Welcome!

Module: Optimization for Data Analysis

Lecture Hours: Wednesday 3pm (LB08); Thursday 2pm (LB01).

Lecturer: George Iosifidis, `www.FutureNetworksTrinity.net`

What is Optimization?

- More than applied maths, it's a language!

What is Optimization?

- More than applied maths, it's a language!
- A language that:
  - allows us to describe a problem with mathematics;
  - find a real-world solution by solving the math problem;
  - expedite the solution using sophisticated algorithms;
  - trade-off solution speed with accuracy or resources spent, etc.

What is Optimization?

- More than applied maths, it's a language!
- A language that:
  - allows us to describe a problem with mathematics;
  - find a real-world solution by solving the math problem;
  - expedite the solution using sophisticated algorithms;
  - trade-off solution speed with accuracy or resources spent, etc.
- Can we learn optimization in 3 months?

What is Optimization?

- More than applied maths, it's a language!
- A language that:
  - allows us to describe a problem with mathematics;
  - find a real-world solution by solving the math problem;
  - expedite the solution using sophisticated algorithms;
  - trade-off solution speed with accuracy or resources spent, etc.
- Can we learn optimization in 3 months?
  - No!
  - Similar to a language; cannot learn all the words.

Who is Using Optimization?

- before 1990: mostly in operations research and economics.
  - production scheduling, inventory control, pricing, equilibrium analysis, etc.
- since 1990: many applications in engineering.
  - signal processing, automatic control, communications, circuit design.
- lately even more applications.
  - networks, data analysis, online advertising, etc.
  - *How Amazon should route its packets to clients?*
  - *How much should Google charge for its sponsored-search ads?*

Used by researchers, engineers, managers, and so on.

Our Focus

- Convex optimization.
  - a general class of techniques with many applications;
  - includes other classes of problems as special case, e.g., linear optimization;
  - convex problems have properties that facilitate their solutions.

Our Focus

- Convex optimization.
  - a general class of techniques with many applications;
  - includes other classes of problems as special case, e.g., linear optimization;
  - convex problems have properties that facilitate their solutions.

- Applications in data analytics.
  - machine learning problems, e.g. regression, classification, etc.
  - will discuss applications in computer systems and networks, if we have time.

