3

Iterative Methods for Nonlinear Problems

In this chapter, we consider iterative methods for the solution of a variety of nonlinear problems. Examples include:

(a) The solution of systems of nonlinear equations that arise in many types of modeling, simulation, and engineering design problems.

(b) Optimization problems, including linear programming, that arise in a broad variety of engineering design, economic modeling, and operations research applications.

(c) Variational inequalities that can be viewed as generalizations of both systems of equations and constrained optimization problems. Variational inequalities can also be used as models of saddle point and other problems arising in the theory of games, and as models for equilibrium studies in diverse fields ranging from economics to traffic engineering.

Nonlinear problems are typically solved by iterative methods, and the convergence analysis of these methods is one of the focal points of this chapter. We use two principal techniques. The first relies on the theory of contraction mappings, and the second is based on showing iterative reduction of the cost function of an underlying optimization problem. Throughout the chapter, we emphasize algorithms that are well suited for parallelization such as methods of the Jacobi and Gauss–Seidel relaxation type. For

optimization problems, we discuss at length gradient projection methods and their scaled versions. We pay special attention to constraint sets that are Cartesian products and lend themselves to parallel calculations. We also develop some of the tools needed for the study of asynchronous parallel algorithms in Chapters 6 and 7.

An important aspect of convex constrained optimization problems is that they can be transformed into dual problems, which in many cases are easier to solve or are more amenable to parallel solution methods. Techniques based on duality, known as decomposition methods, have been widely used for the solution of large problems with special structure. These techniques are also particularly well suited for a parallel computing environment. We discuss a number of decomposition methods, and we delineate some problem structures that are well suited for their application.

In Section 3.1, we consider contraction mappings and associated fixed point problems and develop some broadly applicable tools. In Section 3.2, we study iterative algorithms for the solution of nonlinear optimization problems; these algorithms can be thought of as generalizations of the iterative methods for the solution of linear equations that were presented in Chapter 2. Then, in Section 3.3, we consider constrained optimization problems, with an emphasis on the problem of minimizing a cost function over a convex set. In Section 3.4, we discuss the use of duality transformations of optimization problems to enhance the parallelization of their solution. Finally, in Section 3.5, we consider algorithms for the solution of variational inequalities. Throughout, we comment on the potential for parallelization of the different methods.

3.1 CONTRACTION MAPPINGS

Several iterative algorithms can be written as

$$x(t+1) = T\big(x(t)\big), \qquad t = 0, 1, \ldots, \tag{1.1}$$

where T is a mapping from a subset X of $\Re^n$ into itself and has the property

$$\|T(x) - T(y)\| \leq \alpha \|x - y\|, \qquad \forall x, y \in X. \tag{1.2}$$

Here $\|\cdot\|$ is some norm, and α is a constant belonging to $[0, 1)$. Such a mapping is called a *contraction mapping*, or simply a *contraction*, and iteration (1.1) is called a *contracting iteration*. The scalar α is called the *modulus* of T. A mapping $T : X \mapsto Y$, where $X, Y \subset \Re^n$, that satisfies Eq. (1.2), will also be called a contraction mapping, even if $X \neq Y$.

Let there be given a mapping $T : X \mapsto X$. Any vector $x^* \in X$ satisfying $T(x^*) = x^*$ is called a *fixed point* of T and the iteration $x := T(x)$ can be viewed as an algorithm for finding such a fixed point. The reason is that if the sequence $\{x(t)\}$ converges to some $x^* \in X$ and T is continuous at x^*, then x^* is a fixed point of T. We notice that contraction mappings are automatically continuous.

As an alternative to the contraction assumption (1.2), we will sometimes assume that a mapping $T : X \mapsto X$ has a fixed point $x^* \in X$ and the property

$$\|T(x) - x^*\| \leq \alpha \|x - x^*\|, \qquad \forall x \in X, \tag{1.3}$$

where α, called the *modulus* of T, is again a constant belonging to $[0, 1)$. Clearly, inequality (1.3) is weaker than the contraction condition (1.2). Any mapping T with the above properties will be called a *pseudocontraction* and the corresponding iteration $x := T(x)$ will be called a *pseudocontracting iteration.* (Pseudocontracting iterations will play an important role in the analysis of certain algorithms in Section 3.5.) Notice that the existence of a fixed point is part of the definition of a pseudocontraction and that a pseudocontraction is not necessarily continuous. Figure 3.1.1 shows an example of a contraction and a pseudocontraction.

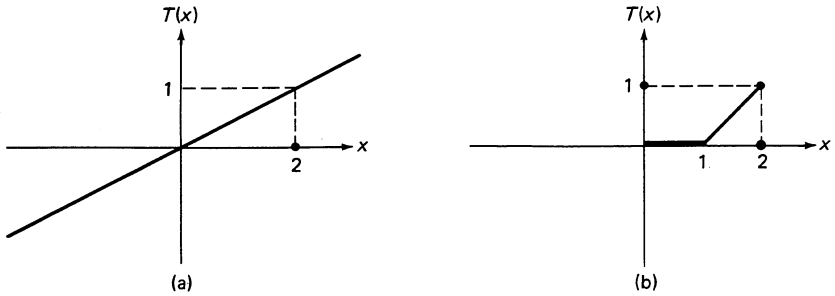

Figure 3.1.1 Illustration of a contraction and a pseudocontraction. (a) The mapping $T : \Re \mapsto \Re$ defined by $T(x) = x/2$ is a contraction with modulus $1/2$, and the iteration $x := T(x)$ converges to zero, which is a fixed point of T. (b) The mapping $T : [0, 2] \mapsto [0, 2]$ defined by $T(x) = \max\{0, x - 1\}$ is not a contraction since $|T(2) - T(1)| = 1$. On the other hand, it has a unique fixed point, equal to zero, and is a pseudocontraction because it is easily seen that $T(x) \leq x/2$ for every $x \in [0, 2]$.

A mapping T could be a contraction (or a pseudocontraction) for some choice of the vector norm $\|\cdot\|$ and, at the same time, fail to be a contraction (respectively, a pseudocontraction) under a different choice of norm. Thus, the proper choice of a norm is critical. Some particularly interesting norms, for our purposes, are the weighted maximum norms. We have already seen an interesting class of mappings that are contractions with respect to a weighted maximum norm: Corollary 6.1 in Section 2.6 shows that a nonnegative matrix M has the property $\rho(M) < 1$ if and only if it is a contraction mapping with respect to some weighted maximum norm.

3.1.1 General Results

The following basic result shows that contraction mappings have a unique fixed point and the corresponding iteration $x := T(x)$ converges to it.

Proposition 1.1. (*Convergence of Contracting Iterations*) Suppose that $T : X \mapsto X$ is a contraction with modulus $\alpha \in [0, 1)$ and that X is a closed subset of $\Re^n$. Then:

(a) (*Existence and Uniqueness of Fixed Points*) The mapping T has a unique fixed point $x^* \in X$.

(b) (*Geometric Convergence*) For every initial vector $x(0) \in X$, the sequence $\{x(t)\}$ generated by $x(t+1) = T\big(x(t)\big)$ converges to x^* geometrically. In particular,

$$\|x(t) - x^*\| \leq \alpha^t \|x(0) - x^*\|, \qquad \forall t \geq 0.$$

Proof.

(a) Fix some $x(0) \in X$ and consider the sequence $\{x(t)\}$ generated by $x(t+1) = T\big(x(t)\big)$. We have, from inequality (1.2),

$$\|x(t+1) - x(t)\| \leq \alpha \|x(t) - x(t-1)\|,$$

for all $t \geq 1$, which implies

$$\|x(t+1) - x(t)\| \leq \alpha^t \|x(1) - x(0)\|, \qquad \forall t \geq 0.$$

It follows that for every $t \geq 0$ and $m \geq 1$, we have

$$\begin{aligned}\|x(t+m) - x(t)\| &\leq \sum_{i=1}^{m} \|x(t+i) - x(t+i-1)\| \\ &\leq \alpha^t (1 + \alpha + \cdots + \alpha^{m-1}) \|x(1) - x(0)\| \leq \frac{\alpha^t}{1-\alpha} \|x(1) - x(0)\|.\end{aligned}$$

Therefore, $\{x(t)\}$ is a Cauchy sequence and must converge to a limit x^* (Prop. A.5 in Appendix A). Furthermore, since X is closed, x^* belongs to X. We have for all $t \geq 1$,

$$\|T(x^*) - x^*\| \leq \|T(x^*) - x(t)\| + \|x(t) - x^*\| \leq \alpha \|x^* - x(t-1)\| + \|x(t) - x^*\|$$

and since $x(t)$ converges to x^*, we obtain $T(x^*) = x^*$. Therefore, the limit x^* of $x(t)$ is a fixed point of T. It is a unique fixed point because if y^* were another fixed point, we would have

$$\|x^* - y^*\| = \|T(x^*) - T(y^*)\| \leq \alpha \|x^* - y^*\|$$

which implies that $x^* = y^*$.

(b) We have

$$\|x(t') - x^*\| = \big\|T\big(x(t'-1)\big) - T(x^*)\big\| \leq \alpha \|x(t'-1) - x^*\|,$$

for all $t' \geq 1$, so by applying this relation successively for $t' = t, t-1, \ldots, 1$, we obtain the desired result. **Q.E.D.**

We now show that the convergence result of the above proposition remains valid for pseudocontractions as well.

