ELEC 539A Lecture Notes

Selected Topics in Signal Processing:

Machine Learning for Signal Processing

Wu-Sheng Lu

Department of Electrical and Computer Engineering

University of Victoria

January 2015

COURSE OUTLINE

- **Instructor:**

Dr. W.-S. Lu
Phone: 8692
E-mail: wslu@ece.uvic.ca
URL: www.ece.uvic.ca/~wslu

- **Office Hours:**

Days: Wednesdays
Time: 14:40 – 16:40
Location: EOW 427

- **Lectures:**

Section: A01/CRN 23843
Days: Tuesdays, Wednesdays, and Fridays
Time: 13:30 – 14:20
Location: ELL 161

- **Text:**

Lecture Notes

- **Assessment:**

Assignments:	30%
Project:	30%
Final:	40%

- **Due Dates for Assignments and Project Report:**

Each assignment due date will be given in class as well as posted in the class web site. To receive course credit, you are required to submit the assignment no later than 4:00 pm on the due day. This course does not use drop-box for assignments and project reports. There are three ways to submit your assignment work:

- submit it to the instructor in class.
- submit by placing it in the instructor's mailbox in EOW 448.
- submit by sliding it underneath the door of EOW 427.

- Project report is due by 4:00 pm on the same day as the final exam.

The course requires CVX – a software package for convex optimization.

- Download current version of CVX from http://cvxr.com/cvx/download/ and a manual from http://cvxr.com/cvx/doc/
- You are strongly encouraged to go through the manual while trying the software on your own as early as possible.
- The course covers basic elements of supervised and unsupervised learning methods with applications in several signal processing problems. Review of key concepts of probability theory and numerical optimization techniques will be provided. A precise list of course contents will be made available at a later point of time.

References

[1] V. N. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[2] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed., Springer, 2000.

[3] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed., Springer, 2009.

[4] H. Stark and J. W. Woods, *Probability and Random Processes with Applications to Signal Processing*, 3rd ed., Prentice Hall, 2002.

[5] A. Antoniou and W.-S. Lu, *Practical Optimization–Algorithms and Engineering Applications*, Springer 2007.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[8] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin, *Learning from Data – A Short Course*, AMLbook.com, 2012.

[9] F. Rosenblatt, "The perception: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386-408, 1958.

[10] G.-X. Yuan, C.-H. Ho, and C.-J. Lin, "Recent advances of large-scale linear classification," *Proc. IEEE*, vol. 100, no. 9, pp. 2584-2603, Sept. 2012.

[11] T. Poggio and S. Smale, "The mathematics of learning: Dealing with data," Notices of the AMS, vol. 50, no. 5, pp. 537-544, May 2003.

[12] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd edition, The Johns Hopkins University Press, 1989.

Chapter 1 Preliminaries

1.1 Probability Theory

We begin by presenting a brief review of the probability theory. The very reason of doing so is that contemporary machine learning methodologies are developed in a probabilistic framework [1] - [3]. In other words, most part of machine leaning makes sense only when you look at it from a statistical perspective.

1.1.1 Definition of Probability

A. *Classical Probability*

There are different kinds of probability: probability as intuition, probability as the ratio of favorable to total outcomes (also known as classical theory of probability), probability as a measure of frequency of occurrence, and probability based on axiomatic theory [4].

Definition 1.1 *Random experiment* and *basic event*

An experiment *E* is said to be a *random experiment* if it can be performed repeatedly and its outcomes are *not* deterministic but probabilistic. An outcome of a random experiment, denoted by ω, is called a *basic event*. ■

Definition 1.2 *Sample space*

The set of all basic events, denoted by Ω, is called *sample space*. ■

Example 1.1

Let *E* be the experiment of picking a ball at random from a box containing *N* identical balls, where the balls have been marked as #1, #2, …, to #*N*. Obviously *E* is random experience because its outcome has *N* possibilities. If the number on the ball that has been picked is *i*, then the outcome is a basic event and is denoted by ω_i. Obviously, the sample space is $\Omega = \{\omega_1, \cdots, \omega_N\}$. ■

Example 1.2

Let *E* be that of observing the average daytime temperature in a city in October. Obviously *E* is a random experiment. If we denote the outcome of experiment *E* by ω_a, then the sample space in this case may be described as $\Omega = \{\omega_a, -\infty < \omega_a < \infty\}$. Note that Ω contains infinite number of basic events. ■

Definition 1.3 *General event*

A general event (or simply called it event), say *A*, consists of several basic events, hence a general event is a subset of sample space Ω. ■

Example 1.3

Continue from Example 1.1. Suppose one randomly picks a ball from the box, one can state that

• Event *A* of getting a ball whose number is no greater than 3 is a general event because it consists of three basic events: $A = \{\omega_1, \omega_2, \omega_3\}$ which is a proper subset of $\Omega = \{\omega_1, \cdots, \omega_N\}$.

• Event *B* of getting a ball with number even numbers is a general event because $B =$

$\{\omega_2, \omega_4, \ldots, \omega_{100}\} \subset \Omega$. ■

The classical probability theory applies to the cases where the random experiment *E* obeys two basic assumptions: (i) the total number of outcomes, *N*, is finite; and (ii) all outcomes (i.e. the results of basic events) are equally likely. Consequently, the probability of a (general) *event A*, denoted by *P*(*A*), is obtained *a priori* (i.e., relating to reasoning that proceeds from theoretical deduction rather than from observation or experience) by counting the number of ways N_e that event *A* can occur, then computing the ratio N_e/N as the probability. This is,

$$P(A) = \frac{N_e}{N} \tag{1.1}$$

Example 1.4

Continue from Example 1.3.

Let *A* be the event of obtaining a ball whose number is no greater than 3. The probability *P*(*A*) = 3/100 = 0.03.

Let *B* be the event of obtaining a ball that is even-numbered. The probability *P*(*B*) = 50/100 = 0.5.

Let *C* be the event of obtaining a ball whose number is a multiple of 3. The probability *P*(*C*) = 33/100 = 0.33.

■

The major problems with the classical theory of probability are that it cannot handle outcomes that are not equally likely; and it cannot deal with infinite number of outcomes.

B. *Probability Based On Axiomatic Theory*

Developed by A. N. Kolmogorov in 1930's, the probability based on axiomatic theory is the one that is followed by most modern texts on the subject. Kolmogorov's probability theory is set-theoretic in that a random event is considered as a subset of a sample space in an abstract setting where the two basic assumptions made in the classical probability theory are no longer necessary.

In order to present Kolmogorov's axiomatic definition of probability, basic elements of *set algebra* and the sigma algebra (σ – algebra) are sketched first.

Set Algebra

• A *set* is a collection of objects (or elements).

• A set *B* is said to be a *subset* of set *A*, and this relation is denoted by $B \subseteq A$, if *B* is contained within set *A*. Let for example set *A* denote all Victoria residents and set *B* be all Victoria residents whose height is between 5.5 and 6.5 feet. Obviously set *B* is a (proper) subset of set *A*.

• Let *A* and *B* be two sets in space Ω. The *union* (sum) of *A* and *B*, denoted by $A \bigcup B$ or *A* + *B*, is the set of elements that are in at least one of the sets *A* or *B*.

• The *intersection* (or set product) of *A* and *B*, denoted by $A \bigcap B$ or *AB*, is the set of elements that are in both *A* and *B*.

• The *empty* set, denoted by ϕ, is a set that contains no objects.

• Let *A* be a set in space Ω, the complement of *A*, denoted by A^c, is a set of all elements that are not in *A*. Obviously, $A \bigcup A^c = \Omega$ and $A \bigcap A^c = \phi$.

• Two sets A and B are said to be equal if both $B \subseteq A$ and $A \subseteq B$ hold.

• The difference of A and B, denoted by $A - B$, is the set of elements that are in A but not in B. It follows that

$$A - B = AB^c \quad \text{and} \quad B - A = BA^c$$

• The *exclusive-or* of sets A and B, denoted by $A \oplus B$, is the set of elements that are in A or B, but not in both. Obviously, we can write

$$A \oplus B = (A - B) \bigcup (B - A)$$

• Sets A and B are said to be disjoint if they have no elements in common, thus $AB = \phi$.

• It can readily be verified that

$$(A \bigcup B)^c = A^c \bigcap B^c \quad \text{and} \quad (A \bigcap B)^c = A^c \bigcup B^c$$

which can be extended by induction to the general case of n sets as

$$\left(\bigcup_{i=1}^{n} A_i \right)^c = \bigcap_{i=1}^{n} A_i^c \quad \text{and} \quad \left(\bigcap_{i=1}^{n} A_i \right)^c = \bigcup_{i=1}^{n} A_i^c$$

Definition 1.4 *σ-Algebra*

Let Ω be a set and we consider a collection of subsets of Ω, denoted by $\mathcal{F}$. $\mathcal{F}$ is said to be a *σ-Algebra* if

1. $\phi \in \mathcal{F}$ and $\Omega \in \mathcal{F}$,

2. If $A \in \mathcal{F}$ and $B \in \mathcal{F}$, then $A \bigcup B \in \mathcal{F}$ and $A \bigcap B \in \mathcal{F}$.

3. If $A \in \mathcal{F}$ then $A^c \in \mathcal{F}$.

4. $\mathcal{F}$ is closed under countable set of unions, intersections, and combinations. Hence if A_1, …, A_n, … belong to $\mathcal{F}$, then

$$\bigcup_{i=1}^{\infty} A_i \in \mathcal{F} \text{ and } \bigcap_{i=1}^{\infty} A_i \in \mathcal{F} \qquad \blacksquare$$

Why is σ-Algebra relevant?

Recall in the classical theory of probability the number of basic events must be finite and all outcomes are equally likely, i.e. the probability of every basic event is the same. The axiomatic probability theory built on σ-algebra allows us to remove these two fundamental limitations. Consider a random experiment and the associated sample space Ω. In the axiomatic probability theory we only consider the collection $\mathcal{F}$ of those subsets of Ω that form a σ-algebra. These subsets are called events. It turns out that $\mathcal{F}$ usually includes all subsets of engineering and science interest.

Definition 1.5 *Axiomatic probability*

Given a sample space Ω and a σ-algebra $\mathcal{F}$ of Ω, probability $P[\cdot]$ is a set function that

assigns to every event $A \in \mathcal{F}$ that obeys the three axioms:

(1) $P[A] \geq 0$.

(2) $P[\Omega] = 1$.

(3) $P[A \cup B] = p[A] + P[B]$ if $A \cap B = \phi$.

From these axioms, one can establish the following important properties:

(4) $P[\phi] = 0$.

(5) $P[A \cap B^c] = P[A] - P[A \cap B]$.

(6) $P[A] = 1 - P[A^c]$.

(7) $P[A \cup B] = P[A] + P[B] - P[A \cap B]$.

(8) $P\left[\bigcup_{i=1}^{n} A_i\right] = \sum_{i=1}^{n} P[A_i]$ if $A_i \cap A_j = \phi$ for all $i \neq j$. ■

Definition 1.6 *Probability space*

The triple of a sample space Ω, a collection of (general) events $\mathcal{F}$ that form a σ-algebra, and a probability measure P, namely $(\Omega, \mathcal{F}, P)$, is called a probability space. ■

1.1.2 Joint Probability, Conditional Probability, and Independence

A. *Joint Probability*

The *joint probability* of two events A and B is defined as the probability of "events A and B both occur", namely $P[A \cap B]$ or P[AB].

B. *Conditional Probability*

Often times people want to know the probability of "event A occurs" given that event B has occurred. This is called *conditional probability* and is denoted by $P[A|B]$. It is intuitively clear that because of the presence of the condition that "event B has occurred", $P[A|B]$ is in general different from $P[A]$.

Example 1.5

Suppose there are four identical balls in a box, which are marked as #1, #2, #3, and #4. The random experiment is to pick up a ball from the box. Now consider two events: Event A is that of obtaining ball #4; and event B is that of obtaining an even-numbered ball. Obviously, we have $P[A] = ¼$ and $P[A|B] = ½$. ■

A question that naturally arises is how to define and compute conditional probability in general circumstances? Let A and B are two events in a random experiment. Each outcome of the random experiment must fall into one of the four cases: (1) A occurs and B does not; (2) B occurs and A does not; (3) A and B both occur; and (4) A and B both do not occur.

Suppose one applies above analysis to Example 1.5, repeats the experiment n times, and denotes the times that each of the four cases occurs by n_1, n_2, n_3, and n_4, respectively. We can state that

- $n_1 + n_2 + n_3 + n_4 = n$
- the frequency of event $B = F_n(B) = \dfrac{n_2 + n_3}{n}$.
- the frequency of event $AB = F_n(AB) = \dfrac{n_3}{n}$.
- given that event B has occurred, the frequency of event $A = F_n(A|B) = \dfrac{n_3}{n_2 + n_3}$.

It follows that

$$F_n(A \mid B) = \frac{F_n(AB)}{F_n(B)} \quad \text{provided that} \quad F_n(B) > 0 \tag{1.2}$$

Based on (1.2), conditional probability is defined as follows.