Our Focus

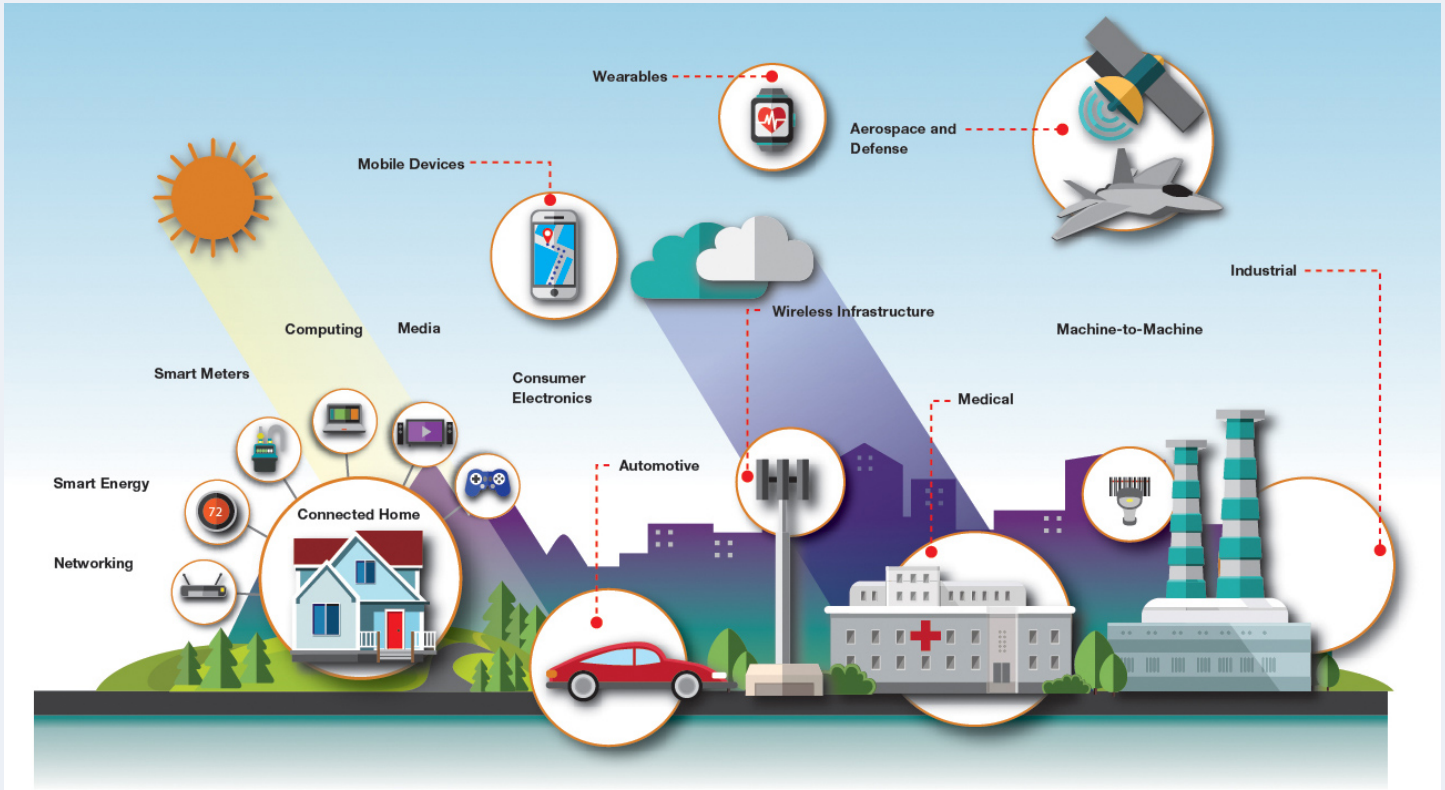
- Convex optimization.
  - a general class of techniques with many applications;
  - includes other classes of problems as special case, e.g., linear optimization;
  - convex problems have properties that facilitate their solutions.
- Applications in data analytics.
  - machine learning problems, e.g. regression, classification, etc.
  - will discuss applications in computer systems and networks, if we have time.
- Algorithms for solving numerically these problems.
  - gradient descent-like iterative algorithms.

Why Bother?

Why Bother?

Can’t we just use our servers and software libraries to solve the problems?

Why Bother?



- New era: IoT, online social networks, mobile computing.
  - More data, more demanding applications, smaller devices.

Why Bother?

- We need a language to describe our problem (**modeling**).
  - Even if a software can solve it, you need to correctly program it!

Why Bother?

- We need a language to describe our problem (**modeling**).
  - Even if a software can solve it, you need to correctly program it!
- We need to understand its structure and properties (**analysis**).
  - transform the problem so as to improve its solution.
  - always important to solve faster, with less computing power.

Why Bother?

- We need a language to describe our problem (**modeling**).
  - Even if a software can solve it, you need to correctly program it!
- We need to understand its structure and properties (**analysis**).
  - transform the problem so as to improve its solution.
  - always important to solve faster, with less computing power.
- We need tailored algorithms to solve our specific problem (**algorithm design**).

Why do we need the algorithms?

Important algorithms for machine-learning; widely used.

- Work always:
  - oftentimes we cannot obtain analytical expressions. Hence, we need a numerical iterative method to calculate the solutions.
- Convergence:
  - usually find the solution fast, with good precision, and scale reasonably well with the problem dimension (*dimension-free*).
- Full control:
  - we can select how close to the solution we wish to be; we can parallelize (in some cases), and so on...

These *iterative algorithms* are one of the main pillars of ML.

Employers will Pay you for...

1. Being able to formulate an optimization problem and use a library to solve it.

2. Write a new algorithm that solves it:
   - faster; or
   - more accurately; or
   - with a cheaper computer;

than their competitors do.

