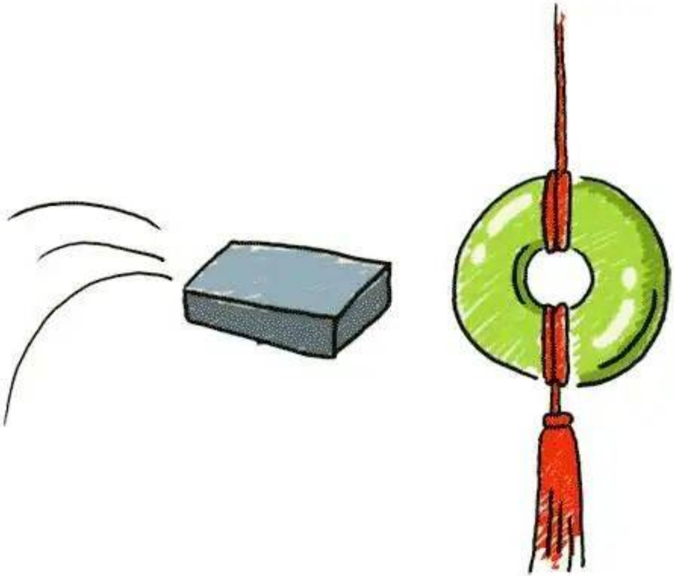
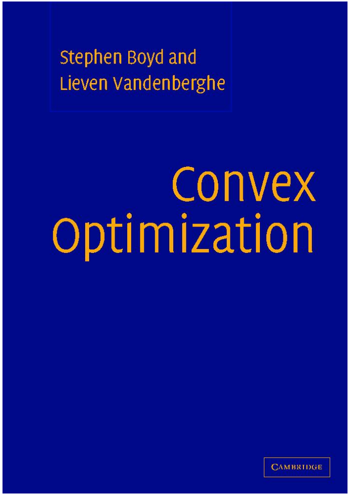


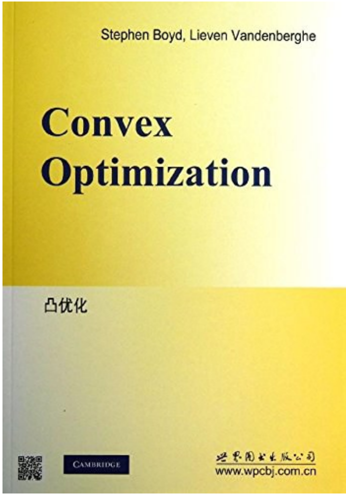
to offer one's own relatively worthless words, opinions, or services in order to attract others' more valuable contributions

reasons why you should NOT take the course (if at all possible)

- your lecturer is an absolute beginner, just like many of you
- this course is very demanding, thus no easy credits
- this course is neither about mathematics, nor about programming
- excellent skills in calculus and linear algebra are crucial
- linear programming is not strictly required as a prerequisite, but you will miss a lot of motivation and experience if you have not learned it

despite of all the disappointments, you are guaranteed to **make great progress** by working hard through the course





the original blue-covered version (uploaded on BB) shall prevail in case of discrepancy

how to achieve the most from the course

▶ read textbook carefully	the most important ever
▶ solve as many exercises as possible	extremely important
▶ attend lectures and participate	very important
▶ complete group project and exam	important

regular performance: 10%

- weekly suggested exercises (not for submission, solutions will be provided)
- biweekly reading report (due on Sundays of even-numbered weeks except week 16)
- lecture attendance (penalty for missing lectures)

group project: 20%

- start to search for possible topics as early as possible
- tentative presentation date: June 1 (week 16)
- project report due on June 5 (week 16)

unit tests: 30%

- a total of 3 tests lasting for approximately 45 minutes each
- tentative test dates: March 16 (week 5), April 13 (week 9), May 11 (week 13)

final exam: 40%

tentative schedule (subject to change)

week numbers	book contents	extra activities
1	1.1 – 1.6, 2.1 – 2.2	
2	2.3 – 2.6, 3.1	
3	3.2 – 3.6	
4	4.1 – 4.3	
5	4.4 – 4.5	unit test
6	4.6 – 4.7, 5.1	
7	5.2 – 5.9	
8	6.1 – 6.5	
9	7.1 – 7.5	unit test
10	8.1 – 8.8, C.1	
11	C.2 – C.5, 9.1 – 9.3	
12	9.4 – 9.7	
13	10.1 – 10.3	unit test
14	10.4, 11.1 – 11.3	
15	11.4 – 11.8	
16	N/A	project presentations

Chapter 1 Introduction

Last update on 2022-02-23 19:04

Table of contents

Mathematical optimization

Least-squares and linear programming

Convex optimization

Nonlinear optimization

Course outline

Mathematical optimization

(mathematical) optimization problem

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

- optimization variables $\quad x = (x_1, \cdots, x_n)$
- objective function $\quad f_0 : \mathbb{R}^n \to \mathbb{R}$
- constraint functions $\quad f_i : \mathbb{R}^n \to \mathbb{R}, \quad i = 1, \cdots, m$

optimal solution x^*

the vector x that gives the smallest objective value among all vectors satisfying 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

