CHAPTER 25

Convex Programming

In the last chapter, we saw that small modifications to the primal–dual interior-point algorithm allow it to be applied to quadratic programming problems as long as the quadratic objective function is convex. In this chapter, we shall go further and allow the objective function to be a general (smooth) convex function. In addition, we shall allow the feasible region to be any convex set given by a finite collection of convex inequalities.

1. Differentiable Functions and Taylor Approximations

In this chapter, all nonlinear functions will be assumed to be twice differentiable, and the second derivatives will be assumed continuous. We begin by reiterating a few definitions and results that were briefly touched on in Chapter 17. First of all, given a real-valued function f defined on a domain in $\mathbb{R}^n$, the vector

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

is called the *gradient* of f at x. The matrix

$$Hf(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(x) \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \frac{\partial^2 f}{\partial x_n \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{bmatrix}$$

is called the *Hessian* of f at x. In dimensions greater than one, the gradient and the Hessian are the analogues of the first and second derivatives of a function in one dimension. In particular, they appear in the three-term Taylor series expansion of f about the point x:

$$f(x + \Delta x) = f(x) + \nabla f(x)^T \Delta x + \frac{1}{2} \Delta x^T Hf(x) \Delta x + r_x(\Delta x).$$

The last term is called the remainder term. The value of this expansion lies in the fact that this remainder is small when Δx is small. To be precise, the remainder has the following property:

R.J. Vanderbei, *Linear Programming*, International Series in Operations Research & Management Science 196, DOI 10.1007/978-1-4614-7630-6_25,

$$\lim_{\Delta x \to 0} \frac{r_x(\Delta x)}{\|\Delta x\|^2} = 0.$$

This result follows immediately from the one-dimensional three-term Taylor series expansion applied to $g(t) = f(x + t\Delta x)$ and the chain rule (see Exercise 25.8).

2. Convex and Concave Functions

There are several equivalent definitions of convexity of a function. The definition that is most expedient for our purposes is the multidimensional generalization of the statement that a function is convex if its second derivative is nonnegative. Hence, we say that a real-valued function defined on a domain in $\mathbb{R}^n$ is *convex* if its Hessian is positive semidefinite everywhere in its domain. A function is called *concave* if its negation is convex.

3. Problem Formulation

We shall study convex optimization problems posed in the following form:

$$\begin{array}{ll} \text{minimize} & c(x) \\ \text{subject to} & a_i(x) \geq b_i, \qquad i = 1, 2, \ldots, m. \end{array}$$

Here, the real-valued function $c(\cdot)$ is assumed to be convex, and the m real-valued functions $a_i(\cdot)$ are assumed to be concave. This formulation is the natural extension of the convex quadratic programming problem studied in the previous chapter, except that we have omitted the nonnegativity constraints on the variables. This omission is only a matter of convenience since, if a given problem involves nonnegative variables, the assertion of their nonnegativity can be incorporated as part of the m nonlinear inequality constraints. Also note that once we allow for general concave inequality constraints, we can take the right-hand sides to be zero by simply incorporating appropriate shifts into the nonlinear constraint functions. Hence, many texts on convex optimization prefer to formulate the constraints in the form $a_i(x) \geq 0$. We have left the constants b_i on the right-hand side for later comparisons with the quadratic programming problem of the previous chapter. Finally, note that many convex and concave functions become infinite in places and therefore have a natural domain that is a strict subset of $\mathbb{R}^n$. This issue is important to address when solving practical problems, but since this chapter is just an introduction to convex optimization, we shall assume that all functions are finite on all of $\mathbb{R}^n$.