Definition 1.7 *Conditional probability*

Let $(\Omega, \mathcal{F}, P)$ be a probability space, $A \in \mathcal{F}, B \in \mathcal{F}$ with $P[B] > 0$. The conditional probability of event A given that event B has occurred is defined by

$$P[A \mid B] = \frac{P[AB]}{P[B]} \tag{1.3}$$

Similarly, the conditional probability of event B given that event A has occurred is defined by

$$P[B \mid A] = \frac{P[AB]}{P[A]} \tag{1.4}$$

provided that $P[A] > 0$. From (1.3) and (1.4) it follows that

$$P[AB] = P[A]P[B \mid A] = P[B]P[A \mid B] \qquad \blacksquare \tag{1.5}$$

Example 1.6

Consider a binary communication system with a two-symbol alphabet, i.e., 0 and 1. Let X and Y be the transmitted and received symbols, respectively. Here the sample space is given by $\Omega = \{(X, Y) : X = 0 \text{ or } 1, Y = 0 \text{ or } 1) = \{(0,0), (0,1), (1,0), (1,1)\}$. Suppose that the communication system is slightly corrupted by noise such that

$$P[Y = 1 \mid X = 1] = 0.92, \quad P[Y = 0 \mid X = 1] = 0.08$$
$$P[Y = 0 \mid X = 0] = 0.92, \quad P[Y = 1 \mid X = 0] = 0.08$$

And by design, $P[X = 0] = P[X = 1] = 0.5$. Under these circumstances, we have

$$P[X=0,Y=0]=P[X=0]\cdot P[Y=0\mid X=0]=0.5\times 0.92=0.46$$
$$P[X=0,Y=1]=P[X=0]\cdot P[Y=1\mid X=0]=0.5\times 0.08=0.04$$
$$P[X=1,Y=0]=P[X=1]\cdot P[Y=0\mid X=1]=0.5\times 0.08=0.04$$
$$P[X=1,Y=1]=P[X=1]\cdot P[Y=1\mid X=1]=0.5\times 0.92=0.46$$

■

Properties of Conditional Probability

Property 1 (Multiplication formula)

Let $A_1, A_2, \ldots, A_n$ be n events with $n \geq 2$ and $P[A_1A_2\cdots A_{n-1}]>0$, then

$$P[A_1A_2\cdots A_n]=P[A_1]\cdot P[A_2\mid A_1]\cdot P[A_3\mid A_1A_2]\cdots P[A_n\mid A_1A_2\cdots A_{n-1}] \tag{1.6}$$

Proof:

Using (1.3), we see that the right-hand side of (1.6) is equal to

$$P[A_1]\cdot\frac{P[A_1A_2]}{P[A_1]}\cdot\frac{P[A_1A_2A_3]}{P[A_1A_2]}\cdots\frac{P[A_1A_2\cdots A_n]}{P[A_1A_2\cdots A_{n-1}]}=P[A_1A_2\cdots A_n] \quad ■$$

Property 2 (Unconditional probability)

Let $A_1, A_2, \ldots, A_n$ be mutually exclusive events such that $\bigcup_{i=1}^{n} A_i=\Omega$ with $P[A_i]>0$ for all i. Let B be any event defined over the probability space of A_i's. Then

$$P[B]=\sum_{i=1}^{n}P[A_i]P[B\mid A_i] \tag{1.7}$$

Proof:

From

$$B=B\Omega=B\bigcup_{i=1}^{n}A_i=\bigcup_{i=1}^{n}BA_i$$

where $\{BA_i\}$ are mutually exclusive, it follows that

$$P[B]=P\left[\bigcup_{i=1}^{n}BA_i\right]=\sum_{i=1}^{n}P[BA_i]=\sum_{i=1}^{n}P[A_i]\cdot P[B\mid A_i] \quad ■$$

Property 3 (Bayes formula)

Let $A_1, A_2, \ldots, A_n$ be mutually exclusive events such that $\bigcup_{i=1}^{n} A_i=\Omega$ with $P[A_i]>0$ for all i. Let B be any event with $P[B]>0$. Then

$$P[A_j\mid B]=\frac{P[B\mid A_j]\cdot P[A_j]}{\sum_{i=1}^{n}P[B\mid A_i]\cdot P[A_i]} \tag{1.8}$$

Proof:

By following the definition of conditional probability and the formula of unconditional probability in (1.7), we obtain

$$P[A_j \mid B] = \frac{P[BA_j]}{P[B]} = \frac{P[B \mid A_j] \cdot P[A_j]}{\sum_{i=1}^{n} P[B \mid A_i] \cdot P[A_i]} \qquad \blacksquare$$

The Bayes formula finds many applications in science and engineering. The terms in (1.8) bear various names: $P[A_j|B]$ is often called *a posteriori* probability of A_j given B; $P[B|A_j]$ is called the *a priori* probability of B given A_j; and $P[A_j]$ is called the *causal* or *a priori* probability of A_j. Typically *a priori* probabilities are estimated from past measurements or presupposed by experience, while *a posteriori* probabilities are measured or computed from observations [4].

Example 1.7

Suppose there are three boxes that look identical, each contains certain number of red and blue balls that are identical except the color. The number of red and blue balls in box i are r_i and b_i, respectively, for $i = 1, 2, 3$. The experiment in question is to randomly pick a box, then randomly pick a ball. The outcome was a red ball. Given that it is a red ball, compute the probability of the ball belongs to box 1.

Solution

Let A_i be the event of the ball in question belongs to box i for $i = 1, 2, 3$. Obviously, $\{A_i$, for $i = 1, 2, 3\}$ are mutually exclusive, $\bigcup_{i=1}^{3} A_i = \Omega$, and $P[A_1] = P[A_2] = P[A_3] = 1/3$. We also define event B as "it is a red ball". With these definitions, the problem we need to address is to compute the conditional probability $P[A_1|B]$.

Clearly, formula (1.8) is applicable if we are able to compute conditional probabilities $P[B|A_i]$ for $i = 1, 2, 3$. These conditional probabilities are found to be

$$P[B \mid A_i] = \frac{r_i}{r_i + b_i} \quad \text{for} \quad i = 1, 2, 3.$$

Hence (1.8) yields

$$P[A_1 \mid B] = \frac{\frac{1}{3}\frac{r_1}{r_1 + b_1}}{\frac{1}{3}\frac{r_1}{r_1 + b_1} + \frac{1}{3}\frac{r_2}{r_2 + b_2} + \frac{1}{3}\frac{r_3}{r_3 + b_3}} = \frac{1}{1 + \frac{r_2(r_1 + b_1)}{r_1(r_2 + b_2)} + \frac{r_3(r_1 + b_1)}{r_1(r_3 + b_3)}} \qquad \blacksquare$$

C. *Independence*

Consider two events A and B. We have seen that in general probability $P[A]$ differs from the conditional probability $P[A|B]$ as long as the occurrence of event B has an impact on occurrence of event A. In other words, if $P[A] = P[A|B]$ happens, then the occurrence of B has no impact on the occurrence of A, and we will say events A and B are mutually independent. Note that $P[A] = P[A|B]$ in conjunction with (1.3) leads to

$$P[AB] = P[A] \cdot P[B] \qquad (1.9)$$

This motivates the following definition.

Definition 1.8 *Independence*

- Events A and B are said to be independent from each other if (1.9) holds.
- n events $A_1, A_2, \ldots, A_n$ are said to be mutually independent if

$$P[A_1 A_2 \cdots A_n] = P[A_1] \cdot P[A_2] \cdots P[A_n] \qquad \blacksquare \qquad (1.10)$$

1.1.3 Random Variables, Distribution Function, and Probability Density

A. *Random Variables*

Let E be a random experiment associated with sample space Ω. Corresponding to each outcome ω, suppose there is a real-valued function $\xi(\omega)$. In probability theory, one is concerned not only with the value of $\xi(\omega)$, but also with the probability of $\xi(\omega)$ taking certain values.

Example 1.8

In a junior-high school a student (that is an ω) is randomly selected, whose height is recorded as $\xi(\omega)$. One of the valid questions in this case would be "what is the probability of the height no greater than x cm?", namely, what is $P[\xi(\omega) \le x]$? ■

Evidently, to address the question in the associated probability space $(\Omega, \mathcal{F}, P)$, it is necessary to assure that $\xi(\omega) \le x$ belongs to $\sigma-$ algebra $\mathcal{F}$ so that $P[\xi(\omega) \le x]$ is well defined. In probability theory functions of this kind are called *random variables*.

Definition 1.9 *Random variables*

Let $\xi(\omega)$ be a real-valued function defined over sample space Ω that is associated with a probability space $(\Omega, \mathcal{F}, P)$. If, for any real x, $(\xi(\omega) \le x)$ is an event, i.e., $(\xi(\omega) \le x) \in \mathcal{F}$, then $\xi(\omega)$ is called a random variable. ■

The introduction of random variables was a major event in the development of modern probability theory as the concept made it possible to extend the probability theory to include studies of random variables and a variety of related issues. One of the issues is *distribution function* of a random variable.

B. *Distribution Function and Probability Density*

Definition 1.10 *Distribution function of a random variable*

Let $\xi(\omega)$ be a random variable, the distribution function of ξ, denoted by $\Phi_\xi(x)$, is defined as the probability of event $(\xi(\omega) \le x)$. Namely,

$$\Phi_\xi(x) = P[\xi(\omega) \le x] \qquad \blacksquare \qquad (1.11)$$

Several basic properties of distribution functions follow:

- $\Phi_\xi(x)$ is monotonically non-decreasing, i.e.,

$$x_2 \geq x_1 \text{ implies that } \Phi_\xi(x_2) \geq \Phi_\xi(x_1) \tag{1.12}$$

- $$\Phi_\xi(-\infty) = 0, \ \Phi_\xi(\infty) = 1 \tag{1.13}$$

- $$P[x_1 < \xi(\omega) \leq x_2] = \Phi_\xi(x_2) - \Phi_\xi(x_1) \tag{1.14}$$

- $$P[\xi(\omega) > x] = 1 - \Phi_\xi(x) \tag{1.15}$$

- $$P[\xi(\omega) = x] = \Phi_\xi(x) - \Phi_\xi(x - 0) \tag{1.16}$$

where $\Phi_\xi(x-0) = \lim_{a \uparrow x} \Phi_\xi(a)$.

Types of distribution functions

There are several types of distribution functions, with the most important being *discrete type* and *continuous type*.

Discrete type distribution

A discrete random variable ξ takes on discrete values x_0, x_1, x_2, … (finite or countable infinite), with probabilities p_0, p_1, p_2, …, respectively. Thus a discrete distribution is characterized by a two-row matrix, known as *density matrix*, with finite or countable infinite columns as

$$\xi \Leftrightarrow \begin{bmatrix} x_0 & x_1 & x_2 & \cdots \\ p_0 & p_1 & p_2 & \cdots \end{bmatrix} \tag{1.17}$$

Obviously we have $p_i \geq 0$ and $\sum_i p_i = 1$. The distribution function in this case is given by

$$F(x) = \sum_{i:\, x_i \leq x} p_i \tag{1.18}$$

which is a staircase function that is continuous from right, see Fig. 1.1.

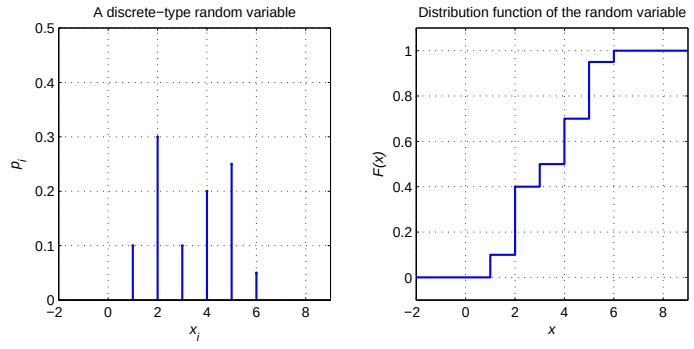

Fig. 1.1 Left: A discrete-type random variable with $x_0 = 1$, $x_1 = 2$, …, $x_5 = 6$ and $p_0 = 0.1$, $p_1 = 0.3$, $p_2 = 0.1$, $p_3 = 0.2$, $p_4 = 0.25$, and $p_5 = 0.05$; Right: The distribution function of the random variable.

Example 1.9 *Bernoulli distribution*

Also known as *binomial distribution*, the Bernoulli distribution is discrete and assumes the form of

$$\begin{bmatrix} 0 & 1 & \cdots & n \\ p_0 & p_1 & \cdots & p_n \end{bmatrix}$$

where

$$p_k = \binom{n}{k} p^k q^{n-k} \quad p \geq 0,\ q \geq 0,\ p+q=1 \tag{1.19}$$

with

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

being the number of combinations of taking k elements out of n elements. Note that p_k in (1.19) are exactly the coefficient of polynomial $(p+q)^n$, hence the name of binomial distribution. The reason it also bears the name of Bernoulli distribution has to do with the well-known *Bernoulli trial* which is a random experiment with exactly two possible outcomes: "success" and "failure". Denote the probability of success and failure by p and q, respectively, and suppose the experiment is repeated n time independently. It is obvious that there are 2^n possible outcomes; the number of outcomes with k successes and $(n-k)$ failures is equal to $\binom{n}{k}$; and the probability of "one such event occurs" is $p^k q^{n-k}$. Therefore, the probability of having k successes and $(n-k)$ failures is precisely given by p_k as seen in (1.19). ■

Continuous type distribution

A distribution of random variable ξ is said to be of continuous type if its distribution function assumes the form

$$\Phi_\xi(x) = \int_{-\infty}^{x} \varphi_\xi(y)\,dy \tag{1.20}$$

where $\varphi_\xi(x) \geq 0$ is called *density function* or *density*. Since $\Phi_\xi(\infty) = 1$, we have

$$\int_{-\infty}^{\infty} \varphi_\xi(y)\,dy = 1$$

Example 1.10 *Uniform distribution*

Let $a < b$. Consider the random variable ξ that takes a value over the interval $[a, b]$ uniformly randomly. The associated distribution, called uniform distribution, is of continuous type with the density function given by

$$\varphi_\xi(x) = \begin{cases} \dfrac{1}{b-a} & \text{if } x \in [a,b] \\ 0 & \text{if } x \notin [a,b] \end{cases} \tag{1.21}$$

Given the density in (1.21), the distribution function is found to be

$$\Phi_\xi(x) = \int_{-\infty}^{x} \varphi_\xi(y)\,dy = \begin{cases} 0 & \text{if } x < a \\ \dfrac{x-a}{b-a} & \text{if } a \le x \le b \\ 1 & \text{if } x > b \end{cases} \tag{1.22}$$

See Figs. 1.2 and 1.3 for plots of the density $\varphi_\xi(x)$ and distribution function $\Phi_\xi(x)$. ■

C. Expectation, Variance and Covariance

An important operation involving random variables is that of finding weighted average of a random variable ξ or a (measurable) function of ξ.

