1. Introduction

- mathematical optimization
- least-squares and linear programming
- convex optimization
- example
- course goals and topics
- nonlinear optimization
- brief history of convex optimization
Mathematical optimization

(mathematical) optimization problem

\[
\begin{align*}
\text{minimize} & \quad f_0(x) \\
\text{subject to} & \quad f_i(x) \leq b_i, \quad i = 1, \ldots, m
\end{align*}
\]

• \(x = (x_1, \ldots, x_n)\): optimization variables

• \(f_0 : \mathbb{R}^n \to \mathbb{R}\): objective function

• \(f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m\): constraint functions

solution or optimal point \(x^*\) has smallest value of \(f_0\) among all vectors that satisfy the constraints
Examples

portfolio optimization

• variables: amounts invested in different assets
• constraints: budget, max./min. investment per asset, minimum return
• objective: overall risk or return variance

device sizing in electronic circuits

• variables: device widths and lengths
• constraints: manufacturing limits, timing requirements, maximum area
• objective: power consumption

data fitting

• variables: model parameters
• constraints: prior information, parameter limits
• objective: measure of misfit or prediction error, plus regularization term
Solving optimization problems

**general optimization problem**

- very difficult to solve
- methods involve some compromise, *e.g.*, very long computation time, or not always finding the solution (which may not matter in practice)

**exceptions:** certain problem classes can be solved efficiently and reliably

- least-squares problems
- linear programming problems
- convex optimization problems
Least-squares

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

solving least-squares problems

- analytical solution: \(x^* = (A^T A)^{-1} A^T b\)
- reliable and efficient algorithms and software
- computation time proportional to \(n^2 k\) (\(A \in \mathbb{R}^{k \times n}\)); less if structured
- 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)
Linear programming

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad a_i^T x \leq b_i, \quad i = 1, \ldots, m
\end{align*}
\]

solving linear programs

• no analytical formula for solution
• reliable and efficient algorithms and software
• computation time proportional to \( n^2 m \) if \( m \geq n \); less with structure
• a mature technology

using linear programming

• not as easy to recognize as least-squares problems
• a few standard tricks used to convert problems into linear programs
  \((e.g., \text{problems involving } \ell_1- \text{ or } \ell_\infty-\text{norms, piecewise-linear functions})\)
Convex optimization problem

minimize $f_0(x)$
subject to $f_i(x) \leq b_i, \ i = 1, \ldots, m$

• objective and constraint functions are convex:

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

if $\alpha + \beta = 1, \ \alpha \geq 0, \ \beta \geq 0$

• includes least-squares problems and linear programs as special cases
solving convex optimization problems

• no analytical solution
• reliable and efficient algorithms
• computation time (roughly) proportional to $\max\{n^3, n^2m, F\}$, where $F$ is cost of evaluating $f_i$’s and their first and second derivatives
• almost a technology

using convex optimization

• often difficult to recognize
• many tricks for transforming problems into convex form
• surprisingly many problems can be solved via convex optimization
Example

$m$ lamps illuminating $n$ (small, flat) patches

\[ I_k = \sum_{j=1}^{m} a_{kj} p_j, \quad a_{kj} = r_{kj}^{-2} \max\{\cos \theta_{kj}, 0\} \]

intensity $I_k$ at patch $k$ depends linearly on lamp powers $p_j$:

**Problem**: achieve desired illumination $I_{\text{des}}$ with bounded lamp powers

\[
\begin{align*}
\text{minimize} & \quad \max_{k=1,\ldots,n} |\log I_k - \log I_{\text{des}}| \\
\text{subject to} & \quad 0 \leq p_j \leq p_{\text{max}}, \quad j = 1, \ldots, m
\end{align*}
\]
how to solve?

1. use uniform power: $p_j = p$, vary $p$

2. use least-squares:

   $$\text{minimize} \quad \sum_{k=1}^{n} (I_k - I_{\text{des}})^2$$

   round $p_j$ if $p_j > p_{\text{max}}$ or $p_j < 0$

3. use weighted least-squares:

   $$\text{minimize} \quad \sum_{k=1}^{n} (I_k - I_{\text{des}})^2 + \sum_{j=1}^{m} w_j (p_j - p_{\text{max}}/2)^2$$

   iteratively adjust weights $w_j$ until $0 \leq p_j \leq p_{\text{max}}$

4. use linear programming:

   $$\text{minimize} \quad \max_{k=1,\ldots,n} |I_k - I_{\text{des}}|$$

   $$\text{subject to} \quad 0 \leq p_j \leq p_{\text{max}}, \quad j = 1, \ldots, m$$

   which can be solved via linear programming

of course these are approximate (suboptimal) ‘solutions’
5. use convex optimization: problem is equivalent to

\[
\begin{align*}
\text{minimize} & \quad f_0(p) = \max_{k=1,\ldots,n} h(I_k/I_{\text{des}}) \\
\text{subject to} & \quad 0 \leq p_j \leq p_{\text{max}}, \quad j = 1, \ldots, m
\end{align*}
\]

with \( h(u) = \max\{u, 1/u\} \)

\( f_0 \) is convex because maximum of convex functions is convex.

\textbf{exact} solution obtained with effort \( \approx \) modest factor \( \times \) least-squares effort.
**additional constraints:** does adding 1 or 2 below complicate the problem?

1. no more than half of total power is in any 10 lamps
2. no more than half of the lamps are on \((p_j > 0)\)

- answer: with (1), still easy to solve; with (2), extremely difficult
- moral: (untrained) intuition doesn’t always work; without the proper background very easy problems can appear quite similar to very difficult problems
Course goals and topics

**goals**

1. recognize/formulate problems (such as the illumination problem) as convex optimization problems
2. develop code for problems of moderate size (1000 lamps, 5000 patches)
3. characterize optimal solution (optimal power distribution), give limits of performance, etc.

**topics**

1. convex sets, functions, optimization problems
2. examples and applications
3. algorithms
Nonlinear optimization

traditional techniques for general nonconvex problems involve compromises

**local optimization methods** (nonlinear programming)
- find a point that minimizes $f_0$ among feasible points near it
- fast, can handle large problems
- require initial guess
- provide no information about distance to (global) optimum

**global optimization methods**
- find the (global) solution
- worst-case complexity grows exponentially with problem size

these algorithms are often based on solving convex subproblems
Brief history of convex optimization

theory (convex analysis): 1900–1970

algorithms

• 1947: simplex algorithm for linear programming (Dantzig)
• 1970s: ellipsoid method and other subgradient methods
• since 2000s: many methods for large-scale convex optimization

applications

• before 1990: mostly in operations research, a few in engineering
• since 1990: many applications in engineering (control, signal processing, communications, circuit design, . . . )
• since 2000s: machine learning and statistics