General form

- Optimization problem:

$$\begin{array}{ll} \text{minimize} & f_0(\boldsymbol{x}) \\ \text{subject to} & f_i(\boldsymbol{x}) \leq b_i, \;\; i = 1, \ldots, m \end{array}$$

- Optimization variables: $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$
- Objective function: $f_0 : \boldsymbol{R}^n \rightarrow R$
- Constraints: $f_i : \boldsymbol{R}^n \rightarrow R, \;\; i = 1, \ldots, m$
- Optimal solution: $\boldsymbol{x}^*$ gives the smallest value for f_0 among all vectors $\boldsymbol{x}$ that satisfy the problem's constraints.

Examples

- Network architecture design
  - variables: locations for servers; capacity of links
  - constraints: data transfer delays, users' demand
  - objective: deployment cost
- Advertisement allocation in social media feeds.
  - variables: users' feed, position in the feed
  - constraints: distance between consecutive ads, relevance to content
  - objective: profit maximization for the platform
- Data fitting.
  - variables: model parameters
  - constraints: prior information, parameter limits
  - objective: measure of misfit or prediction error

Solving Opt Problems

- The general case.
  - very difficult to solve.
  - methods involve some compromise, e.g., very long computation time,
  - or finding an approximate, not exact, solution!

Solving Opt Problems

- The general case.
  - very difficult to solve.
  - methods involve some compromise, e.g., very long computation time,
  - or finding an approximate, not exact, solution!
- Exceptions: certain problem classes can be solved efficiently.
  - least-squares problems
  - linear programming problems
  - convex optimization problems

Least-squares

- Problem formula:

$$\min_{x} \qquad ||Ax - b||_2^2$$

  where $A \in R^{k \times n}$, $x \in R^n$, $b \in R^k$.

Least-squares

- Problem formula:

$$\min_{x} \qquad ||Ax - b||_2^2$$

  where $A \in R^{k \times n}$, $x \in R^n$, $b \in R^k$.

- Properties:
  - there is an analytical solution: $x^* = (A^T A)^{-1} A^T b$;
  - efficient algorithms and software for finding it;
  - computation time can be large, as is proportional to $n^2 k$;
  - quite a mature technology.

Least-squares

- Problem formula:

$$\min_x \qquad ||Ax - b||_2^2$$

  where $A \in R^{k \times n}$, $x \in R^n$, $b \in R^k$.

- Properties:
  - there is an analytical solution: $x^* = (A^T A)^{-1} A^T b$;
  - efficient algorithms and software for finding it;
  - computation time can be large, as is proportional to $n^2 k$;
  - quite a mature technology.

- Using least-squares.
  - least-squares problems are easy to recognize;
  - a few standard techniques increase flexibility, e.g., including weights,adding regularization terms, etc.

Linear programming

- Problem formula:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & a_i^T x \leq b_i, \;\; i = 1, \ldots, m. \end{aligned}$$

Linear programming

- Problem formula:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & a_i^T x \leq b_i, \quad i = 1, \ldots, m. \end{aligned}$$

- Properties
  - no analytical formula for solution;
  - reliable and efficient algorithms and software;
  - computation time proportional to $n^2 m$.

Linear programming

- Problem formula:

$$\begin{array}{ll} \min\limits_{x} & c^T x \\ \text{s.t.} & a_i^T x \leq b_i, \quad i = 1, \ldots, m. \end{array}$$

- Properties
  - no analytical formula for solution;
  - reliable and efficient algorithms and software;
  - computation time proportional to $n^2 m$.

- Using linear programming
  - not as easy to recognize as least-squares problems;
  - a few standard tricks used to convert problems into linear programs.

Convex Problems

- Problem formula:

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

- Convex means that both the objective and the constraint functions are convex:

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

  where $\alpha + \beta = 1$, α, $\beta \geq 0$.

- Convex optimization includes least-squares problems and linear programs as special cases.

Convex Problems

- solving convex optimization problems
  - no analytical solution;
  - reliable and efficient algorithms;
  - computation time proportional to $max\{n^3, n^2 m, F\}$, where F is cost of evaluating f_i's and their first and second derivatives.
  - almost a technology, i.e., increasing number of s/w packages.