At times it will be convenient to use vector notation to consolidate the m constraints into a single inequality. Hence, we sometimes express the problem as

$$\begin{array}{ll} \text{minimize} & c(x) \\ \text{subject to} & A(x) \geq b, \end{array}$$

where $A(\cdot)$ is a function from $\mathbb{R}^n$ into $\mathbb{R}^m$ and b is a vector in $\mathbb{R}^m$. As usual, we let w denote the slack variables that convert the inequality constraints to equalities:

$$\begin{array}{ll} \text{minimize} & c(x) \\ \text{subject to} & A(x) - w = b \\ & \qquad\quad w \geq 0. \end{array}$$

4. Solution via Interior-Point Methods

In this section, we derive an interior-point method for convex programming problems. We start by introducing the associated barrier problem:

$$\begin{array}{ll} \text{minimize} & c(x) - \mu \sum_i \log w_i \\ \text{subject to} & a_i(x) - w_i = b_i, \qquad i = 1, 2, \ldots, m. \end{array}$$

The Lagrangian for this problem is given by

$$L(x, w, y) = c(x) - \mu \sum_i \log w_i + \sum_i y_i (b_i - a_i(x) + w_i).$$

Equating to zero the derivative of L with respect to each of its variables, we get the following set of first-order optimality conditions:

$$\begin{aligned} \frac{\partial L}{\partial x_j} &= \frac{\partial c}{\partial x_j}(x) - \sum_i y_i \frac{\partial a_i}{\partial x_j}(x) \quad &= 0, \qquad j = 1, 2, \ldots, n, \\ \frac{\partial L}{\partial w_i} &= -\frac{\mu}{w_i} + y_i \quad &= 0, \qquad i = 1, 2, \ldots, m, \\ \frac{\partial L}{\partial y_i} &= b_i - a_i(x) + w_i \quad &= 0, \qquad i = 1, 2, \ldots, m. \end{aligned}$$

The next step is to multiply the ith equation in the middle set by w_i and then replace x with $x + \Delta x$, y by $y + \Delta y$, and w by $w + \Delta w$ to get the following system:

$$\begin{aligned} \frac{\partial c}{\partial x_j}(x + \Delta x) - \sum_i (y_i + \Delta y_i) \frac{\partial a_i}{\partial x_j}(x + \Delta x) = 0, \qquad & j = 1, 2, \ldots, n, \\ -\mu + (w_i + \Delta w_i)(y_i + \Delta y_i) = 0, \qquad & i = 1, 2, \ldots, m, \\ b_i - a_i(x + \Delta x) + w_i + \Delta w_i = 0, \qquad & i = 1, 2, \ldots, m. \end{aligned}$$

Now we view this set of equations as a nonlinear system in the "delta" variables and linearize it by replacing each nonlinear function with its two-term Taylor series approximation. For example, $\partial c / \partial x_j (x + \Delta x)$ gets replaced with

$$\frac{\partial c}{\partial x_j}(x + \Delta x) \approx \frac{\partial c}{\partial x_j}(x) + \sum_k \frac{\partial^2 c}{\partial x_j \partial x_k}(x) \Delta x_k.$$

Similarly, $\partial a_i / \partial x_j (x + \Delta x)$ gets replaced with

$$\frac{\partial a_i}{\partial x_j}(x + \Delta x) \approx \frac{\partial a_i}{\partial x_j}(x) + \sum_k \frac{\partial^2 a_i}{\partial x_j \partial x_k}(x) \Delta x_k.$$

Writing the resulting linear system with the delta-variable terms on the left and everything else on the right, we get

$$\begin{aligned}\sum_k \left(-\frac{\partial^2 c}{\partial x_j \partial x_k} + \sum_i y_i \frac{\partial^2 a_i}{\partial x_j \partial x_k} \right) \Delta x_k + \sum_i \frac{\partial a_i}{\partial x_j} \Delta y_i &= \frac{\partial c}{\partial x_j} - \sum_i y_i \frac{\partial a_i}{\partial x_j} \\ y_i \Delta w_i + w_i \Delta y_i &= \mu - w_i y_i \\ \sum_k \frac{\partial a_i}{\partial x_k} \Delta x_k - \Delta w_i &= b_i - a_i + w_i.\end{aligned}$$

(Note that we have omitted the indication that the functions c, a_i, and their derivatives are to be evaluated at x.)

