an LP

LP formulation for quantile regression

when $f(x;\beta_\tau)$ is linearly parametrized in β_τ:

$$f(x;\beta_\tau) = \beta_{\tau,1} g_1(x) + \cdots \beta_{\tau,s} g_s(x)$$

(a nonlinear model f is represented as a sum of s-basis functions g_k)

minimizing pinball loss can be cast as an LP

$$\begin{array}{ll} \text{minimize}_{d,\beta_\tau} & \mathbf{1}^T d \\ \text{subject to} & \tau(y_i - f(x_i;\beta_\tau)) \le d_i,\ i = 1,\ldots,N \\ & (\tau-1)(y_i - f(x_i;\beta_\tau)) \le d_i,\ i = 1,\ldots,N \end{array}$$

(by introducing $d \in \mathbf{R}^N$ as an upper bound of $\max$ function)

it can be an LP because $f(x;\beta_\tau)$ is linear function in $\beta_{\tau,1},\ldots,\beta_{\tau,s}$

Quantile regression: Multiple quantiles

when the goal is to find $\hat{y}_\tau$ associated with a list of τ-quantiles:

$$\tau \in \Gamma = \{\tau_1, \tau_2, \ldots, \tau_m\} \quad e.g., \Gamma = \{0.1, 0.2, \ldots, 0.9\}$$

- we must have separate coefficients β_τ for each τ
- mathematical fact: the value of $\hat{y}_\tau = f(x; \beta_\tau)$ is increasing as τ increase, *e.g.*,

$$\hat{y}_{0.1} \leq \hat{y}_{0.2} \leq \cdots \leq \hat{y}_{0.9}$$

- the coefficients $\beta_{0.1}, \ldots, \beta_{0.9}$ are linked to **non-crossing quantile** condition

quantile regression with non-crossing quantile constraint

$$\begin{array}{ll} \text{minimize}_{\beta_\tau} & (1/N)\sum_{i=1}^{N}\sum_{\tau\in\Gamma}\rho_\tau(y_i - f(x_i;\beta_\tau)) \\ \text{subject to} & f(x_i;\beta_{\tau_1}) \leq f(x_i;\beta_{\tau_2}) \leq \cdots \leq f(x_i;\beta_{\tau_m}),\ i = 1, \ldots, N \end{array}$$

Examples in geoscience

- well test analysis
- decline curve analysis (DCA) and regularization
- quantile decline curve

Well test analysis

motivation:

- when we shut in a producing well, the pressure in the wellbore begins to rise as the surrounding reservoir fluids try to reach equilibrium
- how the pressure increases over time is a *unique* characteristic of the reservoir

problem setting: given N samples of t_i and ΔP_i (change in pressure) for $i \in [N]$

model the pressure change by

$$\Delta \hat{P}(t; kh, s) = \frac{162.6 \cdot q \cdot B \cdot \mu}{kh} \left[\log_{10} \left(\frac{kt}{\phi \mu c_t r_w^2} \right) - 3.23 + 0.869 s \right]$$

where q (flow rate), B (formation volume factor), μ (viscosity), ϕ (porosity), c_t (compressibility), r_w (wellbore radius) are known constants

unknown parameters: $k \cdot h$ (permeability-thickness) and s (skin factor)

Well test analysis: as Nonlinear least-squares

we can group the terms: $A = 162.6 \cdot q \cdot B \cdot \mu$ and $D = \frac{1}{\phi \mu c_t r_w^2}$

$$\Delta \hat{P}(t; kh, s) = \frac{A}{kh} \left[\log_{10}(kDt) - 3.23 + 0.869s \right]$$

physical constraints: $kh > 0$ and $s \geq -5$ (physical limit for stimulated well)

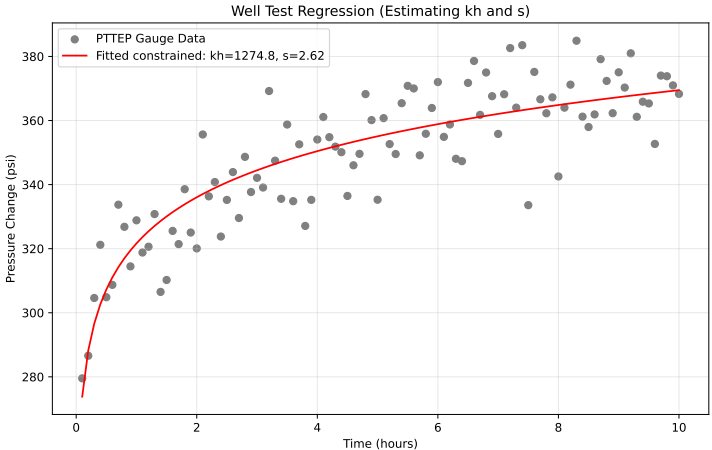
nonlinear least-squares:

$$\underset{kh,s}{\text{minimize}} \quad \frac{1}{N} \sum_{i=1}^{n} \left[\Delta P_i - \left(\frac{A}{kh} \left[\log_{10}(kDt_i) - 3.23 + 0.869s \right] \right) \right]^2$$

- we can try solving NLS with and without physical constraints
- note that if kh is known, the problem becomes linear LS in s

Well test analysis: Fitted result

given a fixed h, ground-truth value $(kh = 1250.0, s = 2.5)$



Decline curve analysis (DCA)

when open a well, the pressure is high, and the oil flows fast; as oil leaves the tank, the pressure drops, and the flow rate declines

questions to be explored:

- forecasting: how much oil will the well produce next month?
- EUR: Estimated uncertainty recovery: what is the total volume we will recover before the well dies?
- mine life: when will the well hit its economic limit where it costs more to run than it earns?

DCA: the Arps equations

J.J.Arps observed that the relative change in production rate follows

$$q(t) = \frac{q_0}{(1 + bD_0 t)^{1/b}}, \quad t > 0 \;\; t \text{ is time}$$

- q_0: initial production rate when the well first starts
- D_0: initial decline rate (how fast the production drops at the every begining, *e.g.*, 10% per year)
- b: the b-factor describes the **curvature** of the decline
  - $b = 0$ (exponential): decline by a constant percentage per year (take the limit $b \to 0$, we get $q(t) = q_0 e^{-Dt}$ using L'Hôpital's rule)
  - $0 < b < 1$ (hyperbolic): a common case where the rate itself slows down over time
  - $b = 1$ (harmonic): special case often seen in high-water-derive reservoirs

DCA: Linearized Arps equations

instead of viewing $q(t)$ as the response, we consider the relative change in production

$$\textbf{decline rate:} \quad D(t) \triangleq -\frac{1}{q(t)}\frac{dq(t)}{dt} = \frac{D_0}{1+bD_0t}$$

the reciprocal of $D(t)$ is linearly parametrized in b

$$\frac{1}{D(t)} = bt + \frac{1}{D_0} = \begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} b \\ \frac{1}{D_0} \end{bmatrix} \quad \triangleq \quad y = Ax$$

- by changing the response variable, we can use a linear model (the estimation part becomes easy)
- calculating dq/dt from raw data can be extremely noisy
- outlier: if dq/dt is near zero (during a temporary production spike or sensor error), the value $\frac{q}{dq/dt}$ goes to ∞

DCA: Nonlinear least-squares

for a single well, we observe data $(t, q(t))$, find $x = (q_0, D_0, b)$ that

$$\underset{x}{\text{minimize}} \quad \sum_{t=1}^{T} (q(t) - \hat{q}(t; x))^2$$

where $\hat{q}(t; x)$ is the production from the assumed model

linear case: when $b = 0$, reduces to linear LS

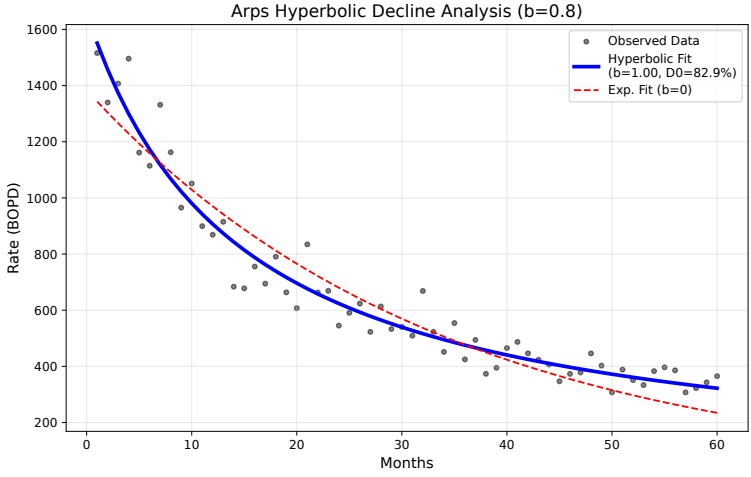
$$\log q(t) = \log(q_0) - D_0 t$$

practical issues:

