Introduction to Mathematical Optimization

Nick Henderson, AJ Friend (Stanford University)
Kevin Carlberg (Meta)

August 5, 2022

Introduction

Outline

Optimization

Optimization find the best choice among a set of options subject to a set of constraints

Formulation in words:

minimize	objective
by varying	variables
subject to	constraints

Applications

- **Portfolio optimization**
  - *Objective*: -expected returns
  - *Variables*: amount of capital to allocate to each available asset
  - *Constraints*: total amount of capital available
- **Transportation problems**
  - *Objective*: transportation cost
  - *Variables*: routes to transport goods between warehouses and outlets
  - *Constraints*: outlets receive proper inventory, vehicle capacity constraints

Applications

- **Control** (model predictive control)
  - *Objective*: difference between model output and desired state over a time horizon
  - *Variables*: control inputs (actuators)
  - *Constraints*: control effort (maximum possible actuation force)
- **Engineering design** (see wing-design example)
  - *Objective*: negative performance (maximize performance)
  - *Variables*: design parameters
  - *Constraints*: manufacturability
- **Model fitting** (statistics, machine learning, and AI)
  - *Objective*: error in model predictions over a training set
  - *Variables*: parameters of the model
  - *Constraints*: model complexity, inductive biases

Much of modern-day AI can be viewed as applied optimization and statistics!

Optimization at the heart of learning-based AI

“People are using the phrase”Artificial Intelligence" in two completely different meanings. A long time ago, there was a debate between people who believed that intelligence was all about **reasoning**. You’d get strings of symbols coming in, and—inside—you’d manipulate strings of symbols using symbolic rules, and then strings of symbols would come out. And then the other school said, no, you have this great big brain with lots of connections, and you change the connection strengths. **It’s not reasoning that’s the essence of intelligence, it’s learning [via optimization]**. And these two schools completely disagreed... What’s happened is, all the progress recently has come from the view of AI that.... **learns from data**. **That second view has completely taken over and it’s what’s making all the progress**." - Geoff Hinton

Mathematical optimization: formulation

"Standard form":

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

- $x \in \mathbf{R}^n$: optimization/decision variable (to be computed)
- $f_0 : \mathbf{R}^n \to \mathbf{R}$: objective/cost function
- $f_i : \mathbf{R}^n \to \mathbf{R}$: inequality constraint functions
- $h_j : \mathbf{R}^n \to \mathbf{R}$: equality constraint functions
- **Feasible set**: $\mathcal{D} = \{x \in \mathbf{R}^n \mid f_i(x) \leq 0,\ i = 1, \ldots, m,\ h_j(x) = 0,\ j = 1, \ldots, p\}$
- **Feasibility problem**: Find $x \in \mathcal{D}$ (determines if the constraints are consistent)
- Specialized fields have other "standard forms"
  - **Reinforcement learning**: Markov Decision Processes (MDP)
  - We won't consider these here, but same principles apply

The big picture

- **Bad news**: *In their fullest generality*, most optimization problems cannot be solved
  - Generally NP-hard
  - Heuristics required, hand-tuning, luck, babysitting
- **Good news**:
  - We can do a lot by *modeling* the problem as a simpler, solvable one
  - Excellent computational tools are available:
    - **Modeling languages** to write problems down (CVX, CVXPY, JuMP, AMPL, GAMS)
    - **Solvers** to obtain solutions (IPOPT, SNOPT, Gurobi, CPLEX, Sedumi, SDPT3)
  - Knowing *a few key problem attributes* facilitates navigating the large set of possible tools and approaches

Key challenge: modeling

Translate real-world problem into standard form

This requires balancing two competing objectives:

1. **Representativeness**
   - Model should closely reflect the actual problem
   - The solution should be **useful**, aligned with the **desired outcome**, and avoid **unintended consequences**
2. **Solvability**
   - Exercise is useless if a solution cannot be computed
   - Time-to-solution constraints (e.g., algorithmic trading) limit model complexity

