Shu Lu *and* Quoc Tran-Dinh

Introduction to Optimization

From Linear Programming to Nonlinear Programming

November 1, 2019

STOR-UNC-Chapel Hill

Chapter 8

Introduction to Nonlinear Programming

8.1 Introduction

So far, we have focused on linear programming and its applications. In this chapter, we discuss nonlinear programming (NLP), which provides a more general framework to capture underlying structures of practical problems, to model complex applications in engineering, artificial intelligence, computer science, and so on. While theory and methods for LPs are well understood and developed, theory and methods for NLPs are still under intensive ongoing research. Here, we give an overview on some nonlinear programming problems at an introductory level.

8.1.1 Mathematical formulation

A general optimization problem in finite dimensions can be formulated as follows:

$$\begin{cases} \min\limits_{x\in\mathbb{R}^n} f_0(x) \\ \text{s.t.} \quad f_i(x) \le 0, \ \ i = 1,\cdots,m, \\ \qquad x \in \mathscr{X}. \end{cases} \tag{8.1}$$

Here, f_i, $i = 0,\cdots,m$ are $m+1$ functions from $\mathbb{R}^n$ to $\mathbb{R}$, and $\mathscr{X}$ is a nonempty, closed subset in $\mathbb{R}^n$. Clearly, maximization problems or problems with constraints of the form $f_i(x) = 0$ or $f_i(x) \ge 0$ can be easily converted into (8.1). We define

$$\mathscr{F} := \{x \in \mathbb{R}^n \mid x \in \mathscr{X},\ f_i(x) \le 0, \ \ i = 1,\cdots,m\} \tag{8.2}$$

as the feasible set of (8.1). In many problems, the feasible set $\mathscr{F}$ is explicitly expressed through constraint functions. In some other cases, $\mathscr{F}$ is given implicitly.

For instance, $\mathscr{F}$ can be the solution set of another problem. When $\mathscr{F} = \mathbb{R}^n$, we say that (8.1) is an unconstrained problem.

There are more than one way to classify optimization problems. Here, we clarify the difference between linear and nonlinear programming problems:

- **A linear programming problem** minimizes or maximizes a linear objective function of finitely many variables (say, $x_1, x_2, \cdots, x_n$) under finitely many linear constraints. In other words, if $f_0(x) = c^T x$ is a linear function, $f_i(x) = a_i^T x - b_i$ is an affine function for each $i = 1, \cdots, m$, and $\mathscr{X}$ is the whole space $\mathbb{R}^n$, the orthant $\mathbb{R}^n_+$, or more generally a polyhedron, then (8.1) is a linear program.
- If $\mathscr{X}$ in (8.1) is specified by constraint functions, and at least some of the constraints or the objective are nonlinear, continuous functions, then (8.1) is a **nonlinear programming problem**.

If $\mathscr{X}$ is specified by some integrality constraints, i.e., if some variables x_i are required to be integers, then (8.1) becomes a discrete optimization problem, or integer programming problem. We will give a basic introduction to integer programming in the next chapter. To read more about different classes of optimization problems, see, e.g., [3].

8.1.2 Applications and examples

Nonlinear programming have many direct applications. Below are some examples:

- **Engineering:** Nonlinear programming can be used to control and stabilize a system; optimize time, energy and material; design optimal structures in civil engineering, and so on.
- **Computer science:** computer vision, machine learning, image processing, graph theory.
- **Economics and finance:** equilibrium price, risk minimization, utility maximization, portfolio selection, expenditure minimization problems, etc.

Here are two specific examples.

1. A simple quadratic program:

$$\begin{cases} \min\limits_{x} \ \sum\limits_{j=1}^{n} (x_j - c_j)^2 \\ \text{s.t.} \ \ \sum\limits_{j=1}^{n} A_{ij}x_j \le b_i, \ i = 1, \cdots, m. \end{cases}$$

The optimal solution of the above problem is the Euclidean projection of the point $c = (c_1, c_2, \cdots, c_n)^T$ onto the polyhedral set $\mathscr{P} := \{x \in \mathbb{R}^n \mid Ax \le b\}$.

2. A sparse solution recovery of a linear system (a related problem has been considered in the previous chapter):

$$\begin{cases} \min\limits_{x} \ \sum\limits_{j=1}^{n} |x_j| \\ \text{s.t.} \ \ \sum\limits_{j=1}^{n} A_{ij}x_j = b_i, \ i = 1, \cdots, m. \end{cases}$$

This problem in the above form is an NLP because the objective function is nonlinear, but it can be converted into an LP by introducing more variables. A related problem is

$$\min_{x} \frac{1}{2} \sum_{i=1}^{m} \left(\sum_{j=1}^{n} A_{ij}x_j - b_i \right)^2 + \lambda \sum_{j=1}^{n} |x_j|,$$

where $\lambda > 0$ is a regularization parameter. The latter problem cannot be converted into an LP. These problems are commonly used in compressive sensing, a signal processing technique.

8.1.3 The focus of this chapter

Nonlinear programming is a very broad class of optimization problems. Here, we focus on the following two subclasses of problems.

- **Convex quadratic programs (QP).** Quadratic programming is a natural extension of linear programming. We will discuss applications of QP in portfolio selection and support vector machines, as well as some basic properties and solution methods.
- **Convex programs (CP).** Convex programs are nonlinear programs with convex objective functions and convex feasible sets. They cover linear programs and convex quadratic programs as special cases. Many problems in compressive sensing, signal and image processing, machine learning, statistics, and data science are convex programs.

8.2 Multivariable calculus, linear algebra, and convexity

First, we recall some basic concepts of multivariable calculus. Then, we review symmetric positive semidefinite matrices. Finally, we provide definitions and basic properties of convex sets and convex functions. See, e.g., [**?**] for details on multivariable calculus.

8.2.1 Multivariable functions: A review

A real-valued function f of n variables $x_1, \cdots, x_n$ is a mapping from the n variables to a scalar, denoted as

$$f : \mathbb{R}^n \to \mathbb{R} \quad \text{or} \quad x \mapsto y = f(x).$$

Here are some examples.

1. The function $f(x_1,x_2) = 4x_1 + 2x_2 + 4$ is a linear function of two variables x_1 and x_2.
2. The function $f(x_1,x_2) = x_1^2 + 2x_2^2 + 3x_1x_2 + 2x_1 + 4x_2$ is a quadratic function of two variables x_1 and x_2.
3. The function $f(x_1,x_2) = \log(1 + x_1^2 + x_2^2)$ is a nonlinear function of two variables x_1 and x_2.
4. The function $f(x) = \sum_{i=1}^n x_i \log(x_i)$ is a nonlinear function of n variables $x_1, x_2, \cdots, x_n$.

The set of points x for which $f(x)$ is defined is called the domain of f, denoted by $\mathrm{dom}\,(f)$. For instance, the domain of the first three functions above is the whole space, while the domain of the last function is $\mathbb{R}^n_{++}$, the set of $x \in \mathbb{R}^n$ with strictly positive entries.

Gradients and Hessian matrices of multivariable functions. The gradient of a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ at a given point $x \in \mathbb{R}^n$ is defined as

$$\nabla f(x) = \left(\frac{\partial f(x)}{\partial x_1}, \cdots, \frac{\partial f(x)}{\partial x_n} \right)^T,$$

which is the vector of all partial derivatives of f with respect to all variables $x_1, x_2, \cdots, x_n$. Here, $\frac{\partial f(x)}{\partial x_n}$ stands for the partial derivative of f with respect to x_n at x. For example, if $f(x) = 2x_1 + 3x_2 + 4$, then $\nabla f(x) = (2,3)^T$. If $f(x) = 2x_1^2 + 4x_2^2 + 3x_1x_2 + 2x_1 + x_2$, then $\nabla f(x) = (4x_1 + 3x_2 + 2, 8x_2 + 3x_1 + 1)^T$. The

gradient of a function $f:\mathbb{R}^n\to\mathbb{R}$ can be considered as a mapping from $\mathbb{R}^n$ to $\mathbb{R}^n$, called the gradient mapping: $\nabla f:\mathbb{R}^n\to\mathbb{R}^n$.

The Hessian matrix of a twice-differentiable function $f:\mathbb{R}^n\to\mathbb{R}$ at x is defined as

$$\nabla^2 f(x)=\begin{bmatrix}\frac{\partial^2 f(x)}{\partial {x_1}^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1\partial x_n}\\ \cdots & \cdots & \cdots\\ \frac{\partial^2 f(x)}{\partial x_n\partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial {x_n}^2}\end{bmatrix},$$

which is a square matrix formed by all the second order partial derivatives of f. When the second order partial derivatives of f are continuous, the Hessian matrix $\nabla^2 f(x)$ is a symmetric matrix, i.e., $\nabla^2 f(x)=\nabla^2 f(x)^T$.