- data may contain outliers due to shut-in period, adjusted choke
- well condition change (pump installation, water breakthrough); this may require refit the decline curve in multi-segment
- with a few samples, the optimization may return b factor that is unrealistic, *e.g.*, $b = 2$ (may fix by posting a constraint)

DCA: Fit result

ground-truth: Arps equation with $b = 0.8$



fitted with `scipy.optimize.curve_fit`

DCA: ℓ_2 regularization

penalize the squared distance of the b factor from a known physical **anchor**, b_{prior} for a shale (unconventional reservoir), expect $b \approx 1 - 1.5$ during first 3-5 years

$$\ell_2\text{-regularized LS:} \quad \underset{x}{\text{minimize}} \quad \sum_{t=1}^{T} (y(t) - \hat{y}(t;x))^2 + \gamma(b - b_{\text{prior}})^2$$

- regularize on the nonlinear Arps model

$$y(t) = q(t), \quad \hat{y}(t;x) = \frac{q_0}{1 + bD_0 t}, \quad x = (b, D_0)$$

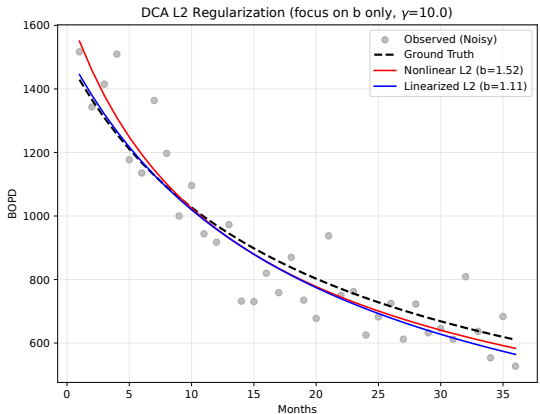
- regularize on the linearized Arps model

$$y(t) = \frac{1}{D(t)}, \quad \hat{y}(t;x) = bt + \frac{1}{D_0}, \quad x = (b, 1/D_0)$$

closed-form solution: $b = (A^T A + \gamma e_1 e_1^T)^{-1}(A^T y + \gamma e_1 b_{\text{prior}})$, $e_1 = (1, 0)$ ✎

DCA: Fit result of ℓ_2 regularization

ground-truth: Arps equation with $b = 1.4$ and set $b_{\text{prior}} = 1.3$



the performance depends on γ (penalty parameter) and the confidence in b_{prior}

Quantile decline curve

produce a family of curves $q_\tau(t)$ that define the probability space of well's performance

- P90 curve ($\tau = 0.1$): the **conservative profile**; there is a 90% chance that the actual production at any time t will be above this line
- P50 curve ($\tau = 0.5$): the **most likely profile**; the median expectation
- P10curve ($\tau = 0.9$): the **upside potential**; only 10% of wells, given these x inputs, will perform this well

modeling:

- response y: production rate (q_t) at time t (STB/d)
- predictor t: time (months on production)
- predictor X:
  - controllable: lateral length, number of stages, total proppant
  - measured: porosity, initial reservoir pressure, depth
  - choke size setting

Quantile decline curve

assume the production rate follows a model $f(t, x; \beta)$

quantile regression minimizes the pinball loss: given quantile list $\Gamma = \{0.1, 0.5, 0.9\}$

$$\text{minimize}_{\beta_\tau} \quad \sum_{\tau \in \Gamma} \sum_{i=1}^{N_{\text{no. well}}} \sum_{t=1}^{T_i} \max\left(\tau(y_{i,t} - f(t, x_i; \beta_\tau)), (\tau - 1)(y_{i,t} - f(t, x_i; \beta_\tau))\right)$$

subject to non-crossing quantile constraint

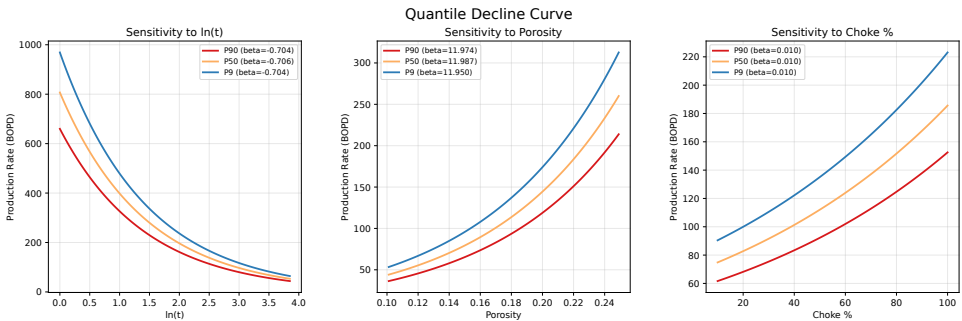
- some static predictors x_i that do not change over time
  - depth of well i
  - amount of proppant (sand) injected into well i during completion
  - geological coordinates of well i
- the time length T_i may vary between wells (well A has 24-month data, while well B has 6 month)
- coefficient β_τ is **global** across wells (represent physics applied to the entire field)

Quantile decline curve

data: 200 wells, $\tau = 0.1, 0.5, 0.9$

$$\log q(t) = \beta_0 + \beta_1 \log(t) + \beta_2 \cdot \text{porosity} + \beta_3 \cdot \text{choke}$$

(choke changes over time, but porosity is static)



(plotted in the original transform - reverse log back to expo)

Numerical tools

Python library

- nonlinear least-squares with constraints
  - `scipy.optimize.curve_fit`
  - `scipy.optimize.least_squares`
  - linear case: `np.linalg.lstsq`
- quantile regression
  - linear case: can be alternatively solved using LP formulation
  - `statsmodel` with `QuantReg` class
  - `Scikit-learn` with `QuantileRegressor` class
  - tree-based models such as LightGBM, XGboost, quantile regression forest
- quantile neural network (QRNN): we can use `PyTorch` implementation

other regression problems: depend on the problem type

if convex, we can use `cvxpy`

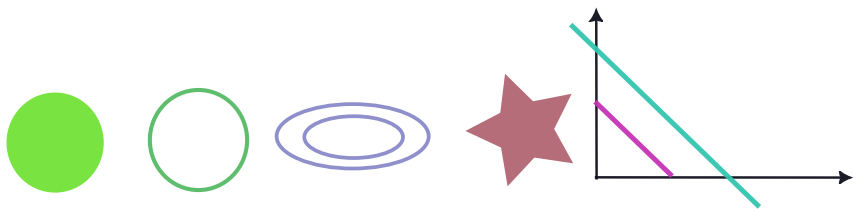
A4: Convex optimization applications

Convex sets

a set $\mathcal{C}$ is said to be convex if for any $x, y \in \mathcal{C}$ we have

$$\theta x + (1-\theta)y \in \mathcal{C}, \quad \text{for all } \ 0 \leq \theta \leq 1$$

which of the following sets are convex ?



fact: an intersection of convex sets is convex (even infinitely many number of intersections)

Convex sets: Examples

examples in $\mathbf{R}^n$

- hyperplane, halfspace
- polyhedron: $\{x \in \mathbf{R}^n \mid a_i^T x \leq b_i,\ i = 1, \ldots, m,\ c_j^T x = d_j,\ j = 1, \ldots, p\}$
- probability simplex: $\{x \mid x \succeq 0,\ \ \mathbf{1}^T x = 1\}$
- norm ball: $\{x \in \mathbf{R}^n \mid \|x\| \leq \alpha\}$ where $\alpha \geq 0$ is given
- norm cone: $\{(x, t) \mid \|x\| \leq t\} \subseteq \mathbf{R}^{n+1}$
- second-order cone: is the norm cone for the ℓ_2-norm
- convex cone: C is a set that $x_1, x_2 \in C$ and $\theta_1, \theta_2 \geq 0$ we have $\theta_1 x_1 + \theta_2 x_2 \in C$

Convex functions

convex function: $f : \mathbf{R}^n \to \mathbf{R}$ is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$$

for all x, y in the domain of f and $0 \leq \theta \leq 1$

loosely speaking, f is convex if it has an upward shape

examples on $\mathbf{R}$:

- affine: $ax + b$ for any $a, b \in \mathbf{R}$
- exponential: e^{ax} for any $a \in \mathbf{R}$
- powers of absolute value: $|x|^p$ for $p \geq 1$
- negative entropy: $x \log x$ on $\mathbf{R}_{++}$