Solving optimization problems

general optimization problems

- very difficult to solve
- methods involve some compromises (e.g. very long computation time, or not always fining the solution)

exceptions: certain problem classes can be solved efficiently and reliably

- least-square problems
- linear programming problems
- convex optimization problems

Least-squares

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

solving least-squares

- analytic solution: $x^* = (A^T A)^{-1} A^T b$
- reliable and efficient algorithms and software
- computation time proportional to $n^2 k$ (when $A \in \mathbb{R}^{k \times n}$); 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

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & a_i^T x \leq b_i, \qquad i = 1, \cdots, m \end{array}$$

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 if structured
- a mature technology

using linear programming

- no as easy to recognize as least-squares problems
- a few standard tricks used to convert problems into linear programs (e.g. problems involving ℓ_1- or ℓ_∞-norms, piecewise linear functions)

Convex optimization problems

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

- objective and constraint functions are convex

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

  if $\alpha, \beta \geq 0$ and $\alpha + \beta = 1$
- includes least-squares problems and linear programs as special cases

solving convex optimization problems

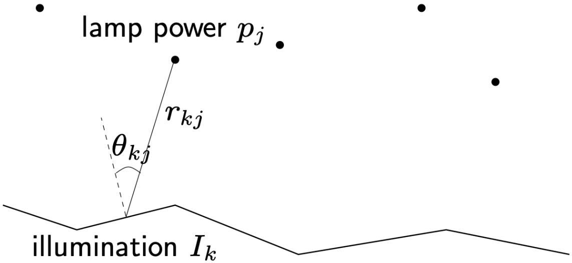
- no analytical solution
- reliable and efficient algorithms
- computation time (roughly) proportional to $\max\{n^3, n^2m, F\}$ where F is the cost of evaluating f_i's and their first and second derivatives
- almost a technology

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 and flat) patches



intensity I_k at patch k depends linearly on lamp powers p_j

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

problem: achieve desired illumination I_{des} (minimizing the maximum percentage deviation/error) with bounded lamp powers

$$
\begin{array}{ll}
\text{minimize} & \max_{1 \le k \le n} \left| \log I_k - \log I_{\text{des}} \right| \\
\text{subject to} & 0 \le p_j \le p_{\max}, \qquad j = 1, \cdots, m
\end{array}
$$

possible solutions

1. use uniform power $p_j = p$, vary p and plot the function
2. use least-squares

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

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

3. use weighted least-squares

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

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

4. use linear programming

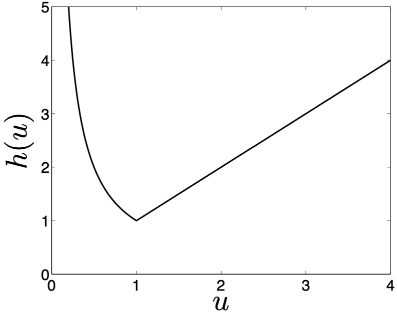
$$\begin{array}{ll} \text{minimize} & \max_{1 \leq k \leq n} \left| I_k - I_{\text{des}} \right| \\ \text{subject to} & 0 \leq p_j \leq p_{\max}, \qquad j = 1, \cdots, m \end{array}$$

which can be solved via linear programming

5. use convex optimization: equivalent formulation of the problem

$$\begin{array}{lll} \text{minimize} & \max\limits_{1\le k\le n} h(I_k/I_{\text{des}}) & \\ \text{subject to} & 0 \le p_j \le p_{\max}, & j = 1, \cdots, m \end{array}$$

with $h(u) = \max\{u, 1/u\}$, implying the objective function is convex



exact solution obtained with effort $\approx$ modest factor $\times$ least-squares effort

additional constraints: does adding either of 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 the first still easy to solve, with the second extremely difficult
- moral: (untrained) intuition does not always work; without the proper background very easy problems can appear quite similar to very difficult problems

Mathematical optimization

Least-squares and linear programming

Convex optimization

Nonlinear optimization

Course outline

Nonlinear optimization

traditional techniques for general nonconvex problems involve compromises

local optimization methods

- find a point that minimizes the objective function 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

the above algorithms are often based on solving convex subproblems

roles of convex optimization in nonconvex problems

- initialization for local optimization
- convex heuristics for nonconvex optimization
- bounds for global optimization

Course outline

theory

- basic convex analysis
- recognize and formulate problems as convex optimization problems
- Lagrangian duality, characterize optimal solutions

applications

- probability and statistics, computational geometry, data fitting

algorithms

- problem types: unconstrained, equality constrained, inequality constrained
- algorithms: Newton's algorithm, interior-point methods

Brief history of convex optimization

theory (convex analysis): ca 1900-1970

algorithms

- 1947: simplex algorithm for linear programming (Dantzig)
- 1960s: early interior-point methods (Fiacco & McCormick, Dikin, ...)
- 1970s: ellipsoid method and other subgradient methods
- 1984: polynomial-time interior-point methods for linear programming (Karmarkar)
- 1994: polynomial-time interior-point methods for nonlinear convex optimization (Nesterov & Nemirovski)

applications

- before 1990: mostly in operations research; few in engineering
- since 1990: many new applications in engineering (control, signal processing, communications, circuit design, ...); new problem classes (semidefinite and second-order cone programming, robust optimization, ...)