Below are some examples:

1. If $f(x)=a^Tx$ then $\nabla f(x)=a$ and $\nabla^2 f(x)=0$, the zero matrix.
2. If $f(x)=\frac{1}{2}x^TQx+q^Tx$ a quadratic function, where Q is symmetric, then

$$\nabla f(x)=Qx+q \quad \text{and} \quad \nabla^2 f(x)=Q.$$

3. If $f(x)=e^{x_1^2+x_2^2}$, then

$$\nabla f(x)=\begin{pmatrix}2x_1e^{x_1^2+x_2^2}\\ 2x_2e^{x_1^2+x_2^2}\end{pmatrix}, \text{ and } \nabla^2 f(x)=\begin{bmatrix}2e^{x_1^2+x_2^2}(1+2x_1^2) & 4x_2x_1e^{x_1^2+x_2^2}\\ 4x_1x_2e^{x_1^2+x_2^2} & 2e^{x_1^2+x_2^2}(1+2x_2^2)\end{bmatrix}.$$

The Hessian matrix of a function $f:\mathbb{R}^n\to\mathbb{R}$ can also be considered as a mapping from $\mathbb{R}^n$ to $\mathbb{R}^{n\times n}$, called the Hessian mapping: $\nabla^2 f:\mathbb{R}^n\to\mathbb{R}^{n\times n}$.

For a differentiable function $F:\mathbb{R}^n\to\mathbb{R}^m$ that takes the input of n independent variables and returns the values of m dependent variables, we write $F=(F_1,F_2,\cdots,F_m)$ with each $F_i:\mathbb{R}^n\to\mathbb{R}$, $i=1,\cdots,m$, being a real-valued function. The gradients of all the component functions F_i form an $m\times n$ matrix as

$$F'(x)=\begin{bmatrix}\nabla F_1(x)^T\\ \nabla F_2(x)^T\\ \vdots\\ \nabla F_1(x)^T\end{bmatrix}=\begin{bmatrix}\frac{\partial F_1}{\partial x_1}(x) & \frac{\partial F_1}{\partial x_2}(x) & \cdots & \frac{\partial F_1}{\partial x_n}(x)\\ \frac{\partial F_2}{\partial x_1}(x) & \frac{\partial F_2}{\partial x_2}(x) & \cdots & \frac{\partial F_2}{\partial x_n}(x)\\ \vdots & \vdots & \cdots & \vdots\\ \frac{\partial F_m}{\partial x_1}(x) & \frac{\partial F_m}{\partial x_2}(x) & \cdots & \frac{\partial F_m}{\partial x_n}(x)\end{bmatrix}.$$

This matrix is called the Jacobian matrix of F at x, and $F'(\cdot):\mathbb{R}^n\to\mathbb{R}^{m\times n}$ is called the Jacobian mapping.

Example 8.1. Consider a function $F:\mathbb{R}^3\to\mathbb{R}^2$ defined as $F(x)=(2x_1^2+x_1x_2+x_3^2,\ x_1x_2+x_2x_3+x_3x_1)$. Here, $F_1(x)=2x_1^2+x_1x_2+x_3^2$ and $F_2(x)=x_1x_2+x_2x_3+x_3x_1)$. Then, the Jacobian mapping $F':\mathbb{R}^3\to\mathbb{R}^{2\times 3}$ is

$$F'(x)=\begin{bmatrix}4x_1+x_2 & x_1 & 2x_3\\ x_2+x_3 & x_1+x_3 & x_2+x_1\end{bmatrix}.$$

The second derivatives of such vector functions are relatively complicated, and we omit them in this course.

8.2.2 Symmetric and positive semidefinite matrices

Given an $n\times n$ square matrix Q in $\mathbb{R}^{n\times n}$. We say that Q is symmetric if $Q=Q^T$. For example, $Q=I$, the identity matrix, is symmetric. The matrix $Q=\begin{bmatrix}1 & 2 & 3\\ 2 & 4 & -2\\ 3 & -2 & 5\end{bmatrix}$ is also a symmetric matrix.

We say that Q is positive semidefinite if for any vector $x\in\mathbb{R}^n$, we have $x^TQx\geq 0$. For example, $Q=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$ and $Q=\begin{bmatrix}4 & 2 & 0\\ 2 & 2 & 1\\ 0 & 1 & 4\end{bmatrix}$ are symmetric positive semidefinite. Indeed, we have

$$x^TQx=\begin{bmatrix}x_1\\ x_2\end{bmatrix}^T\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=x_1^2\geq 0.$$

For any vector $x=(0,x_2)^T\in\mathbb{R}^2$, we have $x^TQx=0$. Similarly, we have

$$x^TQx=\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}^T\begin{bmatrix}4 & 2 & 0\\ 2 & 2 & 1\\ 0 & 1 & 4\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}=4x_1^2+2x_2^2+4x_3^2+4x_1x_2+2x_2x_3=(2x_1+x_2)^2+(x_2+x_3)^2+3x_3^2\geq 0,$$

which shows that Q is positive semidefinite. We note that matrix $Q=\begin{bmatrix}1 & 2\\ 2 & 3\end{bmatrix}$ is not positive semidefinite. Indeed, we consider $x^TQx=x_1^2+4x_1x_2+3x_2^2$. Now, if we take $x_2=-\frac{x_1}{2}\neq 0$, then we have $x^TQx=x_1^2-2x_1^2+\frac{3}{4}x_1^2=-\frac{x_1^2}{4}<0$. Hence, Q is not positive semidefinite. Similarly, $Q=\begin{bmatrix}4 & 2 & 3\\ 2 & 2 & 1\\ 3 & 1 & 1\end{bmatrix}$ is not positive semidefinite. Indeed, let $x\in\mathbb{R}^3$ be a given vector, we have $x^TQx=4x_1^2+2x_2^2+x_3^2+4x_1x_2+6x_1x_3+2x_2x_3$.

Now, let $x_2 = -x_1$ and $x_3 = -x_1$ for any $x_1 \neq 0$. We have $x^T Qx = 4x_1^2 + 2x_1^2 + x_1^2 - 4x_1^2 - 6x_1^2 + 2x_1^2 = -x_1^2 < 0$. Hence, Q is not positive semidefinite.

We say that Q is positive definite if for any vector $x \in \mathbb{R}^n$ with $x \neq 0$, we have $x^T Qx > 0$. For example, $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ is symmetric positive definite. Indeed, $x^T Qx = x_1^2 + x_2^2 > 0$ if $x \neq 0$. Similarly, $Q = \begin{bmatrix} 4 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{bmatrix}$ is also symmetric positive definite since $x^T Qx = 4x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_2x_3 = (x_1 + x_2)^2 + (x_2 + x_3)^2 + 3x_1^2 > 0$ for any $x \neq 0$.

Properties: If we know the eigenvalue decomposition of any symmetric matrix Q as $Q = V\Lambda V^{-1}$, where Λ is a diagonal matrix formed from n eigenvalues λ_i of Q which are all real, and V is an $n \times n$ invertible matrix, then Q is symmetric positive semidefinite if $\min_i\{\lambda_i\} \geq 0$, and symmetric positive definite if $\min_i\{\lambda_i\} > 0$. For example, $Q = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}$ has an eigenvalue decomposition as

$$Q = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}.$$

Here, $\lambda_1 = 4$ and $\lambda_2 = 2$ are both real and positive. Hence, Q is positive definite.

8.2.3 Convex sets and convex functions

We recall the definition of convex sets and convex functions in this subsection.

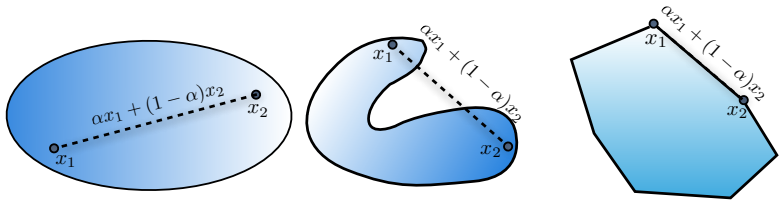
8.2.3.1 Convex sets

Definition 8.1. A set $\mathscr{C}$ in $\mathbb{R}^n$ is said to be convex if for any points $x_1, x_2 \in \mathscr{C}$, and for any $\alpha \in [0,1]$, then $(1-\alpha)x_1 + \alpha x_2 \in \mathscr{C}$.

In other words, $\mathscr{C}$ is convex if for any two points $x_1, x_2 \in \mathscr{C}$, the line segment $[x_1, x_2]$ connecting these two points lies in $\mathscr{C}$. Geometrically, the figure below shows three sets, where the left and the right are convex, while the middle one is nonconvex.

Definition 8.2. The set $\mathscr{P}$ is said to be polyhedral if it is formed as the intersection of a finite number of half-spaces, that is

$$\mathscr{P} = \left\{x \in \mathbb{R}^n \mid a_i^T x \leq b_i,\ i = 1, \cdots, m\right\}.$$



Here are few examples.