Examples of convex functions on $\mathbf{R}^n$

- affine: $a^T x + b$
- norm functions: $\|x\|$
- norms of affine: $\|a^T x + b\|$
- quadratic: $x^T P x + q^T x$ when $P \succeq 0$
- negative entropy: $\sum_{i=1}^n x_i \log x_i$ on $\mathbf{R}^n_{++}$
- log-sum-exponential: $\log\left(\sum_{i=1}^m e^{x_i}\right)$
- sum of K-largest entries: $x_{[1]} + \cdots + x_{[K]}$ where $x_{[1]} \geq x_{[2]} \geq \cdots \geq x_{[n]}$

fact: a α-sublevel set of a convex function f:

$$\mathcal{C}_\alpha = \{x \in \mathbf{R}^n \mid f(x) \leq \alpha\}$$

is a convex set

Connection between convex sets and convex functions

suppose a feasible set is described by inequality constraint functions:

$$\mathcal{C} = \{x \in \mathbf{R}^n \mid f_i(x) \leq 0,\ i = 1, 2, \ldots, m\}$$

fact: $\mathcal{C}$ is a convex set if all $f_i(x)$'s are convex functions

- the constraint of each $f_i(x) \leq 0$ corresponds to a convex set (sublevel set of convex function property)
- $\mathcal{C}$ is the **intersection** of m convex sets, and hence is convex

First- and second-order conditions of convex functions

suppose f is differentiable; then f is convex if and only if

$$\mathbf{dom}\, f \text{ is convex and } f(y) \geq f(x) + \nabla f(x)^T (y - x), \quad \forall x, y \in \mathbf{dom}\, f$$

- the first-order Taylor approximation of f is a **global underestimator** of f if and only if f is convex
- if $\nabla f(x) = 0$ then for all $y \in \mathbf{dom}\, f, f(y) \geq f(x)$,*i.e.*, x is a global minimizer of f

assume that $\nabla^2 f$ exists at each point in $\mathbf{dom}\, f$; then f is convex if and only if

$$\mathbf{dom}\, f \text{ is convex and } \nabla^2 f(x) \succeq 0, \quad \forall x \in \mathbf{dom}\, f$$

f is convex if and only if its Hessian matrix is positive semidefinite

Operations that preserve convexity

preserve convexity of a convex set

- intersection
- image (and inverse image) of affine mapping
- image (and inverse image) of perspective mapping

preserve convexity of a convex function

- non-negative weighted sum
- composition with affine mapping, composition rules
- pointwise maximum and supremum, minimization over one variable
- perspective of a function
- conjugate function (important role in duality theory)

Pointwise maximum

if $f_1, \ldots, f_m$ are convex then their pointwise maximum:

$$f(x) = \max\{f_1(x), f_2(x), \ldots, f_m(x)\}$$

is also convex

example:

- piece-wise linear function:

$$f(x) = \max\{a_1^T x + b_1, a_2^T x + b_2, \ldots, a_m^T x + b_m\}$$

Composition of convex with affine function

if $f(x)$ is convex then $f(Ax + b)$ is also convex

examples:

- $f(x) = \|Ax + b\|$ for any norm $\|\cdot\|$
- $f(x) = \log\left(\sum_{i=1}^{m} e^{a_i^T x + b_i}\right)$ (softplus, smooth approximation of max)
- $f(w) = \max(0, 1 - y_i(w^T x_i + b))$ (hinge loss in SVM)

Supremum over an infinite set of convex functions

if for each $y \in \mathcal{A}$, $f(x, y)$ is convex in x, then the function g, defined as

$$g(x) = \sup_{y \in \mathcal{A}} f(x, y)$$

is convex in x

- distance to farthest point of a set:

$$f(x) = \sup_{y \in C} \|x - y\| \quad \text{is convex}$$

(since $\|x - y\|$ is linear in x (hence, convex in x)

- support function of a set:

$$S_C(x) = \sup \{x^T y \mid y \in C\}$$

(since $x^T y$ is linear in x)

Convex programs

convex optimization problem is one of the form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & a_i^T x = b_i, \quad i = 1, \ldots, p \end{array}$$

where

- objective and constraint functions are **convex**
- equality constraint functions $h_i(x) = a_i^T x - b_i$ must be **affine**

result: an optimal solution of a convex program is a **global** minimizer

Properties of convex problems

convex problems are of interest due to some desirable properties

- many operations preserve convexity of a convex set
  - intersection
  - image (and inverse image) of affine mapping
  - image (and inverse image) of perspective mapping
- many operations preserve convexity of a convex function
  - non-negative weighted sum
  - composition with affine mapping, composition rules
  - pointwise maximum and supremum, minimization over one variable
  - perspective of a function
  - conjugate function (important role in duality theory)
- KKT conditions are *sufficient* and *necessary* for optimality

many optimization problems in engineering are convex programs

LP and QP

a general **linear program** has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

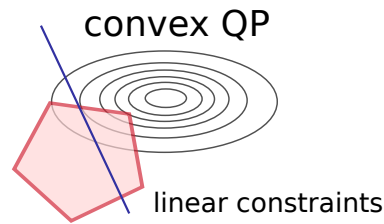
where $G \in \mathbf{R}^{m \times n}$ and $A \in \mathbf{R}^{p \times n}$

a **quadratic program (QP)** is in the form

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x \\ \text{subject to} & Gx \preceq h \\ & Ax = b, \end{array}$$

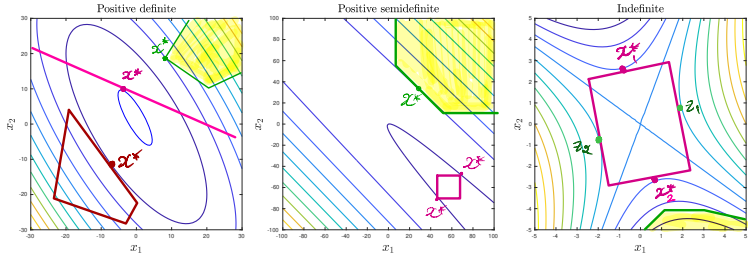
where $P \in \mathbf{S}^n, G \in \mathbf{R}^{m \times n}$ and $A \in \mathbf{R}^{p \times n}$

QP has **linear** constraints



Contour of quadratic objective

consider three cases of P and different feasible sets



verify the location of the optimal solution for each constraint set

- left: a bounded set, a line, an unbounded feasible set
- middle: bounded and unbouded feasible sets, while f is unbounded below
- right: a bounded feasible set, while f is unbounded below and above

QCQP

a **quadratically constrained quadratic program (QCQP)** is in the form

$$\begin{array}{ll} \text{minimize} & (1/x)x^T P_0 x + q_0^T x \\ \text{subject to} & (1/2)x^T P_i x + q_i^T x + r_i \leq 0, \quad i = 1, \ldots, m \\ & Ax = b, \end{array}$$

where P_i's are positive semidefinite, $G \in \mathbf{R}^{m \times n}$ and $A \in \mathbf{R}^{p \times n}$

QCQP has both **linear and quadratic** constraints

Second-order cone programming

SOCP is the problem:

$$\begin{array}{ll} \text{minimize} & f^T x \\ \text{subject to} & \|A_i x + b_i\|_2 \leq c_i^T x + d_i, \quad i = 1, \ldots, m \\ & Fx = g \end{array}$$

with variable $x \in \mathbf{R}^n$

we call a constraint of the form

$$\|Ax + b\|_2 \leq c^T x + d, \quad A \in \mathbf{R}^{k \times n}$$

a **second-order cone constraint** since it requires the affine function $(Ax + b, c^T x + d)$ to lie in the second-order cone in $^{k+1}$

if $c_i = 0, i = 1, \ldots, m$, the SOCP is equivalent to a QCQP

Common convex functions and convexity proof

we list some common convex functions and give simple proof

- estimation (regression, machine learning)
- risk measure
- conic problems

Convex functions in estimation

let $\ell(u)$ be a convex loss function where u is typically a residual error; if $u = y - \hat{y}(\theta)$ is linear in θ then $\ell(y - \hat{y}(\theta))$ is convex in θ

proof: a composition of convex with affine function is convex

example of convex loss function ℓ:

- negative log-likelihood of some distributions
- $\|\cdot\|_2^2$ or any norm: $\ell(\cdot) = \|\cdot\|$
- Huber loss

in which scenario is $\hat{y}(\theta)$ linear in θ?

