Introduction to Mathematical Optimization

Nick Henderson, AJ Friend (Stanford University)
Kevin Carlberg (Sandia National Laboratories)

August 14, 2018

Introduction

Outline

Introduction

Optimization-problem attributes

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*: risk
  - *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
- **Model fitting** (statistics and machine learning)
  - *Objective*: error in model predictions over a training set
  - *Variables*: parameters of the model
  - *Constraints*: model complexity

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

Mathematical optimization: formulation

Standard form:

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

- $x \in \mathbf{R}^n$: 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
- $h_i : \mathbf{R}^n \to \mathbf{R}$: equality constraint functions
- **Feasible set**: $\mathcal{D} = \{x \in \mathbf{R}^n \mid f_i(x) \leq 0,\ i = 1, \ldots, m,\ h_i(x) = 0,\ i = 1, \ldots, p\}$
- **Feasibility problem**: Find $x \in \mathcal{D}$ (determines if the constraints are consistent)

The big picture

- **Bad news**: most optimization problems (in full generality) 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 (e.g., convex)
  - 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

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
2. **Solvability**
   - Exercise is useless if a solution cannot be computed
   - Time-to-solution constraints (e.g., algorithmic trading) limit model complexity

Optimization-problem attributes

Outline

Friends and enemies in mathematical optimization

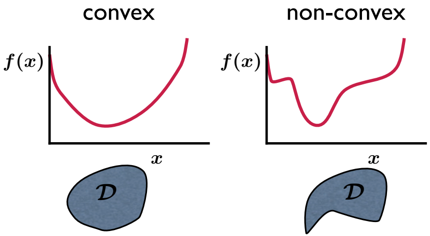
- **Key problem attributes**:
  - Convexity: convex v. non-convex
  - Optimization-variable type: continuous v. discrete
  - Constraints: unconstrained v. constrained
  - Number of optimization variables: low-dimensional v. high-dimensional
- **These attributes dictate**:
  - Ability to find the solution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

Always begin by categorizing your problem accordingly!

Convex v. non-convex

- **Convex problems**:
  - objective and inquality constraint functions are convex

$$g(\alpha x + \beta y) \leq \alpha g(x) + \beta g(y)$$



  - equality constraint functions are linear
- **Examples**:
  - Linear least squares (later today)
  - Linear programming (LP): linear objective and constraints. Common in management, finance, economics.
  - Quadratic programming (QP): quadratic objective, linear constraints.

Convex v. non-convex

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

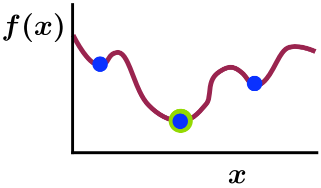


Figure 1: Local and global solutions for a non-convex objective function.

Convex v. non-convex significance

- **Convex**
  - *One unique minimum*: local minimizers are global!
  - *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`, optimization sub-package of SciPy

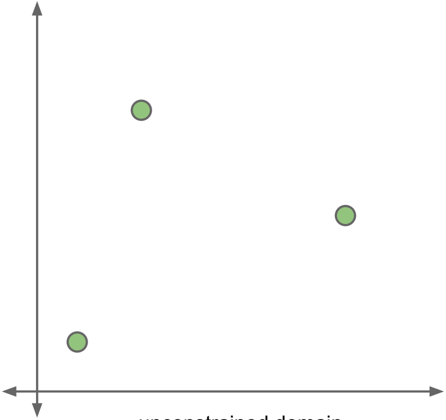
Continuous v. discrete

- **Continuous**
  - For example, $x \in \mathbf{R}^n$
  - Often easier to solve because derivative information can be exploited
- **Examples**
  - parameters in a machine-learning model
  - asset allocation in portfolio optimization
  - position in a coordinate system
  - vehicle speed in a model to minimize fuel consumption
  - wing thickness in aircraft design

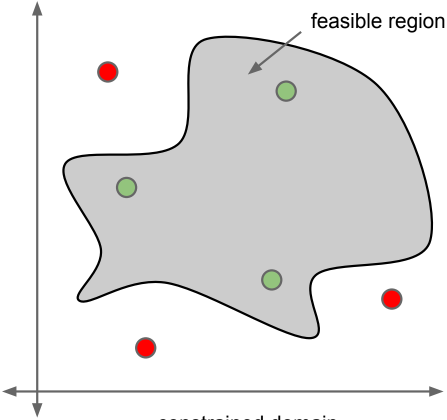
Continuous v. discrete examples

- **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, 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**

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

  - *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

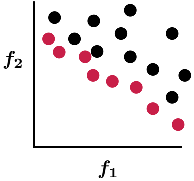
Friends and enemies in mathematical optimization (summary)

- 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 accordingly!

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:

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

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