Proposition 1.2. (*Convergence of Pseudocontracting Iterations*) Suppose that $X \subset \Re^n$ and that the mapping $T : X \mapsto X$ is a pseudocontraction with a fixed point $x^* \in X$ and modulus $\alpha \in [0, 1)$. Then, T has no other fixed points and the sequence $\{x(t)\}$ generated by $x(t+1) = T\big(x(t)\big)$ satisfies

$$\|x(t) - x^*\| \leq \alpha^t \|x(0) - x^*\|, \qquad \forall t \geq 0,$$

for every choice of the initial vector $x(0) \in X$. In particular, the sequence $\{x(t)\}$ converges to x^*.

Proof. Uniqueness of the fixed point follows as in the proof of Prop. 1.1. Now notice that the pseudocontraction condition (1.3) implies that

$$\|x(t) - x^*\| = \big\|T\big(x(t-1)\big) - x^*\big\| \leq \alpha \|x(t-1) - x^*\|,$$

for every $t \geq 1$, and the desired result follows by induction on t. **Q.E.D.**

In order to apply a result such as Prop. 1.2, we often have to show that the mapping T has a fixed point. In some cases, an existence result is obtained from purely topological considerations. The following result, illustrated in Fig. 3.1.2, generalizes the Brouwer Fixed Point Theorem that was used in Section 2.6. Its proof is beyond the scope of this book (see e.g. [DuS63]).

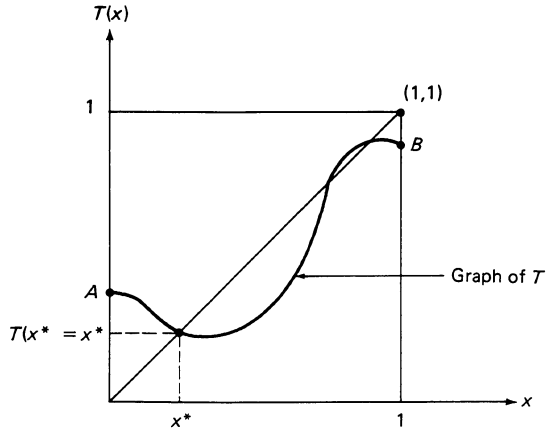

Figure 3.1.2 Illustration of the fixed point theorem (Prop. 1.3) for the case where T maps the unit interval $[0, 1]$ into itself. Here, point $A = \big(0, T(0)\big)$ lies on or above the diagonal of the unit square and point $B = \big(1, T(1)\big)$ lies on or below the diagonal. If T is continuous, its graph must cross the diagonal at some point (x^*, x^*) and such an x^* is a fixed point of T.

Proposition 1.3. (*Leray–Schauder–Tychonoff Fixed Point Theorem*) If $X \subset \Re^n$ is nonempty, convex, and compact, and if $T : X \mapsto X$ is a continuous mapping, then there exists some $x^* \in X$ such that $T(x^*) = x^*$.

The iteration $x := T(x)$ can be implemented in parallel in the obvious manner (see Subsection 1.2.4). However, the parallelization can be wasteful if the mapping T is such that the updating of different components involves a substantial amount of common computations. This issue will be raised, in more specific contexts, in later sections. Efficient parallel implementations are often possible in the case where the set X is a Cartesian product of lower dimensional sets, which we study next.

3.1.2 Contractions Over Cartesian Product Sets

Throughout this subsection, we assume that $X = \prod_{i=1}^m X_i$, where each X_i is a nonempty subset of $\Re^{n_i}$, and where $n_1 + \cdots + n_m = n$. Accordingly, any vector $x \in X$ is decomposed as $x = (x_1, \ldots, x_m)$, with $x_i \in X_i$. We also assume that we are given a norm $\|\cdot\|_i$ on $\Re^{n_i}$ for each i, and that $\Re^n$ is endowed with the norm

$$\|x\| = \max_i \|x_i\|_i, \tag{1.4}$$

which we call a *block–maximum norm.*

Let $T : X \mapsto X$ be a contraction with modulus α, under the above introduced block–maximum norm. Such a mapping will be called a *block–contraction.* Let $T_i : X \mapsto X_i$ be the ith (block)–component of T, that is,

$$T(x) = \big(T_1(x), \ldots, T_m(x)\big).$$

Notice that

$$\|T_i(x) - T_i(y)\|_i \leq \max_j \|T_j(x) - T_j(y)\|_j = \|T(x) - T(y)\| \leq \alpha\|x - y\|, \qquad \forall x, y \in X, \ \forall i. \tag{1.5}$$

Gauss–Seidel Methods

Applying the mapping T, as in the iteration $x(t+1) = T\big(x(t)\big)$, corresponds to updating all components of x simultaneously. A Gauss–Seidel mode of implementation is also possible, whereby the block–components of x are updated one at a time. Due to the assumption that X is a Cartesian product, such a Gauss–Seidel iteration maps the set X into itself and the algorithm is well–defined. We now present a precise description and a proof of convergence of Gauss–Seidel iterations.

The mapping $\hat{T}_i : X \mapsto X$, corresponding to an update of the ith block–component only, is given by

$$\hat{T}_i(x) = \hat{T}_i(x_1, \ldots, x_m) = \big(x_1, \ldots, x_{i-1}, T_i(x), x_{i+1}, \ldots, x_m\big).$$

(The fact that $\hat{T}_i$ maps X into itself is a key consequence of the Cartesian product assumption.) Updating all the block–components of x, one at a time in increasing order, is equivalent to applying the mapping $S : X \mapsto X$, defined by

$$S = \hat{T}_m \circ \hat{T}_{m-1} \circ \cdots \circ \hat{T}_1,$$

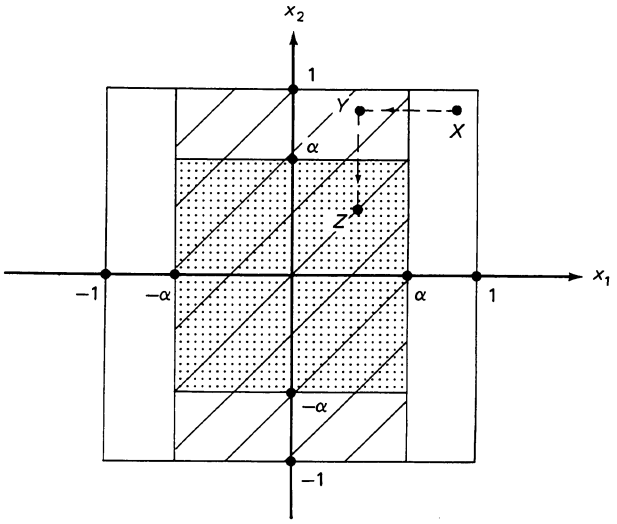

Figure 3.1.3 Illustration of Gauss–Seidel convergence for block–contracting iterations (cf. Prop. 1.4). Let $T : \Re^2 \mapsto \Re^2$ be a contraction, with respect to the maximum norm, with a fixed point $x^* = 0$ and modulus $\alpha < 1$. Consider one iteration of the Gauss–Seidel algorithm, starting from some x such that $\|x\|_\infty \leq 1$. The update of the first component leads to the vector $y = \big(T_1(x), x_2\big)$ that belongs to the shaded region $[-\alpha, \alpha] \times [-1, 1]$. The update of the second component leads to the vector $z = S(x) = \big(y_1, T_2(y)\big) = \big(T_1(x), T_2\big(T_1(x), x_2\big)\big)$, which belongs to the dotted region $[-\alpha, \alpha] \times [-\alpha, \alpha]$. In particular, $\|z\|_\infty \leq \alpha$.

where $\circ$ denotes composition. An equivalent definition of S is given by the equation

$$S_i(x) = T_i\big(S_1(x), \ldots, S_{i-1}(x), x_i, \ldots, x_m\big), \tag{1.6}$$

where $S_i : X \mapsto X_i$ is the ith block–component of S. It is seen that any fixed point of T is also a fixed point of S, and conversely. The mapping S will be called the *Gauss–Seidel mapping based on the mapping* T and the iteration $x(t+1) = S\big(x(t)\big)$ will be called the *Gauss–Seidel algorithm based on the mapping* T.

Proposition 1.4. (*Convergence of Gauss–Seidel Block-Contracting Iterations*) If $T : X \mapsto X$ is a block–contraction, then the Gauss–Seidel mapping S is also a block–contraction, with the same modulus as T. In particular, if X is closed, the sequence of vectors generated by the Gauss–Seidel algorithm based on the mapping T converges to the unique fixed point of T geometrically.

Proof. We use the definition of S [Eq. (1.6)] and the block–contraction assumption [inequality (1.5), in particular] to obtain for every $x, y \in X$

$$\|S_i(x) - S_i(y)\|_i \leq \alpha \max \left\{ \max_{j<i} \|S_j(x) - S_j(y)\|_j \ , \ \max_{j \geq i} \|x_j - y_j\|_j \right\}.$$

A simple induction on i yields $\|S_i(x) - S_i(y)\|_i \leq \alpha \max_j \|x_j - y_j\|_j = \alpha \|x - y\|$ for all i. This proves that S is a block–contraction and the rest follows from the convergence result for contracting iterations (Prop. 1.1). **Q.E.D.**

Proposition 1.4 is illustrated in Fig. 3.1.3. The following result provides a generalization to the case of pseudocontractions.