- $\hat{y}(\theta)$ is a sum of basis functions (polynomials, Fourier, etc.)
- $\hat{y}(\theta)$ is the output of feed-forward NN where all the weights in hidden layers are frozen, and only the weights in output layers are parameters

Sum of K-largest function

for $x \in \mathbf{R}^n$, denote $x_{[k]}$ the k^{th} largest entry with $x_{[1]} \geq x_{[2]} \geq \cdots \geq x_{[n]}$

the sum of K-largest function whered $1 \leq K \leq n$, defined as

$$f(x) = x_{[1]} + x_{[2]} + \cdots + x_{[K]}$$

is convex

- proof: $f(x) = \max\{x_{i1} + x_{i2} + \cdots + x_{ir} \mid 1 \leq i_1 < i_2 < \cdots < i_r \leq n\}$
- f is the maximum of all possible sums of r different components (pointwise maximum of convex is convex)

more generally, $g(x) = \sum_{i=1}^{K} w_i x_{[i]}$ is convex provided that $w_1 \geq w_2 \geq \cdots \geq w_r \geq 0$

when $K = n$, g is called **ordered weighted ℓ_1 norm**

Sum of K-largest function

the function $f(x) = \sum_{i=1}^{K} x_{[i]}$ is the optimal value of the following LP with its dual:

$$\begin{array}{llll} \text{(P)} & \text{maximize}_y & x^T y & \\ & \text{subject to} & 0 \preceq y \preceq \mathbf{1}, & \\ & & \mathbf{1}^T y = K. & \end{array} \qquad \begin{array}{llll} \text{(D)} & \text{minimize}_{t,z} & Kt + \mathbf{1}^T z & \\ & \text{subject to} & t\mathbf{1} + z \succeq x, & \\ & & z \succeq 0. & \end{array}$$

- from the primal, $f(x) = \sup_{y \in C} x^T y$
- the pointwise supremum of linear function in x over y is convex (another convexity proof)
- from the dual, $f(x) \leq c$ is feasible if and only if the followings are feasible

$$Kt + \mathbf{1}^T z \leq c, \;\; t\mathbf{1} + z \succeq x, \;\; z \succeq 0$$

(the sum of K-largest constraint can be presented as linear constraints)

Asset portfolio risk management: Sum-of-largest-K

variable: x (allocation of production rate)

objective: $(1/2)x^T\Sigma x$ (the risk)

constraints:

1. allocation constraints: $0 \preceq x \preceq \mathbf{1},\ \mathbf{1}^T x = 1$
2. demand requirement: $\mu^T x \geq D$ where D is the total required demand
3. physical capacity limit: $0 \preceq x \cdot D \preceq C$ where $C \in \mathbf{R}^n$ is the maximum capacity
4. **bound on K-largest allocations:** $\sum_{i=1}^K x_{[i]} \leq \gamma$ where $0 < \gamma \leq 1$

optimization: quadratic program (QP)

$$\begin{array}{ll} \text{minimize} & (1/2)x^T\Sigma x - \lambda\mu^T x \\ \text{subject to} & 0 \preceq x \preceq \mathbf{1},\ \mathbf{1}^T x = 1 \\ & \mu^T x \geq D,\ \ 0 \preceq x \cdot D \preceq C \\ & \sum_{i=1}^K x_{[i]} \leq \gamma \end{array}$$

with variable $x \in \mathbf{R}^n$

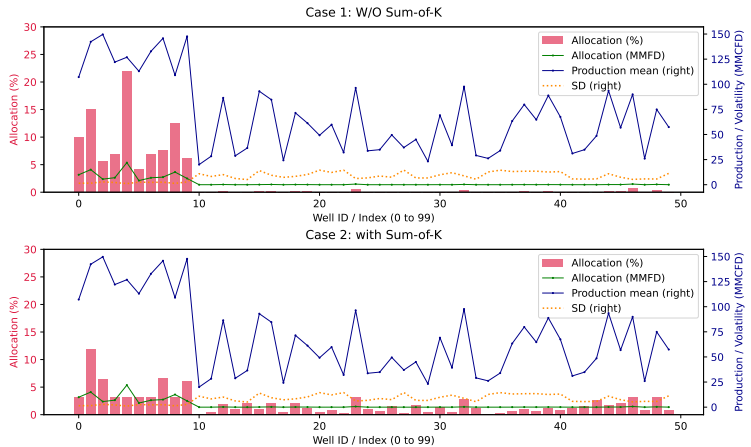
Asset portfolio risk management: Sum-of-largest-K

why should we worry about the K-largest allocations?

- **prevent high-consequence single points of failure:** if the portfolio dominantly allocates high fraction to just a few wells, when one of these wells faces an unplanned shutdown, it would significantly affect the total supply targets
- **reservoir depletion and management:** over-relying on the top few high-performing wells can lead to rapid localized pressure drops or premature water/gas coning in the reservoir ✎ the sum-of-K-largest constraint prevents the optimization from aggressively draining the most attractive assets all at once

Asset portfolio risk management: Sum-of-largest-K result

$n = 50$ with 10 dominant wells, set $K = 10, \gamma = 0.5$



with sum-of-K-largest constraint, the portfolio is more diversified

Sum of K-largest: Connection to CVaR

from duality on page 222, the function can be represented as

$$\begin{aligned}\sum_{i=1}^{K} x_{[i]} &= \underset{t\in\mathbf{R}}{\text{minimize}} \left[K\cdot t + \mathbf{1}^T \max(0, x - t\mathbf{1})\right] \\ &= \underset{t\in\mathbf{R}}{\text{minimize}} \left[K\cdot t + \sum_{i=1}^{n}(x_i - t)_+\right]\end{aligned}$$

Conditional Value-at-Risk (CVaR)

in risk management, let X be a discrete RV representing a loss and with pmf p

definitions:

- **Value-at-Risk** (VAR_α) is the α-quantile of the loss distribution
- **CVaR** (aka Expected Shortfall) is the expected loss given that the loss exceeds VAR_α

CVaR at a confidence level α can be written using the Rockafellar-Uryasev formula:

$$\begin{aligned}\text{CVaR}_\alpha(X) &= \min_{t\in\mathbf{R}} \left[t + \frac{1}{1-\alpha} \sum_{i=1}^{n} p_i (X_i - t)_+ \right] \\ &= \min_{t\in\mathbf{R}} \left[t + \frac{1}{n(1-\alpha)} \sum_{i=1}^{n} (X_i - t)_+ \right] \quad \text{(assume } p_i = 1/n\text{)}\end{aligned}$$

Sum of K-largest: Connection to CVaR

for CVaR, multiply the constant factor $n(1-\alpha)$:

$$n(1-\alpha)\cdot \mathsf{CVaR}_\alpha(X) = \min_{t\in\mathbf{R}} \left\{ \Big[n(1-\alpha)\Big]\cdot t + \sum_{i=1}^{n}(X_i - t)_+ \right\}$$

by setting the vector of losses X equal to asset allocation vector x,

$$K = n(1-\alpha) \quad \Longrightarrow \quad \alpha = 1 - \frac{K}{n} \quad \Longrightarrow \quad \sum_{i=1}^{K} x_{[i]} = K \cdot \mathsf{CVaR}_{1-K/n}(x)$$

- setting $\sum_{i=1}^{5} x_{[i]} \leq 0.4$ means that

$$K = 5, \alpha = 1 - 5/100 = 0.95 \ \ (95\%) \ \ , \mathsf{CVaR}_{0.95}(x) \leq \frac{0.4}{5} = 0.08 \ \ (8\%)$$

- the average allocation of our top 5% most heavily relied-upon wells cannot exceed 8% of the total target production

Robust LP: Uncertainty set approach on objective

consider an LP: minimize $c^T x$ subject to $Ax \preceq b$ but c is uncertain

goal: find a framework that the optimal solution is robust to c

- uncertainty set modeling: assume $c \in \mathcal{U}$ where prior knowledge about the structure of $\mathcal{U}$ is assumed
- use the information of $\mathcal{U}$ to consider the **worst-case** objective

$$w(x) = \sup_{c \in \mathcal{U}} \; c^T x$$

and define a robust LP as the minimization of the worst-case objective instead

$$\begin{array}{ll} \text{minimize}_x & \sup_{c \in \mathcal{U}} \; c^T x \\ \text{subject to} & Ax \preceq b \end{array}$$

the robust LP is convex because $w(x)$ is pointwise maximum of linear function in x

the problem structure of the resulting robust LP is determined by the set $\mathcal{U}$

Robust LP: Uncertainty set approach on objective

example 1: $\mathcal{U} = \{\bar{c} + u \mid \|u\|_\infty \leq \delta \}$

- c has a nominal value plus a perturbation u with maximum magnitude of δ
- the worst-case objective can be derived as

$$c^T x = \bar{c}^T x + u^T x \quad \Rightarrow \quad \sup_{c \in \mathcal{U}} c^T x = \bar{c}^T x + \sup_{\|u\|_\infty \leq \delta} u^T x = \bar{c}^T x + \delta \|x\|_1$$

(using Holder inequality: $x^T y \leq \|x\|_\infty \|y\|_1$ and the inequality can be tight)

robust LP: minimize $\bar{c}^T x + \delta \|x\|_1$ subject to $Ax \preceq b$

the resulting problem can be further cast as an LP

Robust LP: Uncertainty set approach on objective

example 2: $\mathcal{U} = \{c \mid Fc \preceq g\}$

- the description of uncertainty set is generally described by a polyhedron
- *e.g.*, c deviates less than 25% from its nominal and the average of c does not deviate more than 10% from the average of the nominal:

$$0.75\bar{c} \preceq c \preceq 1.25\bar{c}, \quad 0.9(\mathbf{1}^T\bar{c})/n \leq (\mathbf{1}^T c)/n \leq 1.1(\mathbf{1}^T\bar{c})/n$$