As usual, the next step is to solve the middle set of equations for the Δw_i's and then to eliminate them from the system. The reduced system then becomes

$$\begin{aligned}\sum_k \left(-\frac{\partial^2 c}{\partial x_j \partial x_k} + \sum_i y_i \frac{\partial^2 a_i}{\partial x_j \partial x_k} \right) \Delta x_k + \sum_i \frac{\partial a_i}{\partial x_j} \Delta y_i &= \frac{\partial c}{\partial x_j} - \sum_i y_i \frac{\partial a_i}{\partial x_j} \\ \sum_k \frac{\partial a_i}{\partial x_k} \Delta x_k + \frac{w_i}{y_i} \Delta y_i &= b_i - a_i + \frac{\mu}{y_i},\end{aligned}$$

and the equations for the Δw_i's are

$$\Delta w_i = -\frac{w_i}{y_i} \Delta y_i + \frac{\mu}{y_i} - w_i, \qquad i = 1, 2, \ldots, m.$$

At this point it is convenient to put the equations into matrix form. If we generalize our familiar gradient notation by letting $\nabla A(x)$ denote the $m \times n$ matrix whose (i, j)th entry is $\partial a_i / \partial x_j(x)$, then we can write the above system succinctly as follows:

(25.1)
$$\begin{bmatrix} -Hc(x) + \sum_i y_i H a_i(x) & \nabla A(x)^T \\ \nabla A(x) & WY^{-1} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \nabla c(x) - \nabla A(x)^T y \\ b - A(x) + \mu Y^{-1} e \end{bmatrix}.$$

Now that we have step directions, the algorithm is easy to describe—just compute step lengths that preserve strict positivity of the w_i's and the y_i's, step to a new point, and iterate.

5. Successive Quadratic Approximations

It is instructive to notice the similarity between the system given above and the analogous system for the quadratic programming problem posed in the analogous form (see Exercise 24.7). Indeed, a careful matching of terms reveals that the step directions derived here are exactly those that would be obtained if one were to form a certain quadratic approximation at the beginning of each iteration of the interior-point algorithm. Hence, the interior-point method can be thought of as a *successive quadratic programming algorithm*. In order to write this quadratic approximation neatly, let $\bar{x}$ and $\bar{y}$ denote the current primal and dual variables, respectively. Then the quadratic approximation can be written as

$$\begin{array}{ll} \text{minimize} & c(\bar{x}) + \nabla c(\bar{x})^T(x-\bar{x}) \; + \frac{1}{2}(x-\bar{x})^T Hc(\bar{x})(x-\bar{x}) \\ & \qquad\qquad\qquad\qquad - \frac{1}{2}(x-\bar{x})^T \left(\sum_i y_i Ha_i(\bar{x})\right)(x-\bar{x}) \\ \text{subject to} & A(\bar{x}) + \nabla A(\bar{x})(x-\bar{x}) \geq b. \end{array}$$

To verify the equivalence, we first observe that this problem is a quadratic program whose linear objective coefficients are given by

$$\nabla c(\bar{x}) - Hc(\bar{x})\bar{x} + \left(\sum_i y_i Ha_i(\bar{x})\right)\bar{x},$$

whose quadratic objective coefficients are given by

$$Hc(\bar{x}) - \sum_i y_i Ha_i(\bar{x}),$$

and whose right-hand side vector is given by

$$b - A(\bar{x}) + \nabla A(\bar{x})\bar{x}.$$

Substituting these expressions into the appropriate places in (24.8), we get (25.1).

Looking at the quadratic terms in the objective of the quadratic programming approximation, we see that the objective is convex, since we assumed at that start that c is convex, each a_i is concave, and the dual variables multiplying the Hessians of the constraint functions are all strictly positive.