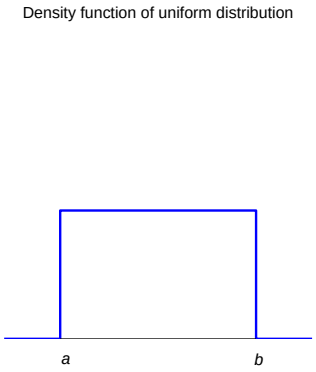

Figure 1.2

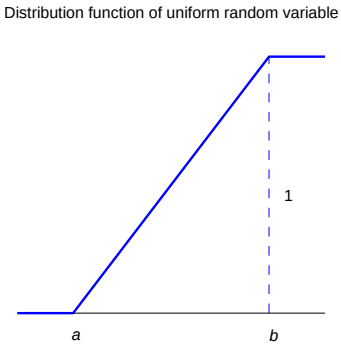

Figure 1.3

Definition 1.11 *Expectation*

The average of a random variable ξ under its probability distribution is called the expectation of ξ, denoted by $E[\xi]$.

For a discrete type random variable ξ that takes values x_0, x_1, x_2, … with probability p_0, p_1, p_2, … (recall the density matrix given by (1.17), namely

$$\begin{bmatrix} x_0 & x_1 & x_2 & \cdots \\ p_0 & p_1 & p_2 & \cdots \end{bmatrix}$$

), the expectation of ξ, also known as mean value of ξ, is defined as

$$E[\xi] = \sum_i x_i p_i$$

Since the random variable ξ here takes values x_i, for the sake of notation convenience we denote the random variable by x, and write the above expression as

$$E[x] = \sum_i x_i p_i \tag{1.23}$$

For a continuous type random variable ξ with density $\varphi_\xi(x)$, its expectation is defined as

$$E[x] = \int_{-\infty}^{\infty} x\varphi_\xi(x)dx \tag{1.24}$$

The concept of expectation can be extended to expectation of a (measurable) function, say $g(\xi)$, of random variable ξ as

$$E[g(x)] = \sum_i g(x_i) p_i \tag{1.25}$$

for discrete-type case, or

$$E[g(x)] = \int_{-\infty}^{\infty} g(x)\varphi_\xi(x)dx \tag{1.26}$$

for continuous-type case. ■

Definition 1.12 *Variance*

The variance of a function g of random variable ξ provides a measure of how much there is in g around its mean value $E[g(x)]$:

$$\text{var}[g(x)] = E\left[\left(g(x) - E[g(x)]\right)^2\right] = E\left[g(x)^2\right] - \left(E[g(x)]\right)^2 \tag{1.27}$$

In particular, with $g(x) = x$ (1.27) yields the variance of a random variable ξ itself as

$$\text{var}[x] = E\left[x^2\right] - \left(E[x]\right)^2 \quad ■ \tag{1.28}$$

Definition 1.13 *Covariance*

Covariance is concerned with two random variables *x* and *y* and quantifies the extent to which *x* and *y* vary together, namely,

$$\text{cov}[x, y] = E_{x,y}\left[\left(x - E[x]\right)\left(y - E[y]\right)\right] = E_{x,y}[xy] - E[x]E[y] \tag{1.29}$$

where $E_{x,y}$ denotes expectation with respect to joint density $\varphi(x, y)$, namely,

$$E_{x,y}[xy] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} xy\,\varphi(x, y)\,dx\,dy \quad ■$$

D. *Independence of Random Variables and Conditional Distribution*

Definition 1.14 *Independence of random variables*

Random variables $\xi_1(\omega),\ldots,\xi_n(\omega)$ with distribution functions $\Phi_1(x_1),\ldots,\ \Phi_n(x_n)$ are said to be *mutually independent* if

$$\Phi(x_1,\ldots,x_n)=\Phi_1(x_1)\cdots\Phi_n(x_n) \tag{1.30}$$

holds, where $\Phi(x_1,\ldots,x_n)$ is the joint distribution of $\xi_1(\omega),\cdots,\xi_n(\omega)$, namely,

$$\Phi(x_1,\ldots,x_n)=P[\xi_1\le x_1,\cdots,\xi_n\le x_n] \qquad \blacksquare$$

From (1.30), it follows that

$$P[\xi_1\le x_1,\cdots,\xi_n\le x_n]=P[\xi_1\le x_1]\cdots P[\xi_n\le x_n]=\prod_{i=1}^{n}P[\xi_i\le x_i] \tag{1.31}$$

In words, if random variables $\xi_1,\ldots,\xi_n$ are mutually independent, the joint probability of the event $(\xi_1\le x_1,\cdots,\xi_n\le x_n)$ is equal to product of the probability of individual event $(\xi_i\le x_i)$.

For continuous type of random variables, (1.30) implies that $\xi_1(\omega),\cdots,\xi_n(\omega)$ are mutually independent if the joint probability density equals the product of the individual probability density almost everywhere (a.e.), i.e.,

$$\varphi(x_1,\ldots,x_n)=\varphi_1(x_1)\cdots\varphi_n(x_n) \qquad \text{(a.e.)} \tag{1.32}$$

Definition 1.15 *Conditional distribution*

For two discrete-type random variables ξ and η with

$$\xi\ \Leftrightarrow\begin{bmatrix}x_0 & x_1 & x_2 & \cdots\\ p_0 & p_1 & p_2 & \cdots\end{bmatrix}\ \text{and}\ \eta\ \Leftrightarrow\begin{bmatrix}y_0 & y_1 & y_2 & \cdots\\ q_0 & q_1 & q_2 & \cdots\end{bmatrix}$$

where $p_i>0$ and $q_i>0$ are assumed, denote the joint probability

$$P[\xi=x_i,\eta=y_j]=p_{i,j}$$

hence we can write

$$p_i=\sum_j p_{i,j}\ \text{ and }\ q_j=\sum_i p_{i,j}$$

which in turn implies that

$$P[\xi=x_i\,|\,\eta=y_j]=\frac{P[\xi=x_i,\eta=y_j]}{P[\eta=y_j]}=\frac{p_{i,j}}{q_j}=\frac{p_{i,j}}{\sum_i p_{i,j}} \tag{1.33}$$

and

$$P[\eta = y_j \mid \xi = x_i] = \frac{p_{i,j}}{p_i} = \frac{p_{i,j}}{\sum_j p_{i,j}} \tag{1.34}$$

The conditional distribution refers to the probability of event $(\xi \le x$ given $\eta = y_j)$ or that of $(\eta \le y$ given $\xi = x_i)$. By using (1.33) and (1.34), we obtain

$$P[(\xi \le x \mid \eta = y_j)] = \frac{\sum_{i:x_i \le x} p_{i,j}}{\sum_i p_{i,j}} \tag{1.35}$$

and

$$P[\eta \le y \mid \xi = x_i] = \frac{\sum_{j:y_j \le y} p_{i,j}}{\sum_j p_{i,j}} \tag{1.36}$$

From (1.35) and (1.36), we see that conditional distributions involving two discrete-type random variables can be expressed (and evaluated) using their joint probability distribution $p_{i,j}$.

For two continuous-type random variables ξ and η with joint probability density $\varphi(x, y)$, the conditional distributions are given by

$$P[\xi \le x \mid \eta = y] = \frac{\int_{-\infty}^{x} \varphi(z, y)\, dz}{\int_{-\infty}^{\infty} \varphi(z, y)\, dz} \tag{1.37}$$

and

$$P[\eta \le y \mid \xi = x] = \frac{\int_{-\infty}^{y} \varphi(x, z)\, dz}{\int_{-\infty}^{\infty} \varphi(x, z)\, dz} \tag{1.38}$$

By writing (1.37) as

$$P[\xi \le x \mid \eta = y] = \int_{-\infty}^{x} \left(\frac{\varphi(z, y)}{\int_{-\infty}^{\infty} \varphi(z, y)\, dz} \right) dz$$

one may interpolate the integrand in the above expression as the *conditional distribution density* and denote it as $\varphi(x \mid y)$:

$$\varphi(x \mid y) = \frac{\varphi(x, y)}{\int_{-\infty}^{\infty} \varphi(z, y)\, dz} \tag{1.39}$$

In this way, (1.37) becomes

$$P[\xi \le x \mid \eta = y] = \int_{-\infty}^{x} \varphi(z \mid y)\, dz$$

Similarly, by defining the conditional distribution density

$$\varphi(y \mid x) = \frac{\varphi(x, y)}{\int_{-\infty}^{\infty} \varphi(x, z)\, dz} \tag{1.40}$$

(1.38) can be expressed as

$$P[\eta \le y \mid \xi = x] = \int_{-\infty}^{y} \varphi(z \mid x)\, dz$$ ■

1.1.4 Gaussian Distribution

Also known as the *normal distribution*, the Gaussian distribution played an extremely important role in the development of probability theory, and is arguably the most useful among all probabilistic distributions. The life span of bulbs produced under practically same manufacturing conditions, for example, obeys a Gaussian distribution. This is also true for several other measures of products that are manufactured in quantity under the same conditions. Gaussian distribution is also encountered in many natural, biological, and social events/phenomena: velocities of gas molecules; errors in measuring a physical object; heights/weights of biological species, yearly precipitations of a certain city, etc. A common thread of these random variables is that they are cumulative synthesis of many small (i.e. of minor importance), independent random components.

A. *One-Dimensional Gaussian Distribution*

Definition 1.16 *Gaussian distribution*

A one-dimensional Gaussian distribution is a continuous-type distribution whose density is given by

$$\varphi_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{1.41}$$

where μ and σ are two real-valued parameters. Hence the Gaussian distribution function is given by

$$\Phi_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-(y-\mu)^2/2\sigma^2} dy \tag{1.42}$$

A common notation for the one-dimensional normal distribution is $\mathcal{N}(x, \mu, \sigma^2)$. ■

By (1.24) and (1.28), the expectation and variance of a Gaussian random variable are found to be

$$E[x] = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} x e^{-(x-\mu)^2/2\sigma^2} dx = \mu \tag{1.43}$$

and

$$\text{var}[x] = E[x^2] - E[x]^2 = \sigma^2 \tag{1.44}$$

respectively. In other words, we see that the Gaussian distribution is characterized by its expectation and variance. Fig. 1.4 illustrates the meaning of μ as the mean value of x using two Gaussian density functions with different μ's, while Fig. 1.5 illustrates the meaning of σ^2 as the variance of x by several Gaussian density functions with different $\sigma's$.

Example 1.11

Let x be a Gaussian random variable with $\mu = 2$ and $\sigma = 1$. Compute the probability of x being in between $\mu - 3$ and $\mu + 3$ (i.e. x falls into the interval [–1, 5]).

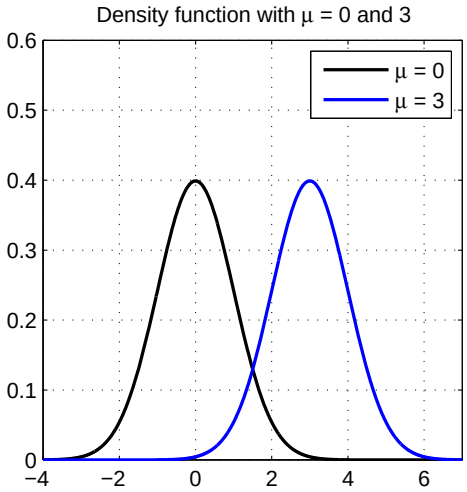

Figure 1.4

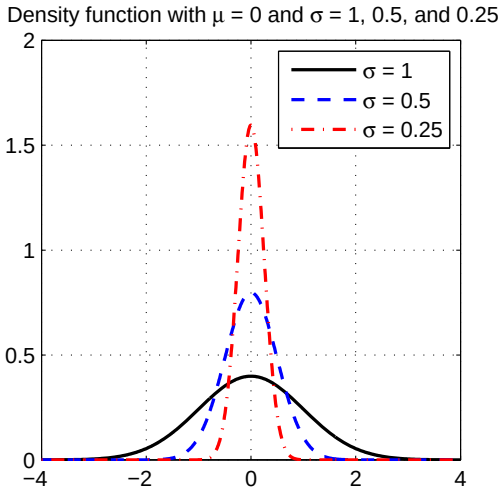

Figure 1.5

Solution

From (1.14) and (1.42) it follows that

$$P[-1 \le \xi \le 5] = \frac{1}{\sqrt{2\pi}} \int_{-1}^{5} e^{-(y-2)^2/2} dy = \frac{1}{\sqrt{2\pi}} \int_{-3}^{3} e^{-z^2/2} dz \approx 99.7\% \qquad \blacksquare$$

B. *Multidimensional Gaussian Distribution*

Let us consider n random variables $\xi_1(\omega), \cdots, \xi_n(\omega)$. The joint distribution of these random variables is defined by

$$\Phi(x_1, x_2, \cdots, x_n) = P[\xi_1(\omega) \le x_1, \xi_2(\omega) \le x_2, \cdots, \xi_n(\omega) \le x_n] \tag{1.45}$$

If $\xi_1(\omega), \cdots, \xi_n(\omega)$ are of continuous type, then $\Phi(x_1, x_2, \cdots, x_n)$ can be expressed as

$$\Phi(\boldsymbol{x}) = \int_{-\infty}^{x_1} \cdots \int_{-\infty}^{x_n} \varphi(y_1, \cdots, y_n)\, dy_n \cdots dy_1 \tag{1.46}$$

where $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$ and $\varphi(x_1, \cdots, x_n)$ is the density function.

As expected, the most important continuous-type multidimensional distribution is Gaussian distribution whose density in matrix notation is given by

$$\varphi(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right\} \tag{1.47}$$

where $\boldsymbol{\Sigma}$ is a symmetric and positive definite matrix of size *n* by *n*, called *covariance matrix*, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$. Oftentimes the density of normal distribution is denoted by $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$. It can be verified that the expectation of Gaussian $\boldsymbol{\xi}$ is $\boldsymbol{\mu}$, and the covariance of between the individual components (as random variables themselves) of $\boldsymbol{\xi}$ is given by $\boldsymbol{\Sigma}$.