- apply duality theory to find the worst-case objective:

$$\text{primal: } \max_{c\in\mathcal{U}} c^T x \quad \text{dual: } \min_{z} g^T z \text{ subject to } -x + F^T z = 0, \ \ z \succeq 0$$

(strong duality holds: primal and dual optimum are equal)

robust LP: the two minimizations can be merged into one over x, z

$$\begin{array}{ll} \text{minimize}_{x,z} & g^T z \\ \text{subject to} & -x + F^T z = 0, \ \ z \succeq 0, \ \ Ax \preceq b \end{array}$$

Robust LP: Uncertainty set approach on constraints

consider an LP: minimize $c^T x$ subject to $Ax \preceq b$ but A is uncertain

goal: find a framework that the optimal solution is robust to A

suppose the row vectors of A, a_i^T for $i = 1, 2, \ldots, m$ are uncertain; each $a_i \in \mathcal{U}_i$ (belong to some uncertainty set)

robust LP: the worst-case of $a_i^T x$ should be still less than b_i:

the constraint $Ax \preceq b$ is replaced by

$$\sup_{a_i \in \mathcal{U}_i} a_i^T x \leq b_i, \quad i = 1, 2, \ldots, m$$

the resulting robust LP depends on the characterization of the supremum

Robust LP: Uncertainty set approach on constraints

example: coefficient a belongs to a box uncertainty set

$$\mathcal{U}_i = \{\, \bar{a}_i + u \mid \|u\|_\infty \le \gamma_i \,\}$$

- use the fact that $y^T x \le \|y\|_\infty \|x\|_1$ (tight when $y = \mathbf{sign}(x)$)

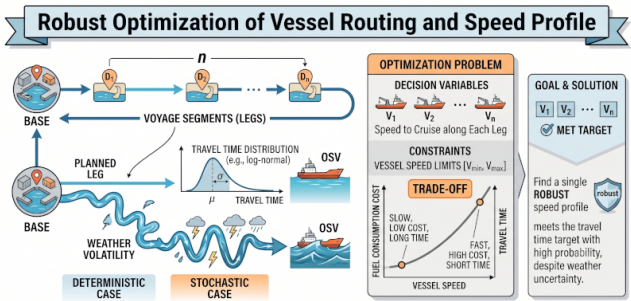
$$\sup_{a_i \in \mathcal{U}_i} a_i^T x = \bar{a}_i^T x + \sup_{\|u\|_\infty \le \gamma_i} u^T x = \bar{a}_i^T x + \gamma_i \|x\|_1$$

(supremum holds when $u = \gamma_i \mathbf{sign}(x)$)

robust LP: can be futher cast as an LP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \bar{a}_i^T x + \gamma_i \|x\|_1 \le b_i, \quad i = 1, 2, \ldots, m \end{array}$$

Robust optimization of vessel routing



goal: decide the speed to cruise the vessel along each leg to meet the travel time target under vessel speed constraints and weather volatility information

- plan the speed of an offshore vessel that travel along a sequence of n voyage segments (legs) starting from base to each destination and back to base
- the vessel travels along each leg under different volatile weather conditions, making the travel time a random quantity
- the trade-off is between speed and fuel economy; increase the speed, finish the trip fast but consume more cost

Robust optimization of vessel routing: Variables

n: the number of legs (travel segments), *e.g.*, base–>1–>2–>3–>4–>base ($n = 5$)

parameters:

- $d \in \mathbf{R}_+^n$: the distance vector of voyage segments
- σ_k^2 for $k \in [n]$: the variance of day per hour of transit (due to weather uncertainty)
- $T_{\max} \in \mathbf{R}_+$: hard deadline for the entire voyage loop (hours)
- $C \in \mathbf{R}_+$: fuel price factor (USD/consumption)

variables:

- $\tau \in \mathbf{R}_+^n$: planned nominal transit time per leg (hours)
- $\Gamma \in \mathbf{R}_+$: total voyage time standard deviation

Robust optimization of vessel routing: Modeling

- the fuel consumption rate per hour scales cubically with speed
- fuel versus time: since speed is $v_k = d_k/\tau_k$,

$$\text{fuel}_k = \text{rate} \times \text{time} \propto \left(\frac{d_k}{\tau_k}\right)^3 \times \tau_k = \frac{d_k^3}{\tau_k^2}$$

- for each leg k, assume there is a random delay ξ_k scales with the planned nominal transit time τ_k:

$$\xi_k = \sigma_k \cdot \tau_k \cdot Z_k \quad \text{where } Z_k \sim \mathcal{N}(0,1)$$

(assume that ξ_k's are uncorrelated)

- the travel time on leg k is $T_k = \tau_k + \xi_k$ and hence, $\mathbf{var}[T_k] = \sigma_k^2 \tau_k^2$
- the total travel time is $T = T_1 + \cdots + T_n$; hence, it is also Gaussian with mean $\sum_k \tau_k$ and the standard deviation of

$$\Gamma = \sqrt{\sum_{k=1}^{n} \sigma_k^2 \tau_k^2} \quad \text{(as } \xi_k\text{'s are uncorrelated)}$$

Robust optimization of vessel routing: Objective and Contraints

objective: the economic cost of fuel consumption ($): $f(\tau) = C \sum_{k=1}^{n} \frac{d_k^3}{\tau_k^2}$ (convex)

constraints:

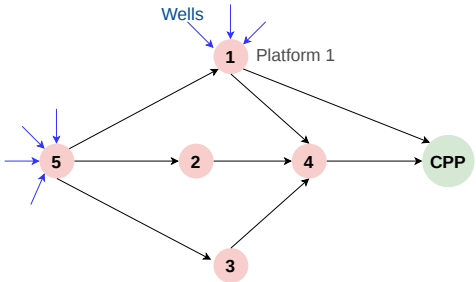
1. vessel speed limit: can be reflected in limit of travel time: $\tau_{\min} \preceq \tau \preceq \tau_{\max}$
2. chance-constrained deadline: $P(T \leq T_{\max}) \geq \alpha$ where α is a confidence level, *e.g.*, 0.95 and use $T \sim \mathcal{N}(\mathbf{1}^T\tau, \Gamma^2)$

$$F(T_{\max}) \geq \alpha \quad \Leftrightarrow \quad T_{\max} \geq F_T^{-1}(\alpha) \quad \Leftrightarrow \quad \mathbf{1}^T\tau + z_{1-\alpha} \cdot \Gamma \leq T_{\max}$$

3. STD relaxation: relax $\sqrt{\sum_{k=1}^{n} \sigma_k^2 \tau_k^2} = \Gamma$ to an inequality in SOCP form:

$$\left\| \begin{bmatrix} \sigma_1\tau_1 \\ \vdots \\ \sigma_n\tau_n \end{bmatrix} \right\|_2 \leq \Gamma \quad \text{(the inequality should be tight at optimum)}$$

Production optimization with network constraints



setting:

- m wellhead platforms routes gas, condensate, and water through a central processing platform
- each platform has n_j wells

wells compete for limited processing capacity at the host platform while their individual performance is dictated by subsea network pressures

goal: maximize daily revenue while satisfying strict facility and reservoir security bounds

Variables

for a network of m platforms and one CPP: $n = \sum_{j=1}^{m} n_j$ is the number of all wells

- $j \in [m]$ is the platform index
- $i \in [n_j]$ is the well index
- $g \in \mathbf{R}_+^n$: gas production rate of well (MMCFD)
- $c \in \mathbf{R}_+^n$: condensate production rate of well (stb/d)
- $w \in \mathbf{R}_+^n$: water production rate of well (stb/d)
- $p \in \mathbf{R}_+^{m+1}$: nodal pressure at platform j (psig)

we model that condensate and water rates are functions of gas rate via

$$c_i = \mathrm{GOR}_i \cdot g_i, \quad w_i = \mathrm{WGR}_i \cdot g_i$$

(GOR = gas-oil ratio and WGR = water-gas ratio)

Objective

the objective is to maximize net daily asset revenue: maximize $f(g, p)$

- revenue $= R_{\text{gas}} \mathbf{1}^T g + R_{\text{cond}} \mathbf{1}^T c$
- environmental cost incurred from processing water: $C_{\text{water}} \mathbf{1}^T w$
- fuel consumption of compressor: offshore platforms do not have access to an onshore electrical grid; to run multi-megawatt compressors that boost gas, they burn a fraction of the produced natural gas as fuel in gas turbines suppose there are N_{comp} compressors and j is the compressor index

$$P_j(g) = a_j \left(\sum_{i \in W_j} g_i \right)^2 + b_j \left(\sum_{i \in W_j} g_i \right) + c_j$$