Importance of modeling for AI

"King Midas specified his objective: I want everything I touch turned to gold [modeling: translated problem into something expressible]. He got exactly what he asked for [optimization problem correctly solved]. Unfortunately, that included his food and his drink and his family members, and he dies in misery and starvation [solution led to unintended consequences]. Many cultures have the same story. The genie grants you three wishes. Always the third wish is "please undo the first two wishes" because I ruined the world." - Stuart Russell

Optimization-problem attributes

Outline

Friends and enemies in mathematical optimization

- **Key problem attributes**:
  - Convexity: convex v. non-convex
  - Optimization-variable type: continuous v. discrete
  - Constraints: unconstrained v. constrained
  - Number of optimization variables: low-dimensional v. high-dimensional
- **These attributes dictate**:
  - Ability to find the solution
  - Problem complexity and computing time
  - Appropriate methods
  - Relevant software

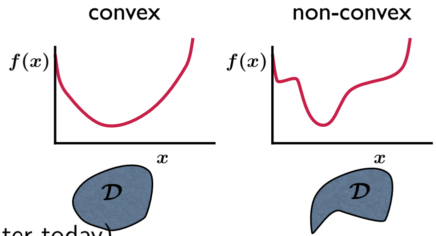
Always begin by categorizing your problem!

Convex v. non-convex

- **Convex problems**:
  - equality constraint functions are affine (linear with offset)
  - objective and inquality constraint functions are convex

  Convex function $f: \; g(tx + (1-t)y) \leq tg(x) + (1-t)g(y), \quad \forall 0 \leq t \leq 1$

  Convex set $\mathcal{D}: \; \sum_i \alpha_i x_i \in S$ for all $\alpha_i, \; x_i$ satisfying $\alpha_i \geq 0, \; \sum_i \alpha_i = 1, \; x_i \in \mathcal{D}$



- **Examples**:
  - Linear least squares (later today)
  - Linear programming (LP): linear objective and constraints (management, finance)
  - Quadratic programming (QP): quadratic objective, linear constraints.

Convex v. non-convex

- **Non-convex problems**:
  1. objective function is nonconvex,
  2. inequality constraint functions are non-convex, **OR**
  3. equality constraints are nonlinear
- **Main challenges**:
  1. Local minimum may not be a global minimum
  2. Cannot verify if we've solved the problem (even if we *have* found the global minimum)

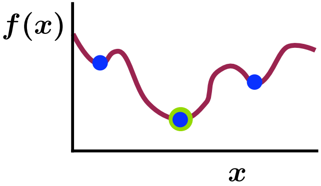


Figure 1: Local and global solutions for a non-convex objective function.

Convex v. non-convex significance

- **Convex**
  - *One unique minimum*: local minimum is the global minimum!
  - *Theory*: convexity theory is powerful
  - *Solution process*: no algorithm tuning or babysitting
  - *Software*: `CVXPY`, a modeling language for convex optimization
- **Non-convex**
  - *Possibly many local minima*: Local minimum may not be global minimum
  - *Theory*: most results ensure convergence to only a *local* minimum
    - This means we **have not really solved the problem!**
  - *Solution process*: often requires significant tuning and babysitting
    - For example, use multiple starting points to try to find global minimum
  - *Software*: `scipy.optimize`, a optimization sub-package of SciPy

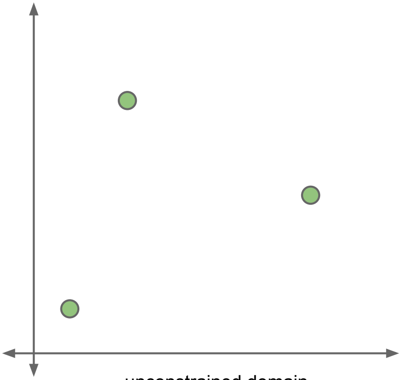
Continuous v. discrete

- **Continuous**
  - For example, $x \in \mathbf{R}^n$
  - Often easier to solve because derivative information can be exploited
- **Examples**
  - parameters in a parametric regression/classification model (e.g., neural network)
  - asset allocation in portfolio optimization
  - position in a coordinate system
  - vehicle speed in a model to minimize fuel consumption
  - wing thickness in aircraft design

