Lecture 1
Optimization models

- numerical optimization paradigm
- complexity & structure
- approximations
- uncertainty, dynamics
- course overview & prerequisites

Numerical optimization paradigm

a standard form:

$$\text{minimize } f(x) \text{ subject to } g_i(x) \le 0, \quad i = 1, \ldots, l$$

where x is the variable, f represents the *objective*, and g_i's the *constraints*

form with feasible set:

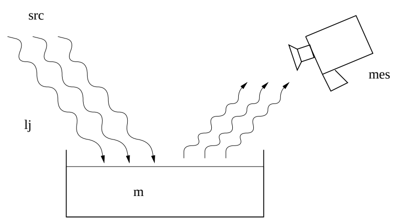
$$\text{minimize } f(x) \text{ subject to } x \in \mathcal{C}$$

where $\mathcal{C}$ is called *feasible set*

- many solution techniques (LP, NLP, neural nets, simulated annealing, evolutionary algorithms, etc)
- has many applications
- improves things a lot (see later)
- suited for software for automated A & D

important issue: tractability (complexity, see later)

Example: spectroscopy with least-squares



- seek concentration x_j of product j in a mix of m products
- source sends light at frequencies λ_j, $i = 1, \ldots, N$
- sensor gets intensity measures $b = (I_j)$ for each frequency

linear model: assume absorbing coefficients is measured in lab, *i.e.*:

a_{ij} is coefficient of absorption of product j at frequency i

define matrix $A = (a_{ij})$; with linear assumption, we have

$$Ax = b$$

Least squares solution

in practice: **no solutions** to $Ax = b$ (errors!)

we say the system of linear equations is inconsistent

we assume **all** errors can be modeled as errors in b, and describe actual data via

$$Ax = b + \Delta b$$

we'll find the smallest norm of errors that make the system consistent

estimation of concentration vector x given by

$$\min \|Ax - b\|_2$$

(here, $\|\cdot\|_2$ is Euclidean norm)

this is an *unconstrained* optimization problem (an easy one, see later)

Related problems

constrained least squares: in previous example, variable x should be positive at optimum, but this is *not* enforced by our solution procedure!

so we want to solve "constrained least-squares" problem

$$\min \|Ax - b\|_2 \ \ x_i \geq 0, \quad i = 1, \ldots, n$$

is it easy? (see later)

problems with uncertainty: in above problem, "data" (*i.e.*, A, b) is not exactly measured—sometimes, large errors in data ... least-squares implicitely assumes no errors in A

what can we do? (see later)

Feasibility

In many cases, we just want to seek a *feasible* solution

problem: find $x \in \mathcal{C}$, or prove $\mathcal{C}$ is empty

(think of $\mathcal{C}$ as reflecting a set of design specs; empty $\mathcal{C}$ means specs are too stringent)

optimization form of feasibility problem: assume $\mathcal{C}$ is described by

$$\mathcal{C} = \{x \mid g_i(x) \leq 0, \quad i = 1, \ldots, l\}$$

then solve

$$\min t \quad g_i(x) \leq t, \quad i = 1, \ldots, l$$

$\mathcal{C}$ is empty iff **global** optimum of above problem is ≤ 0

Linear objective is universal

optimization problem can always be transformed to one with *linear* objective:

original problem: minimize $f(x)$ subject to $x \in C$

with new variable $t \in \mathbf{R}$, rewrite as:

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & f(x) \leq t, \; x \in C \end{array}$$

i.e., new problem has linear objective in variable $z = (x, t)$

(may or may not be a good idea in practice)

Unconstrained optimization is universal

We can always assume the problem is **unconstrained**

Define the "barrier function"

$$\tilde{f}(x) = \begin{cases} f(x) & \text{if } x \in \mathcal{C} \\ +\infty & \text{else} \end{cases}$$

Then constrained optimization problem

$$\text{minimize } f(x) \text{ subject to } x \in \mathcal{C}$$

is equivalent to unconstrained one:

$$\text{minimize } \tilde{f}(x)$$

intuitively, above problem is harder than original one ...

but, some practical algorithms make that idea implementable!

Complexity of optimization problems

Complexity of an optimization problem **roughly** refers to the amount of computer work needed to "solve" it to a given "accuracy", depending on the problems' "size"

measures the "hardness" of a problem class

solving can mean many things: computing the optimum at given relative accuracy, or the solution at given absolute accuracy, etc

