CS675: Convex and Combinatorial Optimization
Fall 2026
Introduction

Instructor: Shaddin Dughmi

Outline

1. Course Overview

2. Administrivia

Outline

1 Course Overview

2 Administrivia

Mathematical Optimization

The task of selecting the “best” configuration of a set of variables from a “feasible” set of configurations.

$$\begin{array}{ll} \text{minimize (or maximize)} & f(x) \\ \text{subject to} & x \in \mathcal{X} \end{array}$$

- Terminology: decision variable(s), objective function, feasible set, optimal solution, optimal value
- Two main classes: continuous and combinatorial

Continuous Optimization Problems

Optimization problems where feasible set $\mathcal{X}$ is a connected subset of Euclidean space, and f is a continuous function.

- Instances typically formulated as follows.

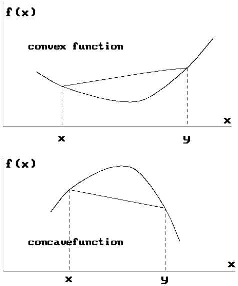
$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g_i(x) \leq b_i, \quad \text{for } i \in \mathcal{C}. \end{array}$$

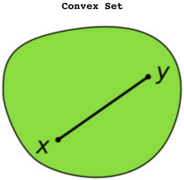
- Objective function $f : \mathbb{R}^n \to \mathbb{R}$.
- Constraint functions $g_i : \mathbb{R}^n \to \mathbb{R}$. The inequality $g_i(x) \leq b_i$ is the i'th constraint.
- In general, intractable to solve efficiently (NP hard)

Convex Optimization Problem

A continuous optimization problem where the minimization objective f is a convex function on $\mathcal{X}$, and $\mathcal{X}$ is a convex set.

- Convex function: $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y)$ for all $x, y \in \mathcal{X}$ and $\alpha \in [0,1]$
- Convex set: $\alpha x + (1-\alpha)y \in \mathcal{X}$, for all $x, y \in \mathcal{X}$ and $\alpha \in [0,1]$
- Convexity of $\mathcal{X}$ implied by convexity of g_i's
- For maximization problems, f should be concave
- Typically solvable efficiently (i.e. in polynomial time)
- Encodes optimization problems from a variety of application areas



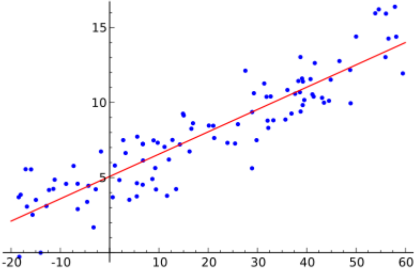


Convex Optimization Example: Least Squares Regression

Given a set of measurements $(a_1, b_1), \ldots, (a_m, b_m)$, where $a_i \in \mathbb{R}^n$ is the i'th input and $b_i \in \mathbb{R}$ is the i'th output, find the linear function $f : \mathbb{R}^n \to \mathbb{R}$ best explaining the relationship between inputs and outputs.

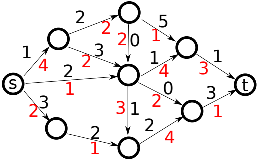
- $f(a) = \langle x, a \rangle$ for some $x \in \mathbb{R}^n$
- Least squares: minimize mean-square error.

$$\text{minimize} \quad ||Ax - b||_2^2$$



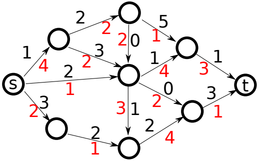
Convex Optimization Example: Minimum Cost Flow

Given a directed network $G = (V, E)$ with cost $c_e \in \mathbb{R}_+$ per unit of traffic on edge e, and capacity d_e, find the minimum cost routing of r divisible units of traffic from s to t.