Proposition 1.5. (*Convergence of Gauss–Seidel Block-Pseudocontracting Iterations*) If a mapping $T : X \mapsto X$ has a fixed point x^* and is a pseudocontraction of modulus α with respect to a block–maximum norm $\|\cdot\|$, then the same is true for the Gauss–Seidel mapping S, that is,

$$\|S(x) - x^*\| \leq \alpha\|x - x^*\|, \qquad \forall x \in X.$$

In particular, the sequence generated by the Gauss–Seidel algorithm based on the mapping T converges to x^* geometrically.

Proof. The proof of the inequality $\|S(x)-x^*\| \leq \alpha\|x-x^*\|$ is the same as for the case of block–contractions (Prop. 1.4), provided that we replace y by x^*. Convergence of the Gauss–Seidel algorithm follows from the convergence result for pseudocontracting iterations (Prop. 1.2), applied to the mapping S. **Q.E.D.**

Component Solution Methods

We now investigate an alternative approach for finding a fixed point of T. We are looking for a solution of the system of equations $x = T(x)$. This system can be decomposed into m smaller systems of equations of the form

$$x_i = T_i(x_1, \ldots, x_m), \qquad i = 1, \ldots, m, \tag{1.7}$$

which have to be solved simultaneously. We will consider an algorithm that solves at each iteration the ith equation in the system (1.7) for x_i, while keeping the other components fixed. There is no established terminology for describing such an algorithm and we will be referring to it as the *component solution method.*

To be more specific, we let $R_i(x)$ be the set of all solutions of the ith equation in the system (1.7), defined by

$$R_i(x) = \Big\{ y_i \in X_i \;\Big|\; y_i = T_i(x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_m) \Big\}. \tag{1.8}$$

The method proceeds as follows. Given a vector $x(t) \in X$, the ith block–component $x_i(t+1)$ of the next vector is chosen to be a solution of the ith equation in the system (1.7), that is,

$$x_i(t+1) \in R_i\big(x(t)\big).$$

The following result shows that $x_i(t+1)$ is uniquely defined if T is a block–contraction.

Proposition 1.6. Suppose that X is closed and that $T : X \mapsto X$ is a block–contraction. Then the set $R_i(x)$ has exactly one element for each i and for each $x \in X$.

Proof. Fix some i and some $x \in X$, and consider the mapping $T_i^x : X_i \mapsto X_i$ defined by

$$T_i^x(y_i) = T_i(x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_m). \tag{1.9}$$

Notice that $R_i(x)$ is, by definition, equal to the set of fixed points of T_i^x. By the block–contraction assumption [cf. inequality (1.5)], we have

$$\|T_i^x(y_i) - T_i^x(z_i)\|_i \leq \alpha \|y_i - z_i\|_i, \qquad \forall y_i, z_i \in X_i$$

and T_i^x is a contraction. The conclusion that $R_i(x)$ is a singleton follows from the existence and uniqueness result for fixed points of contraction mappings (Prop. 1.1). **Q.E.D.**

Assuming that each $R_i(x)$ is a singleton, we define a mapping $Q_i : X \mapsto X_i$ by letting $Q_i(x)$ be equal to the unique element of $R_i(x)$. We then define a mapping $Q : X \mapsto X$ by letting

$$Q(x) = \big(Q_1(x), \ldots, Q_m(x)\big)$$

(see Fig. 3.1.4). The component solution method is then described by

$$x(t+1) = Q\big(x(t)\big), \qquad t = 0, 1, \ldots. \tag{1.10}$$

In this iteration, all block–components of x are updated simultaneously. Alternatively, we could use the Gauss–Seidel algorithm based on the mapping Q, in which the block–components of x are updated one at a time (see Fig. 3.1.5). This will be called the *Gauss–Seidel component solution method.*

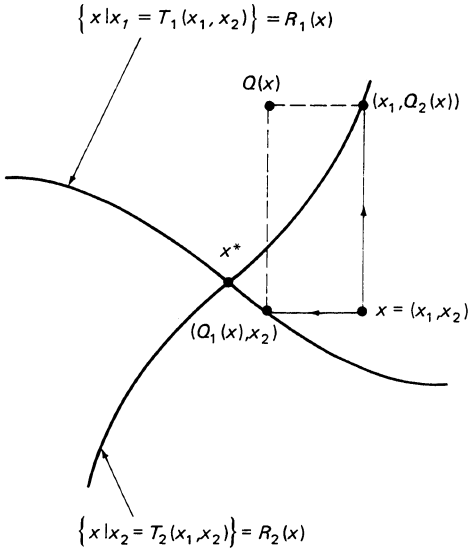

Figure 3.1.4 Illustration of the mapping Q used in the component solution method. Each curve is the set of points where the equation $x_1 = T_1(x_1, x_2)$ [respectively, $x_2 = T_2(x_1, x_2)$] is satisfied. At their intersection x^*, both equations are satisfied and x^* is a fixed point of T. The mapping Q_i corresponds to updating x_i so as to satisfy the ith equation while the other component is fixed.

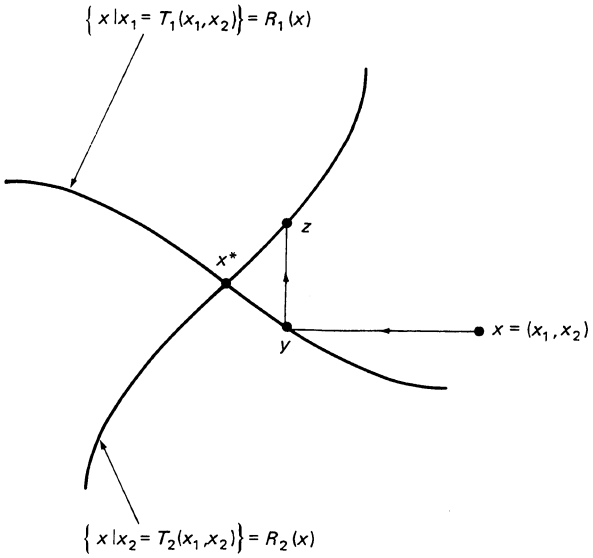

Figure 3.1.5 Illustration of the Gauss–Seidel component solution method. Starting from the vector $x = (x_1, x_2)$, an update of the first component leads to the point $y = \big(Q_1(x), x_2\big)$ and an update of the second component leads to the point $z = \big(y_1, Q_2(y)\big) = \big(Q_1(x), Q_2\big(Q_1(x), x_2\big)\big)$.

Convergence of the component solution method (1.10) and its Gauss–Seidel variant is obtained because Q inherits the contraction property of T, as shown next.

Proposition 1.7. (*Convergence of Component Solution Methods for Block-Contractions*) If $T : X \mapsto X$ is a block–contraction, then Q is also a block–contraction with the same modulus as T. In particular, if X is closed, then the component solution method $x(t+1) = Q\big(x(t)\big)$, as well as the Gauss–Seidel algorithm based on Q, converge to the unique fixed point of T geometrically.

Proof. Let $x = (x_1, \ldots, x_m)$ and $y = (y_1, \ldots, y_m)$. By the definition of Q_i, we have $Q_i(x) \in R_i(x)$ and $Q_i(y) \in R_i(y)$. Therefore,

$$Q_i(x) = T_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big),$$

and

$$Q_i(y) = T_i\big(y_1, \ldots, y_{i-1}, Q_i(y), y_{i+1}, \ldots, y_m\big).$$

Using the block–contraction assumption for T [inequality (1.5)], we obtain

$$\|Q_i(x) - Q_i(y)\|_i \leq \alpha \max\Big\{\|Q_i(x) - Q_i(y)\|_i,\ \max_{j \neq i} \|x_j - y_j\|_j\Big\}.$$

Since $\alpha < 1$, it follows that

$$\|Q_i(x) - Q_i(y)\|_i \leq \alpha \max_{j \neq i} \|x_j - y_j\|_j \leq \alpha \, \|x - y\|.$$

Since this is true for each i, we obtain $\|Q(x) - Q(y)\| \leq \alpha \|x - y\|$, which shows that Q is also a block–contraction. Assuming that X is closed, Q has a unique fixed point x^*. The equation $x^* = Q(x^*)$ is equivalent to $\{x_i^*\} = R_i(x^*)$ for each i. Using the definition of $R_i(x^*)$, we conclude that $x^* = T(x^*)$ and x^* is the unique fixed point of T. The result follows because contracting iterations and their Gauss–Seidel variants are guaranteed to converge geometrically (Props. 1.1 and 1.4). **Q.E.D.**

We now consider the case where T is a pseudocontraction with fixed point x^*. The major difference here is that there is no guarantee that the set $R_i(x)$ is a singleton; consequently, the mapping Q is not, in general, well-defined. In particular, $R_i(x)$ could be empty, in which case, the algorithm breaks down. It is also possible that $R_i(x)$ has many elements; in this case, our approach will be to assume that $Q_i(x)$ is chosen among the elements of $R_i(x)$ by means of some rule that we leave unspecified. The following result provides an easily verifiable condition for the sets $R_i(x)$ to be nonempty.

Proposition 1.8. Suppose that the mapping $T : X \mapsto X$ is continuous and a pseudocontraction with respect to a block–maximum norm $\|\cdot\|$. If each set X_i is closed and convex, then the set $R_i(x)$ is nonempty for each i and for each $x \in X$.