quizz:: complexity of finding the optimum of *least-squares* problem

$$\min_x \|Ax - b\|$$

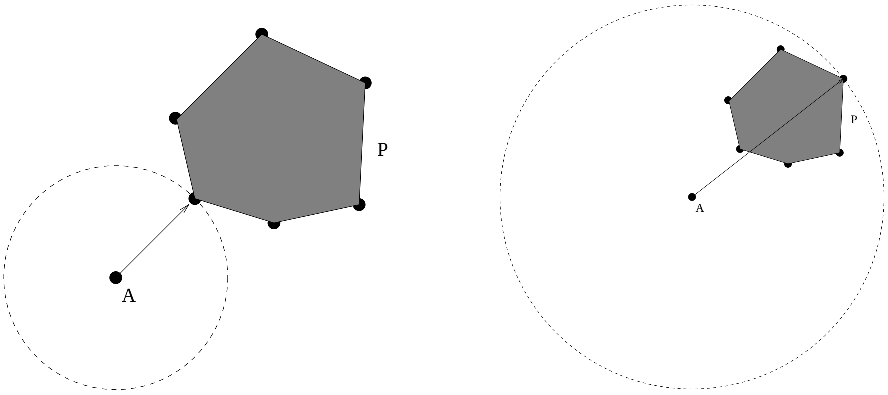
with given absolute accuracy, expressed in terms of row/column size of A?

see later: complexity of solving the constrained least-squares problem?

(answer: not much more than unconstrained problem)

Complexity and convexity

drawing line between hard/easy: NP-hardness



minimizing cost *analyzing worst-case cost*
EASY (convex) HARD (non convex)

fact 1: convexity plays important role in above distinction

(we'll refine this statement)

fact 2: convex optimization can be used to solve *approximately* second (hard) problem

For hard problems, there is trade-off to be made between

– the accuracy of the bound computed

– the effort required to compute this bound

Pitfalls of general paradigm

General framework

$$\text{minimize } f(x) \text{ subject to } g_i(x) \leq 0, \quad i = 1, \ldots, L$$

with arbitrary functions f, g_i's, is not too useful:

- solving problem is very difficult in general
- does not take into account structure
- hard to describe complexity of such problems
- seemingly useful distinctions are void: constrained vs. unconstrained, differentiable or not, etc

We have to specialize it to make tractable

One of our objectives in this course is to identify a problem class that trades off nicely between

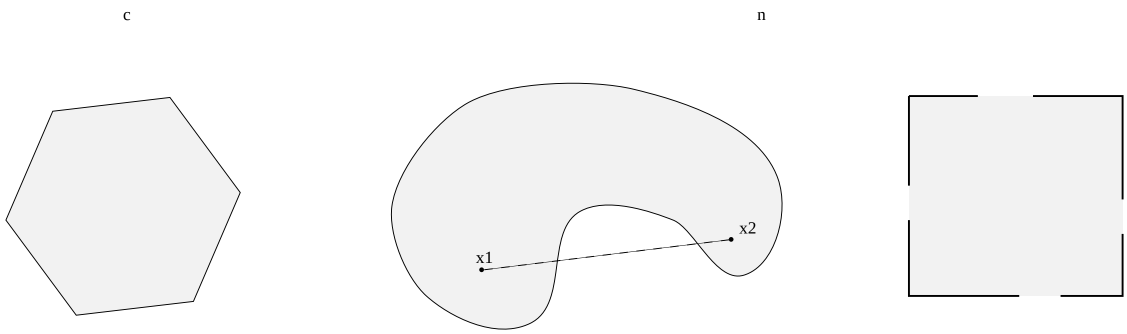
- *applicability:* a great number of interesting problems must be in that class

  (at least, approximately)
- *tractability:* the class must be tractable, *i.e.*, not too hard to solve

Convex optimization problems

$C \subseteq \mathbf{R}^n$ is convex if

$$x, y \in C, \quad \lambda \in [0,1] \Rightarrow \lambda x + (1-\lambda) y \in C$$



$f : \mathbf{R}^n \to \mathbf{R}$ is convex if

$$x, y \in \mathbf{R}^n, \quad \lambda \in [0,1]$$
$$\Downarrow$$
$$f(\lambda x + (1-\lambda) y) \le \lambda f(x) + (1-\lambda) f(y)$$

The problem

minimize $f(x)$ subject to $g_i(x) \le 0, \quad i = 1, \ldots, l$

is a *convex optimization problem* if f, g_i's are convex

Examples

example 1: linearly constrained least-squares

$$\min \|Ax - b\|_2 \quad x_i \geq 0, \quad i = 1, \ldots, n$$

where A is a $m \times n$ matrix, b a m-vector

example 2: linear programming (LP)

$$\min a_0^T x \quad a_i^T x \leq b_i, \quad i = 1, \ldots, l$$

where a_i's are n-vectors, b_i's are scalars

Features of convex optimization

Convex optimization problems

- can be solved numerically with great efficiency (we'll refine this statement later)
- have extensive, useful theory
- occur often in engineering problems
- often go unrecognized

