CS675: Convex and Combinatorial Optimization
Fall 2014
Introduction to Optimization

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.

minimize (or maximize) \( f(x) \)
subject to \( x \in X \)

- 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.

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

- **Objective function** $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

- **Constraint functions** $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$. 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 $f$ 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 \(f : \mathbb{R}^n \rightarrow \mathbb{R}\) best explaining the relationship between inputs and outputs.

\[
f(a) = x^\top a \text{ for some } x \in \mathbb{R}^n
\]

- Least squares: minimize mean-square error.

\[
\text{minimize} \quad \|Ax - b\|_2^2
\]
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 $t$. 

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

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

\[
\begin{align*}
\text{minimize} & \quad \sum_{e \in E} c_e x_e \\
\text{subject to} & \quad \sum_{e \leftarrow v} x_e = \sum_{e \to v} x_e, \quad \text{for } v \in V \setminus \{s, t\}.
\end{align*}
\]

\[
\begin{align*}
\sum_{e \leftarrow s} x_e &= r \\
x_e &\leq d_e, \quad \text{for } e \in E. \\
x_e &\geq 0, \quad \text{for } e \in E.
\end{align*}
\]

Generalizes to traffic-dependent costs. For example

\[
c_e(x_e) = a_e x_e^2 + b_e x_e + c_e.
\]
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)
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
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.

\[
\begin{align*}
\text{minimize} & \quad \sum_{e \in E} c_e x_e \\
\text{subject to} & \quad \sum_{e \leftarrow v} x_e = \sum_{e \rightarrow v} x_e, \quad \text{for } v \in V \setminus \{s, t\}. \\
& \quad \sum_{e \leftarrow s} x_e = 1 \\
& \quad x_e \leq 1, \quad \text{for } e \in E. \\
& \quad x_e \geq 0, \quad \text{for } e \in E.
\end{align*}
\]

The optimum solution of the (linear) convex program above will assign flow only on a single path — namely the 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
  - Convex and combinatorial optimization have become an indispensable part of every algorithmist’s toolkit
- Students interested in theoretical machine learning and AI
  - Convex optimization underlies much of machine learning
  - Submodularity has recently emerged as an important abstraction for feature selection, active learning, planning, and other applications
- 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
- You are looking for a “cookbook” of optimization algorithms, and/or want to learn how to use CPLEX, CVX, etc
  - This is a THEORY class
  - We will bias our attention towards simple yet theoretically insightful algorithms and questions
  - We will not write code
Course Outline

- Weeks 1-5: Convex optimization basics and duality theory
- Weeks 6-7: Combinatorial problems posed as linear and convex programs
- Weeks 8-9: Algorithms for convex optimization
- Weeks 10-11: Matroid theory and optimization
- Weeks 12-13: Submodular Function optimization
- Week 14: Semidefinite programming and constraint satisfaction problems
- Week 15: Additional topics
Outline

1. Course Overview
2. Administrivia
Basic Information

- **Lecture time:** Tuesdays and Thursdays 11:00am - 12:20 pm
- **Lecture place:** THH 121
- **Instructor:** Shaddin Dughmi
  - Email: shaddin@usc.edu
  - Office: SAL 234
  - Office Hours: TBD
- **TA:** Ruixin Qiang
  - Email: rqiang@usc.edu
  - Office Hours: TBA
- **Course Homepage:** www.cs.usc.edu/people/shaddin/cs675fa14
- **References:** Convex Optimization by Boyd and Vandenberghe, and Combinatorial Optimization by Korte and Vygen. (Available online through USC libraries. Will place on reserve)
- **Additional References:** 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
  - CS303 and you did really well
Requirements and Grading

- This is an advanced elective class, so grade is not the point.
  - I assume you want to learn this stuff.

- 4 homeworks, 75% of grade.
  - Proof based.
  - Challenging.
  - Discussion allowed, even encouraged, but must write up solutions independently.

- Research project or final, 25% of grade. Project suggestions will be posted on website.

- 3 late days allowed total (use in integer amounts)
Survey

- Name
- Email
- Department
- Degree
- Relevant coursework/background
- Research project idea

Email Ruixin: rqiang@usc.edu