- Any polyhedral set $\mathscr{P} := \{x \in \mathbb{R}^n \mid Ax \leq b\}$ is convex. Indeed, if $x, y \in \mathscr{P}$, we have $Ax \leq b$ and $Ay \leq b$. Now, for any $\alpha \in [0,1]$, we have $A((1-\alpha)x + \alpha y) = (1-\alpha)Ax + \alpha Ay \leq (1-\alpha)b + \alpha b = b$. Hence, $(1-\alpha)x + \alpha y \in \mathscr{P}$. We say that $\mathscr{P}$ is convex. We note that half-spaces and hyperplane are special polyhedra. Hence, they are convex. (We note that our definition here may be different from geometry, where polyhedra in geometry may not be convex, like star-shapes).
- The whole space $\mathbb{R}^n$ and empty set are convex. It is trivial to prove.
- The ball $\mathscr{B} = \{x \in \mathbb{R}^n \mid \sum_{j=1}^n (x_i - a_i)^2 \leq r\}$ is convex. Indeed, we can write this ball using Euclidean norm as $\mathscr{B} = \{x \in \mathbb{R}^n \mid \|x-a\| \leq r\}$. For any $x, y \in \mathscr{B}$, we have $\|x-a\| \leq r$, and $\|y-a\| \leq r$. Taking any $\alpha \in [0,1]$, we consider the point $z = (1-\alpha)x + \alpha y$. We have $\|z-a\| = \|(1-\alpha)x + \alpha y - a\| = \|(1-\alpha)(x-a) + \alpha(y-a)\| \leq \|(1-\alpha)(x-a)\| + \|\alpha(y-a)\| = (1-\alpha)\|x-a\| + \alpha\|y-a\| \leq (1-\alpha)r + \alpha r = r$. Here, the first inequality follows from the triangle inequality. Hence, we conclude that $z \in \mathscr{B}$, which means that $\mathscr{B}$ is convex by definition.

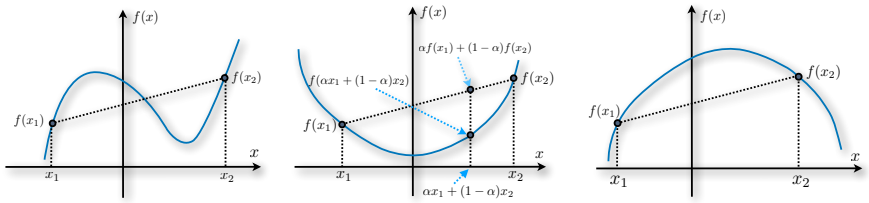
8.2.3.2 Convex functions

Definition 8.3. A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be convex if for any x_1, x_2, and $\alpha \in [0,1]$, then

$$f((1-\alpha)x_1 + \alpha x_2) \leq (1-\alpha)f(x_1) + \alpha f(x_2),$$

In other words, f is convex if the tangent line of f at any point x lies below its graph. A function f is said to be concave if $-f$ is convex. Geometrically, we can illustrate convex functions in the figure below, where the left function is nonconvex, the middle one is convex, and the right plot is concave. Below are few examples.

1. Constant functions $f(x) = c$ and linear functions $f(x) = a^T x$ are convex.



2. Consider a quadratic function $f(x) = \frac{1}{2}x^T Qx + q^T$, where Q is an $n \times n$ symmetric matrix, $q \in \mathbb{R}^n$. This function is convex if and only if Q is symmetric and positive semidefinite.
3. The norm function $f(x) = \|x\|$ is convex. Indeed, for any points $x, y \in \mathbb{R}^n$, and any $\alpha \in [0,1]$, we consider $z = (1-\alpha)x + \alpha y$. By the triangle inequality, we have $\|z\| = \|(1-\alpha)x + \alpha y\| \leq \|(1-\alpha)x\| + \|\alpha y\| = (1-\alpha)\|x\| + \alpha\|y\|$. Hence, $f((1-\alpha)x+\alpha y) \leq (1-\alpha)f(x)+\alpha f(y)$, which shows that f is convex.

We provide some simple properties of convex functions:

- If f and g are convex, and $\beta, \gamma \geq 0$, then $\beta f + \gamma g$ is convex.
- If $f : \mathbb{R} \to \mathbb{R}$ is twice differentiable, then f is convex if and only if $f''(x) \geq 0$ for all x.
- If $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable, then f is convex iff the Hessian $\nabla^2 f(x)$ is positive semidefinite for all x.

One of the most important properties of a convex function is that any local minimum is a global one as described below. We consider the optimization problem

$$\min_x f(x) \;\; \text{s.t. } x \in \mathcal{X}, \tag{8.3}$$

where f is a convex function and $\mathcal{X}$ is a nonempty, closed and convex set. We say that $x^* \in \mathcal{X}$ is a local optimal solution of this problem if there exists a neighborhood $\mathcal{V}(x^*)$ of x^* in $\mathbb{R}^n$ such that

$$f(x^*) \leq f(x), \;\; \forall x \in \mathcal{V}(x^*) \cap \mathcal{X}.$$

We say that $x^* \in \mathcal{X}$ is a global optimal solution of this problem if

$$f(x^*) \leq f(x), \;\; \forall x \in \mathcal{X}.$$

The following theorem is key in convex optimization.

Theorem 8.1. *For the convex optimization problem* (8.3)*, every local optimal solution x^* (if exists) is a global optimal solution. The solution set of* (8.3) *is also convex.*

As we have seen from calculus that if x^* is an extreme point of a function f, then $f'(x^*) = 0$ due to Fermat's rule. For example, $x^* = 2$ is an extreme point (a minimum point) of $f(x) = x^2 - 4x$. Then, we have $f'(x^*) = 2x^* - 4 = 0$. This concept becomes important in convex optimization. Let us begin by considering an unconstrained convex optimization problem:

$$\min_{x\in\mathbb{R}^n} f(x) \tag{8.4}$$

where f is a convex function and differentiable.

Theorem 8.2. *A $x^* \in \mathbb{R}^n$ is an optimal solution of the unconstrained convex optimization problem* (8.4) *if and only if*

$$\nabla f(x^*) = 0. \tag{8.5}$$

The condition (8.5) is called the optimality condition, or simply, Fertmat's rule.

Example 8.2. Let us consider the following convex optimization problem:

$$\min_{x\in\mathbb{R}^2} f(x) = x_1^2 - 4x_1x_2 + 6x_2^2 - 2x_1.$$

Since f is convex and differentiable, using Theorem 8.2, we can show that $x^* \in \mathbb{R}^2$ is an optimal solution of this problem iff $\nabla f(x^*) = 0$. That is,

$$\nabla f(x^*) = \begin{pmatrix} 2x_1^* - 4x_2^* - 2 \\ -4x_1^* + 12x_2^* \end{pmatrix} = 0.$$

This is a linear system of x_1^* and x_2^*. Solve this system we obtain $(x_1^*, x_2^*)^T = (3,1)^T$, which is an optimal solution of the above problem with the optimal value $f(x^*) = -3$. □

For the constrained convex problem (8.11), the optimality condition becomes more complicated. That is, $x^* \in \mathscr{X}$ is an optimal solution of (8.11) if and only if

$$\nabla f(x^*)^T(y - x^*) \geq 0, \;\; \forall y \in \mathscr{X}. \tag{8.6}$$

This condition is no longer an equation, but it is an inequation, and is often hard to analytically solve it.

8.3 Convex quadratic programming

Mathematically, a quadratic program is a nonlinear optimization problem of the following form:

$$\begin{cases} \min\limits_{x} \ \frac{1}{2}x^T Qx + q^T x \\ \text{s.t.} \ \ Ax = b, \\ \qquad x \geq 0. \end{cases} \tag{8.7}$$

where Q is an $n \times n$ symmetric matrix, A is an $m \times n$ matrix, $b \in \mathbb{R}^m$ and $q \in \mathbb{R}^n$ are two given vectors. This problem is called a standard quadratic program. When $Q = 0$, this problem reduces to a standard linear program. Hence, we can consider QP as an extension of LP.

The objective function $f(x) = \frac{1}{2}x^T Qx + q^T x$ can be convex or nonconvex. It is clear that $\nabla^2 f(x) = Q$. Hence, f is convex if and only if Q is positive definite. In this section, we only consider convex quadratic programs, where Q is symmetric positive semidefinite. Note that if Q is non-symmetric, we can always symmetrize it using the fact that $x^T Qx = \frac{1}{2}(x^T(Q+Q^T)x$, where the right-hand side is symmetric.

As an example, we consider the following least-squares problem:

$$\min_{x\in\mathbb{R}^n}\left\{\frac{1}{2}\|Ax-b\|_2^2 = \frac{1}{2}\sum_{i=1}^{m}\Big(\sum_{j=1}^{b}A_{ij}x_j - b_i\Big)^2\right\}$$

Let us define $Q = A^T A$ (symmetric and positive semidefinite), and $q = -A^T b$, then we can write this problem as

$$\min_x \frac{1}{2}x^T Qx + q^T x.$$

This problem is a convex quadratic program without constraint.

Properties of convex quadratic programming: We consider some basic properties of convex quadratic programming.

1. Any optimal solution x^* of the convex quadratic problem (8.7) is global optimal solution, i.e., $f(x^*) \leq f(x)$ for all feasible solutions x.
2. The solution set is convex.
3. If Q is symmetric positive definite and the problem is feasible, then it always has a unique solution.

Now, we consider two special cases of QPs: Unconstrained QPs and Equality constrained QPs.

8.3.1 Unconstrained quadratic programming

We consider the following unconstrained convex QP problem:

$$\min_{x\in\mathbb{R}^n}\left\{f(x)=\tfrac{1}{2}x^TQx+q^Tx\right\},$$

where Q is a symmetric positive semidefinite matrix of the size $n\times n$, and $q\in\mathbb{R}^n$.