An important special case of multidimensional Gaussian distribution is when the individual random variables are mutually independent, the density function in this case becomes separable as

$$\varphi(x_1, \cdots, x_n) = \varphi_1(x_1) \cdots \varphi(x_n)$$

with each $\varphi_i(x_i)$ being a one-dimensional Gaussian density, i.e.,

$$\varphi_i(x_i) = \frac{1}{\sigma_i \sqrt{2\pi}} e^{-(x_i - \mu_i)^2 / 2\sigma_i^2}$$

Therefore the probability density of *n* mutually independent Gaussian distribution is given by

$$\varphi(x_1, \cdots, x_n) = \prod_{i=1}^{n} \frac{1}{\sigma_i \sqrt{2\pi}} e^{-(x_i - \mu_i)^2 / 2\sigma_i^2} = \frac{1}{(2\pi)^{n/2}} \frac{1}{\sigma_1 \cdots \sigma_n} \exp\left(-\frac{1}{2} \sum_{i=1}^{n} \frac{(x_i - \mu_i)^2}{\sigma_i^2} \right) \tag{1.48}$$

It can be readily verified that (1.47) coincides with (1.48) by assigning $\boldsymbol{\mu} = \begin{bmatrix} \mu_1 & \cdots & \mu_n \end{bmatrix}$ and

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{bmatrix}$$

In other words, (1.47) becomes joint density of *n* independent Gaussian distributions if and only if $\boldsymbol{\Sigma}$ is *diagonal*.

1.1.5 Likelihood Function and Log-Likelihood

Consider a continuous-type distribution with density $\varphi(x, \boldsymbol{w})$ where vector $\boldsymbol{w}$ collects the parameters involved. Let $\mathcal{D}$ be a data set $\mathcal{D} = \{x_1, x_2, \ldots, x_N\}$ that are drawn independently from the same distribution. In literature, a data set of this type is said to be *independently and identically distributed* (i.i.d.). Because of its probabilistic independence, given parameter $\boldsymbol{w}$ the probability

density of an i.i.d. data set $\mathcal{D}$ assumes the form

$$p(\mathcal{D} \mid \boldsymbol{w}) = \prod_{i=1}^{N} \varphi(x_i, \boldsymbol{w}) \tag{1.49}$$

which is called *likelihood function* of the probability distribution. A popular approach for determining the parameters of a probability distribution using an observed data is to maximize the likelihood function with respect to the parameters. Because logarithm is a monotonically increasing function and logarithm of the likelihood function simplifies subsequent mathematical analysis, one maximizes the log-likelihood instead:

$$\log p(\mathcal{D} \mid \boldsymbol{w}) = \sum_{i=1}^{n} \log \varphi(x_i, \boldsymbol{w}) \tag{1.50}$$

Example 1.12

Consider the 1-D Gaussian distribution in (1.41), i.e.

$$\varphi_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

Following (1.50), its log-likelihood function for an observed i.i.d. data set, denoted by $L(\mu, \sigma^2, \mathcal{D})$, is given by

$$L(\mu, \sigma^2, \mathcal{D}) = -\frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i - \mu)^2 - \frac{N}{2}\log\sigma^2 - \frac{N}{2}\log(2\pi)$$

Maximizing $L(\mu, \sigma^2, \mathcal{D})$ with respect to $\{\mu, \sigma^2\}$ yields *maximum likelihood* (ML) estimates

$$\mu_{ML} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{1.51}$$

and

$$\sigma^2_{ML} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_{ML})^2 \qquad \blacksquare \tag{1.52}$$

1.2 Optimization Methods

1.2.1 Unconstrained Optimization [5]

Unconstrained optimization studies problems of the form

$$\underset{\boldsymbol{x} \in R^n}{\text{minimize}} \quad f(\boldsymbol{x}) \tag{1.53}$$

where $f(\boldsymbol{x})$ is a read-valued twice continuously differentiable function.

- Gradient of $f(\boldsymbol{x})$ is defined by

$$\nabla f(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix} \dfrac{\partial f}{\partial x_1} \\ \dfrac{\partial f}{\partial x_2} \\ \vdots \\ \dfrac{\partial f}{\partial x_n} \end{bmatrix}$$

- Hessian of $f(\boldsymbol{x})$ is defined b by

$$\nabla^2 f(\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{x}) = \left\{ \frac{\partial^2 f}{\partial x_i \partial x_j} \right\}_{i,j=1,2,\ldots,n}$$

Note that Hessian is always square and symmetric.

- A point $\boldsymbol{x}$ is called a *stationary point* if $\nabla f(\boldsymbol{x}) = \boldsymbol{0}$. A geometric interpretation of the concept in the case of n = 1 or 2 is that $\boldsymbol{x}$ is a stationary point if the tangent (plane) of $f(\boldsymbol{x})$ at $\boldsymbol{x}$ is in parallel with the $\boldsymbol{x}$-axis (plane).

A. *First-Order Necessary Condition*

If $\boldsymbol{x}^*$ is a local minimum point (minimizer), then

$$\nabla f(\boldsymbol{x}^*) = \boldsymbol{0} \tag{1.54}$$

In other words, if $\boldsymbol{x}^*$ is a minimum point, then it must be a stationary point.

◊ A geometric interpretation of the 1st-order necessary condition when n = 1 or 2 is that the tangent plane of $f(\boldsymbol{x})$ at a minimizer must be in parallel to the $\boldsymbol{x}$-axis (plane).

◊ $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ is not sufficient for $\boldsymbol{x}^*$ to be a local minimizer.

B. *Second-Order Necessary Conditions*

If $\boldsymbol{x}^*$ is a local minimum point (minimizer), then

$$\nabla f(\boldsymbol{x}^*) = \boldsymbol{0} \quad \text{and} \quad \nabla^2 f(\boldsymbol{x}^*) \succeq \boldsymbol{0} \tag{1.55}$$

◊ $\nabla^2 f(\boldsymbol{x}^*) \succeq \boldsymbol{0}$ (i.e. the Hessian $\nabla^2 f(\boldsymbol{x}^*)$ is positive semidefinite if and only if the eigenvalues of $\nabla^2 f(\boldsymbol{x}^*)$ are nonnegative.

◊ $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ and $\nabla^2 f(\boldsymbol{x}^*) \succeq \boldsymbol{0}$ are not sufficient for $\boldsymbol{x}^*$ to be a local minimizer.

C. *Second-Order Sufficient Conditions*

Point $\boldsymbol{x}^*$ is a local minimizer, if

$$\nabla f(\boldsymbol{x}^*) = \mathbf{0} \quad \text{and} \quad \nabla^2 f(\boldsymbol{x}^*) \succ \mathbf{0} \tag{1.56}$$

D. ***Convex Sets and Convex Functions***

- A set $S \subseteq R^n$ is said to be convex if

$$\boldsymbol{x}_1, \boldsymbol{x}_2 \in S \Rightarrow \alpha \boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2 \in S \quad \text{for} \quad 0 \le \alpha \le 1 \tag{1.57}$$

- A function $f(\boldsymbol{x})$ is said to be convex over convex set S if

$$f(\alpha \boldsymbol{x}_1 + (1-\alpha)\boldsymbol{x}_2) \le \alpha f(\boldsymbol{x}_1) + (1-\alpha) f(\boldsymbol{x}_2) \tag{1.58}$$

for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$ and $0 \le \alpha \le 1$.

- A function $f(\boldsymbol{x}) \in C^1$ (i.e. continuously differentiable) is convex if and only if

$$f(\boldsymbol{x}_1) \ge f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T (\boldsymbol{x}_1 - \boldsymbol{x}) \tag{1.59}$$

for any $\boldsymbol{x}, \boldsymbol{x}_1 \in S$. Property (1.59) is an important one as it actually characterizes convex functions and has a geometrical interpretation – all tangents of $f(\boldsymbol{x})$ lie below the function graph $(\boldsymbol{x}, f(\boldsymbol{x}))$ as illustrated in Fig. 1.6.

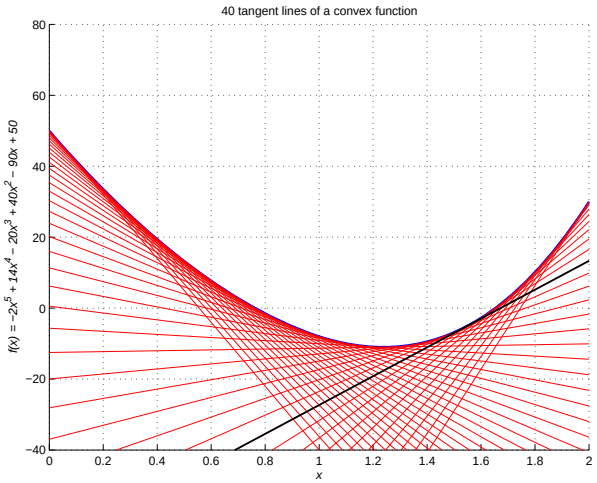

Figure 1.6 All supporting tangents of a convex function lie underneath it.

- A function $f(\boldsymbol{x}) \in C^2$ (i.e. twice continuously differentiable) is convex over a convex set S (which contains more than one point) if and only if $\nabla^2 f(\boldsymbol{x}) \succeq \mathbf{0}$ for $\boldsymbol{x} \in S$.

• If function $f(\boldsymbol{x}) \in C^1$ is convex over a convex set S and $\boldsymbol{x}^*$ is a point in S that satisfies $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$, then $\boldsymbol{x}^*$ is a global minimizer in S. This is because for any $\boldsymbol{x} \in S$ we have

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*) + \nabla f(\boldsymbol{x}^*)^T(\boldsymbol{x} - \boldsymbol{x}^*) = f(\boldsymbol{x}^*)$$

◊ In other words, for smooth convex functions, the first-order necessary condition $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ is sufficient for $\boldsymbol{x}^*$ to be a minimizer.

E. ***A General Structure of Unconstrained Optimization Algorithms***

Step 1: Choose an initial point $\boldsymbol{x}_0$ and a convergence tolerance ε. Set a counter k = 0 for the number of iterations.

Step 2: Determine a search direction $\boldsymbol{d}_k$ for reducing $f(\boldsymbol{x})$ from $f(\boldsymbol{x}_k)$.

Step 3: Determine a step size $\alpha_k \geq 0$ such that $f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k)$ is minimized, and construct

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k .$$

Step 4: If $\|\alpha_k \boldsymbol{d}_k\| < \varepsilon$, stop and output a solution $\boldsymbol{x}_{k+1}$, otherwise set $k := k + 1$ and repeat from Step 2.

F. ***A Backtracking Algorithm for Line Search (BLS)*** [6]

Assume $\boldsymbol{d}_k$ is a descent direction of $f(\boldsymbol{x})$ at $\boldsymbol{x}_k$.

Step 1: Select constants $\rho \in (0, 0.5)$ and $\gamma \in (0, 1)$ (e.g. $\rho = 0.1,\ \gamma = 0.5$). Set $\alpha = 1$.

Step 2: While

$$f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k) > f(\boldsymbol{x}_k) + \rho \alpha \boldsymbol{g}_k^T \boldsymbol{d}_k \qquad (1.60)$$

Set $\alpha = \gamma \alpha$.

Step 3: Output $\alpha_k = \alpha$.

◊ How BLS works: the right-hand side of (1.60) is a line in α passing through point (0, $f(\boldsymbol{x}_k)$) with a negative slop because $\boldsymbol{d}_k$ is a descent direction:

$$\rho \boldsymbol{g}_k^T \boldsymbol{d}_k = \rho \left. \frac{df(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k)}{d\alpha} \right|_{\alpha = 0} < 0$$

See Fig. 1.7 for an illustration. Note from the figure that a value of α less than α_0 may be

considered acceptable because such an α satisfies

$$f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k) < f(\boldsymbol{x}_k) + \rho \alpha \boldsymbol{g}_k^T \boldsymbol{d}_k < f(\boldsymbol{x}_k)$$

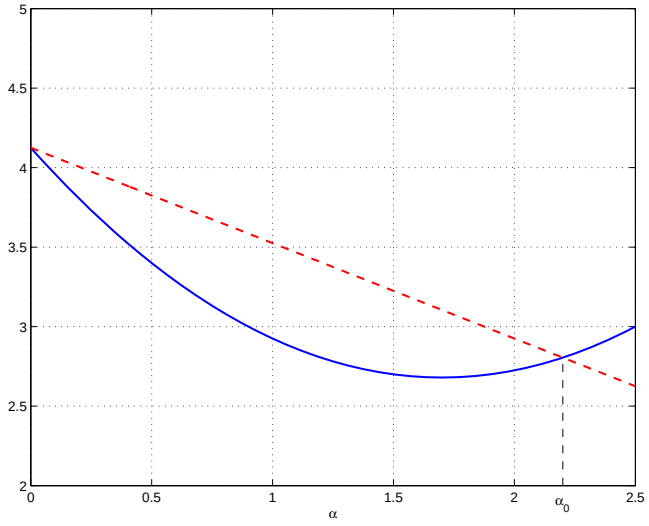

Figure 1.7 Both $f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k)$ (blue curve) and $f(\boldsymbol{x}_k) + \rho \alpha \boldsymbol{g}_k^T \boldsymbol{d}_k$ (red dashed line) are shown as functions of α. An acceptable α is found by gradually reducing α to a value less than α_0 (which is signified by $f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k) < f(\boldsymbol{x}_k) + \rho \alpha \boldsymbol{g}_k^T \boldsymbol{d}_k$).

• Two MATLAB functions, namely `bt_lsearch.m` and `inex_lsearch.m`, for line search are available from the link below:

`http://www.ece.uvic.ca/~wslu/Talk.html`

G. ***The Steepest Descent Method (SDM)***

• The SDM was originated by A. Cauchy (1848). It remains to be one of the most popular minimization techniques because its simplicity although the algorithm is usually slower relative to another popular algorithm –Newton's algorithm.

• The SMD follows the general algorithmic structure described above, where the search direction is computed as

$$\boldsymbol{d}_k = -\nabla f(\boldsymbol{x}_k) \qquad (1.61)$$

This choice of $\boldsymbol{d}_k$ may be understood by the Taylor expansion of $f(\boldsymbol{x})$ at $\boldsymbol{x}_k$ up to its 1st-order:

$$f(\boldsymbol{x}_k + \boldsymbol{d}) \approx f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d} \quad \text{(assuming } \|\boldsymbol{d}\| \text{ is small)}$$

From which we see the value of $f(\boldsymbol{x})$ is reduced at highest rate at $\boldsymbol{x}_k + \boldsymbol{d}_k$ if $\boldsymbol{d}_k = -\nabla f(\boldsymbol{x}_k)$ is taken.