  - W_j is the subset of wells routing into compressor j
  - $a_j, b_j, c_j \geq 0$ are empirical quadratic coefficients derived from compressor's characteristics

Objective

option 1: neglect the fuel consumption

$$f_1(g) = \sum_{i=1}^{n} \left(R_{\text{gas}} + R_{\text{cond}} \cdot \text{GOR}_i - C_{\text{water}} \cdot \text{WGR}_i\right) g_i$$

the objective is linear in g

justification: 1) if the compressors are running at fixed speed, it consumes relatively constant fuel gas or 2) absorb this cost in lower revenue coefficient, R_{gas}

option 2: include the fuel consumption

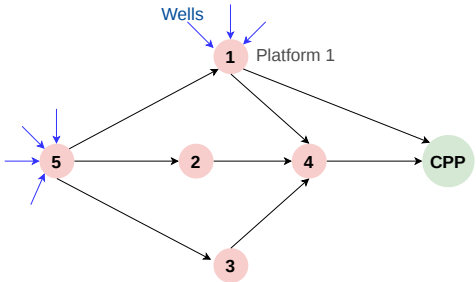
$$f_2(g) = f_1(g) - \sum_{j=1}^{N_{\text{comp}}} a_j \left(\sum_{i \in W_j} g_i\right)^2 + b_j \left(\sum_{i \in W_j} g_i\right) + c_j$$

the objective is quadratic in g

Constraints

1. total gas handling capacity: the export compressors cannot exceed a maximum throughput: $\sum_{i=1}^{n} g_i \leq Q_{\text{gas\_max}}$
2. produced water disposal limit: $\sum_{i=1}^{n} \text{WGR}_i \cdot g_i \leq Q_{\text{water\_max}}$
3. slug/liquid capacity: total liquid (condensate+water) entering the reception must not flood the system: $\sum_{i=1}^{n} (\text{GOR}_i + \text{WGR}_i) g_i \leq Q_{\text{liquid\_max}}$
4. well operational limit: $G_{\text{min}} \preceq g \preceq G_{\text{max}}$
5. **pressure dictates the flow rates:** well's flow rate (g) and pressure (p) are bound together by the laws of fluid mechanics
6. **network flow:** the platform network topology defines the flow layout specifying the upstream and downstream sides and the mass balance constraint

Platform network



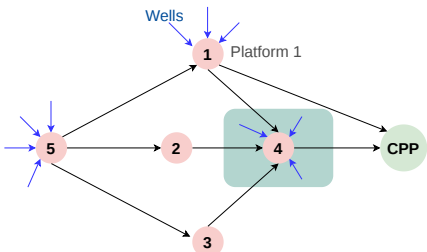
- $j \in [m]$ is the platform index
- each platform has n_j wells connected
- pipelines connect between platforms

- upstream and downstream are designated from the flow (high to low pressure)
- modeled as a directed graph of $m + 1$ nodes and pipelines are the set of direct edges from node i to j
- CPP (central processing platform) is the sink node, collecting all field production

Mass balance constraint

variable: pipeline flow variable

- g_{ud} is defined as the volumetric gas flow rate travelling *through* the specific pipeline that connects node u to node d
- g_{ud} is not the flow of a single well but the cumulative sum of all fluids passing through that pipe segment



Kirchhoff's current law for fluids:

at every node j which has n_j wells

$$\sum_{i \in [n_j]} g_i + \sum_{u \in \text{upstream}} g_{uj} = \sum_{d \in \text{downstream}} g_{jd}$$

Pressure/flow rate constraint

the Weymouth equation:

$$g_{ud}^2 = K_{ud}\left(p_u^2 - p_d^2\right)$$

- g_{ud} is gas flow rate from upsteam to downstream
- K_{ud} is a constant based on pipeline diameter, length friction, and gas gravity
- p_u, p_d are upstream/downstream pressures

the quadratic equality constraints from Weymouth equation is **non-convex**

- for gas to flow from u to d, we need $p_u > p_d$; the bigger the pressure gap, the more gas the pipe can physically clear
- K_{ud} acts as a hydraulic conductivity; if the pipeline is long, $_{ud}$ is small, meaning that we need a high pressure difference to push the gas through

Pressure/flow rate SOCP relaxed constraint

apply a **convex relaxation**

$$g_{ud}^2 \leq K_{ud}\left(p_u^2 - p_d^2\right)$$

when the objective favors pushing large g to maximize the production, we hope the relaxation becomes tight at optimum

we further introduce auxiliary variables: $\phi_u = p_u^2$ and $\phi_d = p_d^2$ and re-arrange terms:

$$\left\| \begin{matrix} g_{ud} \\ K_{ud}(\phi_u - \phi_d) - 1 \end{matrix} \right\|_2 \leq K_{ud}(\phi_u - \phi_d) + 1$$

this is a **second-order cone** constraint in variables g_{ud}, ϕ_u, ϕ_d

(use identity: $(2x)^2 + (y-1)^2 \leq (y+1)^2 \Leftrightarrow x^2 \leq y$ where $x = g_{ud}$ and $y = K\Delta\phi$)

Pressure physical constraints

1. the fixed boundary condition at CPP: p_{cpp} is fixed constant by facility valves
2. maximum pressure limits: $p_j \leq p_{\max,j}$ for $j \in [m]$ (metallurgical wall)
3. inflow performance relationship (IPR): the pressure at wellhead relates to reservoir pressure and flow rate by

$$p_{\text{wh},i} \leq P_{\text{reservoir},i} - \alpha \cdot g_i$$

(if g_i increases, $p_{\text{wh},i}$ drops because of reservoir depletion; the maximum pressure a well can ever hits is its static reservoir shut-in pressure, $P_{\text{reservoir},i}$)

for fluid to physically flow from well i into the platform manifold j, the wellhead pressure must be higher than (or equal to) the platform manifold pressure

$$p_j \leq p_{\text{wh},i} \quad \Rightarrow \quad p_j \leq P_{\text{reservoir},i} - \alpha \cdot g_i, \quad \Leftrightarrow \quad \sqrt{\phi}_j \leq P_{\text{reservoir},i} - \alpha \cdot g_i \quad j \in [m]$$

Production optimization with network constraints

combining all objective and constraints, we observe that

- the flow rate of one node affects the constraint of neighboring nodes and has to satisfy the Weymouth relaxation
- the problem parameters highly affect problem feasibility; if the values are not realistic, the mass balance constraints are hard to acheive
- even for a simple network where mass balance constraint can be achieve; however, the relaxation is not tight at optimum

Structured convex problems

some structures that are amenable for parallel and distributed algorithms

- **separable sum**

$$\underset{x_1,\dots,x_m}{\text{minimize}} \quad f(x) := \sum_{i=1}^{m} f_i(x_i)$$

it is obvious that we can minimize over x_i independently

- **global consensus**

$$\underset{x}{\text{minimize}} \quad f(x) := \sum_{i=1}^{m} f_i(x)$$

f_i is a local objective; x is the global variable

consensus form: add a consensus constraint that makes all local x_i's agree

$$\underset{x_1,\dots,x_m}{\text{minimize}} \quad \sum_{i=1}^{m} f_i(x) \quad \text{subject to} \quad x_1 = x_2 = \cdots = x_m$$

Structured convex problems

- **global exchange**

$$\underset{x}{\text{minimize}} \quad \sum_{i=1}^{m} f_i(x_i) \quad \text{subject to} \quad \sum_{i=1}^{m} x_i = 0$$

interpretation: x_i's are quantities of commodities exchanged among m agents

goal: minimize total social cost subject to the market clearing

- **allocation**

$$\underset{x}{\text{minimize}} \quad \sum_{i=1}^{m} f_i(x_i) \quad \text{subject to} \quad x_i \geq 0, \ \sum_{i=1}^{m} x_i = b$$

interpretation: x_i's are non-negative resources allocated to m activities

goal: minimize each activity cost while the total resource is limited to a budget

Distributed model fitting

a problem of fitting y using a linear model Ax using a loss function l

$$\underset{x}{\text{minimize}} \quad l(Ax - y) + r(x)$$

$l(Ax - y) = \sum_{i=1}^{N} l_i(a_i^T x - y_i)$ represents the model cost due to error $Ax - y$

r is a separable function representing **regularization**, *e.g.*, $\|\cdot\|_1, \|\cdot\|_2^2$

this is an example of global consensus

a common model parameter x that makes the model fits with *all* data samples

Scalarized multi-objective optimization

a common form of multi-objective problem: for a given $\gamma > 0$,

$$\text{minimize} \quad f(x) + \gamma g(x)$$

- we desire both f and g to be small but they are weighed in by a given weight, γ (or often called penalty parameter)
- as γ is higher, we penalize more on g, then the minimized g is smaller; in this case, we care less about f
- appear in model performance evaluation where two diffferent metrics are desired to be small
- example 1: minimize model error + model complexity
- example 2: minimize system tracking error + input power

