ORF 523
Convex and Conic Optimization

Amir Ali Ahmadi
Princeton, ORFE

Lecture 1
What is this course about?

- The mathematical and algorithmic theory of making optimal decisions subject to constraints.

Common theme of every optimization problem:

- You make decisions and choose one of many alternatives.
- You hope to maximize or minimize something (you have an objective).
- You cannot make arbitrary decisions. Life puts constraints on you.

- This pretty much encompasses everything that you do when you are awake. But let’s see a few concrete examples...
Examples of optimization problems

**In finance**

- In what proportions to invest in 500 stocks?
  - To maximize return.
  - To minimize risk.
- No more than 1/5 of your money in any one stock.
- Transactions costs < $70.
- Return rate > 2%.

**In control engineering**

- How to drive an autonomous vehicle from A to B?
  - To minimize fuel consumption.
  - To minimize travel time.
  - Distance to closest obstacle > 2 meters.
  - Speed < 40 miles/hr.
  - Path needs to be smooth (no sudden changes in direction).
Examples of optimization problems

In economics

- How to play a strategic game?
  - To maximize payoff.
  - To maximize social welfare.
  - Be at a (Nash) equilibrium.
  - Randomize between no more than five strategies.

In machine learning

- How to assign likelihoods to emails being spam?
  - To minimize probability of a false positive.
  - To penalize overfitting on training set.
  - Probability of false negative < .15.
  - Misclassification error on training set < 5%.
So the question is **not**

- Which problems are optimization problems?
  (The answer would be everything.)

The right question is

- *Which optimization problems can we solve?*

This is what this course is about.

We will formalize what we mean by “solve”.

We’ll see some of the most successful modern optimization tools available to solve a broad class of problems.

We will also see problems that we simply cannot solve.

- Nevertheless, we’ll introduce strategies for dealing with them.

There will be many applications...
Prerequisites

- Linear optimization (e.g., at the level of ORF 522)
  - Familiarity with modeling, linear programming, and basic concepts of optimization.
- Linear algebra
- Multivariate calculus
- Familiarity with MATLAB
  - Easy to pick up
Tentative list of topics

- Optimality conditions in nonlinear programming
- Convex analysis (a good dose)
- Duality and infeasibility certificates
- Computational complexity
  - Focus on complexity in numerical optimization
- Conic programming
- More in depth coverage of semidefinite programming
- A module on combinatorial optimization
- Selected topics:
  - Robust optimization
  - Polynomial optimization
  - Sum of squares programming
  - The ellipsoid method
  - Optimization in dynamical systems
Agenda for today

- Meet your teaching staff & classmates
- Get your hands dirty with algorithms
  - Game 1
  - Game 2
- Course logistics and expectations
Meet your teaching staff

- **Amir Ali Ahmadi** (Amir Ali, or Amirali, is my first name)
  - I am an Assistant Professor at ORFE. I come here from MIT, EECS, after a fellowship at IBM Research.
  - Office hours: **Tue, 5:30-7:30 PM, Sherrerd 329.**
  - [http://aaa.princeton.edu/](http://aaa.princeton.edu/)  

- **Bachir El Khadir**
  - Grad student at ORFE
  - Office hours: **Wed, 4-6 PM, Sherrerd 005**
  - bkhadir@p...
Let’s get to the games!
Meet your fellow Princetonians!

- The green check marks tell you when your visitors are available.
- You want to meet as many of them as you can, for 15 minutes each.

<table>
<thead>
<tr>
<th>February 2016</th>
<th>Mon 1</th>
<th>Tue 2</th>
<th>Wed 3</th>
<th>Thu 4</th>
<th>Fri 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.00 PM - 3.15 PM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3.15 PM - 3.30 PM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3.30 PM - 3.45 PM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3.45 PM - 4.00 PM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4.00 PM - 4.15 PM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

- 20 visitors, 20 time slots. How many can you meet?
Let me start things off for you. Here is 15 meetings:

- Can you do better? How much better?
- You all get a copy of this Doodle on the handout. You have 8 minutes!
You tell me, I draw...
A good attempt

- 18 meetings!

Can you do better?
An even better attempt

- 19 meetings!

**Can you do better?**

**How would you convince someone that it’s impossible to do better?**
19 is the best possible!

- Proof by magic:

- Do you see what’s happening?
19 is the best possible!

- There are 19 red arrows.
- Each green checkmark “.touches” at least one of them (by going either up or left).
- If you could choose 20 green checkmarks, at least two of them would have to touch the same arrow.
- And here is the magic: such a proof is always possible!
A related problem: shipping oil!

Before we get to our second game, let’s look at another problem which may look more familiar to you.

- **Rules of the problem:**
  - Cannot exceed capacity on the edges.
  - For each node, except for S and T, flow in = flow out (i.e., no storage).