Proof. Let x^* and α be the fixed point and the modulus of T, respectively. As in the proof of Prop. 1.6 we consider the mapping $T_i^x : X_i \mapsto X_i$, defined by

$$T_i^x(y_i) = T_i\big(x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_m\big).$$

Fix some i and some $x \in X$. Consider the set

$$Y_i = \Big\{ y_i \;\Big|\; \|y_i - x_i^*\|_i \leq \max_{j \neq i} \|x_j - x_j^*\|_j \Big\} \cap X_i.$$

Notice that for every $y_i \in Y_i$, we have

$$\|T_i^x(y_i) - x_i^*\|_i \leq \alpha \max\Big\{ \|y_i - x_i^*\|_i, \; \max_{j \neq i} \|x_j - x_j^*\|_j \Big\} \leq \max_{j \neq i} \|x_j - x_j^*\|_j,$$

which shows that $T_i^x(y_i) \in Y_i$. The set Y_i is closed and convex because it is the intersection of two closed and convex sets. Furthermore, Y_i is bounded and is therefore compact. Finally, T_i^x is continuous and the Leray–Schauder–Tychonoff Fixed Point Theorem (Prop. 1.3) shows that T_i^x has a fixed point. From the definition of T_i^x, such a fixed point is an element of $R_i(x)$. **Q.E.D.**

The following result is an analog of Prop. 1.7.

Proposition 1.9. (*Convergence of Component Solution Methods for Block-Pseudocontractions*) Suppose that the mapping $T : X \mapsto X$ has a fixed point x^* and is a

pseudocontraction with respect to a block–maximum norm $\|\cdot\|$. Suppose that for every i and $x \in X$, the set $R_i(x)$ is nonempty. Then, the mapping Q is also a pseudocontraction, with respect to the same norm, and x^* is its unique fixed point. In particular, the sequence $\{x(t)\}$ generated by the component solution method $x(t+1) = Q\big(x(t)\big)$, or by the Gauss–Seidel algorithm based on Q, converges to x^* geometrically.

Proof. Since $Q_i(x) \in R_i(x)$, we have

$$Q_i(x) = T_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big).$$

Using the pseudocontraction assumption on T, we obtain

$$\|Q_i(x) - x_i^*\|_i \leq \alpha \max\Big\{\|Q_i(x) - x_i^*\|_i, \max_{j \neq i} \|x_j - x_j^*\|_j\Big\},$$

where $\alpha \in [0, 1)$ is the modulus of T. It follows that $\|Q(x) - x^*\| \leq \alpha\|x - x^*\|$ for every $x \in X$. In particular, x^* is the unique fixed point of Q and Q is a pseudocontraction. The desired conclusions follow from the convergence result for pseudocontracting iterations (Prop. 1.2) and their Gauss–Seidel variants (Prop. 1.5), applied to the mapping Q. **Q.E.D.**

Results similar to those proved so far can also be obtained if T is a *monotone* mapping, that is, if T satisfies $T(x) \leq T(y)$ for every x, y such that $x \leq y$ (Exercise 1.4).

3.1.3 Some Useful Contraction Mappings

We assume again that $X \subset \Re^n$ and that X is decomposed as a Cartesian product of lower dimensional sets $X_i \subset \Re^{n_i}$, $i = 1, \ldots, m$. We consider a mapping $T : X \mapsto \Re^n$, whose ith block–component T_i is of the form

$$T_i(x) = x_i - \gamma G_i^{-1} f_i(x). \tag{1.11}$$

Here each f_i is a function from $\Re^n$ into $\Re^{n_i}$, γ is some scalar, and G_i is an invertible symmetric matrix of dimensions $n_i \times n_i$. Mappings of this form are very common in iterative methods for optimization and solution of systems of equations or variational inequalities, and they will keep recurring in subsequent sections of this chapter. We collect here certain sufficient conditions for such mappings to be block–contractions. These conditions will be invoked in later sections in order to establish the convergence of certain iterative methods.

The general nature of the conditions to be considered is best illustrated in the simple case where $X = \Re^n$ and $n_i = 1$, $G_i = 1$ for each i, and the mapping f has the form $f(x) = Ax$, where A is an $n \times n$ matrix. We then have

$$T(x) = x - \gamma Ax = (I - \gamma A)x,$$

which is reminiscent of iterative algorithms for linear equations [see Section 2.4, Eq. (4.9), for example]. The mapping T is a contraction with respect to the maximum norm $\|\cdot\|_\infty$ if and only if $\|I - \gamma A\|_\infty < 1$. From the formula for the maximum norm (Prop. A.13 in Appendix A), an equivalent condition is

$$\max_i \Big\{ |1 - \gamma a_{ii}| + \sum_{j \neq i} |\gamma a_{ij}| \Big\} < 1. \tag{1.12}$$

Assuming that γ is positive and small enough so that $\gamma |a_{ii}| \leq 1$ for each i, the expression in Eq. (1.12) is equal to

$$\max_i \Big\{ 1 - \gamma a_{ii} + \gamma \sum_{j \neq i} |a_{ij}| \Big\}.$$

It follows that for γ positive and small enough, the mapping T is a contraction if and only if $a_{ii} > 0$ and

$$\sum_{j \neq i} |a_{ij}| < a_{ii}, \qquad \forall i, \tag{1.13}$$

which is a diagonal dominance condition on A. Notice that if $f(x) = Ax$ then $\nabla_j f_i(x) = a_{ij}$. This suggests that the appropriate generalization of conditions (1.12) and (1.13) to the case of nonlinear functions f should be to replace a_{ij} by $\nabla_j f_i(x)$. Indeed, all of the conditions to be introduced in the sequel can be interpreted as diagonal dominance assumptions on the matrix ∇f of partial derivatives of f.

In the general case where the block–components of f have dimension $n_i \geq 1$, we use the notation $\nabla_j f_i(x)$ to denote the matrix of dimension $n_j \times n_i$ whose entries are the partial derivatives of the components of f_i with respect to the components of x_j. In particular, the kth column of $\nabla_j f_i(x)$ is the gradient vector of the kth component of f_i, when viewed as a function of x_j.

Since each $\nabla_j f_i(x)$ is a matrix rather than a scalar, a diagonal dominance condition on $\nabla f(x)$ should involve the norms of these matrices. The most suitable norms for such a purpose are induced matrix norms (see Appendix A) corresponding to the underlying vector norm on $\Re^n$. To be more specific, let $\|\cdot\|_i$ be an arbitrary norm on $\Re^{n_i}$, for each i, and let $\|\cdot\|$ be the corresponding block–maximum norm. With any matrix A of dimension $n_i \times n_j$, we associate the induced matrix norm

$$\|A\|_{ij} = \max_{x \neq 0} \frac{\|Ax\|_i}{\|x\|_j} = \max_{\|x\|_j = 1} \|Ax\|_i.$$

We are now ready to generalize the diagonal dominance conditions (1.12) and (1.13) to the case where f is nonlinear and the dimension n_i of each block–component is possibly larger than 1.

Proposition 1.10. Suppose that X is convex. If $f : \Re^n \mapsto \Re^n$ is continuously differentiable and there exists a scalar $\alpha \in [0, 1)$ such that

$$\left\|I - \gamma G_i^{-1}\big(\nabla_i f_i(x)\big)'\right\|_{ii} + \sum_{j \neq i}\left\|\gamma G_i^{-1}\big(\nabla_j f_i(x)\big)'\right\|_{ij} \leq \alpha, \qquad \forall x \in X,\ \forall i, \quad (1.14)$$

then the mapping $T : X \mapsto \Re^n$ defined by $T_i(x) = x_i - \gamma G_i^{-1} f_i(x)$ is a contraction with respect to the block–maximum norm $\|\cdot\|$.

Proof. We fix some i and $x, y \in X$, and we define a function $g_i : [0, 1] \mapsto \Re^{n_i}$ by

$$g_i(t) = tx_i + (1 - t)y_i - \gamma G_i^{-1} f_i\big(tx + (1 - t)y\big).$$

Notice that g_i is continuously differentiable. Let dg_i/dt be the n_i–dimensional vector consisting of the derivatives of the components of g_i. We then have

$$\begin{aligned}\|T_i(x) - T_i(y)\|_i = \|g_i(1) - g_i(0)\|_i &= \left\|\int_0^1 \frac{dg_i(t)}{dt}\,dt\right\|_i \\ &\leq \int_0^1 \left\|\frac{dg_i}{dt}(t)\right\|_i dt \leq \max_{t\in[0,1]}\left\|\frac{dg_i}{dt}(t)\right\|_i.\end{aligned}$$

It, therefore, suffices to bound the norm of dg_i/dt. The chain rule yields

$$\begin{aligned}\left\|\frac{dg_i}{dt}(t)\right\|_i &= \left\|x_i - y_i - \gamma G_i^{-1}\Big(\nabla f_i\big(tx + (1 - t)y\big)\Big)'(x - y)\right\|_i \\ &= \left\|\left[I - \gamma G_i^{-1}\Big(\nabla_i f_i\big(tx + (1 - t)y\big)\Big)'\right](x_i - y_i)\right. \\ &\qquad \left. - \sum_{j \neq i} \gamma G_i^{-1}\Big(\nabla_j f_i\big(tx + (1 - t)y\big)\Big)'(x_j - y_j)\right\|_i \\ &\leq \left\|I - \gamma G_i^{-1}\Big(\nabla_i f_i\big(tx + (1 - t)y\big)\Big)'\right\|_{ii} \cdot \|x_i - y_i\|_i \\ &\qquad + \sum_{j \neq i}\left\|\gamma G_i^{-1}\Big(\nabla_j f_i\big(tx + (1 - t)y\big)\Big)'\right\|_{ij} \cdot \|x_j - y_j\|_j \\ &\leq \alpha \max_j \|x_j - y_j\|_j = \alpha\|x - y\|,\end{aligned}$$

which establishes the contraction property. [We have used the assumption (1.14) with x replaced by $tx + (1 - t)y$; this vector belongs to X because X is assumed convex.] **Q.E.D.**

The condition (1.14) in the previous proposition is generally hard to verify. It is shown in the following that if $n_i = 1$ for each i, then this condition simplifies considerably and bears closer resemblance to the diagonal dominance condition (1.13).