A point $x^*\in\mathbb{R}^n$ is an optimal solution to this problem if and only if

$$\nabla f(x^*)=Qx^*+q=0.$$

If Q is invertible, then we can compute the solution x^* as $x^*=-Q^{-1}q$. Here, the inverse Q^{-1} can be computed by using Cholesky decomposition, or we can solve $Qx^*=-q$ using a conjugate gradient method [1]. If Q is not invertible, then other methods should be conducted to solve $Qx^*+q=0$ such as QR factorization.

Example 8.3. We consider the unconstrained convex QP problem

$$\min_{x\in\mathbb{R}^3}\left\{f(x)=\frac{1}{2}(x_1^2+4x_2^2+3x_3^2-2x_1x_2+2x_2x_3)+3x_1+2x_2-x_3\right\}.$$

In this case, we have $Q=\begin{bmatrix}1&-1&0\\-1&4&1\\0&1&3\end{bmatrix}$ and $q=(3,2,-1)^T$. Hence, the solution of this problem can be computed by solving the following linear system:

$$\begin{bmatrix}1&-1&0\\-1&4&1\\0&1&3\end{bmatrix}\begin{pmatrix}x_1^*\\x_2^*\\x_3^*\end{pmatrix}=-\begin{pmatrix}3\\2\\-1\end{pmatrix}.$$

Solving this system, we obtain $x^*=(-4,-2,1)^T$ is an optimal solution of the QP problem. Moreover, the optimal value is $f(x^*)=22.5$.

8.3.2 Quadratic programming with equality constraints

$$\min_x \tfrac{1}{2}x^TQx+q^Tx \quad \text{subject to } Ax=b. \tag{8.8}$$

Here Q, q, A and b are defined as above. We note that we are only interested in the case $m\le n$ and the m rows of A are linearly independent in this constrained

problem. Otherwise, the system $Ax = b$ may not have solution or has only one solution.

A point x^* is an optimal solution to this problem if and only if there exists a vector $y^* \in \mathbb{R}^m$ such that

$$\begin{cases} Qx^* + q + A^T y^* &= 0 \\ Ax - b &= 0. \end{cases}, \quad \text{which is equivalent to} \quad \begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ y^* \end{bmatrix} = \begin{bmatrix} -q \\ b \end{bmatrix}$$

Here, y^* is called the vector of Lagrange multipliers, and this condition is called the optimality condition.

If Q is invertible (i.e., Q is symmetric positive definite), and rows of A are linearly independent, then we can solve this system explicitly as follows. From $Qx^* + A^T y^* + q = 0$ we have $x^* = -Q^{-1}(A^T y^* + q)$. Substitute this into the second equation, we have $-AQ^{-1}A^T y^* - AQ^{-1}q = b$, which implies $AQ^{-1}A^T y^* = -AQ^{-1}q - b$. Hence, we obtain $y^* = -(AQ^{-1}A^T)^{-1}(AQ^{-1}q + b)$. Plug this expression into $x^* = -Q^{-1}(A^T y^* + q)$, we obtain

$$x^* = Q^{-1}A^T(AQ^{-1}A^T)^{-1}(AQ^{-1}q + b) - Q^{-1}q.$$

In summary, the solution of (8.8) can explicitly be written as

$$\begin{cases} x^* = Q^{-1}A^T(AQ^{-1}A^T)^{-1}(AQ^{-1}q + b) - Q^{-1}q \\ y^* = -(AQ^{-1}A^T)^{-1}(AQ^{-1}q + b). \end{cases}$$

Computing this solution requires the inverse Q^{-1} of Q, and $(AQ^{-1}A^T)^{-1}$, and some matrix-vector multiplications. Hence, in practice, other efficient methods to solve directly the linear system are often used.

Example 8.4. We consider the following quadratic program with one linear constraint:

$$\min_{x_1, x_2} \left\{ f(x) = \frac{1}{2}(x_1^2 + 2x_2^2 - 2x_1x_2) + 2x_1 - 3x_2 \right\} \quad \text{s.t. } x_1 + x_2 = 4.$$

Here, since matrix $Q = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}$, $q = (2, -3)^T$, $A = [1, 1]$ and $b = 4$, we can write down the optimality condition above as

$$\begin{cases} x_1^* - x_2^* + 2 + y^* & = 0, \\ -x_1^* + 2x_2^* - 3 + y^* & = 0, \\ x_1^* + x_2^* & = 4, \end{cases}$$

where y is the Lagrange multiplier. Solve this system, we obtain $x^* = (1.4, 2.6)^T$ and $y^* = -0.8$. Hence, x^* is the optimal solution, and y^* is the optimal multiplier.

8.3.3 Solution methods

There are two common methods to solve standard convex QP problem (8.7):

- *Active set methods:* Can be considered as extensions of the LP simplex methods.
- *Interior-point methods:* These methods are iterative methods based on logarithmic barriers.

Both methods are widely used, and have implemented in several software packages including `quadprog` in Matlab. Let us consider a standard primal-dual interior-point method to solve (8.7).

8.3.3.1 A primal-dual interior-point method for convex QPs

We first define a barrier problem associated with (8.7) as follows:

$$\min_x \left\{ \mathscr{B}_\mu(x) = \frac{1}{2}x^T Qx + q^T x - \mu \sum_{j=1}^n \log(x_j) \mid Ax = b \right\}, \tag{8.9}$$

where $\mu > 0$ is a penalty parameter. The solution of this problem $x^*(\mu)$ generates a set $\{x^*(\mu) \mid \mu > 0\}$ called the central path of the barrier method. We can write the optimality condition of this barrier problem as follows:

$$\begin{cases} Qx + q + A^T y - \mu \mathrm{diag}(X^{-1}) & = 0 \\ Ax - b & = 0. \end{cases}$$

Here, $X = \mathrm{diag}(x)$. Let us define $S := \mu X^{-1}$ as a slack variable. Then, $s = \mathrm{diag}(S)$ and $x \otimes s = \mu \mathbf{e}$, where $\mathbf{e} = (1, 1, \cdots, 1)^T$ is the vector of all ones, and $\otimes$ is component-wise product between two vectors. We write this optimality condition as

$$\begin{cases} Qx + A^T y - s = -q \\ Ax = b \\ x \otimes s = \mu \mathbf{e}. \end{cases}$$

Clearly, this is a nonlinear system due to $x \otimes s = XS\mathbf{e} = \mu \mathbf{e}$.

In the infeasible primal-dual interior-point methods, we linearize the nonlinear system $x \otimes s = \mu \mathbf{e}$ to obtain $X\Delta s + S\Delta x = \mu \mathbf{e} - XS\mathbf{e}$. Hence, the linear system for computing a Newton direction is defined as

$$\begin{bmatrix} Q & A^T & -\mathbb{I} \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} r^x \\ r^y \\ r^s \end{pmatrix} \quad \text{where} \quad \begin{cases} r^x & := -Qx - q - A^T y \\ r^y & := b - Ax \\ r^s & := \mu \mathbf{e} - XS\mathbf{e}. \end{cases}$$

Then, the next iteration is updated as

$$x_+ := x + \alpha_p \Delta x, \;\; y_+ := y + \alpha_p \Delta y, \;\; s_+ := s + \alpha_d \Delta s,$$

where $\alpha_p \in (0,1]$ and $\alpha_d \in (0,1]$ are given step-sizes chosen such that $x_+ > 0$ and $s_+ > 0$. Very often, the parameter μ is replaced by $\sigma\mu$ for a given $\sigma \in (0,1]$ and $\mu = \frac{s^T x}{n}$ measuring the duality gap.

Now, we can describe a primal-dual interior point method as follows: Algorithm 1 is called an infeasible primal-dual interior-point method since (x_k, y_k, s_k) may not be feasible to the primal and dual problems. If we start from a feasible point (x_0, y_0, s_0) such that $Qx_0 + A^T y_0 - s_0 = -q$, $Ax_0 - b = 0$, $x_0 > 0$ and $s_0 > 0$, then we can maintain $r_k^x = 0$ and $r_k^y = 0$ at each iteration. In this case, we obtain a feasible primal-dual IP method. We note that practical primal-dual IP methods are often more sophisticated than Algorithm 1 in order to accelerate their performance. A well-known variant is perhaps the Mehrotra predictor-corrector IP method [5]. More details about interior-point methods can be found in [2, 4, 5].

8.3.3.2 Implementation details

The main step of Algorithm 1 is the solution of the linear system at Step 7. This system is often solved by a substitution method combining with a Cholesky decomposition. Let us describe this method as follows: From the last equation $X_k \Delta s + S_k \Delta_x = r_k^s$, we compute Δs as $\Delta s = X_k^{-1}(r_k^s - S_k \Delta x)$. Substituting Δs into the first equation $Q\Delta x + A^T \Delta y - \Delta s = r_k^x$, we have

Algorithm 1 (*Primal-dual interior-point algorithm for QPs*)

1: **Input:** A tolerance $\varepsilon > 0$ and $\sigma \in (0,1]$.