H. *Why Is SDM Slow?*

The slow convergence of the SDM has to do with the "condition" of the Hessian at $\boldsymbol{x}_k$: the progress made by the kth SDM iteration relative to the preceding progress can be estimated as

$$f(x_{k+1}) - f(x^*) \le \left(\frac{1-r}{1+r}\right)^2 \left(f(x_k) - f(x^*)\right) \tag{1.62}$$

where r is the ratio of the smallest over largest eigenvalues of the Hessian at $\boldsymbol{x}_k$ (which is denoted by $\boldsymbol{H}_k$). From (1.62) the SDM is expected to be slow when ratio r is small, or stated in another way, when the *condition number* of $\boldsymbol{H}_k$ is large.

◊ The condition number of a square nonsingular matrix is the ratio of its largest over smallest eigenvalues. A square nonsingular matrix is said to be *ill-conditioned* if its condition number is very large.

I. *Scaling and Normalization*

The eigenvalues of Hessian of an objective function and, in turn, the performance of the steepest-descent method tend to depend to a large extent on the choice of variables. For example, in one and the same two-dimensional problem, whether the contours look nearly circular or extremely elliptical has to do with how the units of design components $\{x_i,\ i = 1,\ldots,\ n\}$ are chosen. Consequently, the rate of convergence can often be improved by scaling the variables properly through variable transformation. A possible approach to scaling is to let

$$\boldsymbol{x} = \boldsymbol{T}\boldsymbol{y}$$

where $\boldsymbol{T}$ is an $n \times n$ diagonal matrix, and then solve the problem

$$\underset{y}{\text{minimize}} \quad h(\boldsymbol{y}) = f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{T}\boldsymbol{y}}$$

The gradient and Hessian of the new problem are calculated as

$$\boldsymbol{g}_h = \boldsymbol{T}\boldsymbol{g}_x \quad \text{and} \quad \boldsymbol{H}_h = \boldsymbol{T}^T\boldsymbol{H}\boldsymbol{T} \tag{1.63}$$

respectively. We note that both the steepest-descent direction and the eigenvalues associated with the transformed problem are altered, hence the performance of an SDM is expected to be improved by adequate scaling.

The choice of $\boldsymbol{T}$ tends to depend heavily on the problem at hand and, as a result, no general rules can be stated. For many problems in machine learning, however, the variables are always related to real-world quantities (e.g. number of rooms in a typical house, age of a house, total living area of a house in feet^2, etc.) and are known to vary in certain ranges. In cases like these, it is often beneficial to normalize each variable x, assuming to be in the range $[a, b]$, to a standard range say $[-0.5, 0.5]$, by linear transform

$$\overline{x} = \frac{x-\mu}{b-a} \quad \text{with} \quad \mu = \frac{a+b}{2} \tag{1.64}$$

J. *Newton's Method*

Newton's method is well known for its fast convergence at a cost of increased complexity.

- Newton's method also follows the general algorithmic structure as outlined in Sec. 1.2.1.E,

where the search direction is computed as

$$\boldsymbol{d}_k = -\left(\nabla^2 f(\boldsymbol{x}_k)\right)^{-1} \nabla f(\boldsymbol{x}_k), \quad \text{i.e.,} \quad \boldsymbol{d}_k = -\boldsymbol{H}_k^{-1}\boldsymbol{g}_k \tag{1.65}$$

◊ This choice of $\boldsymbol{d}_k$ may be understood as the minimizer of the 2nd-order approximate of $f(\boldsymbol{x})$ in a small vicinity of $\boldsymbol{x}_k$:

$$f(\boldsymbol{x}_k + \boldsymbol{d}) \approx f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d} + \tfrac{1}{2}\boldsymbol{d}^T \nabla^2 f(\boldsymbol{x}_k)\boldsymbol{d}$$

Figure 1.8 illustrates the first two Newton steps for the convex function

$$f(x) = x^2 + 0.05e^{-x} + 8$$

With $x_0 = -5$, the first two Newton iterations yield $x_1 = -3.15$ and $x_2 = -0.7929$ while the global minimizer is at $x^* = 0.0242$.

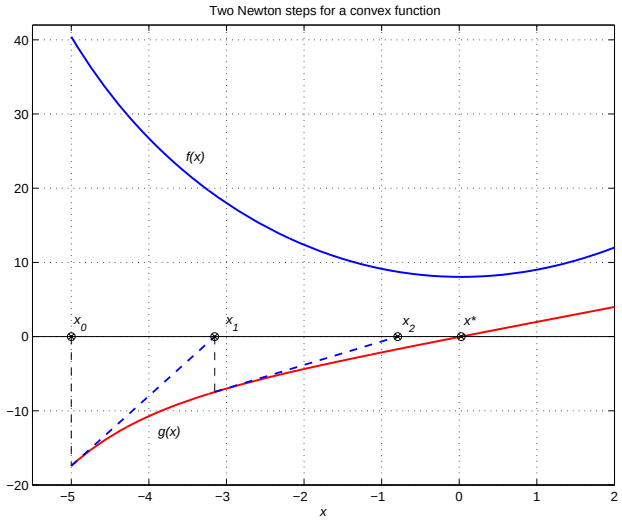

Figure 1.8 Two Newton steps for a convex function.

• It is important to note that Newton's direction given by (1.65) is effective only when $\boldsymbol{H}_k$ is *positive definite*. In case $\boldsymbol{H}_k$ is not positive definite, the search direction in Newton's method is modified to

$$\boldsymbol{d}_k = -\hat{\boldsymbol{H}}_k^{-1}\boldsymbol{g}_k$$

where $\hat{\boldsymbol{H}}_k = \dfrac{\boldsymbol{H}_k + \beta \boldsymbol{I}}{1+\beta}$ with a sufficiently large β so that $\boldsymbol{H}_k + \beta \boldsymbol{I} \succ \boldsymbol{0}$.

K. ***Quasi-Newton Methods***

One may take a unifying look at the SDM and Newton method by writing their search direction as

$$\boldsymbol{d}_k = -\boldsymbol{S}_k \boldsymbol{g}_k \tag{1.66}$$

where

$$\boldsymbol{S}_k = \begin{cases} \boldsymbol{I} & \text{for steepest descent} \\ \boldsymbol{H}_k^{-1} & \text{for Newton} \end{cases}$$

Newton's method is known for its fast convergence and solution accuracy. However, this fast convergence is achieved at the cost of increased computational complexity which has largely to do with the computation of the inverse Hessian in every Newton iteration. Quasi-Newton algorithms are developed to provide convergent rates comparable with that of Newton algorithm with reduced complexity. Actually quasi-Newton algorithms are based only on gradient information and do not require explicit evaluation of the Hessian and its inverse.

Quasi-Newton algorithms follow the general algorithmic structure described earlier, where search direction $\boldsymbol{d}_k$ is evaluated using (1.66) where matrix $\boldsymbol{S}_k$ is updated in each iteration using gradient information. Two most well-known quasi-Newton algorithms are those developed by Davidon, Fletcher, and Powell (DFP), and by Broyden, Fletcher, Goldfarb, and Shanno (BFGS), respectively. The BFGS and DFP formulas for updating matrix $\boldsymbol{S}_k$ are given below.

◊ Quantities required: $\boldsymbol{\delta}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k, \quad \boldsymbol{\gamma}_k = \boldsymbol{g}_{k+1} - \boldsymbol{g}_k$

◊ Davidon-Fletcher-Powell (DFP) updating formula: $\boldsymbol{S}_0 = \boldsymbol{I}$, and

$$\boldsymbol{S}_{k+1} = \boldsymbol{S}_k + \frac{\boldsymbol{\delta}_k \boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k} - \frac{\boldsymbol{S}_k \boldsymbol{\gamma}_k \boldsymbol{\gamma}_k^T \boldsymbol{S}_k}{\boldsymbol{\gamma}_k^T \boldsymbol{S}_k \boldsymbol{\gamma}_k} \tag{1.67}$$

◊ Broyden-Fletcher-Goldfarb-Shanno (BFGS) updating formula: $\boldsymbol{S}_0 = \boldsymbol{I}$, and

$$\boldsymbol{S}_{k+1} = \boldsymbol{S}_k + \left(1 + \frac{\boldsymbol{\gamma}_k^T \boldsymbol{S}_k \boldsymbol{\gamma}_k}{\boldsymbol{\gamma}_k^T \boldsymbol{\delta}_k}\right) \frac{\boldsymbol{\delta}_k \boldsymbol{\delta}_k^T}{\boldsymbol{\gamma}_k^T \boldsymbol{\delta}_k} - \frac{\boldsymbol{\delta}_k \boldsymbol{\gamma}_k^T \boldsymbol{S}_k + \boldsymbol{S}_k \boldsymbol{\gamma}_k \boldsymbol{\delta}_k^T}{\boldsymbol{\gamma}_k^T \boldsymbol{\delta}_k} \tag{1.68}$$

1.2.2 Constrained Optimization

A general constrained optimization problem assumes the form

$$\begin{aligned} &\text{minimize} \quad f(\boldsymbol{x}) \\ &\text{subject to}: \; a_i(\boldsymbol{x}) = 0 \quad \text{for } i = 1, 2, ..., p \\ &\qquad\qquad c_j(\boldsymbol{x}) \le 0 \quad \text{for } j = 1, 2, ..., q \end{aligned} \tag{1.69a-c}$$

◊ Feasible region: $\{\boldsymbol{x} : a_i(\boldsymbol{x}) = 0 \text{ for } i = 1, 2, ..., p, \text{ and } c_j(\boldsymbol{x}) \le 0 \text{ for } j = 1, 2, ..., q\}$

◊ A point $\boldsymbol{x}$ is said to be *feasible* if it is in the feasible region (inside or on the boundary).

◊ An inequality constraint $c_j(\boldsymbol{x}) \le 0$ is said to be *active* at a feasible point $\boldsymbol{x}$, if $c_j(\boldsymbol{x}) = 0$. An inequality constraint $c_j(\boldsymbol{x}) \le 0$ is said to be *inactive* at a feasible point $\boldsymbol{x}$ if $c_j(\boldsymbol{x}) < 0$.

A. *Karush-Kuhn-Tucker (KKT) Conditions*

The Karush-Kuhn-Tucker (KKT) conditions are a set of *first-order necessary conditions* for point $\boldsymbol{x}^*$ to be a local minimizer of problem (1.69). These conditions are fundamentally important in the theory and practice of constrained optimization. The KKT conditions can be stated as follows.

KKT Conditions

If $\boldsymbol{x}^*$ is a local minimizer of the constrained problem (1.69), then the following conditions, known as *KKT conditions*, are satisfied:

(a) $$a_i(\boldsymbol{x}^*) = 0 \quad \text{for } i = 1, 2, ..., p \tag{1.70a}$$

(b) $$c_j(\boldsymbol{x}^*) \le 0 \text{ for } j = 1, 2, ..., q \tag{1.70b}$$

(c) there exist Lagrange multipliers λ_i^* for $1 \le i \le p$ and μ_j^* for $1 \le j \le q$ such that

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{p} \lambda_i^* \nabla a_i(\boldsymbol{x}^*) + \sum_{j=1}^{q} \mu_j^* \nabla c_j(\boldsymbol{x}^*) = \mathbf{0} \tag{1.70c}$$

(d) $$\mu_j^* c_j(\boldsymbol{x}^*) = 0 \text{ for } j = 1, 2, ..., q \tag{1.70d}$$

(e) $$\mu_j^* \ge 0 \text{ for } j = 1, 2, ..., q \tag{1.70e}$$

◊ Conditions (1.70a) and (1.70b) are merely the constraints imposed in the problem, see (1.69b) and (1.69c).

◊ Condition (1.70c) states that at a local minimizer the gradient of the objective function is a linear combination of the gradients of the constraint functions, and the coefficients in the combination are (up to a sign) Lagrange multipliers .

◊ If we define the *Lagrangian* of problem (1.69) as

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \sum_{i=1}^{p} \lambda_i a_i(\boldsymbol{x}) + \sum_{j=1}^{q} \mu_j c_j(\boldsymbol{x}) \tag{1.71}$$

then (1.70c) can be expressed as

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0} \tag{1.72}$$

◊ Conditions in (1.70d) are called *complementarity conditions*. It follows that if the *k*th inequality constraint is inactive at $\boldsymbol{x}^*$, then $\mu_k^* = 0$. In other words, the equation in (1.70c) only includes those terms $\nabla c_j(\boldsymbol{x}^*)$ with $c_j(\boldsymbol{x})$ active at $\boldsymbol{x}^*$.

◊ The total number of equations in the KKT conditions, namely *p* (from (1.70a)) + *n* (from (1.70c)) + *q* (from (1.70d)) , are equal to the total number of components in unknown vectors $\{\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*\}$.

B. ***Convex Programming Problems***

Definition 1.17 *Convex programming*

A constrained minimization problem (1.69) is said to be a *convex programming* (CP) problem if the objective function is convex and the feasible region defined by the constraints, i.e., $\{\boldsymbol{x}: a_i(\boldsymbol{x}) = 0$ for $i = 1, 2, \ldots, p$, and $c_j(\boldsymbol{x}) \le 0$ for $j = 1, 2, \ldots, q\}$ is a convex set. In words, a CP problem is a problem of minimizing a convex function over a convex region. ■

- It can be shown that the feasible region $\{\boldsymbol{x}: a_i(\boldsymbol{x}) = 0$ for $i = 1, 2, \ldots, p$, and $c_j(\boldsymbol{x}) \le 0$ for $j = 1,$

2, …,q} is convex if (i) the equality constraint functions $a_i(\boldsymbol{x})$ are *affine*, and (ii) the inequality constraint functions $c_j(\boldsymbol{x})$ are *convex*.