6. Merit Functions

It is perhaps a little late to bring this up, but here's a small piece of advice: *always test your knowledge on the simplest possible example*. With that in mind, consider the following trivial convex optimization problem:

$$\text{minimize } \sqrt{1+x^2}.$$

This problem has no constraints. Looking at the graph of the objective function, which looks like a smoothed out version of $|x|$, we see that the optimal solution is $x^* = 0$. What could be easier! There are no y_i's nor any w_i's and equation (25.1) becomes just

$$-Hc(x)\Delta x = \nabla c(x),$$

where $c(x) = \sqrt{1+x^2}$. Taking the first and second derivatives, we get

$$\nabla c(x) = \frac{x}{\sqrt{1+x^2}} \quad \text{and} \quad Hc(x) = \frac{1}{(1+x^2)^{3/2}}.$$

Substituting these expressions into the equation for Δx and simplifying, we get that

$$\Delta x = -x(1+x^2).$$

Since there are no nonnegative variables that need to be kept positive, we can take unshortened steps. Hence, letting $x^{(k)}$ denote our current point and $x^{(k+1)}$ denote the next point, we have that

$$x^{(k+1)} = x^{(k)} + \Delta x = -(x^{(k)})^3.$$

That is, the algorithm says to start at any point $x^{(0)}$ and then replace this point with the negative of its cube, replace that with the negative of its cube, and so on.

The question is: does this sequence converge to zero? It is easy to see that the answer is yes if $|x^{(0)}| < 1$ but no otherwise. For example, if we start with $x^{(0)} = 1/2$, then the sequence of iterates is

k	$x^{(k)}$
0	0.50000000
1	−0.12500000
2	0.00195313
3	−0.00000001

If, on the other hand, we start at $x^{(0)} = 2$, then we get the following wildly divergent sequence:

k	$x^{(k)}$
0	2
1	−8
2	512
3	−134,217,728

Here is what goes wrong in this example. For problems without constraints, our algorithm has an especially simple description:

> From the current point, use the first three terms of a Taylor series expansion to make a quadratic approximation to the objective function. The next point is the minimum of this quadratic approximation function.

Figure 25.1 shows a graph of the objective function together with the quadratic approximation at $x^{(0)} = 2$. It is easy to see that the next iterate is at -8. Also, the further from zero that one starts, the more the function looks like a straight line and hence the further the minimum will be to the other side.

How do we remedy this nonconvergence? The key insight is the observation that the steps are always in the correct direction (i.e, a descent direction) but they are too long—we need to shorten them. A standard technique for shortening steps in situations like this is to introduce a function called a *merit function* and to shorten steps as needed to ensure that this merit function is always monotonically decreasing. For the example above, and in fact for any unconstrained optimization problem, we can use the objective function itself as the merit function. But, for problems with constraints, one needs to use something a little different from just the objective function. For example, one can use the logarithmic barrier function plus a constant times the square of the Euclidean norm of the infeasibility vector:

$$\Psi(x, w) := c(x) - \sum_i \log(w_i) + \beta \|b - A(x) + w\|^2.$$

Here, β is a positive real number. One can show that for β sufficiently large the step directions are always descent directions for this merit function.

A summary of the algorithm is shown in Figure 25.2.

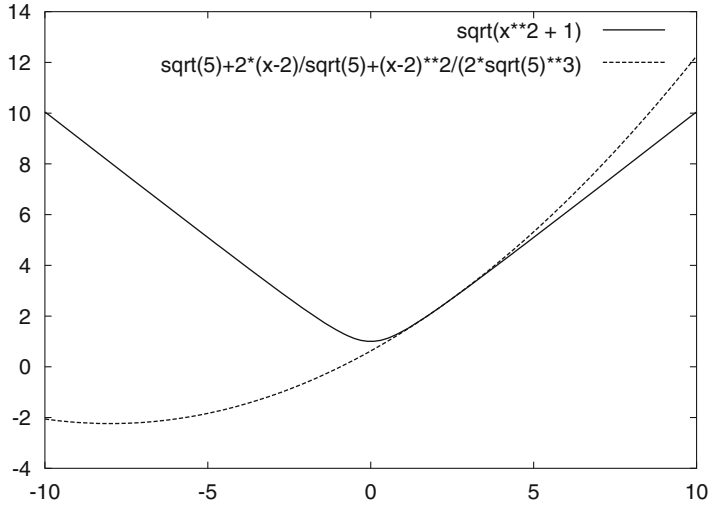

FIGURE 25.1. The function $c(x) = \sqrt{1+x^2}$ and its quadratic approximation at $x = 2$.