- **Goal:** ship as much oil as you can from S to T.
A couple of good attempts

\[4 + 4 + 4 = 12\]
- Can you do better?

\[5 + 4 + 4 = 13\]
- Can you do better?
- How can you convince someone that it's impossible to do better?
13 is the best possible!

- Proof by magic:
  - The rabbit is the red “cut”!
  - Any flow from S to T must cross the red curve.
  - So it can have value at most 13.
    - And here is the magic: such a proof is always possible!

\[6 + 1 + 2 + 4 = 13\]

- What does any of this have to do with the Doodle problem?
The idea of reductions
They’ll come up often
A real-world instance of max-flow
- How long do you think an optimization solver would take (on my laptop) to find the best solution here?
- How many lines of code do you think you have to write for it?
- How would someone who hasn’t seen optimization approach this?
  - Trial and error?
  - Push a little flow here, a little there...
  - Do you think they are likely to find the best solution?
  - How would they certify it?
A bit of history behind this map

- From a secret report by Harris and Ross (1955) written for the Air Force.
- Railway network of the Western Soviet Union going to Eastern Europe.
- Declassified in 1999.
- Look at the min-cut on the map (called the “bottleneck”!
- There are 44 vertices, 105 edges, and the max flow is 163K.

- Harris and Ross gave a heuristic which happened to solve the problem optimally in this case.
- Later that year (1955), the famous Ford-Fulkerson algorithm came out of the RAND corporation. The algorithm always finds the best solution (for rational edge costs).

More on this history: [Sch05]
Let’s look at our second problem

...and tell me which one you thought was easier
Robust-to-noise communication

- You are given a set of letters from an alphabet.
- Want to use them for communication over a noisy channel.
- Some letters look similar and can be confused at the receiving end because of noise. (Notion of similarity can be formalized; e.g., think of Hamming distance.)

Let’s draw a graph whose nodes are our letters. There is an edge between two nodes if and only if the letters can be confused.

We want to pick the maximum number of letters that we can safely use for communication (i.e., no two should be prone to confusion).

- What are we looking for in this graph?

- The largest “stable set” (aka “independent set”)!
Let me start things off for you. Here is a stable set of size 3:

You all get a copy of this graph on the handout.

You have 8 minutes!
You tell me, I draw...
A couple of good attempts

- Can you do better?

Size 4
A couple of good attempts

Can you do better?
A couple of good attempts

- Tired of trying?
- Is this the best possible?

Size 5
5 is the best possible!

- Proof by magic?
- Unfortunately not 😞
- No magician in the world has pulled out such a rabbit to this day! (By this we mean a rabbit that would work on all graphs.)

- Of course there is always a proof:
  - Try all possible subsets of 6 nodes.
  - There are 924 of them.
  - Observe that none of them work.

- But this is no magic. It impresses nobody. We want a “short” proof. (We will formalize what this means.) Like the one in our Doodle/max-flow examples.

- Let’s appreciate this further...
What our graph can look like with 32 letters

- Maximum stable set anyone? ;)
- Is there a stable set of size 16?
- Want to try all possibilities? There are **over 600 million** of them!!
- If the graph had 100 nodes, there would be **over $10^{18}$ possibilities** to try!
But there is some good news

- Even though finding the best solution always may be too much to hope for, techniques from optimization (and in particular from the area of *convex optimization*) often allow us to find high-quality solutions with performance guarantees.

- For example, an optimization algorithm may quickly find a stable set of size 15 for you.

- You really want to know if 16 is impossible. Instead, another optimization algorithm (or sometimes the same one) tells you that 18 is impossible.

- This is very useful information! You know you got 15, and no one can do better than 18.

**We sill see a lot of convex optimization in this class!**
A related problem: capacity of a graph

Suppose instead of single letters, we wanted to send k-tuples of letters. How many k-tuples can we collect such that no two among them can be confused?

Two k-tuple can be confused if for each $1 \leq i \leq k$, their $i$-th letters can be confused (or are equal).

The answer is the stability number of the “k-th power of the graph”, in the sense of strong graph products (k-th Kronecker power of the adjacency matrix).

Example: $k=2$. 

\[ A_G \]

\[ A_G^2 = A_G \otimes A_G \]
Capacity of a graph

\[ \Theta(G) = \lim_{k \to \infty} \sqrt[k]{\alpha(G^k)} \]

(\(\alpha\): size of max stable set)

- \(G\)
- \(G^2\)
- \(G^3\)

\[ \Theta(C_5) = \sqrt{5} \]

\[ \Theta(C_7) = ? \text{ (open)} \]

Using semidefinite programming

(Automatic "A" in ORF 523)
Which of the two problems was harder for you?

Not always obvious. A lot of research in optimization and computer science goes into distinguishing the “tractable” problems from the “intractable” ones.

The two brain teasers actually just gave you a taste of the P vs. NP problem. (If you haven’t seen these concepts formally, that’s OK. You will soon.)

The first problem we can solve efficiently (in “polynomial time”).

The second problem: no one knows. If you do, you literally get $1M!