Proposition 1.11. Assume the following:

(a) We have $n_i = 1$, for each i, the set X is convex, and the function $f : \Re^n \mapsto \Re^n$ is continuously differentiable.

(b) There exists a positive constant K such that

$$\nabla_i f_i(x) \leq K, \qquad \forall x \in X, \ \forall i.$$

(c) There exists some $\beta > 0$ such that

$$\sum_{j \neq i} |\nabla_j f_i(x)| \leq \nabla_i f_i(x) - \beta, \qquad \forall x \in X, \ \forall i, \tag{1.15}$$

Then, the mapping $T : X \mapsto \Re^n$ defined by $T(x) = x - \gamma f(x)$ is a contraction with respect to the maximum norm, provided that $0 < \gamma < 1/K$.

Proof. Under the assumption $0 < \gamma < 1/K$, we have

$$|1-\gamma\nabla_i f_i(x)|+\gamma \sum_{j \neq i} |\nabla_j f_i(x)| = 1-\gamma\Big(\nabla_i f_i(x)-\sum_{j \neq i} |\nabla_j f_i(x)|\Big) \leq 1-\gamma\beta < 1, \tag{1.16}$$

which shows that inequality (1.14) holds. The result follows from Prop. 1.10. **Q.E.D.**

A minor generalization of Prop. 1.11 is provided in Exercise 1.3.

The next two results are based on a particular choice of norms, namely weighted quadratic norms. For motivation purposes, let us temporarily consider the case where there is only one block–component and consider a mapping $T : X \mapsto \Re^n$ given by

$$T(x) = x - \gamma G^{-1} f(x), \qquad \forall x \in X, \tag{1.17}$$

where G is a symmetric positive definite matrix. The effect of G in Eq. (1.17) is to scale the direction in which x is changed when T is applied. Accordingly, it is reasonable to consider a norm that scales the components of x in a corresponding fashion. To this effect, we introduce the norm $\| \cdot \|_G$, defined by

$$\|x\|_G = \left(x'Gx\right)^{1/2}$$

and we look for conditions under which T is a contraction with respect to $\|\cdot\|_G$. An easy calculation yields

$$\begin{aligned}\|T(x)-T(y)\|_G^2 &= \Big((x-y)-\gamma G^{-1}\big(f(x)-f(y)\big)\Big)' G\Big((x-y)-\gamma G^{-1}\big(f(x)-f(y)\big)\Big)\\ &= \|x-y\|_G^2+\gamma^2\big(f(x)-f(y)\big)'G^{-1}\big(f(x)-f(y)\big)\\ &\quad -2\gamma\big(f(x)-f(y)\big)'(x-y).\end{aligned}$$

If γ is chosen very small and the norm of $f(x)-f(y)$ is of the order of $\|x-y\|_G$, then the term involving γ^2 can be neglected. We then see that for T to be a contraction, it is sufficient to assume that γ is positive and small enough, and that

$$\big(f(x)-f(y)\big)'(x-y)\geq \alpha\|x-y\|_G^2, \qquad \forall x,y\in X, \tag{1.18}$$

where α is some positive constant. Inequality (1.18) is called a *strong monotonicity* condition and its significance will be explored further in Section 3.5. Let us simply notice here that if f is the linear function $f(x)=Ax$, then strong monotonicity is equivalent to the positive definiteness of A.

We now return to the general case where $X=\prod_{i=1}^m X_i\subset\prod_{i=1}^m \Re^{n_i}$, and where T is given by $T_i(x)=x_i-\gamma G_i^{-1}f_i(x)$ for each i. We assume that each matrix G_i is symmetric and positive definite. For each i, we define a norm $\|\cdot\|_i$ on $\Re^{n_i}$ by

$$\|x_i\|_i=\big(x_i'G_ix_i\big)^{1/2}.$$

These norms define a block–maximum norm $\|\cdot\|$ given by $\|x\|=\max_i\|x_i\|_i$. In keeping with the discussion in the preceding paragraph, we shall impose a bound on the magnitude of $f(x)-f(y)$ and a monotonicity condition similar to (1.18).

Proposition 1.12. Suppose that each G_i is symmetric positive definite and let the norms $\|\cdot\|_i$ and $\|\cdot\|$ be as above. Suppose that there exist positive constants A_1, A_2, A_3, with $A_3<A_2$, such that for each i and for each $x,y\in X$, we have

$$\|f_i(x)-f_i(y)\|_i\leq A_1\|x-y\|, \tag{1.19}$$

and

$$\big(f_i(x)-f_i(y)\big)'(x_i-y_i)\geq A_2\|x_i-y_i\|_i^2-A_3\|x-y\|^2. \tag{1.20}$$

Then, provided that γ is positive and small enough, the mapping $T:X\mapsto\Re^n$, defined by $T_i(x)=x_i-\gamma G_i^{-1}f_i(x)$, is a contraction with respect to the block–maximum norm $\|\cdot\|$.

Proof. Let A_4 be a positive constant such that $x_i' G_i^{-1} x_i \leq A_4 x_i' G_i x_i$ for every $x_i \in \Re^{n_i}$. [The existence of such a constant follows from the positive definiteness of G_i; see Prop. A.28(c) in Appendix A.] Assuming that $0 < \gamma < 1/(2A_2)$, we have

$$\begin{aligned}\|T_i(x) - T_i(y)\|_i^2 =& \|x_i - y_i\|_i^2 + \gamma^2 \big(f_i(x) - f_i(y)\big)' G_i^{-1} \big(f_i(x) - f_i(y)\big) \\ & - 2\gamma \big(f_i(x) - f_i(y)\big)' (x_i - y_i) \\ \leq & \|x_i - y_i\|_i^2 + A_1^2 A_4 \gamma^2 \|x - y\|^2 - 2\gamma A_2 \|x_i - y_i\|_i^2 + 2\gamma A_3 \|x - y\|^2 \\ \leq & \Big(1 - 2\gamma A_2 + A_1^2 A_4 \gamma^2 + 2\gamma A_3\Big) \|x - y\|^2.\end{aligned}$$

If γ is also smaller than $2(A_2 - A_3)/(A_1^2 A_4)$, which is possible because $A_2 > A_3$, the expression $1 - 2\gamma A_2 + A_1^2 A_4 \gamma^2 + 2\gamma A_3$ is smaller than 1, which proves the result. **Q.E.D.**

We now simplify the conditions of Prop. 1.12, for the case where f is continuously differentiable. In the proof of our next result, we use the fact that if G_i is symmetric and positive definite, then the norm $\|x_i\|_i = (x_i' G_i x_i)^{1/2}$ is also equal to $\|G_i^{1/2} x_i\|_2$, where $G_i^{1/2}$ is a symmetric square root of G_i and $\|\cdot\|_2$ is the Euclidean norm (see Props. A.27 and A.28 in Appendix A).

Proposition 1.13. Assume the following:

(a) The set X is convex and the function $f : \Re^n \mapsto \Re^n$ is continuously differentiable.

(b) For each i, the matrix G_i is symmetric and positive definite.

(c) There exists a constant K such that $\|\nabla f(x)\|_2 \leq K$ for every $x \in X$.

(d) There exist some $\delta > 0$ and $\epsilon > 0$ such that $\nabla_i f_i(x)' - \delta G_i$ is nonnegative definite, for every i and $x \in X$, and such that

$$\sum_{j \neq i} \left\| G_j^{-1/2} \nabla_j f_i(x) G_i^{-1/2} \right\|_2 \leq \delta(1 - \epsilon), \qquad \forall i, \ \forall x \in X. \tag{1.21}$$

Then, provided that γ is positive and small enough, the mapping $T : X \mapsto \Re^n$, defined by $T_i(x) = x_i - \gamma G_i^{-1} f_i(x)$, is a contraction with respect to the block–maximum norm $\|x\| = \max_i (x_i' G_i x_i)^{1/2}$.