2: **Initialization:** Find an initial point $(x_0, y_0, s_0) \in \mathbb{R}^n_{++} \times \mathbb{R}^m \times \mathbb{R}^n_{++}$.

3: **Main loop:**

4: Compute the duality gap $\mu_k := \frac{x_k^T s_k}{n}$.

5: If $\mu_k \leq \varepsilon$, then terminate, and (x_k, y_k, s_k) is an approximate solution.

6: Compute the residual $r_k^x := Qx_k - q - A^T y_k$, $r_k^y := b - Ax_k$ and $r_k^s := \sigma\mu_k \mathbf{e} - X_k S_k \mathbf{e}$.

7: Solve the linear system

$$\begin{bmatrix} Q & A^T & -\mathbb{I} \\ A & 0 & 0 \\ S_k & 0 & X_k \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} r_k^x \\ r_k^y \\ r_k^s \end{pmatrix}$$

to obtain Δx_k, Δy_k and Δs_k.

8: Find step-sizes $\alpha_k^p, \alpha_k^d \in (0,1]$ from the conditions $x_k + \alpha_k^p \Delta x_k > 0$ and $s_k + \alpha_k^d \Delta s_k > 0$.

9: Update

$$x_{k+1} := x_k + \alpha_k^p \Delta x_k, \;\; y_{k+1} := y_k + \alpha_k^p \Delta y_k, \quad s_{k+1} := s_k + \alpha_k^d \Delta s_k.$$

10: **End**

$$(Q + X_k^{-1} S_k)\Delta x + A^T \Delta y = r_k^x + X_k^{-1} r_k^s.$$

This implies that $\Delta x = (Q + X_k^{-1} S_k)^{-1} \left(r_k^x + X_k^{-1} r_k^s - A^T \Delta y \right)$. Substituting Δx into the second equation $A\Delta x = r_k^y$, we get $A(Q + X_k^{-1} S_k)^{-1} \left(r_k^x + X_k^{-1} r_k^s - A^T \Delta y \right) = r_k^y$, which leads to

$$A(Q + X_k^{-1} S_k)^{-1} A^T \Delta y = (Q + X_k^{-1} S_k)^{-1} \left(r_k^x + X_k^{-1} r_k^s \right) - r_k^y.$$

Let us define $H_k := A(Q + X_k^{-1} S_k)^{-1} A^T$ and $h_k := (Q + X_k^{-1} S_k)^{-1} \left(r_k^x + X_k^{-1} r_k^s \right) - r_k^y$. Since matrix H_k is symmetric positive definite, we can compute its Cholesky decomposition as $H_k := L_k L_k^T$, where L_k is a lower triangular matrix. Then the last linear system becomes

$$L_k L_k^T \Delta y = h_k.$$

This system can be solved by both back-substitution and forward-substitution methods. Since Q is symmetric positive semidefinite, we can compute its eigen-decomposition once in advance as $Q = U\Lambda U^T$, and apply a change of variable to convert the QP problem (8.7) into a form so that Q is a diagonal matrix. Without loss of generality, we can assume that Q is a diagonal matrix. Hence, computing $(Q + X_k^{-1}S_k)^{-1}$ now can be done efficiently, since this matrix is in fact a diagonal matrix.

To compute the stepsizes α_k^p and α_k^d, we can apply the following rules:

$$\begin{aligned}\alpha_k^p &:= 0.99 \min\left\{1, \min\left\{-\tfrac{(x_k)_i}{(\Delta x_k)_i} \mid (\Delta x_k)_i < 0\right\}\right\},\\ \alpha_k^d &:= 0.99 \min\left\{1, \min\left\{-\tfrac{(s_k)_i}{(\Delta s_k)_i} \mid (\Delta s_k)_i < 0\right\}\right\}.\end{aligned}$$

Here, we clip these step-sizes by a factor 0.99 to make sure that $x_{k+1} > 0$ and $s_{k+1} > 0$.

8.3.4 Applications

We present three applications in this section: Least-squares problem, portfolio selection, and support vector machine.

8.3.4.1 Application 1: Least squares problem

Problem description: We consider a linear model

$$b = x_0 + x_1 a_1 + \cdots + x_n a_n + \varepsilon,$$

where $a = (1, a_1, \cdots, a_n)^T \in \mathbb{R}^{n+1}$ is an input vector, and $b \in \mathbb{R}$ is the response, $x = (x_0, \cdots, x_n)^T \in \mathbb{R}^{n+1}$ is a parameter vector we need to estimate, and ε is an additive Gaussian white noise. Assume that we have a collection of m data points $(a^{(i)}, b_i)$ where $a^{(i)} \in \mathbb{R}^{n+1}$ generated from the above linear model. The noise vector ε has components $\varepsilon_i = b_i - (x_0 + x_1 a_1^{(i)} + \cdots + x_n a_n^{(i)})$ for $i = 1, \cdots, m$. Our goal is to minimize the sum of square errors $\sum_{i=1}^m \varepsilon_i^2$ over all possible parameter vector $x \in \mathbb{R}^{n+1}$.

Problem formulation: By minimizing the sum of squares of the noise vector $\varepsilon = (\varepsilon_1, \cdots, \varepsilon_m)^T$, we obtain the following least-squares problem:

$$\min_{x=(x_0,\cdots,x_n)^T} \frac{1}{2}\sum_{i=1}^m \left(y_i - x_0 - \sum_{j=1}^n a_j^{(i)} x_j\right)^2.$$

Let us define

$$x = \begin{pmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \cdots \\ b_m \end{pmatrix}, \quad \text{and } A = \begin{bmatrix} 1 & a_1^{(1)} & \cdots & a_n^{(1)} \\ \cdots & \cdots & \cdots & \cdots \\ 1 & a_1^{(m)} & \cdots & a_n^{(m)} \end{bmatrix}.$$

Then, we can write the above least-squares problem into the matrix form as

$$\min_x \frac{1}{2}\|Ax-b\|_2^2,$$

where $\|u\|_2^2 = \sum_{i=1}^m u_i^2$ is the square of the Euclidean norm. We note that we use $\|\cdot\|_2$ to denote the Euclidean norm. But sometimes, we drop the subscript 2 to have $\|\cdot\|$ which remains the Euclidean norm.

Solution: A vector $x^* \in \mathbb{R}^{n+1}$ is a solution of this problem if and only if it solves the following normal equation:

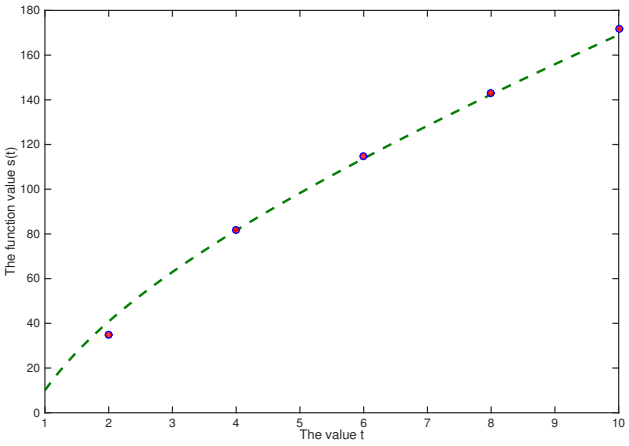
$$A^TAx = A^Tb.$$

This is a special square linear system of $n+1$ variables, and matrix A^TA is symmetric positive semidefinite.

- If A^TA is invertible, then the system has a unique solution $x^* = (A^TA)^{-1}A^Tb$. This is often the case if $m \geq n+1$, meaning that we have more than n data points.
- If A^TA is not invertible, then the system has many solutions. In this case, we need to regularize the problem or to use pseudo-inverse.

Concrete example: We consider a concrete example as follows. Given a function $s(t) = x_1t + x_2\log(t)$, where $t \in [1,10]$, and x_1 and x_2 are two parameters. We do not know x_1 and x_2 but by experiment, we obtain the following dataset in given in the table, where $\hat{s}(t)$ is an approximation of $s(t)$ at a given t. We note that we only obtain measured values $\hat{s}(t)$, not the exact values $s(t)$ (see the figure below). Now,

t	2	4	6	8	10
$\hat{s}(t)$	35	82	115	143	172



we formulate the problem of estimating the parameters x_1 and x_2 as a least-squares problem.

The error between $\hat{s}(t)$ and $s(t)$ is computed as $\varepsilon(t) = s(t) - \hat{s}(t)$. Using five data points as given in the table, we minimize the sum of square error as $\sum_{i=1}^{5} \varepsilon(t_i)^2$. Since $s(t) = x_1 t + x_2 \log(t)$, we can write this problem as a least-squares problem:

$$\min_{x_1,x_2} \frac{1}{2} \sum_{i=1}^{5} (x_1 t_i + x_2 \log(t_i) - \hat{s}(t_i))^2.$$

If we define

$$A = \begin{bmatrix} 2 & \log(2) \\ 4 & \log(4) \\ 6 & \log(6) \\ 8 & \log(8) \\ 10 & \log(10) \end{bmatrix} = \begin{bmatrix} 2 & 0.6931 \\ 4 & 1.3863 \\ 6 & 1.7918 \\ 8 & 2.0794 \\ 10 & 2.3026 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} 35 \\ 82 \\ 115 \\ 143 \\ 172 \end{bmatrix},$$

then we can write the above problem as $\min_{x \in \mathbb{R}^2} \frac{1}{2}\|Ax - b\|^2$. In this case, we can solve this problem to obtain the unique solution $x^* = (A^T A)^{-1} A^T b = (11.4489, 24.9939)^T$. Hence, we can write the given function as $s(t) = 11.4489t + 24.9939\log(t)$. Now, let say, we can use this function to evaluate s at other points of t such as $s(1) = 11.4489$, $s(3) = 61.8053$, and $s(5) = 97.4706$.