Multi-objective optimization

setting: minimizing $f_0 : \mathbf{R}^n \to \mathbf{R}^m$ (vector-valued function) over a feasible set

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & x \in \mathcal{C} \end{array}$$

a vector optimization has a **vector-valued** objective function

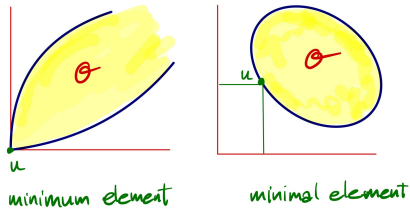
- example: $f_0(x) = (\text{fuel,time})$ the energy used and time spent of a vehicle parameter x
- require a generalized inequality definition for comparing any two vectors of $f_0(x)$

$$\begin{bmatrix} 5 \\ 2 \end{bmatrix} \preceq \begin{bmatrix} 10 \\ 3 \end{bmatrix} \quad \text{but} \quad \begin{bmatrix} 5 \\ 2 \end{bmatrix} \not\preceq \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

here, for $f_0(x) \in \mathbf{R}^n$, we typically use the **non-negative orthant** to define $\preceq$

Achievable objective values

define $\mathcal{O} = \{ f_0(x) \mid x \in \mathcal{C} \}$ the set of objective values of feasible points



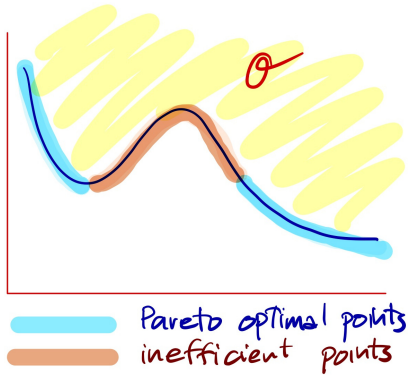
- u is said to be the **minimum** element of $\mathcal{O}$ if $u \preceq v$, for every $v \in \mathcal{O}$
- u is said to be a **minimal** element of $\mathcal{O}$ if $v \in \mathcal{O}$, $v \preceq u$ only if $v = u$
- if $\mathcal{O}$ has a minimum point (then it is unique) and

$$\exists \text{ feasible } x \quad \text{such that} \quad f_0(x) \preceq f_0(y), \quad \text{for all feasible } y$$

then we say x is **optimal**

Pareto optimal points

consider when $\mathcal{O}$ does not have a minimum element



- x is called **Pareto optimal** (or efficient) if $f_0(x)$ is a minimal element of $\mathcal{O}$
- a technique to extract pareto optimal points: scalarization (more on this later)

Overview of available methods

- unconstrained problems: gradient descent, Newton, quasi Newton, trust-region
- convex programs: interior point, gradient projection, ellipsoid method
- convex programs of certain structures: proximal methods
- linear programming: simplex, interior point
- quadratic programming: interior point, active set, conjugate gradient, augmented Lagrangian

Softwares

MATLAB: cvx

- CVX is a MATLAB-based modeling system for convex optimization
- `http://cvxr.com/cvx/`

Python

- **CVXPY**: Python-embedded modeling language for convex optimization problems available at `https://www.cvxpy.org/` by Stephen Boyd group
- **CVXOPT**: Python-based package for convex optimization available at `http://cvxopt.org/` by M. Andersen, J. Dahl and L. Vandenberghe

References

1. S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, 2004
2. G. Calafiore and L. El Ghaoui, *Optimization Models*, Cambridge University Press, 2014
3. Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, KAP, 2004
4. R.T. Rockfellar, *Convex Analysis*, Princeton University Press, 1970
5. S.Boyd, N. Parikh, E. Chu, B.Peleato, and J. Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, Foundations and Trends in Machine Learning, 2011

A5: Optimization case studies by PTTEP

Optimal location of wellhead platform

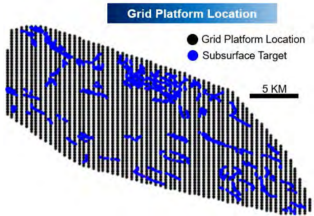


Figure 3—Grid Platform location to cover subsurface target

Platform / Well	W1	W2	W3	W4	W5	W6
P1	1	1	0	0	1	0
P2	0	0	1	1	1	0
P3	0	1	1	0	0	0
P4	1	0	0	0	1	0
P5	0	0	0	0	1	1
P6	1	0	0	0	0	0
P7	0	0	0	0	0	1

source: https://onepetro.org/SPEAPOG/proceedings-abstract/21APOG/21APOG/D011S009R004/470152

setting:

- there are P platforms and W wells connected by a covarage matrix
- each well has different hydrocarbon resources and distance to the connected platform
- each platform's total gas resource must pass the minimum cutoff constraint

goal: K platforms that maximize the hydrocarbon resource with smallest total distance

Variables

variables:

- $A \in \{0,1\}^{P \times W}$: assignment matrix where a_{ij} shows that the well j is assigned to platform i
- $w \in \{0,1\}^{W}$: $e_i = 1$ if the well i is selected
- $p \in \{0,1\}^{P}$: $p_i = 1$ if the platform i is selected

parameters:

- $K \in \mathbf{Z}_+$: a target number of platforms
- $g \in \mathbf{R}^{W}$: the reserve coefficient vector (hydrocarbon value)
- $D \in \mathbf{R}^{P \times W}$: distance matrix with d_{ij} the distance from well j to platform i
- $N_{\text{well}} \in \mathbf{Z}^{P}$: number of wells in the platform can handle
- $\text{OGIP}_{\text{cut-off}}$: original gas in place threshold

Constraints

1. target number of platforms: $\mathbf{1}^T p = K$
2. platform coverage: $\sum_{i=1}^{P} C_{ij} p_i \geq w_j, \quad j = 1, 2, \ldots, W$
3. platform assignment: only one platform can be assigned to a selected well

$$\sum_{i=1}^{P} a_{ij} = w_j, \quad j = 1, 2, \ldots, W \quad \Leftrightarrow \quad \mathbf{1}^T A = w$$

4. only selected platform can be assigned:

$$\sum_{j=1}^{W} a_{ij} \leq p_i N_{\text{well},i}, \quad i = 1, 2, \ldots, P \quad \Leftrightarrow \quad A\mathbf{1} \preceq N_{\text{well}}$$

5. OGIP constraint:

$$\sum_{j=1}^{W} a_{ij} g_j \geq \text{OGIP}_{\text{cut-off}} \cdot p_i, \quad i = 1, 2, \ldots, P \quad \Leftrightarrow \quad Ag \succeq p$$

6. physical constraint: $A \preceq C$ (can only be assigned if covered by C)

Objective

we maximize: hydrocarbon value - total distance to platform

$$\sum_{j=1}^{W} g_j w_j - \lambda \sum_{j=1}^{W} \sum_{i=1}^{P} d_{ij} a_{ij}$$

the objective in vector form

$$g^T w - \lambda \mathbf{1}^T (D \odot A) \mathbf{1}$$

- $\lambda > 0$ is a trade-off constant between two objectives
- the objective is linear in w, A (variables)

the optimization problem is a pure linear IP (integer programming)

Results: without and with $A \preceq C$

```
Solver Status: optimal
Optimal hydrocarbon value: 596.792
Optimal distance: 20.179
Optimal Objective Value with lambda: 556.43

----Coverage matrix -----
[[1 1 0 0 1 0]
 [0 0 1 1 1 0]
 [0 1 1 0 0 0]
 [1 0 0 0 1 0]
 [0 0 0 0 1 1]
 [1 0 0 0 0 0]
 [0 0 0 0 0 1]]
--- Selection Vectors ---
Selected Platforms (p): [1 1 0 0 1 0 0]
Selected platform 1-based index: [1 2 5]
Selected Wells (w):     [1 1 1 1 1 1]

--- Assignment Matrix A (P x W) ---
[[1 0 0 0 1 0]
 [0 0 0 1 0 1]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 1 1 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]]

--- Sanity Checks ---
Platform 1 assigned gas = 153.1 (Must be >= 100.0)
Platform 2 assigned gas = 175.5 (Must be >= 100.0)
Platform 5 assigned gas = 268.3 (Must be >= 100.0)
```

```
Solver Status: optimal
Optimal hydrocarbon value: 596.792
Optimal distance: 36.508
Optimal Objective Value with lambda: 523.78

----Coverage matrix -----
[[1 1 0 0 1 0]
 [0 0 1 1 1 0]
 [0 1 1 0 0 0]
 [1 0 0 0 1 0]
 [0 0 0 0 1 1]
 [1 0 0 0 0 0]
 [0 0 0 0 0 1]]
--- Selection Vectors ---
Selected Platforms (p): [1 1 0 0 1 0 0]
Selected platform index (1-based index): [1 2 5]
Selected Wells (w):     [1 1 1 1 1 1]

--- Assignment Matrix A (P x W) ---
[[1 1 0 0 0 0]
 [0 0 1 1 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 0 1 1]
 [0 0 0 0 0 0]
 [0 0 0 0 0 0]]

--- Sanity Checks ---
Platform 1 assigned gas = 232.5 (Must be >= 100.0)
Platform 2 assigned gas = 233.1 (Must be >= 100.0)
Platform 5 assigned gas = 131.2 (Must be >= 100.0)
```