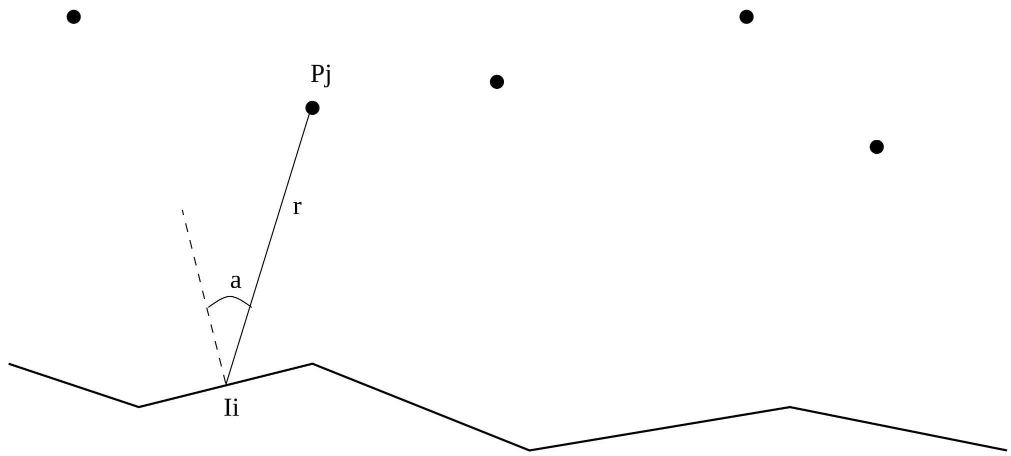
convex optimization is **tractable** in theory and practice:

there exist algorithms s.t.

- computation time small, grows gracefully with problem size
- global solutions attained
- non heuristic stopping criteria; provable lower bounds
- handle nondifferentiable as well as smooth problems

Example: illumination problem

m lamps illuminating n (small, flat) patches



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

lamp power limits: $0 \le p_j \le p_{\mathsf{max}}$

problem:

$$\underset{p_j}{\text{minimize}} \quad \max_{i=1,\ldots,n} \; |\log I_i - \log I_{\mathsf{des}}|$$

How to solve?

1. uniform power: $p_i = p$, vary p
   could try heuristic adjustment of powers
2. least squares: minimize $\sum_j (I_j - I_{\mathsf{des}})^2$
   (closed form, widely available, reliable software, fast)
   what if $p_i \geq p_{\mathsf{max}}$ or $p_i \leq 0$?
   could 'saturate' or add weights:
   $$\text{minimize} \sum_j (I_j - I_{\mathsf{des}})^2 + \sum_i w_i (p_i - p_{\mathsf{max}}/2)^2$$
3. linear programming

...of course these are approximate 'solutions'

in fact this problem can be formulated as a convex optimization problem, hence is readily solved

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

By the end of semester, you will be able to

1. develop in few hours code for problems with 10s of lamps, 100s of patches
2. develop in one week code that quickly solves problems with 100s of lamps, 1000s of patches
3. characterize optimal power distribution, give limits of performance, etc.

Two additional constraints

1. no more than half total power is in any 10 lamps
2. no more than half of the lamps are on

Does adding (1) or (2) complicate the problem?

With (1), still easy to solve

With (2), **extremely difficult** to solve

moral:
without the proper background (*i.e.*, this course) very easy problems can appear quite similar to very difficult problems.
(Untrained) intuition doesn't always work.

What fraction of 'real' problems are convex?

- by no means all
- many more than are recognized
- convex optimization plays important role in nonconvex optimization (more later)

Analog: linear programs

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

- no "closed form" solution
- very large LPs solved very quickly in practice
- extensive, useful theory

How many problems are LPs?

1940s: "the real world is nonlinear, hence LP silly"
a few examples known (in planning)
many examples found **after** LP became widely known. . .

(we guess) same story for convex optimization

convex optimization

- handles* some problems very well
- can say a lot about it

(simulated annealing, genetic algorithms, neural networks, ...)

- handle† many problems
- can say very little about it

* means a lot — global solutions, always works, worst case computation time, etc.

† means much less — local solutions (sometimes), no complexity theory, etc.

Approximations for hard problems

For hard problems we have to use approximations

- *global* approximation: upper or lower bound on objective
- *local* approximation: we don't know

"classical" approach: linearize! But ... leads to a **local** approximation

"modern" approach: convexify! leads to a **global** approximation

Aims of this course

Learn how to recognize problems that are "easy", which means *here* convex optimization problems with a special structure

Learn how to approximate, with "easy" convex problems, those problems that are not easy

Describe a number of examples in engineering, finance, operations research, etc

Emphasize how the approach can be applied to *decision problems with uncertain data,* in a static and dynamic setting

What we will cover

- recognizing & exploiting convexity in engineering context
- ideas of convex optimization
- a few algorithms (*less is more*, Le Corbusier) extremal on the run time/code time tradeoff curve
- applications in problems with uncertain data

What we won't do

- details of convex **analysis**
- details of optimization **theory** (regularity conditions, constraint qualifications, ...)
- encyclopedia of algorithms

Prerequisites

- good knowledge of linear algebra
  (we'll review some of it)
- elementary probability
- exposure to engineering
  (mechanical, electrical, civil, ...)
- elementary analysis (norms, limits, ...)
- knowledge of Matlab, or willingness to learn

Not required but helps

- exposure to optimization
- numerical linear algebra