Hence a general CP problem assumes the form

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to}: && \boldsymbol{a}_i^T \boldsymbol{x} = b_i \quad \text{for } i = 1, 2, ..., p \\ & && c_j(\boldsymbol{x}) \le 0 \quad \text{for } j = 1, 2, ..., q \end{aligned} \tag{1.73a-c}$$

where $f(\boldsymbol{x})$ and $c_j(\boldsymbol{x})$ for j = 1, 2, …, q are convex.

- An important property of CP problems is that the KKT conditions become *both necessary and sufficient*. Therefore, for CP problems $\{\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*\}$ satisfying the KKT conditions gives a local solution (minimizer) $\boldsymbol{x}^*$.
- Another important property of CP problems is that any local minimizer of a CP problem is a global minimizer. By combining the above two properties, one concludes that $\boldsymbol{x}^*$ is a global minimizer of CP problem (1.73) if and only if the KKT conditions are satisfied.

C. ***Important Classes of CP Problems***

- ***Linear programming*** (*LP*)

Linear programming problems are of great importance both theoretically and practically. Many problems in science and engineering can be accurately or approximately modeled as LP problems.

The *standard-form* LP problem is given by

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) = \boldsymbol{c}^T \boldsymbol{x} \\ &\text{subject to}: && \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & && \boldsymbol{x} \ge \boldsymbol{0} \end{aligned} \tag{1.74a-c}$$

Another class of popular LP problem know as *alternative-form* LP problems, assumes the form

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) = \boldsymbol{c}^T \boldsymbol{x} \\ &\text{subject to}: && \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b} \end{aligned} \tag{1.75a-b}$$

- ***Quadratic programming*** (*QP*)

$$\begin{aligned} &\text{minimize} && \tfrac{1}{2} \boldsymbol{x}^T \boldsymbol{H}\boldsymbol{x} + \boldsymbol{p}^T \boldsymbol{x} \\ &\text{subject to:} && \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b} \end{aligned} \tag{1.76a-b}$$

where $\boldsymbol{H}$ is symmetric and positive semidefinite ($\boldsymbol{H} \succeq \boldsymbol{0}$).

- ***QP with quadratic constraints***

$$\begin{aligned} &\text{minimize} && \tfrac{1}{2} \boldsymbol{x}^T \boldsymbol{H}\boldsymbol{x} + \boldsymbol{p}^T \boldsymbol{x} \\ &\text{subject to:} && \boldsymbol{x}^T \boldsymbol{Q}_i \boldsymbol{x} + \boldsymbol{q}_i^T \boldsymbol{x} + r_i \le 0 \quad \text{for } 1 \le i \le q \end{aligned} \tag{1.77a-b}$$

where $\boldsymbol{H}$ and $\boldsymbol{Q}_i$ are symmetric and positive semidefinite.

- ***Semidefinite programming*** (*SDP*)

◊ **Primal SDP**

$$\begin{aligned} &\text{minimize} && \boldsymbol{C}\cdot\boldsymbol{X} \\ &\text{subject to:} && \boldsymbol{A}_i\cdot\boldsymbol{X}=b_i \quad \text{for } 1\le i\le p \\ & && \boldsymbol{X}\succeq\boldsymbol{0} \end{aligned} \tag{1.78a-c}$$

where variable $\boldsymbol{X}$ and data $\boldsymbol{A}_i$ and $\boldsymbol{C}$ are symmetric matrices and $\boldsymbol{C}\cdot\boldsymbol{X}$ denotes the inner product in the space of real symmetric matrices defined by

$$\boldsymbol{C}\cdot\boldsymbol{X}=\sum_{i=1}^{n}\sum_{j=1}^{n}c_{i,j}x_{i,j}$$

◊ **Dual SDP**

$$\begin{aligned} &\text{minimize} && \boldsymbol{c}^T\boldsymbol{x} \\ &\text{subject to:} && \boldsymbol{F}(\boldsymbol{x})\succeq\boldsymbol{0} \end{aligned} \tag{1.79a-b}$$

where $\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{F}_0+\sum_{i=1}^{p}x_i\boldsymbol{F}_i$.

- ***Second-order cone programming*** (*SOCP*)

$$\begin{aligned} &\text{minimize} && \boldsymbol{b}^T\boldsymbol{x} \\ &\text{subject to:} && \left\|\boldsymbol{A}_j^T\boldsymbol{x}+\boldsymbol{c}_j\right\|\le\boldsymbol{b}_j^T\boldsymbol{x}+d_j \quad \text{for } 1\le j\le q \end{aligned} \tag{1.80a-b}$$

D. ***Lagrange Dual*** [6]

Duality is an important concept associated with constrained convex problems. To study the Lagrange dual, we need a concept called *Lagrange dual function*. Consider the general CP problem

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to:} && \boldsymbol{a}_i^T\boldsymbol{x}=b_i \quad \text{for } 1\le i\le p \\ & && c_j(\boldsymbol{x})\le 0 \quad \text{for } 1\le j\le q \end{aligned} \tag{1.81a-c}$$

where $f(\boldsymbol{x})$ and $c_j(\boldsymbol{x})$ are convex and recall its Lagrangian

$$L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu})=f(\boldsymbol{x})+\sum_{i=1}^{p}\lambda_i\left(\boldsymbol{a}_i^T\boldsymbol{x}-b_i\right)+\sum_{j=1}^{q}\mu_i c_j(\boldsymbol{x})$$

Definition 1.18 *Lagrange dual function*

The *Lagrange dual function* of problem (1.81) is defined as

$$q(\boldsymbol{\lambda},\boldsymbol{\mu})=\inf_{\boldsymbol{x}}L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu}) \tag{1.82}$$

for $\boldsymbol{\lambda}\in R^p$ and $\boldsymbol{\mu}\in R^q$ with $\boldsymbol{\mu}\ge\boldsymbol{0}$. Note that the Lagrangian is *convex* with respect to $\boldsymbol{x}$. If

$L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu})$ is unbounded from below for some $\{\boldsymbol{\lambda},\ \boldsymbol{\mu}\}$, then the value of $q(\boldsymbol{\lambda},\boldsymbol{\mu})$ is assigned to $-\infty$.

Property 1 $q(\boldsymbol{\lambda},\boldsymbol{\mu})$ is a concave function with respect to $\{\boldsymbol{\lambda},\ \boldsymbol{\mu}\}$. The property follows from that fact that for $\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2 \in R^p$ and $\boldsymbol{\mu}_1,\boldsymbol{\mu}_2 \in R^q$ with $\boldsymbol{\mu}_1,\boldsymbol{\mu}_2 \geq \mathbf{0}$ and for $t \in (0,1)$, we have

$$
\begin{aligned}
&q(t\boldsymbol{\lambda}_1+(1-t)\boldsymbol{\lambda}_2, t\boldsymbol{\mu}_1+(1-t)\boldsymbol{\mu}_2) = \inf_{\boldsymbol{x}} L(\boldsymbol{x}, t\boldsymbol{\lambda}_1+(1-t)\boldsymbol{\lambda}_1, t\boldsymbol{\mu}_1+(1-t)\boldsymbol{\mu}_2) \\
&= \inf_{\boldsymbol{x}}\left[(t+1-t)f(\boldsymbol{x}) + \sum_{i=1}^{p}(t\lambda_{1,i}+(1-t)\lambda_{2,i})(\boldsymbol{a}_i^T\boldsymbol{x}-b_i) + \sum_{j=1}^{q}(t\mu_{1,j}+(1-t)\mu_{2,j})c_j(\boldsymbol{x})\right] \\
&\geq t\cdot\inf_{\boldsymbol{x}}\left[f(\boldsymbol{x}) + \sum_{i=1}^{p}\lambda_{1,i}(\boldsymbol{a}_i^T\boldsymbol{x}-b_i) + \sum_{j=1}^{q}\mu_{1,j}c_j(\boldsymbol{x})\right] + (1-t)\cdot\inf_{\boldsymbol{x}}\left[f(\boldsymbol{x}) + \sum_{i=1}^{p}\lambda_{2,i}(\boldsymbol{a}_i^T\boldsymbol{x}-b_i) + \sum_{j=1}^{q}\mu_{2,j}c_j(\boldsymbol{x})\right] \\
&= t\cdot q(\boldsymbol{\lambda}_1,\boldsymbol{\mu}_1) + (1-t)\cdot q(\boldsymbol{\lambda}_2,\boldsymbol{\mu}_2)
\end{aligned}
$$

Definition 1.19 *Lagrange dual problem*

The *Lagrange dual problem* with respect to problem (1.81) is defined as

$$
\begin{aligned}
&\underset{\boldsymbol{\lambda},\boldsymbol{\mu}}{\text{maximize}} \quad q(\boldsymbol{\lambda},\boldsymbol{\mu}) \\
&\text{subject to:} \quad \boldsymbol{\mu} \geq \mathbf{0}
\end{aligned}
\tag{1.83a-b}
$$

Property 2 For any $\boldsymbol{x}$ feasible for problem (1.81) and $\{\boldsymbol{\lambda},\ \boldsymbol{\mu}\}$ feasible for problem (1.83), we have

$$f(\boldsymbol{x}) \geq q(\boldsymbol{\lambda},\boldsymbol{\mu}) \tag{1.84}$$

This is because

$$L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu}) = f(\boldsymbol{x}) + \sum_{i=1}^{p}\lambda_i\left(\boldsymbol{a}_i^T\boldsymbol{x}-b_i\right) + \sum_{j=1}^{q}\mu_i\, c_j(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{j=1}^{q}\mu_i\, c_j(\boldsymbol{x}) \leq f(\boldsymbol{x})$$

thus

$$q(\boldsymbol{\lambda},\boldsymbol{\mu}) = \inf_{\boldsymbol{x}} L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu}) \leq L(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu}) \leq f(\boldsymbol{x})$$

We call the convex minimization problem in (1.81) the *primal* problem and the concave maximization problem in (1.83) the *dual* problem. The property (1.84) leads naturally to the concept of *duality gap* between the primal and dual objectives, which us defined by

$$\delta(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu}) = f(\boldsymbol{x}) - q(\boldsymbol{\lambda},\boldsymbol{\mu}) \tag{1.85}$$

It follows that for feasible $\{\boldsymbol{x},\boldsymbol{\lambda},\ \boldsymbol{\mu}\}$ the duality gap is always nonnegative.

Property 3 Let $\boldsymbol{x}^*$ be a solution of the primal problem in (1.81). Then the dual function at any feasible $\{\boldsymbol{\lambda},\ \boldsymbol{\mu}\}$ serves as a lower bound of the optimal value of the primal objective, $f(\boldsymbol{x}^*)$, namely,

$$f(\boldsymbol{x}^*) \geq q(\boldsymbol{\lambda}, \boldsymbol{\mu}) \tag{1.86}$$

This property follows immediately from (1.84) by taking the minimum of $f(\boldsymbol{x})$ on its left-hand side.

- A question that naturally arises is what is the tightest lower bound the dual function $q(\boldsymbol{\lambda}, \boldsymbol{\mu})$ can offer? Obviously the answer is found by maximizing the dual function $q(\boldsymbol{\lambda}, \boldsymbol{\mu})$ on the right-hand side of (1.86) subject to $\boldsymbol{\mu} \geq \mathbf{0}$, which is exactly the Lagrange dual problem as formulated in (1.83). Therefore, if we denote the solution of (1.83) by $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$, then we have

$$f(\boldsymbol{x}^*) \geq q(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \tag{1.87}$$

Based on (1.79), we now introduce the concept of *strong* and *weak* duality as follows.

Definition 1.20 *Strong and weak duality*

Let $\boldsymbol{x}^*$ and $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ be solutions of primal problem (1.81) and dual problem (1.83), respectively. We say strong duality holds if $f(\boldsymbol{x}^*) = q(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$, i.e., the optimal duality gap is zero; and a weak duality holds if $f(\boldsymbol{x}^*) > q(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$. ■

- It can be shown that if the primal problem is strictly feasible, i.e., there exists $\boldsymbol{x}$ satisfying

$$\begin{aligned} \boldsymbol{a}_i^T \boldsymbol{x} &= b_i \quad \text{for } 1 \leq i \leq p \\ c_j(\boldsymbol{x}) &< 0 \quad \text{for } 1 \leq j \leq q \end{aligned}$$

(this is to say that the interior of the feasible region of problem (1.81) is nonempty), then strong duality holds, i.e., the optimal duality gap is zero.

Example 1.13 Find the Lagrange dual of the LP problem

$$\begin{aligned} &\text{minimize} \quad \boldsymbol{c}^T \boldsymbol{x} \\ &\text{subject to:} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \ \boldsymbol{x} \geq \mathbf{0} \end{aligned}$$

Solution We write $\boldsymbol{x} \geq \mathbf{0}$ as $-\boldsymbol{x} \leq \mathbf{0}$ hence the Lagrangian of the LP problem is given by

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{c}^T \boldsymbol{x} + (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T \boldsymbol{\lambda} - \boldsymbol{x}^T \boldsymbol{\mu}$$

thus

$$q(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \inf_{\boldsymbol{x}} \left\{ \boldsymbol{c}^T \boldsymbol{x} + (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T \boldsymbol{\lambda} - \boldsymbol{x}^T \boldsymbol{\mu} \right\} = \inf_{\boldsymbol{x}} \left\{ (\boldsymbol{c} + \boldsymbol{A}^T \boldsymbol{\lambda} - \boldsymbol{\mu})^T \boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{\lambda} \right\} \tag{1.88}$$

For given $\{\boldsymbol{\lambda}, \boldsymbol{\mu}\}$ such that $\boldsymbol{c} + \boldsymbol{A}^T \boldsymbol{\lambda} - \boldsymbol{\mu} \neq \mathbf{0}$, by (1.88) we have $q(\boldsymbol{\lambda}, \boldsymbol{\mu}) = -\infty$. Therefore to deal with a well-defined dual function $q(\boldsymbol{\lambda}, \boldsymbol{\mu})$ we assume $\boldsymbol{c} + \boldsymbol{A}^T \boldsymbol{\lambda} - \boldsymbol{\mu} = \mathbf{0}$ which leads to

$$q(\boldsymbol{\lambda},\boldsymbol{\mu}) = \inf_{\boldsymbol{x}}(-\boldsymbol{b}^T\boldsymbol{\lambda}) = -\boldsymbol{b}^T\boldsymbol{\lambda}$$

and the Lagrange dual of the LP problem in question is given by

$$\begin{aligned} \underset{\boldsymbol{\lambda},\boldsymbol{\mu}}{\text{maximize}} \quad & -\boldsymbol{b}^T\boldsymbol{\lambda} \\ \text{subject to:} \quad & \boldsymbol{\mu} \geq \boldsymbol{0} \end{aligned}$$

Since $\boldsymbol{c} + \boldsymbol{A}^T\boldsymbol{\lambda} - \boldsymbol{\mu} = \boldsymbol{0}$, $\boldsymbol{\mu} = \boldsymbol{c} + \boldsymbol{A}^T\boldsymbol{\lambda}$ so the above problem becomes

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{maximize}} \quad & -\boldsymbol{b}^T\boldsymbol{\lambda} \\ \text{subject to:} \quad & -\boldsymbol{c} - \boldsymbol{A}^T\boldsymbol{\lambda} \leq \boldsymbol{0} \end{aligned}$$

which is equivalent to

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad & \boldsymbol{b}^T\boldsymbol{\lambda} \\ \text{subject to:} \quad & (-\boldsymbol{A}^T)\boldsymbol{\lambda} \leq \boldsymbol{c} \end{aligned}$$

Example 1.14 Find the Lagrange dual of the QP problem

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{p}^T\boldsymbol{x} \\ \text{subject to:} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \end{aligned} \tag{1.89a-b}$$

where $\boldsymbol{H}$ is positive definite.