8.3.4.2 Application 2: Markowitz's portfolio selection

Problem description: Assume that we have n assets or stocks over a period of time. Let x_i denote the amount of asset i held throughout the period, with x_i in dollars at the price at the beginning of the period.

Let us denote p_i the relative price change of asset i over the period (i.e., the change in price over the price at the beginning of the period).

- The overall return on the portfolio is $r = p^T x$ (in dollars).
- The optimization variable is the vector of portfolio $x = (x_1, \cdots, x_n)$.
- The constraints include the total budget invested B, i.e., $\sum_{i=1}^n x_i = B$, and the non-negativity of x_i.
- We note that the price is random. Hence, p is a random vector, with known mean $\bar{p}$ and covariance Σ.

Therefore, with the portfolio x, the return r is a scalar random variable with mean $\bar{p}^T x$ and variance $x^T \Sigma x$. The classical Markowitz model tries to minimize the variance while satisfying the lower bound on the overall return $r_{\min}$, and the upper bound u.

Problem formulation: Finally, we can formulate this problem as

$$\begin{cases} \min\limits_x \ x^T \Sigma x \\ \text{s.t.} \ \ \bar{p}^T x \geq r_{\min} \\ \qquad \sum_{i=1}^n x_i = B, \\ \qquad 0 \leq x_i \leq u_i, \ i = 1, \cdots, n. \end{cases}$$

This problem is a constrained convex quadratic program. If we do not want to invest all the budget, we can replace the budget constraint by $\sum_{i=1}^n x_i \leq B$. The problem remains a constrained convex QP.

A numerical example: We consider three assets: AS1, AS2 and AS3, where the covariance matrix is

$$\Sigma = \begin{bmatrix} 3 & 1 & -0.5 \\ 1 & 2 & -0.4 \\ -0.5 & -0.4 & 1 \end{bmatrix},$$

the mean of the relative price change: $\bar{p} = (1.3, 1.2, 1.08)^T$, the budget is $B = 1$, and the upper bound: $u = 0.75$ and the lower bound on the overall return: $r_{\min} = 1.2$. We can model this problem either in GAMS or in Matlab. For example, here is a small Matlab script with three lines.

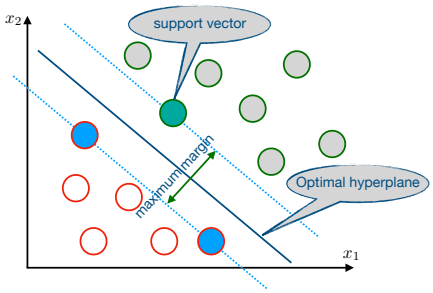
```
>> S = [3 1 -0.5; 1 2 -0.4; -0.5 -0.4 1];
>> A = -[1.3, 1.2, 1.08]; b = -1.2; C = [1, 1, 1]; d = 1;
>> [x, sig] = quadprog(S, zeros(3,1), A, b, C, d,
```

```
zeros(3,1), 0.75*ones(3,1));
```

Running the above code, we get $x = (0.4202, 0.2296, 0.3502)^T$ and the optimal variance is $\sigma = 0.3696$.

8.3.4.3 Application 3: Support vector machine

Problem description: We are given a set of data points $\left\{(\mathbf{x}^{(i)}, y_i)\right\}_{i=1}^m$, where $\mathbf{x} \in \mathbb{R}^n$ and the label $y_i \in \{-1, 1\}$. Assume that we draw $\left\{\mathbf{x}^{(i)}\right\}$ in the $\mathbb{R}^n$ space, and want to find a hyperplane $\mathscr{H} : \mathbf{w}^T\mathbf{x} + b = 0$ that separates all the points associated with a label $y_i = +1$ into one half-space and the others into the other half-space, and $\mathbf{w} \in \mathbb{R}^n$ is its normal vector (later, we call it a support vector), and $b \in \mathbb{R}$ is an intercept.



Problem formulation: There are points which may stay on this hyperplane, and we cannot classify them correctly. We want to build a hyperplane that has maximum margin to the points in this dataset. We assume that there are two hyperplanes $\mathbf{w}^T\mathbf{x} + b = 1$ and $\mathbf{w}^T\mathbf{x} + b = -1$ that are parallel to $\mathscr{H}$ and such they separate this dataset into two half-spaces. That is

$$\begin{cases} \mathbf{w}^T\mathbf{x}^{(i)} + b \geq 1 & \text{if } y_i = 1, \\ \mathbf{w}^T\mathbf{x}^{(i)} + b \leq -1 & \text{otherwise.} \end{cases} \quad \Leftrightarrow \quad y_i(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \ \ \forall i = 1, \cdots, m.$$

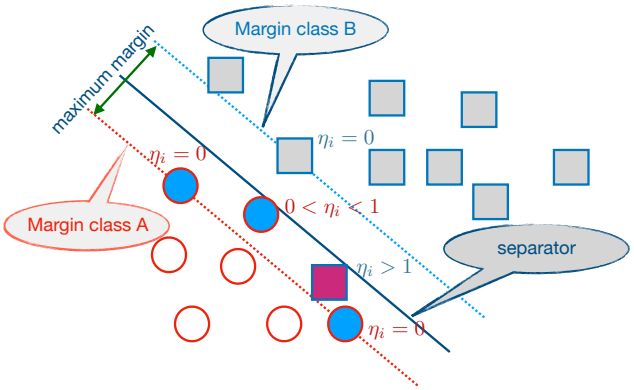
The distance between the two latter hyperplanes is given by $\frac{2}{\|\mathbf{w}\|}$ and we want to maximize it. This problem turns out to be minimizing $\|\mathbf{w}\|_2$, or equivalently, minimizing $\|\mathbf{w}\|_2^2$.

QP formulation: We can formulate this support vector machine problem as a special quadratic program:

$$\min_{\mathbf{w} \in \mathbb{R}^n} \frac{1}{2}\|\mathbf{w}\|_2^2 \ \text{ subject to } \ y_i(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \ \ \forall i = 1, \cdots, m. \tag{8.10}$$

This is a convex quadratic programming problem.

Handling the misclassification case: In reality, we do not have perfect classification of this dataset, we accept some misclassification of a few points.

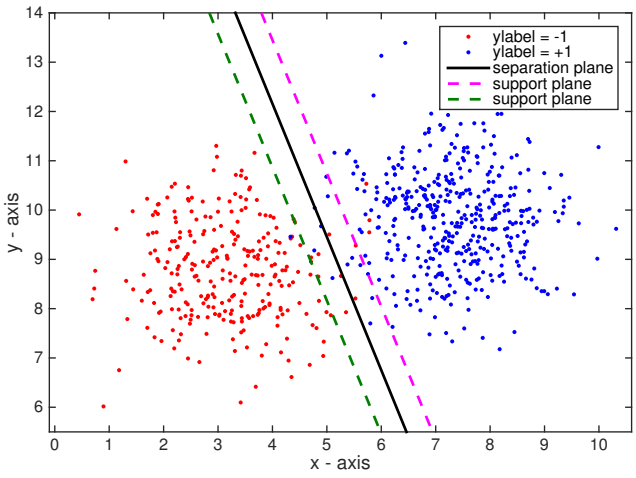


We can relax this problem into the following with some penalty constant $C > 0$:

$$\min_{\mathbf{w}\in\mathbb{R}^n,\eta\in\mathbb{R}^m_+}\left\{\frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{m}\eta_i \;\;|\;\; y_i(\mathbf{w}^T\mathbf{x}^{(i)}+b) \geq 1-\eta_i,\;\; \forall i = 1,\cdots,m\right\}.$$

This problem is again a quadratic program with the joint variable $(\mathbf{w},\eta) \in \mathbb{R}^{n+m}$.

Numerical example: We consider a dataset of (X,y) where $X \in \mathbb{R}^{711\times 2}$ represents observed measurements and $y \in \{-1,1\}$ provides the labels. We apply SVM method to solve this problem using $C = 1000$, we obtain the following result, where $b = -17.7893$, and $w = (2.0934, 0.7745)^T$.



8.4 Convex optimization

In this section, we discuss some basic solution methods to solve convex optimization problems and representative applications.

8.4.1 Mathematical formulations

We consider the following nonlinear programming problem:

$$f_0^* = \begin{cases} \min\limits_x \ f_0(x) \\ \text{s.t.} \ \ Ax = b \\ \qquad f_i(x) \le 0, \ \ i = 1, \cdots, l, \end{cases} \tag{8.11}$$

where f_i for $i = 0, \cdots, l$ are convex functions from $\mathbb{R}^n$ to $\mathbb{R}$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ are given. This problem is called a convex program.