Proof. We shall verify that the assumptions of Prop. 1.12 are satisfied. Let us fix some i and some $x, y \in X$. We define a function $g : [0, 1] \mapsto \Re$ by

$$g(t) = f_i\big(tx + (1 - t)y\big)'(x_i - y_i).$$

The chain rule yields

$$\frac{dg}{dt}(t) = (x - y)'\nabla f_i\big(tx + (1 - t)y\big)(x_i - y_i).$$

Furthermore, the mean value theorem shows that there exists some $t_0 \in [0, 1]$ such that

$$g(1) - g(0) = \frac{dg}{dt}(t_0).$$

Let $z = t_0 x + (1 - t_0)y$ and note that z belongs to X because X is convex. We then have

$$\begin{aligned}
\big(f_i(x) - f_i(y)\big)'(x_i - y_i) &= g(1) - g(0) = \frac{dg}{dt}(t_0) \\
&= (x - y)'\nabla f_i(z)(x_i - y_i) \\
&= (x_i - y_i)'\nabla_i f_i(z)(x_i - y_i) + \sum_{j \neq i}(x_j - y_j)'\nabla_j f_i(z)(x_i - y_i) \\
&\geq \delta\|x_i - y_i\|_i^2 + \sum_{j \neq i}(x_j - y_j)'G_j^{1/2}\left(G_j^{-1/2}\nabla_j f_i(z)G_i^{-1/2}\right)G_i^{1/2}(x_i - y_i) \\
&\geq \delta\|x_i - y_i\|_i^2 - \sum_{j \neq i}\|G_j^{1/2}(x_j - y_j)\|_2 \cdot \|G_j^{-1/2}\nabla_j f_i(z)G_i^{-1/2}\|_2 \cdot \|G_i^{1/2}(x_i - y_i)\|_2 \\
&\geq \delta\|x_i - y_i\|_i^2 - \|x - y\|^2 \sum_{j \neq i}\|G_j^{-1/2}\nabla_j f_i(z)G_i^{-1/2}\|_2 \\
&\geq \delta\|x_i - y_i\|_i^2 - \delta(1 - \epsilon)\|x - y\|^2,
\end{aligned}$$

where in the last two steps we used the definition of the norm $\|\cdot\|$ and inequality (1.21). This shows that condition (1.20) is satisfied with $A_2 = \delta$ and $A_3 = \delta(1 - \epsilon) < A_2$.

Condition (1.19) in Prop. 1.12 is a simple consequence of condition (c) and the mean value theorem. We conclude that Prop. 1.12 applies and shows that T is a contraction. **Q.E.D.**

EXERCISES

1.1. Show that if $T : X \mapsto X$ is a contraction but X is not closed, then T need not have a fixed point.

1.2. **(a)** Construct an example of a mapping T satisfying the assumptions of Prop. 1.8 and such that for some $x \in X$ and some i, the set $R_i(x)$ has more than one element.

(b) Construct an example to show that Prop. 1.8 is false without the assumption that X is convex.

(c) Construct an example to show that Prop. 1.8 is false without the assumption that the mapping T is continuous.

1.3. Let a function $f : \Re^n \mapsto \Re^n$ satisfy the assumptions of Prop. 1.11 except that the condition (1.15) is replaced by

$$\sum_{j \neq i} w_j |\nabla_j f_i(x)| \leq w_i \nabla_i f_i(x) - \beta, \qquad \forall x \in X, \ \forall i,$$

where $w_1, \ldots, w_m$ are positive scalars. Show that for γ positive and small enough, the mapping T defined by $T(x) = x - \gamma f(x)$ is a contraction mapping with respect to a suitable norm.

1.4. **(Monotone Mappings.)** A mapping $T : \Re^n \mapsto \Re^n$ is called monotone if it satisfies $T(x) \leq T(y)$ for every $x, y \in \Re^n$ such that $x \leq y$. Suppose that T is monotone, continuous, has a unique fixed point x^*, and that there exist two vectors y^*, $z^* \in \Re^n$ such that $y^* \leq z^*$ and $T(y^*) \geq y^*$, $T(z^*) \leq z^*$. Let $H = \{x \mid y^* \leq x \leq z^*\}$.

(a) Show that the sequence $\{x(t)\}$ generated by the iteration $x(t+1) = T\big(x(t)\big)$ converges to x^* if $x(0)$ is equal to either y^* or z^*. Furthermore, $x^* \in H$.

(b) Show that the conclusion of part (a) remains valid for every $x(0) \in H$.

(c) Show that the sequence of vectors generated by the Gauss–Seidel algorithm based on the mapping T converges to x^* for every initial vector $x(0)$ belonging to H.

(d) We define $\hat{T}_i : \Re^n \mapsto \Re^n$ by

$$\hat{T}_i(x) = \hat{T}_i(x_1, \ldots, x_n) = \big(x_1, \ldots, x_{i-1}, T_i(x), x_{i+1}, \ldots, x_m\big). \tag{1.22}$$

Consider the iteration $x(t+1) = \hat{T}_i\big(x(t)\big)$ and show that it converges to a finite vector for every $x(0) \in H$. *Hint:* This is essentially a one-dimensional iteration.

(e) For any $x \in H$ and any i, let $Q_i(x)$ be the limit of $x_i(t)$, where $\{x(t)\}$ is the sequence generated by the iteration of part (d), initialized with $x(0) = x$. Show that the mapping $Q = (Q_1, \ldots, Q_n)$ is monotone on the set H. Construct an example showing that Q can be discontinuous.

(f) Let Q be as in part (e). Show that the sequence generated by the iteration $x(t+1) = Q\big(x(t)\big)$ converges to x^* for every $x(0) \in H$, and that the same is true for the Gauss–Seidel algorithm based on Q.

3.2 UNCONSTRAINED OPTIMIZATION

In this section, we consider algorithms for minimizing a continuously differentiable cost function $F : \Re^n \mapsto \Re$. We have $\nabla F(x^*) = 0$ for every vector x^* that minimizes F (Prop. A.34 in Appendix A). In view of this fact, the minimization of F is related to the problem of solving the system $\nabla F(x) = 0$ of generally nonlinear equations. In fact, most iterative optimization algorithms are aimed at finding a solution of the equation $\nabla F(x) = 0$ without any guarantees that such a solution is a global minimizer of F. We will thus settle with this as our objective. In Chapter 2, we studied iterative algorithms for the solution of linear equations; the algorithms in this section can be viewed as their natural extensions to a nonlinear setting.

There are two main approaches for proving convergence of an algorithm for nonlinear optimization. In the *descent* approach, one shows that the value of the cost function keeps decreasing toward its minimal value. In an alternative approach, a suitable norm is introduced and one shows that the distance of the current iterate from a minimizing point decreases with each iteration. In what follows, both approaches will be considered.

3.2.1 The Main Algorithms

The algorithms to be presented can be motivated from the iterative algorithms for solving linear equations that were introduced in Section 2.4. Suppose that we are solving the linear system $Ax = b$, where A is a symmetric positive definite matrix. This is equivalent to minimizing a cost function F defined by $F(x) = \frac{1}{2}x'Ax - x'b$. In this context, $\nabla F(x) = Ax - b$ and $\nabla^2 F(x) = A$. We now recall the iterative algorithms introduced in Section 2.4 and the following generalizations suggest themselves:

Jacobi Algorithm. (Generalizing the JOR algorithm for linear equations):

$$x(t+1) = x(t) - \gamma\big[D\big(x(t)\big)\big]^{-1}\nabla F(x(t)), \tag{2.1}$$

where γ is a positive stepsize, and where $D(x)$ is a diagonal matrix whose ith diagonal entry is $\nabla^2_{ii}F(x)$, assumed to be nonzero for each i.

Gauss-Seidel Algorithm. (Generalizing the SOR algorithm for linear equations):

$$x_i(t+1) = x_i(t) - \gamma\frac{\nabla_i F\big(z(i,t)\big)}{\nabla^2_{ii}F\big(z(i,t)\big)}, \qquad i = 1,\ldots,n, \tag{2.2}$$

where $z(i,t) = \big(x_1(t+1),\ldots,x_{i-1}(t+1),x_i(t),\ldots,x_n(t)\big)$.

Gradient Algorithm. (Generalizing Richardson's algorithm for linear equations):

$$x(t+1) = x(t) - \gamma\nabla F\big(x(t)\big). \tag{2.3}$$

A Gauss–Seidel variant of the gradient algorithm is obtained if Eq. (2.3) is replaced by

$$x_i(t+1) = x_i(t) - \gamma\nabla_i F\big(z(i,t)\big), \qquad i = 1,\ldots,n, \tag{2.4}$$

where $z(i,t)$ is defined as in the Gauss–Seidel algorithm.

Given some $x \in \Re^n$ such that $\nabla F(x) \neq 0$, any vector $s \in \Re^n$ with the property $s'\nabla F(x) < 0$ is called a *descent direction*. The reason is that $s'\nabla F(x)$ is the directional derivative of F along the direction s and therefore, if γ is a sufficiently small positive

constant, then $F(x + \gamma s) < F(x)$. Any algorithm that, given a current vector x satisfying $\nabla F(x) \neq 0$, updates x along a descent direction is called a *descent algorithm.* The gradient algorithm (2.3) is certainly a descent algorithm; in fact, it is often called the *steepest descent* algorithm because the direction of update is such that F tends to decrease as fast as possible, in the sense that $-\nabla F(x)/\|\nabla F(x)\|_2$ minimizes the directional derivative $s'\nabla F(x)$ over all directions s with $\|s\|_2 = 1$. The Gauss–Seidel variant (2.4) of the gradient algorithm is also a descent algorithm, and the same property holds for the Jacobi and Gauss–Seidel algorithms of Eqs. (2.1) and (2.2), respectively, under the assumption that $\nabla^2_{ii}F(x) > 0$ for all $x \in \Re^n$. We can think of the Jacobi algorithm as a *scaled* version of the gradient algorithm, whereby the ith component of the update $-\gamma\nabla F\big(x(t)\big)$ is scaled by a factor of $1/\nabla^2_{ii}F\big(x(t)\big)$. One can consider more general scaling methods and this leads to the following algorithm.