More importantly, your algorithm immediately translates to an efficient algorithm for thousands of other problems no one knows how to solve.
Modelling problems as a mathematical program
Let’s revisit our first game

- What were your decision variables?
- What were your constraints?
- What was your objective function?

Variables: \( x_e \) (one per edge)

\[
\begin{align*}
\max & \sum_{e} x_e \\
\text{s.t.} & \quad x_e \leq 1, \quad \forall e \\
\sum_{e} x_e & \leq 1, \quad \forall v \\
\sum_{e} (1 - x_e) & = 0, \quad \forall e
\end{align*}
\]
Let’s revisit our second game

- What were your decision variables?
- What were your constraints?
- What was your objective function?
Why one hard and one easy? How can you tell?

Variables: \( \chi_e \) (one per edge)

\[
G(\mathcal{V}, \mathcal{E}) \quad \max \sum_{e \in \mathcal{E}} \chi_e \\
\chi_e \geq 0 \quad \forall e \in \mathcal{E} \\
\sum_{e \in \mathcal{E}} \chi_e \leq 1 \quad \forall v \in \mathcal{V} \\
\chi_e \cdot (1 - \chi_e) = 0 \quad \forall e \in \mathcal{E}
\]

**Caution:** just because we can write something as a mathematical program, it doesn’t mean we can solve it.
Fermat’s Last Theorem

- Can you give me three positive integers $x, y, z$ such that

\[ x^2 + y^2 = z^2? \]

- Sure:

$\begin{align*}
(3, 4, 5) & \quad (5, 12, 13) & \quad (8, 15, 17) & \quad (7, 24, 25) \\
(20, 21, 29) & \quad (12, 35, 37) & \quad (9, 40, 41) & \quad (28, 45, 53)
\end{align*}$

And there are infinitely many more...

- How about $x^3 + y^3 = z^3$?

- How about $x^4 + y^4 = z^4$?

- How about $x^5 + y^5 = z^5$?
Fermat’s conjecture (1637):  
For $n \geq 3$, the equation $x^n + y^n = z^n$ has no solution over positive integers.

Proved in 1994 (357 years later!) by Andrew Wiles. (Was on the faculty in our math department until a few years ago.)
Fermat’s Last Theorem

Fermat’s conjecture (1637):
For $n \geq 3$, the equation $x^n + y^n = z^n$ has no solution over positive integers.

Consider the following optimization problem (mathematical program):

\[
\begin{align*}
\min_{x, y, z, n} & \quad (x^n + y^n - z^n)^2 \\
\text{s.t.} & \quad x \geq 1, \quad y \geq 1, \quad z \geq 1, \quad n \geq 3,
\end{align*}
\]

\[
\sin^2(\pi t) = 0.
\]

Innocent-looking optimization problem: 4 variables, 5 constraints.

If you could show the optimal value is non-zero, you would prove Fermat’s conjecture!
Course objectives

The skills I hope you acquire:

- Ability to view your own field through the lens of optimization and computation
  - To help you, we’ll draw basic applications from operations research, dynamical systems, finance, machine learning, engineering, ...

- Comfort with proofs in convex analysis.

- Improved coding abilities (in MATLAB, CVX, YALMIP)
  - There will be a computational component on every homework (usually the fun part of the homework)

- Ability to recognize hard and easy optimization problems.

- Ability to rigorously show an optimization problem is hard.

- Solid understanding of conic optimization, in particular semidefinite programming.

- Familiarity with selected topics: robust optimization, polynomial optimization, optimization in dynamical systems, etc.
Software you need to download

▪ Right away:

MATLAB (the 2017 version does not currently work with CVX. Download the 2016 version).

http://www.princeton.edu/software/licenses/software/software/matlab/

▪ In the next couple of weeks (will likely appear on HW#2):

CVX

http://cvxr.com/cvx/

▪ Towards the end of the course:

YALMIP

http://users.isy.liu.se/johanl/ylmip/
Course logistics

- Course website:  aaa.princeton.edu/orf523

(should go live in its updated version in a couple of days)

- Your grade:
  - 50% homework (5 or 6 total – biweekly, can drop your lowest score, no extensions allowed)
    - Collaboration policy: you can and are encouraged. Turn in individual psets. Write the name of your collaborators.
  - 20% Midterm exams (in class – 80 mins, a single double-sided page of cheatsheet allowed)
    - Exam 1: in March, Exam 2: in April, dates TBA. Not cumulative. Can drop the lower score.
  - 30% Final exam/assignment (think of it as a longer, cumulative homework that needs to be done with no collaboration). In rare cases, may be replaced with a project.

- Textbooks
  - What matters primarily is class notes. You are expected to take good notes. (I teach on the blackboard most of the time.) Georgina Hall (former TA) has provided lecture outlines. I will also occasionally post the notes that I use to prepare for lecture.
  - Four references will be posted on the course website if you want to read further – all should be free to download online.
Image credits and references