Convex Problems

- solving convex optimization problems
  - no analytical solution;
  - reliable and efficient algorithms;
  - computation time proportional to $max\{n^3, n^2 m, F\}$, where F is cost of evaluating f_i's and their first and second derivatives.
  - almost a technology, i.e., increasing number of s/w packages.
- Using convex optimization
  - often difficult to recognize
  - many tricks for transforming problems into convex form
  - surprisingly many problems can be solved via convex optimization

Structure of this Course

- Program (tentative):
  1. Week 1 (23-24/1): Intro and Convex sets;
  2. Week 2 (30-31/1: Convex Functions and Convex Optimization Problems;
  3. Week 3 (6-7/2): Convex Optimization Problems;
  4. Week 4 (13-14/2): Convex Problems;
  5. Week 5 (20-21/2): Duality;
  6. Week 6 (27-28/2): Review and Exercises;
  7. Week 7 (6-7/3): Reading Week;
  8. Week 8a (13/3): Mid-term Exams;
  9. Week 8b (14/3): Gradient Algorithms ;
  10. Week 9a (20/3): Guest Lecture: Dr. Jakub Marecek;
  11. Week 9b (21/3): Advanced Gradient Algorithms;
  12. Week 10 (27-28/3): Advanced Gradient Algorithms;
  13. Week 11 (3-4/4): Classification Problems;
  14. Week 12 (10-11/4): Review and Exercises.

More about this course

- **Main Textbook**: Convex Optimization by S. Boyd and L. Vandenberghe.
  - Available online; very important reference!
  - Web link: `http://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf`
- **Blackboard**: slides, notes, scripts, additional references.

More about this course

- **Main Textbook**: Convex Optimization by S. Boyd and L. Vandenberghe.
  - Available online; very important reference!
  - Web link: `http://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf`
- **Blackboard**: slides, notes, scripts, additional references.
- **Some tips**:
  - It's a math class, do study every week.
  - Evaluation with mid-terms (30%) and final exams (70%).
  - Need to practice; your responsibility.
  - Might have some extra lectures for solving exercises.
  - Office hours: Thursday 3.30-5pm (Lloyd, ground floor).
  - Questions sent to email:george.iosifidis@scss.tcd.ie.

More about this course

- **Other Textbooks**: An Introduction to Optimization by E. Chong and S. H. Zak.
- Good reference; dedicates first 5 chapters in mathematical review.
- Background you need to have:
  - Know how to answer a question with math (formalism; proofs).
  - Algebra; matrix operations; norms; derivatives; gradients; Taylor series; etc.
  - Suggestion: start with Appendix A from Boyd's book.

Matlab and CVX

Matlab can be used to solve the problems we are discussing here.

CVX is a library that facilitates expression and solution of convex problems.

We will be using these throughout the module.

CVX

Complete (and very nice!) tutorial:

- The CVX Users' Guide; Release 2.1, March 2017.
- Michael C. Grant, Stephen P. Boyd; CVX Research, Inc.
- Library and tutorial are available online.

CVX is a modeling system for constructing and solving convex programs.

Supports the solution of:

- linear programs; quadratic programs, second-order cone programs, semidefinite programs; problems with nondifferentiable functions; mixed integer programs; etc.

It is based on and uses Matlab, offering:

- a stylized interface for expressing optimization problems;
- automated transformation of these problems to solvable formats.

CVX

As an example, consider the following optimization problem:

$$\min_x ||Ax - b||_2 \qquad \text{s.t.} \qquad lb \leq x \leq ub$$

Solution with Matlab:

- transform it to a quadratic program, and run a standard Matlab/Octave solver:

$$x^* = quadprog(2 \cdot A' \cdot A, -2 \cdot A' \cdot b, [\,], [\,], [\,], [\,], lb, ub);$$

Solution with CVX: use *disciplined programming*.

```
cvx_begin
    variable x(n)
    minimize ( norm(A * x - b) )
    subject to
        lb <= x <= ub
cvx_end
```

CVX

More examples can be found in the tutorial – check also the online forum!

CVX presumes you know how to formulate your optimization problem.

CVX cannot run unless you follow its scripting rules.

It is not suitable for very large scale problems, but:

- can be combined with other Matlab commands;
- can be used to solve sub-problems of larger problems;
- can be used to solve smaller instances of the actual problem (trials and tests).