Scaled Gradient Algorithm.

$$x(t+1) = x(t) - \gamma\big(D(t)\big)^{-1}\nabla F\big(x(t)\big), \tag{2.5}$$

where $D(t)$ is a scaling matrix. Quite often, $D(t)$ is chosen diagonal, which simplifies the task of inverting it. If $D(t)$ is indeed diagonal, its entries are positive, γ is positive, and $\nabla F\big(x(t)\big) \neq 0$, then it is seen that $\gamma\nabla F\big(x(t)\big)'\big(D(t)\big)^{-1}\nabla F\big(x(t)\big) > 0$ and the scaled gradient algorithm is a descent algorithm (see Fig. 3.2.1).

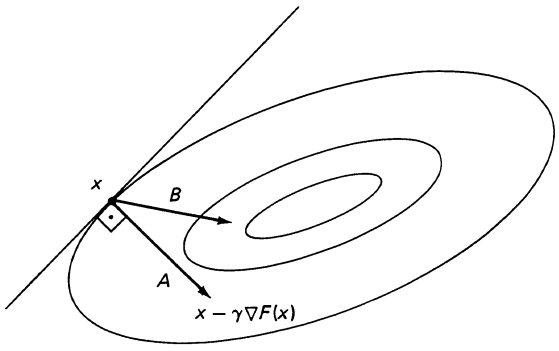

Figure 3.2.1 Descent directions of the gradient and the scaled gradient algorithms. The curves shown are sets of points where the value of F is constant. The vector A indicates the steepest descent direction. The vector B is another descent direction obtained by positive scaling of the components of A, as in the scaled gradient iteration $x := x - \gamma D^{-1}\nabla F(x)$, with D being diagonal and with positive diagonal entries. With proper scaling, the direction of B is preferable to that of A.

The parallel implementation of algorithms such as the Jacobi and gradient algorithms of Eqs. (2.1) and (2.3), respectively, is straightforward. We assign to the ith processor the task of updating the ith component of x. After each update, each processor communicates the newly computed value to those processors that require it. We notice that the ith processor has to know the current value of x_j only if $\nabla_i F$ or $\nabla^2_{ii}F$ depends on x_j. For many large problems, $\nabla_i F$ and $\nabla^2_{ii}F$ depend on only a few of the remaining components, the corresponding dependency graph is sparse, and the communication requirements of such algorithms are greatly reduced. The Gauss–Seidel algorithms of Eqs. (2.2) and (2.4) are generally unsuitable for parallel implementation except when the

dependency graph is sparse, in which case, the coloring scheme discussed in Subsection 1.2.4 is applicable.

A related class of algorithms is obtained if instead of using a constant stepsize γ, we use a stepsize that leads to the largest possible reduction of the value of F. For example, in a modification of the gradient algorithm, we can let $x(t+1)$ be equal to $x(t) - \gamma(t)\nabla F\big(x(t)\big)$, where $\gamma(t)$ is the value of γ that minimizes $F\big(x(t) - \gamma\nabla F(x(t))\big)$ with respect to γ. Such algorithms often converge faster; on the other hand, the one–dimensional minimization that has to be carried out at each stage is not easily parallelizable in general. For this reason, in what follows, we concentrate attention to the case of a constant stepsize. We refer to the sources given at the end of the chapter for convergence analysis using other stepsize rules.

Newton and Approximate Newton Methods

Let us assume that F is twice continuously differentiable. An important method for nonlinear optimization is *Newton's algorithm*, described by the equation

$$x(t+1) = x(t) - \gamma\Big(\nabla^2 F\big(x(t)\big)\Big)^{-1}\nabla F\big(x(t)\big). \tag{2.6}$$

We notice that if $F(x) = \frac{1}{2}x'Ax - x'b$ and if $\gamma = 1$, then $x(t+1) = x(t) - A^{-1}\big(Ax(t) - b\big)$ $= A^{-1}b$, which proves convergence in a single step. Accordingly, it can be shown that for nonquadratic problems, Newton's algorithm converges much faster (under certain assumptions) than the previously introduced algorithms, see e.g. [OrR70]. As an illustration of this fact, assume that F is twice continuously differentiable and has a local minimum x^* for which $\nabla^2 F(x^*)$ is positive definite. Suppose that we are given $x(t)$ which is close enough to x^* so that $\nabla^2 F\big(x(t)\big)$ is invertible. Let $x(t+1)$ be the vector generated by the Newton iteration (2.6) with $\gamma = 1$. We then have

$$\begin{aligned} x(t+1) - x^* &= \big[\nabla^2 F\big(x(t)\big)\big]^{-1}\big[\nabla^2 F\big(x(t)\big)\big(x(t) - x^*\big) - \nabla F\big(x(t)\big)\big] \\ &= \big[\nabla^2 F\big(x(t)\big)\big]^{-1}\int_0^1 \big[\nabla^2 F\big(x(t)\big) - \nabla^2 F\big(x^* + \xi\big(x(t) - x^*\big)\big)\big]\,d\xi\,\big(x(t) - x^*\big), \end{aligned}$$

from which we obtain for any norm $\|\cdot\|$

$$\begin{aligned} &\|x(t+1) - x^*\| \\ &\leq \Big\|\big[\nabla^2 F\big(x(t)\big)\big]^{-1}\Big\| \cdot \Big(\int_0^1 \big\|\nabla^2 F\big(x(t)\big) - \nabla^2 F\big(x^* + \xi\big(x(t) - x^*\big)\big)\big\|\,d\xi\Big) \cdot \|x(t) - x^*\|. \end{aligned}$$

Using the continuity of $\nabla^2 F(x)$, it follows that given any $\alpha \in (0,1)$ there exists some $\epsilon > 0$ such that

$$\|x(t+1) - x^*\| \leq \alpha\|x(t) - x^*\|,$$

for all $x(t)$ with $\|x(t) - x^*\| \leq \epsilon$. This is in contrast with the other algorithms we have been studying for which the preceding inequality cannot be proved for an arbitrarily small value of α, and establishes the faster convergence of Newton's algorithm. On the other hand, the Newton iteration (2.6) involves a matrix inverse that greatly amplifies the computational requirements per stage. The Jacobi algorithm (2.1) can be viewed as an approximation of Newton's algorithm in which the off–diagonal entries of the matrix $\nabla^2 F$ are ignored, thereby making the matrix inversion very easy. More generally, in scaled gradient algorithms of the form $x(t+1) = x(t) - \gamma\big(D(t)\big)^{-1}\nabla F\big(x(t)\big)$, it is usually desired to let $D(t)$ be an approximation of $\nabla^2 F\big(x(t)\big)$ that is easy to invert.

We now describe a related and frequently more practical class of algorithms, the *approximate Newton methods*, which are based on the Newton iteration (2.6) except that the inversion of the matrix $\nabla^2 F\big(x(t)\big)$ is not carried out to completion. Let

$$H = \nabla^2 F\big(x(t)\big)$$

and

$$g = \nabla F\big(x(t)\big).$$

Equation (2.6) becomes

$$x(t+1) = x(t) + \gamma s, \tag{2.7}$$

where s is computed by solving the linear system $Hs = -g$. In an approximate Newton method, we employ an iterative algorithm for solving the system $Hs = -g$ and we terminate this algorithm after only a few iterations, before it converges. (Some common choices of iterative algorithms are the SOR method of Section 2.4 and the conjugate gradient method of Section 2.7, which, incidentally, are well–suited for parallelization.) This provides us with a direction vector $\hat{s}$ that is an approximation of s, and Eq. (2.7) is replaced by

$$x(t+1) = x(t) + \gamma \hat{s}. \tag{2.8}$$

A remarkable fact is that the vector $\hat{s}$ is guaranteed, under certain assumptions, to be a direction of descent, which we proceed to demonstrate.

Suppose that $g \neq 0$ and that H is positive definite. (Conditions for H to be positive definite are provided by Prop. A.41 in Appendix A.) Furthermore, H is symmetric. Several iterative methods for solving the system $Hs = -g$ have the property that successive iterates reduce the value of the quadratic form $\frac{1}{2}s'Hs + g's$. This is the case, for example, for SOR (see the argument preceding Prop. 6.10 and Fig. 2.6.5, in Section 2.6) and for the conjugate gradient method (see Prop. 7.1 and Exercise 7.4 in Section 2.7). Therefore, if the iterative algorithm is initialized with $s = 0$ (or, more generally, with any s such that $\frac{1}{2}s'Hs \leq -g's$), the vector $\hat{s}$ produced after any finite number of iterations satisfies

$$\tfrac{1}{2}\hat{s}'H\hat{s} + g'\hat{s} < 0. \tag{2.9}$$

Since H is assumed positive definite, we obtain $g'\hat{s} < 0$, which shows that $\hat{s}$ is a direction of descent.