- If there is no constraint, then we obtain an unconstrained convex problem

$$\min_{x\in\mathbb{R}^n} f_0(x).$$

- If we write the feasible domain $\mathscr{D} = \{x \in \mathbb{R}^n \mid Ax = b, \ f_i(x) \le 0, \ i = 1, \cdots n\}$, then $\mathscr{D}$ is a convex set in $\mathbb{R}^p$. We can rewrite the above constrained problem as

$$\min_{x\in\mathscr{D}} f_0(x).$$

8.4.1.1 Basic properties of convex programs

A convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is allowed to take the $+\infty$ as its values. In this case, $\mathrm{dom}(f) := \{x \in \mathbb{R}^n \mid f(x) < +\infty\}$ is called the domain of f (also called effective domain). If $f(x) > -\infty$ for all $x \in \mathrm{dom}(f)$, then we say that f is proper. The set $\mathrm{epi}(f) := \left\{(x,t) \in \mathbb{R}^{n+1} \mid f(x) \le t, \ x \in \mathrm{dom}(f), \ t \in \mathbb{R}\right\}$ is called the epigraph of f. If $\mathrm{epi}(f)$ is a closed set in $\mathbb{R}^{n+1}$, then we say that f is closed. For a proper, closed and convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, and $x \in \mathrm{dom}(f)$, we consider

$$\partial f(x) := \left\{w \in \mathbb{R}^n \mid f(y) \ge f(x) + w^T(y-x), \ \ \forall y \in \mathrm{dom}(f)\right\}.$$

If this is a nonempty set, then we say that f is subdifferentiable at x, and $\partial f(x)$ is called the subdifferential of f at x. Any $w \in \partial f(x)$ is called a subgradient of f at

x. If f is differentiable at x, then $\partial f(x) = \{\nabla f(x)\}$, which has a unique element, called the gradient of f at x.

- Any optimal solution x^* is a global optimal solution, i.e., $f_0(x^*) \leq f(x)$ for any $x \in \mathscr{D}$ (x is feasible).
- If the solution set of the problem is nonempty, then it is convex.

For a proper, closed and convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, we have

$$x^* = \arg\min_{x \in \mathbb{R}^n} f(x) \quad \text{if and only if} \quad 0 \in \partial f(x^*).$$

This optimality condition can be considered as a generalization of Fermat's rule.

8.4.1.2 Optimality condition

Let us consider the general convex optimization problem (8.11). We can write the Lagrange function associated with this problem as

$$\mathscr{L}(x,y,z) = f_0(x) + y^T(Ax-b) + \sum_{i=1}^{l} z_i f_i(x),$$

where $y \in \mathbb{R}^m$ and $z \in \mathbb{R}^l$ are two Lagrange multiplier vectors. The dual function is defined as

$$d(y,z) = \min_x \left\{ \mathscr{L}(x,y,z) = f_0(x) + y^T(Ax-b) + \sum_{i=1}^{l} z_i f_i(x) \right\}. \tag{8.12}$$

Clearly, this dual function is concave no matter what $f_i's$ are convex or nonconvex for $i = 0, \cdots, l$. The dual problem is define as

$$d^* = \max_{y \in \mathbb{R}^m, z \in \mathbb{R}^l_+} d(y,z). \tag{8.13}$$

A point (x^*, y^*, z^*) is called a saddle point of $\mathscr{L}$ if

$$\mathscr{L}(x^*,y,z) \leq \mathscr{L}(x^*,y^*,z^*) \leq \mathscr{L}(x,y^*,z^*), \ \ \forall x \in \mathbb{R}^n, \ y \in \mathbb{R}^m, z \in \mathbb{R}^l.$$

If $f_i's$ are differentiable, then we can write down the optimality condition, also called the KKT (Karush-Kuhn-Tucker) condition as

$$\begin{cases} \nabla f_0(x) + A^T y + \sum_{i=1}^{l} z_i \nabla f_i(x) = 0 \\ Ax = b \\ f_i(x) \leq 0, \ \ z_i \geq 0, \ \ f_i(x) z_i = 0, \ i = 1, \cdots, l \end{cases} \tag{8.14}$$

The conditions $f_i(x)z_i = 0$ for $i = 1, \cdots, l$ at the last line are called the complementarity slackness. Practically, solving this KKT condition remains challenging, but it provides an intermediate tool to study the properties of optimal solutions.

For general convex problem (8.11), the KKT condition is necessary and sufficient for (x^*, y^*, z^*) to be an optimal solution of (8.11) under the Slater condition:

$$\{x \in \mathbb{R}^n \mid Ax = b,\ f_i(x) < 0,\ \ i = 1, \cdots, l\} \neq \emptyset. \tag{8.15}$$

As special case, when $Ax = b$ is absent, then the Slater condition becomes: there exists $x \in \mathbb{R}^n$ such that $f_i(x) < 0$ for all $i = 1, \cdots, l$.

8.4.2 Concrete applications

8.4.2.1 Application 1: LASSO problem in statistics

Problem description: A linear regression model generates the response vector b using a linear model of the form $b = a^T x + \varepsilon$, where a is a regressor, and x is a vector of parameters in $\mathbb{R}^n$ (which is unknown), and ε is a Gaussian noise. We often obtain a set of data points $\{(a_i, b_i)\}$ with both the input $a_i \in \mathbb{R}^n$ and the response $b_i \in \mathbb{R}$ for $i = 1, \cdots, m$. In this case, the sum of squares of errors is computed by

$$E_n = \sum_{i=1}^{m} (a_i^T x - b_i)^2 = \|Ax - b\|^2,$$

where A is a matrix whose rows are a_i, and $b = (b_1, \cdots, b_m)^T$. The goal is

- to minimize this error, and
- to select as few as possible the number of parameters x_j for $j = 1, \cdots, n$.

Penalized LASSO (least absolute shrinkage and selection operator): We formulate this problem as

$$\min_x \frac{1}{2}\|Ax - b\|_2^2 + \lambda \sum_{j=1}^{n} |x_j|.$$

where $\lambda > 0$ is a given parameter to trade-off the number of nonzero entries in x and the error in $\frac{1}{2}\|Ax - b\|_2^2$.

8.4.2.2 Application 2: Robust linear programming

Problem description: We consider the following linear program:

$$\min_x z = c^T x \quad \text{s.t. } a_i^T x \leq b_i, \text{ for } i = 1, \cdots, m.$$

We assume that, for each $i = 1, \cdots, m$, the coefficient vector a_i is varying in a unit ball centered at a given vector $\bar{a}_i$ of radius 1. That is,

$$a_i \in B_1(\bar{a}_i) := \{a_i \in \mathbb{R}^n \mid \|a_i - \bar{a}_i\| \leq 1\}.$$

The goal is to solve this linear problem such that the optimal solution remains satisfying the constraints for all $a_i \in B_1(\bar{a}_i)$, while minimizing the objective z.

Problem formulation: The robust linear constraint $a_i^T x \leq b_i$ for all $a_i \in B_1(\bar{a}_i)$ becomes $\max_{a_i \in B_1(\bar{a}_i)} a_i^T x \leq b_i$. Solving the right-hand side directly, we get

$$\bar{a}_i^T x + \|x\| \leq b_i, \;\; i = 1, \cdots, m.$$

Hence, we can formulate the robust linear program as follows:

$$\min_x z = c^T x \quad \text{s.t. } a_i^T x + \|x\| \leq b_i, \text{ for } i = 1, \cdots, m.$$

One can check that this is a convex problem (Exercise).

8.4.3 Solution methods

8.4.3.1 Unconstrained problems with smooth objective function:

We consider the following unconstrained convex program:

$$f^* = \min_{x \in \mathbb{R}^n} f(x), \tag{8.16}$$

where f is a convex function and smooth (i.e., it is differentiable, and its gradient is continuous). The gradient ∇f is Lipschitz continuous. That is, there exists a constant $L_f \in [0, +\infty)$ such that $\|\nabla f(x) - \nabla f(y)\| \leq L_f \|x - y\|$ for all $x, y \in \text{dom}(f)$.

Gradient method: One of the simplest methods to solve (8.16) is the gradient method, which is an iterative method. It generates a sequence of vectors $\{x^k\}$ in $\mathbb{R}^n$ such that $\{f(x^k)\}$ converges to f^* the optimal value of the given problem (8.16).

Starting from an initial point $x^0 \in \mathbb{R}^n$, it updates x^{k+1} from the current x^k at each iteration k as

$$x^{k+1} = x^k - \alpha_k \nabla f(x^k),$$

where $\alpha_k \in (0, \frac{2}{L_f})$ is a given step-size. The optimal choice is $\alpha_k = \frac{1}{L_f}$.

Example - *Least-squares problem*: We consider the following least-squares problem

$$\min_x \frac{1}{2}\|Ax-b\|^2.$$

The objective function $f(x) = \frac{1}{2}\|Ax-b\|^2$ is convex and smooth. Its gradient is $\nabla f(x) = A^T(Ax-b)$, which is Lipschitz continuous with $L_f = \lambda_{\max}(A^TA)$, the maximum eigenvalue of A^TA.