Solution The Lagrangian of the QP problem is given by

$$L(\boldsymbol{x},\boldsymbol{\mu}) = \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{p}^T\boldsymbol{x} + \boldsymbol{\mu}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$$

Hence

$$q(\boldsymbol{\mu}) = \inf_{\boldsymbol{x}}\left\{\tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{p}^T\boldsymbol{x} + \boldsymbol{\mu}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\right\} \tag{1.90}$$

where the infimum (defined as the largest lower bound of the objective function) is attained at $\boldsymbol{x} = -\boldsymbol{H}^{-1}(\boldsymbol{p} + \boldsymbol{A}^T\boldsymbol{\mu})$. By substituting this solution into (1.90), we obtain

$$q(\boldsymbol{\mu}) = -\tfrac{1}{2}\boldsymbol{\mu}^T\boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{A}^T\boldsymbol{\mu} - \boldsymbol{\mu}^T(\boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{p} + \boldsymbol{b}) - \tfrac{1}{2}\boldsymbol{p}^T\boldsymbol{H}^{-1}\boldsymbol{p} \tag{1.91}$$

If we let $\boldsymbol{P} = \boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{A}^T$, $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{p} + \boldsymbol{b}$ and neglect the constant term in (1.91), the dual function becomes $q(\boldsymbol{\mu}) = -\tfrac{1}{2}\boldsymbol{\mu}^T\boldsymbol{P}\boldsymbol{\mu} - \boldsymbol{\mu}^T\boldsymbol{t}$, hence the Lagrange dual of (1.89) is given by

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\boldsymbol{\mu}^T\boldsymbol{P}\boldsymbol{\mu} + \boldsymbol{\mu}^T\boldsymbol{t} \\ \text{subject to:} \quad & \boldsymbol{\mu} \geq \boldsymbol{0} \end{aligned} \tag{1.92}$$

Note that by definition matrix $\boldsymbol{P}$ in (1.92) is positive definite, therefore the dual problem is also a convex QP problem, but with simpler constraints in comparison with the primal problem in (1.89). In addition, if the number of constraints involved in the primal problem is smaller than *n*, so is the size of the dual problem. ■

E. *CVX*

• *The material of this section is largely based on the User Guide version 1.22 of the software written by M. Grant and S. Boyd.*

- **What is CVX?**

Designed by Michael Grant and Stephen Boyd, with input from Yinyu Ye, `CVX` is a modeling system for disciplined convex programming that are convex optimization problems described by a limited set of construction rules. `CVX` solves standard problems such as LP, QP, SOCP, and SDP problems. Compared to directly using a solver such as SeDuMi for these problems, `CVX` greatly simplifies the task of specifying the problem. `CVX` also solves more complex convex optimization problems, including many involving nonsmooth functions, such as l_1 norm.

- **Examples**

Example 1.15 We first consider the most basic convex optimization problem, least-squares. In a least-squares problem, we seek $\boldsymbol{x} \in R^n$ that minimizes $\| \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \|_2$, where $\boldsymbol{A} \in R^{m \times n}$ is skinny and full rank (i.e., $m \geq n$ and rank($\boldsymbol{A}$) = n). Let us create some test problem data for m, n, $\boldsymbol{A}$, and $\boldsymbol{b}$ in MATLAB:

```
m = 16; n = 8;
A = randn(m,n);
b = randn(m,1);
```

(We chose small values of m and n to keep the output readable.) Then the least squares solution $\boldsymbol{x} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}$ is easily computed using the backslash operator:

```
x_ls = A\b;
```

Using `cvx`, the same problem can be solved as follows:

```
cvx_begin
   variable x(n);
   minimize( norm(A*x-b) );
cvx_end
```

When MATLAB reaches command `cvx_end`, the least-squares problem is solved, and the Matlab variable $\boldsymbol{x}$ is overwritten with the solution of the least-squares problem, i.e., $(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}$. Now `x` is an ordinary length-n numerical vector, identical to what would be obtained in the traditional approach, at least to within the accuracy of the solver. In addition, two additional Matlab variables are created:

• `cvx_optval`, which contains the value of the objective function; i.e., $\| \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \|_2$;

• `cvx_status`, which contains a string describing the status of the calculation. In this case, `cvx_status` would contain the string `Solved`. See Appendix C of the user guide for a list of the possible values of `cvx_status` and their meaning.

• `cvx_slvtol`: the tolerance level achieved by the solver.

• `cvx_slvitr`: the number of iterations taken by the solver.

Example 1.16 Suppose we wish to add some simple upper and lower bounds to the

least-squares problem above. That is, we wish to solve the CP problem

$$\begin{aligned} &\text{minimize} && \| \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \|_2 \\ &\text{subject to:} && \boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u} \end{aligned}$$

where $\boldsymbol{l}$ and $\boldsymbol{u}$ are given data, vectors with the same dimension as the variable $\boldsymbol{x}$. We can no longer use the simple backslash notation to solve this problem, but it can be transformed into a QP, which can be solved without difficulty if you have some form of QP software available.

Let us provide some numeric values for $\boldsymbol{l}$ and $\boldsymbol{u}$:

```
bnds = randn(n,2);
l = min(bnds,[],2);
u = max(bnds,[],2);
```

Then if you have the MATLAB Optimization Toolbox, you can use the quadprog function to solve the problem as follows:

```
x_qp = quadprog(2*A'*A,-2*A'*b,[],[],[],[],l,u );
```

This actually minimizes the square of the norm, which is the same as minimizing the norm itself. In contrast, the cvx specification is given by

```
cvx_begin
   variable x(n);
   minimize(norm(A*x-b));
   subject to
   l <= x <= u;
cvx_end
```

Example 1.17 cvx supports constraints more complex than simple bounds as seen in Example 1.16. For example, let us define new matrices $\boldsymbol{C}$ and $\boldsymbol{d}$ in MATLAB as follows:

```
p = 4;
C = randn(p,n);
d = randn(p,1);
```

Now let us add an equality constraint and a nonlinear inequality constraint to the original least-squares problem:

```
cvx_begin
   variable x(n);
   minimize(norm(A*x-b));
   subject to
   C*x == d;
   norm(x,Inf) <= 1;
cvx_end
```

Example 1.18 For our final example in this section, let us show how traditional MATLAB code and cvx specifications can be mixed to form and solve multiple optimization problems. The following code solves the problem of minimizing $\| \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \|_2 + \gamma \| \boldsymbol{x} \|_1$, for a logarithmically spaced vector of (positive) values of γ. This gives us points on the optimal trade-off curve

between $\| \boldsymbol{Ax} - \boldsymbol{b} \|_2$ and $\| \boldsymbol{x} \|_1$. An example of this curve is given in Fig. 1.9.

```
gamma = logspace(-2,2,20);
l2norm = zeros(size(gamma));
l1norm = zeros(size(gamma));
fprintf(1, ' gamma norm(x,1) norm(A*x-b)\n');
fprintf(1, '-------------------------------------\n');
for k = 1:length(gamma),
  fprintf(1,'%8.4e', gamma(k));
  cvx_begin quiet
     variable x(n);
     minimize(norm(A*x-b)+gamma(k)*norm(x,1));
  cvx_end
  l1norm(k) = norm(x,1);
  l2norm(k) = norm(A*x-b);
  fprintf(1,' %8.4e %8.4e\n',l1norm(k),l2norm(k));
end
plot(l1norm, l2norm);
xlabel('norm (x,1)');
ylabel('norm (A*x-b)');
grid
```

- **Basic rules**

(1) All cvx models must be preceded by command `cvx_begin` and terminated with commend `cvx_end`.

Command `cvx_begin` accepts several variants. For example, `cvx_begin quiet` prevents the model from producing screen output while it is been solved.

(2) Data Types for Variables

All variables must be declared using command `variable` (or `variables`). Variables can be real or complex; and scalar, vector, matrix, or n-dimensional array. Variables can be matrices with structure,

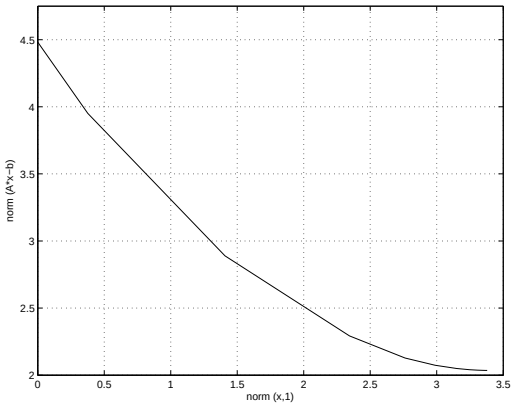

Fig. 1.9

like symmetry or bandedness. The variable structure is given by including descriptive keywords

following the name and size of the variable. Here are some examples of cvx variable:

```
variable w(50) complex;
variable X(20,10);
variable Y(50,50) symmetric;
variable Z(100,100) hermitian toeplitz;
```

- **Structure Keywords**

Version 1.22 of cvx supports the following structure keywords:

```
banded(lb,ub) complex diagonal hankel hermitian lower_bidiagonal
lower_hessenberg lower_triangular scaled_identity skew_symmetric
symmetric toeplitz tridiagonal upper_bidiagonal upper_hankel
upper_hessenberg upper_triangular
```

(3) Objective functions

Declaring an objective function requires the use of the minimize or maximize function, as appropriate. The objective function in a call to minimize must be convex; the objective function in a call to maximize must be concave. At most one objective function may be declared in a given cvx specification, and the objective function must have a scalar value.

If no objective function is specified, the problem is interpreted as a *feasibility* problem, which is the same as performing a minimization with the objective function set to zero. In this case, cvx_optval is either 0, if a feasible point is found, or +Inf, if the constraints are not feasible.

(4) Constraints

Version 1.22 of cvx supports the following constraint types:

• Equality == constraints, where both the left- and right-hand sides are affine functions of the optimization variables.

• Less-than <= inequality constraints, where the left-hand expression is convex, and the right-hand expression is concave.

• Greater-than >= constraints, where the left-hand expression is concave, and the right-hand expression is convex.

These equality and inequality operators work for arrays. When both sides of the constraint are arrays of the same size, the constraint is imposed elementwise. For example, if a and b are $m \times n$ matrices, then a<=b is interpreted by cvx as mn (scalar) inequalities, i.e., each entry of a must be less than or equal to the corresponding entry of b. cvx also handles cases where one side is a scalar and the other is an array. This is interpreted as a constraint for each element of the array, with the (same) scalar appearing on the other side. As an example, if a is an $m \times n$ matrix, then a>=0 is interpreted as mn inequalities: each element of the matrix must be nonnegative.

(5) Functions

The base cvx function library includes a variety of convex, concave, and affine functions which accept cvx variables or expressions as arguments. Many are common MATLAB functions such as sum, trace, diag, sqrt, max, and min, re-implemented as needed to support cvx; others are new functions not found in MATLAB. A complete list of the functions in the base library can be found in §B of the user guide. It's also possible to add your own new functions; see §5 of the user guide.

(6) Sets

◊ `cvx` supports the definition and use of convex sets. The base library includes the cone of positive semidefinite $n \times n$ matrices, the second-order or Lorentz cone, and various norm balls. A complete list of sets supplied in the base library is given in §B of the user guide.

◊ Since MATLAB does not have a set membership operator, `cvx` adopts a slightly different syntax to require that an expression is in a set. To represent a set, a function is used to return an unnamed variable that is required to be in the set. Consider, for example, $\boldsymbol{S}_+^n$, the cone of symmetric positive semidefinite $n \times n$ matrices. In `cvx`, this is represented by function `semidefinite(n)`, which returns an unnamed new variable, that is constrained to be positive semidefinite. To require that the matrix expression `X` be symmetric positive semidefinite, the syntax `X == semidefinite(n)` is used. The literal meaning of this is that `X` is constrained to be equal to some unnamed variable, which is required to be an $n \times n$ symmetric positive semidefinite matrix. This is equivalent to saying that `X` must be symmetric positive semidefinite.

As an example, consider the constraint that a (matrix) variable `X` is a correlation matrix, i.e., it is symmetric, has unit diagonal elements, and is positive semidefinite.

◊ In `cvx` we can declare such a variable and impose such constraints using

```
variable X(n,n) symmetric;
X == semidefinite(n);
diag(X) == ones(n,1);
```

The second line here imposes the constraint that `X` be positive semidefinite (you can read `'=='` here as `'is'`, so the second line can be read as `'X is positive semidefinite'`). The left-hand side of the third line is a vector containing the diagonal elements of `X`, whose elements we require to be equal to one. Incidentally, `cvx` allows us to simplify the third line to

```
diag(X) == 1;
```

because `cvx` follows the MATLAB convention of handling array/scalar comparisons by comparing each element of the array independently with the scalar.