Result

without versus with $A \preceq C$

without $A \preceq C$

- the optimal hydrocarbon are the same (as all wells are selected)
- the distance is smaller (better)
- the optimal value with λ-trade-off is higher (better)
- **catch!** but the assignment matrix is not realizable!
  - well 6 is assigned to platform 2
  - but well 6 can only be covered by platform 5 and 7 (check from C)

Discussion

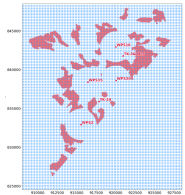
without $A \preceq C$, it might encounter a loophole that we assign A to maximize gas and minimize distance but forget to realize the reality check that $a_{ij} = 1$ only if $c_{ij} = 1$ too

the constraint 2 catch: $\sum_{i=1}^{P} C_{ij} p_i \geq w_j$

- if we select well j, we must build at least one platform $p_i = 1$ that is mathematically capable of covering it $c_{ij} = 1$
- it dictates which platform are built (p), but it says nothing about how A is filled out

Well combination optimization

setting:



Well	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	G12	G13	G14	G15
W1															
W2															
W3															
W4															
W5															
W6															

- given information about n_w wells and 2D n_g grids via a coverage matrix C
- $c_{ij} = 1$ if the well i is laid out on grid j
- each grid j has different value
- some grids may be covered by more than one well

goal: select K wells that maximize the total value while avoiding the number of overlapped grids

Well combination: Formulation 1

variables:

- $x \in \{0,1\}^{n_g}$: $x_i = 1$ if grid i is selected
- $y \in \{0,1\}^{n_w}$: $y_i = 1$ if well i is selected

parameters:

- $w \in \mathbf{R}^{n_g}$: the value coefficient of each grid
- $C \in \{0,1\}^{n_w \times n_g}$: the coverage matrix
- $K \in \mathbf{Z}_+$: target number of selected wells

objective: maximize $w^T x$

constraint:

1. number of selected wells: $\mathbf{1}^T y = K$
2. well and grid coverage: $\sum_{i=1}^{n_w} c_{ij} y_i \geq x_j, \quad j = 1, 2, \ldots, n_g$

the problem is pure IP where n_g can be in order of million (x)

Well combination: Formulation 2 (BQP)

motivation: avoid using grid variable (higher dimension)

- from the value vector $w \in \mathbf{R}^{n_g}$ and C, we can compute the value of each well, described by $s \in \mathbf{R}^{n_w}$
- from C, we can construct the overlap matrix $O \in \mathbf{Z}^{n_w \times n_w}$ where O_{ij} is the number of overlapped grids between well i and j
- when y is given, we can count the number of total overlapped grids occured in the selected well; let's denote it by

$$h(y) = \sum_{ij} O_{ij} y_i y_j = y^T O y$$

formulation: weighted objective between the total value and number of overlaps

$$\begin{array}{ll} \text{maximize}_y & s^T y - \lambda y^T O y \\ \text{subject to} & \mathbf{1}^T y = K \end{array}$$

the problem becomes **binary quadratic program (BQP)** in $y \in \{0,1\}^{n_w}$

Well combination: Formulation 3

the quadratic function $y^T Oy$ can be linearized using **McCormick Envelopes**

- introduce $z_{ij} = y_i y_j$ (aim $z_{ij} = 1$ only when $y_i = y_j = 1$)
- force z_{ij} to be binary without using product form

$$z_{ij} \leq y_i, \quad z_{ij} \leq y_j, \quad z_{ij} \geq y_i + y_j - 1, \quad z_{ij} \geq 0$$

- as the objective is to minimize $y^T Oy$, it naturally drives z_{ij} to zero
- the third inequality tends to force $z_{ij} = 1$ *only* when $y_i = y_j = 1$

Well combination: Formulation 3

the problem becomes linear IP

$$\begin{array}{ll} \text{maximize}_{y,z} & s^T y - \lambda \sum_{ij} O_{ij} z_{ij} \\ \text{subject to} & \mathbf{1}^T y = K \\ & z_{ij} \le y_i, \;\; z_{ij} \le y_j, \\ & z_{ij} \ge y_i + y_j - 1, \quad z_{ij} \ge 0 \end{array}$$

with variables $y \in \{0,1\}^{n_w}$ and $Z \in \{0,1\}^{n_w \times n_w}$

while the problem is linearized, n_w^2 variables are added (from Z)

Well combination: Formulation 4

motivation: avoid using grid variable by re-arranging the objective in terms of y

- we can compute how many wells cover a grid j: $\sum_{i=1}^{n_w} C_{ij} y_i$ (ranges in $0, 1, \ldots$)
- the status of a grid j is chosen is capped by minimum operation

$$\min\left(\sum_{i=1}^{n_w} C_{ij} y_i, 1\right) \quad \text{(which is nonlinear in } y \text{ and its value is either 0 or 1)}$$

the problem becomes nonlinear IP

$$\begin{array}{ll} \text{maximize}_y & \sum_{j=1}^{n_g} w_j \min\left(\sum_{i=1}^{n_w} C_{ij} y_i, 1\right) \\ \text{subject to} & \mathbf{1}^T y = K \end{array}$$

with variables $y \in \{0, 1\}^{n_w}$

Well combination: Formulation 4

CP-SAT solver has conditional implication that can handle how to evaluate the objective function more effectively without introducing more variables

Well / Grid	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
W1										
W2										
W3										
W4										
W5										
Grid value	59	74	64	59	48	68	49	90	97	45
Selected well (0-based Indices): [1 4]										
Active z (selected profiles with their covering wells):										
[(0, 1), (1,), (1, 2), (3, 4), (4,)]										

introduce compressed profiles (dict):

- profile of covering wells
- combined grid value

Compressed grid profiles

```
Grids 1 has the profile: (1,) (Only Well 1)
Grids 2 has the profile: (1,2) (Well 1 and Welll 2)
Grids 3 has the profile: (2,) (Only Well 2)
Grids 4 has the profile: (2,3) (Well 2 and Well 3)
...
Grids 9 and 10 have the profile: (5,) (Well 5) --> add w9+w10
```

when C has a staircase structure:

the grids in C naturally group into two distinct types of structural columns:

1. exclusive zones: grids covered by exactly one well (e.g., only well 1, only well 2).
2. overlap zones: grids covered by exactly two consecutive wells

let's calculate exactly how many unique keys will end up in the compressed profiles dictionary for $n_w = 100$:

- single-well keys: every well has an exclusive zone; this creates profiles like (0,), (1,), (2,), ..., (99,).

$$\text{Total} = 100 \text{ profiles}$$

- consecutive overlap keys: every consecutive pair of wells shares a few overlap zone; this creates profiles like (0, 1), (1, 2), (2, 3), ..., (98, 99).

$$\text{Total} = 99 \text{ profiles}$$

the dictionary z will contain at most 199 unique keys

it does not matter if n_g is 1 million, or 10 million (for this structure of C)

Profile activation variable

for each covering wells in the compressed profiles, create z_k

```
z_k = model.NewBoolVar(f'z_profile_{covering_wells}')
```

- if $z_k = 1$, it means this grid profile is successfully covered by our well selection, and we get to claim its value using conditional implication

  ```
  model.Add(sum(y[i] for i in covering_wells) >= 1).OnlyEnforceIf(z_k)
  ```

  (at least one well is selected)
- if $z_k = 0$, no wells in this profile can be selected

  ```
  model.Add(sum(y[i] for i in covering_wells) == 0).OnlyEnforceIf(z_k.Not())
  ```

the objective function is $\sum_{k=1}^{\text{no. profiles}} \text{combined weight}_k \cdot z_k$

variables are y (length of n_w) and z (length = number of compressed profiles)

Well combination: Formulation 4 results

10 grids and 5 wells
compress profiles and combined weight:

Well / Grid	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
W1										
W2										
W3										
W4										
W5										
Grid value	59	74	64	59	48	68	49	90	97	45
Selected well (0-based Indices): [1 4]										
Active z (selected profiles with their covering wells):										
[(0, 1), (1,), (1, 2), (3, 4), (4,)]										

(0-based indices)
(0,) : 59
(0,1) : 74
(1,) : 64
(1,2) : 59
(2,) : 48
(2,3) : 68
(3,) : 49
(3,4) : 90
(4,) : 97+45

- well 2 and 5 are selected (index 1 and 4)
- these selected well are covered in the active profiles