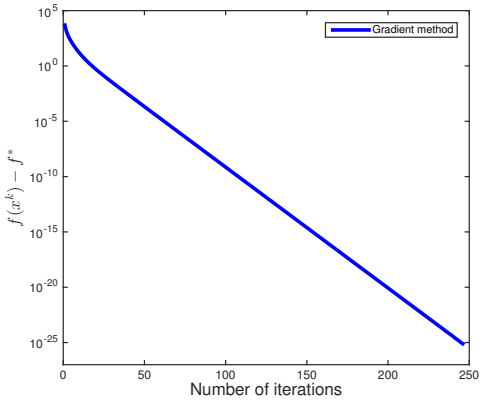
If we apply the above gradient method to solve this problem, we have

$$x^{k+1} = x^k - \frac{1}{\lambda_{\max}(A^TA)}A^T(Ax^k-b).$$

Example 8.5. Consider an example with

- $m = 50$, $n = 200$.
- `A = randn(m, n)`
- $b = Ax^\natural + \texttt{randn}(0.01)$, where $x^\natural$ is given.

The plot shows the decrease of the objective residual $f(x^k) - f^*$ along the iteration k.



8.4.3.2 Methods for solving convex programming

Methods for convex optimization vary from one class of problems to another. Below are some classes of methods.

- **Nonsmooth optimization methods:** Subgradient, bundle, mirror descent, and proximal methods are usually used to solve nonsmooth convex programs.
- **First-order methods:** Gradient, fast gradient, conditional gradient, coordinate descent, stochastic gradient descent methods using only function values and

gradients are called first order methods. These methods are widely used in large-scale applications nowadays (e.g., machine learning, deep learning, and data sciences).

- **Second order methods:** Newton and quasi-Newton methods are also used to solve convex programs which often have fast convergence speed. These methods are recently combined with stochastic methods to obtain scalable methods for solving large-scale applications in machine learning and statistical learning.
- **Interior-point methods:** Interior-point methods can be used to solve different classes of convex problems such as cone programing or geometric programming.

8.4.4 Software

Solvers for convex programming: There are several software packages for certain classes of convex programming problems. Below are few examples.

- **Conic programming:** LPs, QPs, cone programs can be solved by several interior-point solvers including SeDuMi, SDPT3, SDPA, and Mosek.
- **First-order methods:** TFOCS is an open-source software package based on first-order methods to solve different convex optimization problems. This package was developed by S. Becker using Matlab, and is available at http://cvxr.com/tfocs/http://cvxr.com/tfocs/http://c

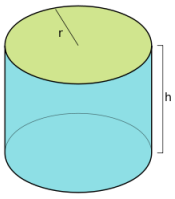
Modeling languages for convex programming:

There exist several modeling languages for convex optimization. The following two packages are perhaps widely used:

- **CVX:** This software package is implemented in Matlab by M. Grant and S. Boyd. Initially, it was open-source and is available for users with free-of-charge. It can be downloaded from http://cvxr.comhttp://cvxr.comhttp://cvxr.com.
- **YALMIP:** This software package is also implemented in Matlab by Johan Löfberg. It is also open-source and is available for users with free-of-charge. It can be downloaded from https://yalmip.github.iohttps://yalmip.github.iohttps://yalmip.github.io.

8.5 Exercises

Exercise 8.1. We are going to make an open-topped cylinder as given in the figure below. Here $r \geq 0$ is the radius of the bottom, and $h \geq 0$ is the height of the cylinder (in inches). The material using to make the bottom and the side area costs \$2 and



$1.5 per square inch, respectively. Assume that we would like to keep the volume of this cylinder at least 100 cubic inches.

(a) Formulate this problem as an optimization problem for finding r and h such that it minimizes the total material cost.

(b) Solve the optimization problem obtained in part (a) assuming that the volume is fixed at 100 cubic inches.

Exercise 8.2. (a) Given the following symmetric matrices:

$$Q_1 = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}, \quad Q_2 = \begin{bmatrix} 1 & 4 \\ 4 & 16 \end{bmatrix}, \quad Q_3 = \begin{bmatrix} 1 & 4 \\ 4 & 17 \end{bmatrix}, \quad Q_4 = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix}, \quad Q_5 = \begin{bmatrix} 6 & 8 & 8 \\ 8 & 6 & 2 \\ 8 & 2 & 10 \end{bmatrix}$$

Which of these matrices is: (i) positive semidefinite, (ii) positive definite, and (iii) not positive semidefinite? Prove your answer.

(b) Prove that for any matrix A in $\mathbb{R}^{m\times n}$, the matrix $Q = A^TA$ is symmetric and positive semidefinite.

(c) A function $f : \mathbb{R}^3 \to \mathbb{R}$ is defined as $f(x) = x_1^2 + 2x_2^2 + x_3^2 - 2x_1x_2 + x_1x_3 + 3x_2x_3 + 2x_1 + x_2 + 3x_3$. Show that we can write $f(x)$ as $f(x) = \frac{1}{2}x^TQx + q^Tx$, where Q is a 3×3 symmetric matrix, and q is a vector in $\mathbb{R}^3$. Is Q positive definite or positive semidefinite? Prove your answer.

Exercise 8.3. The following data is obtained as the rounded values $\bar{f}(t_i)$ from a cubic function $f(t) = a_0 + a_1t + a_2t^2 + a_3t^3$ at given time periods t_i for $i = 1, 2, \cdots, 10$. Since we do not know the coefficients a_0, a_1, a_2 and a_3, and we want to evaluate

Time t_i [in seconds]	1	2	3	4	5	6	7	8	9	10
$\bar{f}(t_i)$	120	140	160	180	200	220	240	260	280	301

them using the above data. Because the the data is rounded, we no longer have $f(t_i) = \bar{f}(t_i)$, but we instead have $f(t_i) \approx \bar{f}(t_i)$.

(a) Formulate this problem as an optimization problem by minimizing the sum of square of all the differences $f(t_i) - \bar{f}(t_i)$ for $i = 1, \cdots, 10$. Here, the vector of variables $x = (a_0, a_1, a_2, a_3)^T$ is in $\mathbb{R}^4$.

(b) Is it a quadratic program? If so, write down the matrix Q in the form $\frac{1}{2}x^T Qx + q^T x$, and show that Q is symmetric positive semidefinite.

(c) Solve this problem using Fermat's rule.

Exercise 8.4. A communication company plans to place 10 broadcast stations at given locations $a^{(1)}, \cdots, a^{(10)}$ in 10 regions. Each station has its own maximum *signal coverage area* of radius r_i (in feet). The radius r_i of each station i is at least $\underline{r}$ feet and at most $\bar{r}$ feet ($i = 1, \cdots, 10$). To cover each square ft of the *signal coverage area* of the i-th station, a cost of $\$c_i$ is incurred. The company can remotely control these stations by placing a central station at a location x. This location should be covered by all stations $a^{(i)}$ in their *signal coverage area*, but it must be at most r_0 feet far from the company's headquarter located at a given place $a^{(0)}$. The goal is to find the location x of the central station and the radius r_i of the i-th station to minimize the total cost. The input data is given as follows:

- The location of all stations is $a^{(1)} = (0,4)$, $a^{(2)} = (1,5)$, $a^{(3)} = (2,3)$, $a^{(4)} = (2,1)$, $a^{(5)} = (3,6)$, $a^{(6)} = (4,5)$, $a^{(7)} = (4,1)$, $a^{(8)} = (5,2)$, $a^{(9)} = (6,5)$ and $a^{(10)} = (7,4)$.
- The location of the headquarter is $a^{(0)} = (4,4)$, the upper and lower bounds of the radius are $r_0 = 1$, $\bar{r} = 5$, and $\underline{r} = 0.02$.
- The unit costs are $c = [1; 2; 1.2; 2.5; 2.1; 1.1; 1.8; 1.4; 1.35; 1.82]^T$.

The unit of the distances r_i, $\underline{r}$, $\bar{r}$, r_0 and the coordinates of $a^{(i)}$ are in 10^5 feet, and the unit cost c_i is in dollars. The distance is measured by the Euclidean distance, which is defined as $\|x-y\| = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2}$ for any two points $x = (x_1, x_2)$ and $y = (y_1, y_2)$ in $\mathbb{R}^2$.

(a) Formulate this problem into an optimization problem where the variables are the radius r_i and the location x of the central station. This cost excludes the cost of building the central station.

(b) Now, we replace each *signal coverage area* from a circle to a square, where the center of this square is $a^{(i)}$ and the length of its edges is $2r_i$ for $i = 0, \cdots, 10$. Re-

formulate the problem in part (a) into an optimization problem? Is it a quadratic program or a linear program? Show your result in detail.

References

1. G.H. Golub and C.F. van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 3rd edition, 1996.
2. Y. Nesterov and A. Nemirovski. *Interior-point Polynomial Algorithms in Convex Programming*. Society for Industrial Mathematics, 1994.
3. Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.
4. J. Renegar. *A mathematical view of interior-point methods in convex optimization*, volume 2. SIAM, 2001.
5. S.J. Wright. *Primal-Dual Interior-Point Methods*. SIAM Publications, Philadelphia, 1997.