1. Introduction

- mathematical optimization
- least squares and linear programming
- convex optimization
- example

Mathematical optimization

(Mathematical) optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \end{array}$$

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

solution $x^\star$ 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

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

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: $x^\star = (A^T A)^{-1} A^T b$ (if A has full column rank)
- reliable and efficient algorithms and software
- computation time proportional to pn^2 ($A \in \mathbf{R}^{p \times n}$); less if structured
- a mature technology

Using least squares

- least squares problems are easy to recognize
- a few standard techniques increase flexibility (*e.g.*, weights, regularization)

Linear programming

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & a_i^T x + b_i \leq 0, \quad i = 1, \ldots, m \end{array}$$

Solving linear programs

- no analytical formula for solution
- reliable and efficient algorithms and software
- computation time (roughly) proportional to mn^2 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.*, problems involving ℓ_1- or ℓ_∞-norms, piecewise-linear functions)

Convex optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \end{array}$$

- objective and constraint functions are convex:

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

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

- includes least squares problems and linear programs as special cases

Convex optimization

Solving convex optimization problems

- no analytical solution
- reliable and efficient algorithms
- computation time (roughly) proportional to

$$\max\{n^3, n^2 m, F\},$$

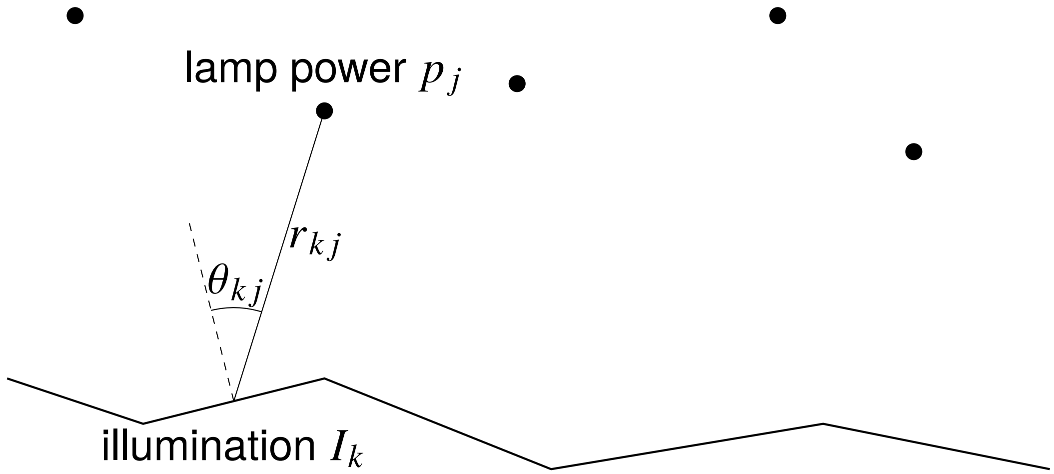
  where F is cost of evaluating f_i's and their first and second derivatives
- almost a technology

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

- n lamps illuminating m (small, flat) patches



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

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

Problem: achieve desired illumination I_{des} with bounded lamp powers

$$\begin{array}{ll} \text{minimize} & \max_{k=1,\ldots,m} |\log I_k - \log I_{\mathrm{des}}| \\ \text{subject to} & 0 \leq p_j \leq p_{\max}, \quad j = 1, \ldots, n \end{array}$$

How to solve?

1. use uniform power: $p_j = p$, vary p
2. use least squares: solve

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

   and round p_j if $p_j > p_{\max}$ or $p_j < 0$
3. use weighted least squares:

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

   iteratively adjust weights w_j until $0 \le p_j \le p_{\max}$
4. use linear programming:

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

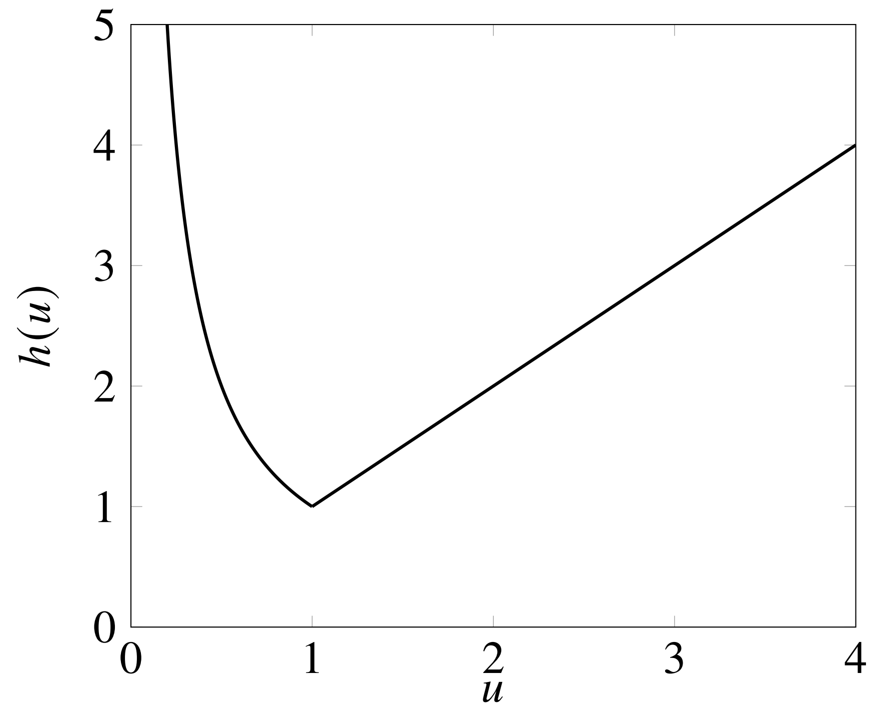
   which can be solved via linear programming

of course these are approximate (suboptimal) “solutions”

5. use convex optimization: problem is equivalent to

$$\begin{array}{ll}\text{minimize} & f_0(p) = \max_{k=1,\ldots,m} h(I_k/I_{\text{des}}) \\ \text{subject to} & 0 \le p_j \le p_{\max}, \quad j = 1, \ldots, n\end{array}$$

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



f_0 is convex because maximum of convex functions is convex

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, duality
2. examples and applications
3. algorithms

Nonlinear optimization

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, other subgradient methods
- 1980s and 1990s: polynomial-time interior-point methods for convex optimization (Karmarkar 1984, Nesterov & Nemirovski 1994)
- 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