◊ Sets can be combined in affine expressions, and we can constrain an affine expression to be in a convex set. For example, we can impose constraints of the form

```
A*X*A'-X == B*semidefinite(n)*B';
```

where `X` is an $n \times n$ symmetric variable matrix, and $\boldsymbol{A}$ and $\boldsymbol{B}$ are $n \times n$ constant matrices. This constraint requires that $\boldsymbol{AXA}^T - \boldsymbol{X} = \boldsymbol{BYB}^T$, for some $Y \in \boldsymbol{S}_+^n$.

Example 1.19 Standard-form LP

```
function x = lps_cvx(c,A,b)
n = length(c);
cvx_begin quiet
  variable x(n);
  minimize(c'*x);
  subject to
  A*x == b;
```

```
  x >= 0;
cvx_end
```

Example 1.20 Alternative-form LP

```
function x = lpa_cvx(c,A,b)
n = length(c);
cvx_begin quiet
  variable x(n);
  minimize(c'*x);
  subject to
  A*x >= b;
cvx_end
```

Example 1.21 Convex QP

```
function x = qp_cvx(H,p,A,b,E,d)
n = length(p);
cvx_begin quiet
  variable x(n);
  minimize(0.5*x'*H*x+x'*p);
  subject to
  A*x == b;
  E*x >= d;
cvx_end
```

Example 1.22 SDP

```
% Program: sdp_cvx.m
% To solve semidefinite programming (SDP) problem
%           minimize   c'*x
%           subject to: F0 + x(1)*F1 + ... + x(p)*Fp >= 0
% using cvx, where ">= 0" means being positive semidefinite,
% and x = [x1 x2 ... xp]'.
% Input:
% c: constant vector of length p defining objective function.
% F = [F0 F1 ... Fp] with each Fi a symmetric square matrix.
% Output:
% x: solution of the SDP problem.
% Written by W.-S. Lu, University of Victoria.
% Last modified: Oct. 29, 2012.
% Example: Solve the SDP problem involved in Example 14.1 from PO, where
% c = [0 0 0 -1]';
% F = [F0 F1 F2 F3 F4]
%  = [-2.0  0.5  0.6  0  -1  0  0 0 -1 0  0  0 -1  0  0;
%    0.5 -2.0 -0.4 -1  0  0  0 0  0 0  0 -1  0 -1  0;
%    0.6 -0.4 -3.0  0  0  0 -1 0  0 0 -1  0  0  0 -1];
% x = sdp_cvx(c,F);
function x = sdp_cvx(c,F)
```

```
p = length(c);
[n,m] = size(F);
Fx = F(:,1:n);
cvx_begin quiet
  variable x(p);
  minimize(c'*x);
  subject to
  for i = 1:p,
     Fx = Fx + x(i)*F(:,(i*n+1):((i+1)*n));
  end
  Fx == semidefinite(n);
cvx_end
```

Example 1.23 SOCP

```
% Program: socp_cvx.m
% To solve second-order cone programming (SOCP) problem
%       minimize b'*x
%       subject to: ||Ai'*x + ci|| <= bi'*x + di
%               for i = 1, ..., q
% using cvx, where x = [x1 x2 ... xm]' and Ai is a matrix of size m by ni.
Thus there are a total of q
% 2nd-order (Lorentz) cones, each with size ni.
% Input:
% A = [A1 A2 ... Aq]
% B = [b1 b2 ... bq]
% b: constant vector of length m defining objective function
% c = [c1; c2; ...; cq]
% d = [d1 d2 ... dq]
% k = [n1 n2 ... nq]
% Output:
% x: solution of the SOCP problem.
% Written by W.-S. Lu, University of Victoria.
% Last modified: Oct. 30, 2012.
% Example: Solve the SOCP problem in Example 14.5 from PO, where
% b = [1 0 0 0 0]';
% A = [0  0  0   0  0  0;
%   -1  0  0.5   0  0  0;
%    0  1  0   1  0  0;
%    1  0  0   0 -0.7071 -0.3536;
%    0  -1  0   0 -0.7071  0.3536];
% z = zeros(5,1);B = [b z z];
% c = [0 0 -0.5 0 4.2426 -0.7071]';
% d = [0 1 1];
% k = [2 2 2];
% x = socp_cvx(A,B,b,c,d,k);
function x = socp_cvx(A,B,b,c,d,k)
```

```
m = length(b);
q = length(d);
t = 0;
cvx_begin quiet
  variable x(m);
  minimize(b'*x);
  subject to
  for i = 1:q,
     Ai = A(:,(t+1):(t+k(i)));
     bi = B(:,i);
     ci = c((t+1):(t+k(i)));
     di = d(i);
     norm(Ai'*x+ci) <= bi'*x + di;
     t = t + k(i);
  end
cvx_end
```

Problems

1.1 [7] Suppose we have three colored boxes *r* (red), *b* (blue), and *g* (green). Box *r* contains 3 apples, 4 oranges, and 3 limes; box *b* contains 1 apple, 1 orange, and 0 limes; and box *g* contains 3 apples, 3 oranges, and 4 limes. If a box is chosen at random with probabilities $p(r) = 0.2$, $p(b) = 0.2$, and $p(g) = 0.6$, and a piece of fruit is removed from the box (with equal probability of selecting any of the items in the box), then

(i) what is the probability of selecting an apple?

(ii) If we observe that the selected fruit is an orange, what is the probability that it came from the green box?

1.2 [7] There are 2 opaque bags. One bag has two blue balls and the other has one blue ball and one red ball. Suppose you pick a bag at random, and then pick a ball from that bag at random. You found the ball you picked was a blue ball. Now you pick the second ball from that same bag. Compute the probability that this ball is also blue?

1.3 Suppose we have a box that contains *A* red balls and *B* blue balls, one takes balls out of the box one by one, what is the probability that the balls that remain in the box are all red balls?

1.4 Prove the equality in Eq. (1.27).

1.5 Let *x* be a random variable with mean (expectation) $E[x] = \mu$ and variance $\mathrm{Var}[x] = \sigma^2$. Compute and express the mean and variance of random variable $ax + b$ in terms of μ and σ^2, where *a* and *b* are constants.

1.6 Let x denote a set of temperature readings in Celsius (C). Treating *x* as a random variable, its average (expectation) and variance are known to be 22.5° C and $\sigma^2 = 0.4$. If we convert x to a set of temperature readings *y* in Fahrenheit (F) using $F = (9/5)C + 32$, what are the average and variance of the temperature data in Fahrenheit?

1.7 Let ξ_1 and ξ_2 be two independent random variables, find the distribution function of random variable $\xi = \xi_1 + \xi_2$ where

(i) ξ_1 and ξ_2 are uniform distribution over intervals (–5, 1) and (1, 5), respectively.

(ii) ξ_1 and ξ_2 are random variables with density functions $\varphi_1(x) = \varphi_2(x) = \dfrac{1}{2\alpha} e^{-|x|/\alpha}$ with some $\alpha > 0$.

1.8 Consider the convex QP problem

$$\underset{x}{\text{minimize}} \quad \frac{1}{2}\boldsymbol{x}^T \begin{bmatrix} 10 & -9 \\ -9 & 8.15 \end{bmatrix} \boldsymbol{x} + \boldsymbol{x}^T \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{P1.1}$$

(i) Apply SDM to solve the problem with initial point $\boldsymbol{x}_0 = [1\ 1]^T$. Exact line search is carried out using the closed-form formula

$$\alpha_k = \frac{\boldsymbol{g}_k^T \boldsymbol{g}_k}{\boldsymbol{g}_k^T \boldsymbol{H} \boldsymbol{g}_k}$$

where $\boldsymbol{H}$ is the Hessian of the objective function and $\boldsymbol{g}_k$ is the gradient of the objective function at $\boldsymbol{x}_k$. How many iterations the SDM has to take to achieve a solution $\boldsymbol{x}_s$ with the accuracy of $\|\boldsymbol{x}_s - \boldsymbol{x}^*\|_2 \le 10^{-5}$ where $\boldsymbol{x}^*$ is the true minimizer of the problem which is known to be $\boldsymbol{x}^* = [-16.3 \quad -18]^T$?

(ii) Now solve the problem in part (i) by scaling the variable $\boldsymbol{x}$ with $\boldsymbol{x} = \boldsymbol{T}\,\boldsymbol{y}$ first, where matrix $\boldsymbol{T}$ is given by

$$\boldsymbol{T} = \begin{bmatrix} 0.25 & 4.5 \\ 0 & 5 \end{bmatrix}$$

Working with the scaled QP problem, how many SDM iterations are needed to achieve a solution with the same accuracy as in part (i)?

Explain why such a scaling matrix can improve SDM's performance? If this $\boldsymbol{T}$ was not given, how do you find a similar $\boldsymbol{T}$ to properly scale an ill-conditioned QP problem?

To earn credit, include the solution details, your MATLAB code, and all numerical results.

1.9 Consider the one-variable *polynomial curve fitting* problem where a polynomial model of the form

$$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_m x^m = \sum_{i=0}^{m} w_i x^i \qquad \text{(P1.1)}$$

is used to fit a data set $\{(x_k, y_k), \text{for } k = 1, \ldots, n\}$, where $\boldsymbol{w} = [w_0 \quad w_1 \quad \cdots \quad w_m]^T$. The function $y(x, \boldsymbol{w})$ is a polynomial of order m in x, whose behavior for a fixed order m is determined by its coefficients $\{w_i, i = 0, 1, \ldots, m\}$. Therefore, the *polynomial curve fitting* problem is about to find "optimal" $\{w_i, i = 0, 1, \ldots, m\}$ such that the polynomial $y(x, \boldsymbol{w})$, when evaluated at $\{x_k, k = 1, 2, \ldots, n\}$, produces $\{y(x_k, \boldsymbol{w}), k = 1, \ldots, n\}$ that is the *closest* to the given data $\{y_k, k = 1, 2, \ldots, n\}$ in a certain sense. The *least-squares* (LS) polynomial curve fitting is one of the most popular approach to deal with polynomial modeling problems, which in the present case amounts to finding an $(m + 1)$-dimensional $\boldsymbol{w}$ that solves the unconstrained problem

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad e(\boldsymbol{w}) = \frac{1}{2} \sum_{k=1}^{n} |y(x_k, \boldsymbol{w}) - y_k|^2 \qquad \text{(P1.2)}$$

where $y(x, \boldsymbol{w})$ is given by (P1.1). Show that the optimal $\boldsymbol{w}$ satisfies the linear system of equations

$\boldsymbol{A}\boldsymbol{w} = \boldsymbol{t}$ where, for real-valued data set $\{(x_k, y_k), \text{for } k = 1,\ldots,n\}$, $\boldsymbol{A} = (a_{i,j}) \in R^{(m+1)\times(m+1)}$ with i, $j = 0, 1, \ldots, m$; and $\boldsymbol{t} = (t_i) \in R^{(m+1)\times 1}$ with $i = 0, 1, \ldots, m$ are determined by the data set as

$$a_{i,j} = \sum_{k=1}^{n} (x_k)^{i+j} \quad \text{and} \quad t_i = \sum_{k=1}^{n} (x_k)^i y_k \tag{P1.3}$$

To get credit, include all derivation details.

1.10 The purpose of this problem is to perform numerical evaluation of the LS data fitting method.

(i) Data preparation: the data to be used in what follows was generated from a 2nd order polynomial model that were contaminated by a small amount of Gaussian noise. Use the MATLAB code below to produce the data.

```
w_target = [9.5   -5.6   1];   % here the vector is associated with a poly with increasing order.
p_target = fliplr(w_target);   % function polyval requires poly coefficients with decreasing order.
x = [0.5   2.5   3   3.95   4.05];
randn('state',16)
y = polyval(p_target,x) + 0.4*randn(1,5);
```

To see how the noisy data are related to the target polynomial, use MATLAB to plot the following:

```
t = 0:4.5/999:4.5;
z = polyval(p_target,t);
figure(1)
plot(t,z,'k')
hold on
plot(x,y,'ro','linewidth',1.5)
```

(ii) Apply the polynomial data fitting method from Prob. 1.7 to the data set {x, y} generated in part (i) with a 2nd – order polynomial $y(x, \boldsymbol{w})$ (i.e. $m = 2$, see Eq. (P1.1)). To earn credit:

• Compute and report the numerical values of $\boldsymbol{w}$ for the $y(x, \boldsymbol{w})$ that solves the minimization problem in (P1.2);

• Plot $y(t, \boldsymbol{w})$ versus t = 0:4.5/999:4.5 in Figure 1 generated in part (i) with the data set also displayed in the figure;

• Evaluate the approximation error, namely the value of the minimized objective function $e(\boldsymbol{w})$ (see (P1.2)) and comment the performance of the 2nd-order model obtained.

1.11 Repeat Problem 1.8 with a 4th-order polynomial $y(x, \boldsymbol{w})$.

1.12 Use CVX to solve the following LP and convex QP problems

(i)

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) = 2x_1 + 9x_2 + 3x_3 \\ \text{subject to:} \quad & -2x_1 + 2x_2 + x_3 - x_4 = 1 \\ & x_1 + 4x_2 - x_3 - x_5 = 1 \\ & x_i \geq 0 \quad \text{for } i = 1, 2, ..., 5 \end{aligned}$$

(ii)

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) = x_1 + x_2 \\ \text{subject to:} \quad & x_1 \leq 2 \\ & x_2 \leq 2 \\ & -x_2 \leq 5 \\ & -2x_1 + x_2 \leq 2 \\ & 2x_1 + x_2 \leq 4 \end{aligned}$$

(iii)

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) = (x_1 - x_3)^2 + (x_2 - x_4)^2 \\ \text{subject to:} \quad & -x_1 \leq 0 \\ & -x_2 \leq 0 \\ & x_1 + 2x_2 \leq 2 \\ & -x_4 \leq -2 \\ & -x_3 - x_4 \leq -3 \\ & x_3 + 2x_4 \leq 6 \end{aligned}$$

To earn credit, include CVX code and report numerical values of the solutions with an accuracy of no-less-than-eight decimal places.