7. Parting Words

A story is never over, but every book must have an end. So, we stop here mindful of the fact that there are many interesting things left unsaid and topics unexplored. We hope we have motivated the reader to pursue the story further without our assistance—by reading other books and/or research papers and even perhaps making his or her own contributions. Cheers.

Exercises

25.1 *Piecewise Linear Approximation.* Given real numbers $b_1 < b_2 < \cdots < b_k$, let f be a continuous function on $\mathbb{R}$ that is linear on each interval $[b_i, b_{i+1}]$, $i = 0, 1, \ldots, k$ (for convenience we let $b_0 = -\infty$ and $b_{k+1} = \infty$). Such a function is called *piecewise linear* and the numbers b_i are called *breakpoints*. Piecewise linear functions are often used to approximate (continuous) nonlinear functions. The purpose of this exercise is to show how and why.

(a) Every piecewise linear function can be written as a sum of a constant plus a linear term plus a sum of absolute value terms:

$$f(x) = d + a_0 x + \sum_{i=1}^{k} a_i |x - b_i|.$$

Let c_i denote the slope of f on the interval $[b_i, b_{i+1}]$. Derive an explicit expression for each of the a_j's (including a_0) in terms of the c_i's.

initialize (x, w, y) so that $(w, y) > 0$

while (not optimal) {
  set up QP subproblem:
    $A = \nabla A(x)$
    $b = b - A(x) + \nabla A(x)x$
    $c = \nabla c(x) - Hc(x)x + (\sum_i y_i Ha_i(x))\, x$
    $Q = Hc(x) - \sum_i y_i Ha_i(x)$
  $\rho = b - Ax + w$
  $\sigma = c - A^T y + Qx$
  $\gamma = y^T w$
  $\mu = \delta \dfrac{\gamma}{n+m}$
  solve:
    $$\begin{bmatrix} -Q & A^T \\ A & Y^{-1}W \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} c - A^T y + Qx \\ b - Ax - \mu Y^{-1} e \end{bmatrix}$$
  $\Delta w = Y^{-1}(\mu e - YWe - W\Delta y)$
  $\theta = r \left(\max_{ij} \left\{ -\dfrac{\Delta x_j}{x_j}, -\dfrac{\Delta w_i}{w_i}, -\dfrac{\Delta y_i}{y_i} \right\} \right)^{-1} \wedge 1$
  do {
    $x^{\text{new}} = x + \theta \Delta x,$
    $w^{\text{new}} = w + \theta \Delta w$
    $y^{\text{new}} = y + \theta \Delta y$
    $\theta \leftarrow \theta/2$
  }while ($\Psi(x^{\text{new}}, w^{\text{new}}) \geq \Psi(x, w)$)
}

FIGURE 25.2. The path-following method for convex programming problems.

(b) In terms of the c_i's, give necessary and sufficient conditions for f to be convex.
(c) In terms of the a_j's, give necessary and sufficient conditions for f to be convex.
(d) Assuming that f is convex and is a term in the objective function for a linearly constrained optimization problem, derive an equivalent linear programming formulation involving at most k extra variables and constraints.
(e) Repeat the first four parts of this problem using $\max(x - b_i, 0)$ in place of $|x - b_i|$.

25.2 Let f be the function of 2 real variables defined by

$$f(x, y) = x^2 - 2xy + y^2.$$

Show that f is convex.

25.3 A function f of 2 real variables is called a *monomial* if it has the form

$$f(x, y) = x^m y^n$$

for some nonnegative integers m and n. Which monomials are convex?

25.4 Let ϕ be a convex function of a single real variable. Let f be a function defined on $\mathbb{R}^n$ by the formula

$$f(x) = \phi(a^T x + b),$$

where a is an n-vector and b is a scalar. Show that f is convex.

25.5 Which of the following functions are convex (assume that the domain of the function is all of $\mathbb{R}^n$ unless specified otherwise)?