Convex Optimization Example: Minimum Cost Flow

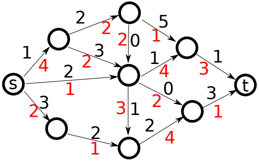
Given a directed network $G = (V, E)$ with cost $c_e \in \mathbb{R}_+$ per unit of traffic on edge e, and capacity d_e, find the minimum cost routing of r divisible units of traffic from s to t.



$$
\begin{array}{lll}
\text{minimize} & \sum_{e \in E} c_e x_e & \\
\text{subject to} & \sum_{e \leftarrow v} x_e = \sum_{e \rightarrow v} x_e, & \text{for } v \in V \setminus \{s, t\}. \\
 & \sum_{e \leftarrow s} x_e = r & \\
 & x_e \leq d_e, & \text{for } e \in E. \\
 & x_e \geq 0, & \text{for } e \in E.
\end{array}
$$

Convex Optimization Example: Minimum Cost Flow

Given a directed network $G = (V, E)$ with cost $c_e \in \mathbb{R}_+$ per unit of traffic on edge e, and capacity d_e, find the minimum cost routing of r divisible units of traffic from s to t.



$$\begin{array}{lll} \text{minimize} & \sum_{e \in E} c_e x_e & \\ \text{subject to} & \sum_{e \leftarrow v} x_e = \sum_{e \rightarrow v} x_e, & \text{for } v \in V \setminus \{s,t\}. \\ & \sum_{e \leftarrow s} x_e = r & \\ & x_e \le d_e, & \text{for } e \in E. \\ & x_e \ge 0, & \text{for } e \in E. \end{array}$$

Generalizes to traffic-dependent costs. For example $c_e(x_e) = a_e x_e^2 + b_e x_e + c_e$.

Combinatorial Optimization

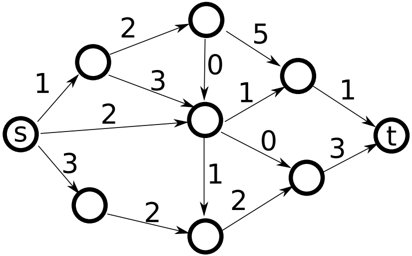
Combinatorial Optimization Problem

An optimization problem where the feasible set $\mathcal{X}$ is finite.

- e.g. $\mathcal{X}$ is the set of paths in a network, assignments of tasks to workers, etc...
- Again, NP-hard in general, but many are efficiently solvable (either exactly or approximately)

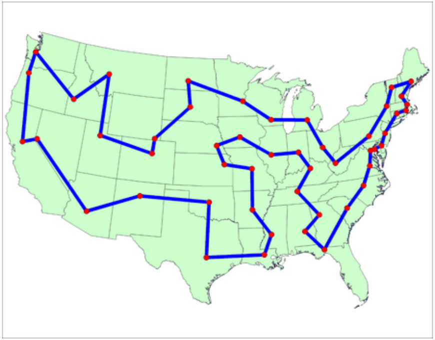
Combinatorial Optimization Example: Shortest Path

Given a directed network $G = (V, E)$ with cost $c_e \in \mathbb{R}_+$ on edge e, find the minimum cost path from s to t.



Combinatorial Optimization Example: Traveling Salesman Problem

Given a set of cities V, with $d(u, v)$ denoting the distance between cities u and v, find the minimum length tour that visits all cities.

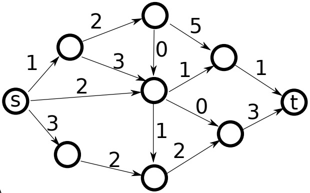


Continuous vs Combinatorial Optimization

- Some optimization problems are best formulated as one or the other
- Many problems, particularly in computer science and operations research, can be formulated as both
- This dual perspective can lead to structural insights and better algorithms

Example: Shortest Path

The shortest path problem can be encoded as a minimum cost flow problem, using distances as the edge costs, unit capacities, and desired flow rate 1



$$
\begin{array}{lll}
\text{minimize} & \sum_{e\in E} c_e x_e & \\
\text{subject to} & \sum_{e\leftarrow v} x_e = \sum_{e\to v} x_e, & \text{for } v \in V \setminus \{s,t\}\,. \\
& \sum_{e\leftarrow s} x_e = {\color{red}1} & \\
& x_e \le {\color{red}1}, & \text{for } e \in E. \\
& x_e \ge 0, & \text{for } e \in E.
\end{array}
$$

The optimum solution of the (linear) convex program above will assign flow only on a shortest path.