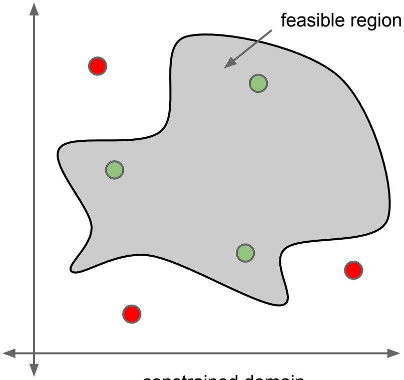
Continuous v. discrete examples

- **Discrete**
  - For example, $x \in \{0, 1, 2, 3, \ldots\}$ or $x \in \{0, 1\}$
  - Always non-convex
  - Often NP-hard
  - Often reformulated as a sequence of continuous problems (e.g., branch and bound)
  - *Sub-types*: combinatorial optimization, (mixed) integer programming
- **Examples of discrete variables**
  - binary selector for facility location, e.g., $x_{ij} = 1$ if and only if resource i is placed in location j and zero otherwise
  - integer representing the number of warehouses to build
  - integer representing the number of people allocated to a task

Unconstrained v. constrained (domain)



unconstrained domain
(all points considered acceptable)



constrained domain
(only green points acceptable)

Unconstrained v. constrained (problem)

- **Unconstrained problems**

$$\text{minimize} \quad f_0(x)$$

  - easier to solve

- **Constrained problems**

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

  - *linear equality constraints*: can apply **null-space/reduced-space methods** to reformulate as an unconstrained problem.
  - *otherwise*: can apply **interior-point methods**, which reformulate as a sequence of unconstrained problems

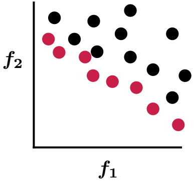
Friends and enemies in mathematical optimization (summary)

- Convexity:
  - convex: local solutions are global
  - non-convex: local solutions are not global
- Optimization-variable type:
  - continuous: gradients facilitate computing the solution
  - discrete: cannot compute gradients, NP-hard
- Constraints:
  - unconstrained: simpler algorithms
  - constrained: more complex algorithms; must consider feasibility
- Number of optimization variables:
  - low-dimensional: can solve even without gradients
  - high-dimensional: requires gradients to be solvable in practice

Always begin by categorizing your problem!

Single-objective v. multi-objective

- What if we care about two competing objectives f_1 and f_2?
  - *Example*: f_1=risk, f_2=negative expected return
- **Pareto front**: set of candidate solutions among which no solution is better than any other solution in both objectives



Each candidate solution is plotted in terms of both objectives.

Pareto-optimal points plotted in red

- Often solved using **evolutionary algorithms**
- Can also minimize the composite objective function for many different values of a:

$$\text{minimize } a \cdot f_1(x) + f_2(x)$$

  - **Warning**: this captures only points on the **convex hull** of the Pareto front

This course

Theory, methods, and software for problems exihibiting the characteristics below

- Convexity:
  - convex: local solutions are global
  - non-convex: local solutions are not global
- Optimization-variable type:
  - continuous: gradients facilitate computing the solution
  - discrete: cannot compute gradients, NP-hard
- Constraints:
  - unconstrained: simpler algorithms
  - constrained: more complex algorithms; must consider feasibility
- Number of optimization variables:
  - low-dimensional: can solve even without gradients
  - high-dimensional: requires gradients to be solvable in practice