(a) $4x^2 - 12xy + 9y^2$
(b) $x^2 + 2xy + y^2$
(c) $x^2 y^2$
(d) $x^2 - y^2$
(e) e^{x-y}
(f) $e^{x^2-y^2}$
(g) $\frac{x^2}{y}$ on $\{(x, y) : y > 0\}$

25.6 Given a symmetric square matrix A, the quadratic form $x^T Ax = \sum_{i,j} a_{ij} x_i x_j$ generalizes the notion of the square of a variable. The generalization of the notion of the fourth power of a variable is an expression of the form

$$f(x) = \sum_{i,j,k,l} a_{ijkl} x_i x_j x_k x_l.$$

The four-dimensional array of numbers $A = \{a_{ijkl} : 1 \leq i \leq n, 1 \leq j \leq n, 1 \leq k \leq n, 1 \leq l \leq n\}$ is called a *4-tensor*. As with quadratic expressions, we may assume that A is symmetric:

$$a_{ijkl} = a_{jkli} = \cdots = a_{lkij}$$

(i.e., given i, j, k, l, all $4! = 24$ premutations must give the same value for the tensor).

(a) Give conditions on the 4-tensor A to guarantee that f is convex.
(b) Suppose that some variables, say y_i's, are related to some other variables, say x_j's, in a linear fashion:

$$y_i = \sum_j f_{ij} x_j.$$

Express $\sum_i y_i^4$ in terms of the x_j's. In particular, give an explicit expression for the 4-tensor and show that it satisfies the conditions derived in part (a).

25.7 Consider the problem

$$\begin{array}{ll} \text{minimize} & ax_1 + x_2 \\ \text{subject to} & \sqrt{\epsilon^2 + x_1^2} \leq x_2. \end{array}$$

where $-1 < a < 1$.

(a) Graph the feasible set: $\left\{(x_1, x_2) : \sqrt{\epsilon^2 + x_1^2} \leq x_2\right\}$. Is the problem convex?

(b) Following the steps in the middle of p. 391 of the text, write down the first-order optimality conditions for the barrier problem associated with barrier parameter $\mu > 0$.

(c) Solve explicitly the first-order optimality conditions. Let $(x_1(\mu), x_2(\mu))$ denote the solution.

(d) Graph the central path, $(x_1(\mu), x_2(\mu))$, as μ varies from 0 to ∞.

25.8 *Multidimensional Taylor's series expansion.* Given a function $g(t)$ defined for real values of t, the three-term Taylor's series expansion with remainder is

$$g(t + \Delta t) = g(t) + g'(t)\Delta t + \frac{1}{2}g''(t)\Delta t^2 + r_t(\Delta t).$$

The remainder term satisfies

$$\lim_{\Delta t \to 0} \frac{r_t(\Delta t)}{\Delta t^2} = 0.$$

Let f be a smooth function defined on $\mathbb{R}^n$. Apply the three-term Taylor's series expansion to $g(t) = f(x + t\Delta x)$ to show that

$$f(x + \Delta x) = f(x) + \nabla f(x)^T \Delta x + \frac{1}{2}\Delta x^T H f(x) \Delta x + r_x(\Delta x).$$

25.9 Consider the following convex programming problem:

$$\begin{array}{ll} \text{minimize} & x_2 \\ \text{subject to} & x_1^2 + x_2^2 \leq 1. \end{array}$$

(a) Find the quadratic subproblem if the current primal solution is $(\bar{x}_1, \bar{x}_2) = (1/2, -2/3)$ and the current dual solution is $\bar{y} = 2$.

(b) Show that for arbitrary current primal and dual solutions, the feasible set for the convex programming problem is contained within the feasible set for the quadratic approximation.

Notes

Interior-point methods for nonlinear programming can be traced back to the pioneering work of Fiacco and McCormick (1968). For more on interior-point methods for convex programming, see Nesterov and Nemirovsky (1993) or den Hertog (1994).

The fact that the step directions are descent directions for the merit function Ψ is proved in Vanderbei and Shanno (1999).