Course Goals

- Recognize and model convex optimization problems, and develop a general understanding of the relevant algorithms.
- Formulate combinatorial optimization problems as convex programs
- Use both the discrete and continuous perspectives to design algorithms and gain structural insights for optimization problems

Who Should Take this Class

- Anyone planning to do research in the design and analysis of algorithms for discrete problems
  - Convex and combinatorial optimization have become an indispensible part of every algorithmist's toolkit
- Students interested in theoretical machine learning and AI
  - Convex optimization appears often in machine learning and AI
  - Submodularity is an important abstraction for feature selection, active learning, planning, and other applications
  - Though ML training often involves non-convex problems, which are out of scope for this class, it often borrows ideas and insights from convex optimization
- Anyone else who solves or reasons about optimization problems: electrical engineers, control theorists, operations researchers, economists . . .
  - If there are applications in your field you would like to hear more about, let me know.

Who Should Not Take this Class

- You don't satisfy the prerequisites "in practice"
- You are merely looking for a "cookbook" of optimization algorithms, and/or want to learn how to use Gurobi, CPLEX, CVX, etc
    - This is primarily a theory class
    - We will bias our attention towards simple yet theoretically insightful algorithms and questions
    - I will not expect you to write code or learn how to use any specific software, though you are welcome to do so on your own time if it helps in your understanding.

Course Outline (Tentative)

- Weeks 1-5: Convex optimization basics and duality theory
- Weeks 6-7: Combinatorial problems posed as linear and convex programs
- Weeks 8-10: Algorithms for convex optimization
- Weeks 11-12: Matroid theory and optimization
- Weeks 13-14: Submodular Function optimization
- Week 15: Additional topics (SDP, constraint satisfaction, other?)

Outline

1 Course Overview

2 Administrivia

Basic Information

- Lecture time: Mondays and Wednesdays 2:00pm - 3:50pm
- Lecture place: KAP 158
- Instructor: Shaddin Dughmi
    - Email: shaddin@usc.edu
    - Office: GCS
    - Office Hours: TBD
- TA: TBD
    - Email: TBD
    - Office Hours: TBD
- Course Home: TBD (maybe website, maybe brightspace)
- References: Convex Optimization by Boyd and Vandenberghe, and Combinatorial Optimization by Korte and Vygen. (Available online through USC libraries.)
- Additional References: Vishnoi, Schrijver, Luenberger and Ye (available online through USC libraries)

Prerequisites

- Mathematical maturity: Be good at proofs, at the graduate level.
- Linear algebra at advanced undergrad / beginning grad level
- Exposure to algorithms or optimization at advanced undergrad / beginning grad level
  - CS570 or equivalent, or
  - CS270 and you did really well

Requirements and Grading (Subject to Change)

- This is an advanced elective class, so grade is not the point.
  - I assume you want to learn this stuff.
  - I will assign letter grades generously.
- Bi-weekly homeworks (roughly), 7% of grade
  - Proof based.
  - Challenging.
  - Discussion with classmates allowed, even encouraged, but must write up solutions independently.
  - 6 late days allowed total (use in integer amounts)
- Peer grading: 3% (details TBD)
- Two quizzes, 20% each, and final exam worth 30%
- Research project 20%. Can be theoretical, applied, or in between.

Policy on AI Use

- I used to assign most of the grade to homework, with no quizzes/exams, but temptation to use AI is too strong.
- This semester, homework is worth very little in order to reduce this temptation. I will also reduce the volume of homework.
- Homework is mainly useful as practice for quizzes / exam:
  - I will design the quizzes and exam to be very close to some homework problems.
  - This should serve as incentive for you to do the homework yourself.
- Officially, you are prohibited from using AI in doing your homework. Hopefully, the incentive stucture makes it easy to adhere to this policy.
  - If we strongly suspect AI use, this would count as cheating and be subject to academic integrity procedures.