3.2.2 Convergence Analysis Using the Descent Approach

We now study the convergence of the previous algorithms using the descent approach. The proofs given will be generalized to the context of partially asynchronous algorithms in Chapter 7. The main line of argument is simple: we first show that each update reduces the value of the cost function by an amount that is bounded away from zero if the magnitude of the update is bounded away from zero. Given that the cost function is bounded below, it follows that the magnitude of the updates converges to zero. Then, one uses the formula for the updates to show that $\nabla F\big(x(t)\big)$ must also converge to zero.

The following assumption on F will be used in most of the results of this section.

Assumption 2.1.

(a) There holds $F(x) \geq 0$ for every $x \in \Re^n$.

(b) (*Lipschitz Continuity of* ∇F) The function F is continuously differentiable and there exists a constant K such that

$$\|\nabla F(x) - \nabla F(y)\|_2 \leq K\|x - y\|_2, \qquad \forall x, y \in \Re^n.$$

A key consequence of Assumption 2.1 is provided by Prop. A.32 in Appendix A, which we repeat here for easier reference.

Lemma 2.1. (*Descent Lemma*) If F satisfies the Lipschitz condition of Assumption 2.1(b), then

$$F(x + y) \leq F(x) + y'\nabla F(x) + \frac{K}{2}\|y\|_2^2, \qquad \forall x, y \in \Re^n.$$

The following convergence result covers a wide class of descent algorithms.

Proposition 2.1. (*Convergence of Descent Algorithms*) Suppose that Assumption 2.1 holds and let K_1 and K_2 be positive constants. Consider the sequence $\{x(t)\}$ generated by an algorithm of the form

$$x(t+1) = x(t) + \gamma s(t), \tag{2.10}$$

where $s(t)$ satisfies

$$\|s(t)\|_2 \geq K_1\big\|\nabla F\big(x(t)\big)\big\|_2, \qquad \forall t, \tag{2.11}$$

and

$$s(t)'\nabla F\big(x(t)\big) \leq -K_2\|s(t)\|_2^2, \qquad \forall t. \tag{2.12}$$

If $0 < \gamma < 2K_2/K$, then

$$\lim_{t\to\infty} \nabla F\big(x(t)\big) = 0.$$

Proof. Using the Descent Lemma and the assumption (2.12), we obtain

$$\begin{aligned} F\big(x(t+1)\big) &\leq F\big(x(t)\big) + \gamma s(t)'\nabla F\big(x(t)\big) + \frac{K}{2}\gamma^2\|s(t)\|_2^2 \\ &\leq F\big(x(t)\big) - \gamma\left(K_2 - \frac{K\gamma}{2}\right)\|s(t)\|_2^2. \end{aligned}$$

Let $\beta = \gamma(K_2 - K\gamma/2)$, which, by our assumptions on γ, is positive. We have one such inequality for every $t \geq 0$. Adding these inequalities and using the nonnegativity condition of Assumption 2.1(a), we obtain

$$0 \leq F\big(x(t+1)\big) \leq F\big(x(0)\big) - \beta\sum_{\tau=0}^{t}\|s(\tau)\|_2^2.$$

Since this inequality is true for all t, we obtain

$$\sum_{\tau=0}^{\infty}\|s(\tau)\|_2^2 \leq \frac{1}{\beta}F\big(x(0)\big) < \infty.$$

This implies that $\lim_{t\to\infty} s(t) = 0$ and Eq. (2.11) shows that $\lim_{t\to\infty}\nabla F\big(x(t)\big) = 0$. **Q.E.D.**

Assumption 2.1 is stronger than necessary for Prop. 2.1 to hold. For example, instead of the nonnegativity condition on F, we could only assume that F is bounded below. The Lipschitz condition on ∇F can also be weakened somewhat (see Exercise 2.1). Besides Assumption 2.1, Prop. 2.1 involves two additional conditions, inequalities (2.11) and (2.12). Inequality (2.11) implies that $s(t) \neq 0$, and, therefore, $x(t+1) \neq x(t)$ whenever $\nabla F\big(x(t)\big) \neq 0$. Such a condition is necessary if the algorithm is to make any progress at all. Inequality (2.12) implies that the direction of update in Eq. (2.10) is a descent direction.

The conditions of Prop. 2.1 can be verified for a variety of algorithms, as we now show.

(a) For the gradient algorithm (2.4), we have $s(t) = -\nabla F\big(x(t)\big)$. Thus, $K_1 = K_2 = 1$, and we have convergence for $0 < \gamma < 2/K$.

(b) Consider the scaled gradient algorithm (2.5) for which $s(t) = -\big(D(t)\big)^{-1}\nabla F\big(x(t)\big)$. Assume that the sequence $\{D(t)\}$ is bounded and that for some $K_2 > 0$, the matrix $D(t) - K_2 I$ is nonnegative definite for each t. We then have

$$K_2\|s(t)\|_2^2 \leq s(t)'D(t)s(t) = -s(t)'\nabla F\big(x(t)\big)$$

and inequality (2.12) is satisfied. Let $K_1 = 1/\sup_t \|D(t)\|_2$. We have $D(t)s(t) = -\nabla F\big(x(t)\big)$, which implies that $\|D(t)\|_2 \cdot \|s(t)\|_2 \geq \big\|\nabla F\big(x(t)\big)\big\|_2$. From the latter inequality, we obtain $\|s(t)\|_2 \geq K_1\big\|\nabla F\big(x(t)\big)\big\|_2$ and inequality (2.11) is satisfied.

(c) Assume that F is twice continuously differentiable and consider the Jacobi algorithm (2.1). This is a special case of the scaled gradient algorithm (2.5), with $D(t)$ diagonal. Assuming that $\nabla_{ii}^2 F(x)$ is bounded above by $1/K_1$ and below by K_2 for some positive constants K_1 and K_2, the discussion in (b) applies.

(d) Consider the approximate Newton method (2.8) and again let $g = \nabla F\big(x(t)\big)$ and $H = \nabla^2 F\big(x(t)\big)$. Whenever $g \neq 0$, we assume that $\hat{s}$ is chosen to satisfy $\frac{1}{2}\hat{s}'H\hat{s} + g'\hat{s} < 0$ [cf. Eq. (2.9)] and $\|\hat{s}\|_2 \geq K_1\|g\|_2$. Then, inequality (2.11) holds. We assume that F is twice continuously differentiable and that there exists a constant K_2 such that $\frac{1}{2}\nabla^2 F(x) - K_2 I$ is nonnegative definite for every x. We then have

$$-g'\hat{s} > \frac{1}{2}\hat{s}'H\hat{s} \geq K_2\|\hat{s}\|_2^2$$

and inequality (2.12) is also satisfied.

Convergence of algorithms of the Gauss–Seidel type can also be proved by an argument similar to that in Prop. 2.1.

Proposition 2.2. (*Convergence of the Gauss–Seidel Algorithm*) Suppose that Assumption 2.1 holds and that F is twice differentiable. Assume that there exist constants $d_i, D_i > 0$ such that $0 < d_i \leq \nabla_{ii}^2 F(x) \leq D_i$ for all $x \in \Re^n$. If $0 < \gamma < 2d_i/D_i$ for all i, and if the sequence $\{x(t)\}$ is generated by the Gauss–Seidel algorithm (2.2), then $\lim_{t\to\infty} \nabla F(x(t)) = 0$.

Proof. Let $s^i(t)$ be a vector with all components equal to zero, except for the ith component, which is equal to $-\nabla_i F\big(z(i,t)\big)/\nabla_{ii}^2 F\big(z(i,t)\big)$. Notice that $z(i+1,t) = z(i,t) + \gamma s^i(t)$ for $1 \leq i < n$ and $x(t+1) = z(n,t) + \gamma s^n(t)$. We use the Descent Lemma (Lemma 2.1), with the function F viewed as a function of the single variable x_i; we also use the bound $|\nabla_i F(x) - \nabla_i F(y)| \leq D_i|x_i - y_i|$, which is valid for all x and y that differ only in the ith component. Then

$$F\big(z(i,t)+\gamma s^i(t)\big) \le F\big(z(i,t)\big) + \gamma s^i(t)'\nabla F\big(z(i,t)\big) + \gamma^2\frac{D_i}{2}\|s^i(t)\|_2^2$$

$$\le F\big(z(i,t)\big) - \gamma d_i\|s^i(t)\|_2^2 + \gamma^2\frac{D_i}{2}\|s^i(t)\|_2^2$$

$$= F\big(z(i,t)\big) - \gamma\left(d_i - \gamma\frac{D_i}{2}\right)\|s^i(t)\|_2^2.$$

With our assumption on γ, the quantity $\gamma\big(d_i - \gamma D_i/2\big)$ is positive. Thus, the steps in the proof of Prop. 2.1 remain valid and we conclude that

$$\sum_{t=0}^{\infty}\sum_{i=1}^{n}\|s^i(t)\|_2^2 < \infty.$$

Therefore, $\lim_{t\to\infty} s^i(t) = 0$ for all i, which implies that $\lim_{t\to\infty}\nabla_i F\big(z(i,t)\big) = 0$ and that $\lim_{t\to\infty}\big(z(i,t) - x(t)\big) = 0$. Using the Lipschitz continuity of ∇F [Assumption 2.1(b)], we obtain $\lim_{t\to\infty}\big(\nabla_i F\big(z(i,t)\big) - \nabla_i F\big(x(t)\big)\big) = 0$, from which we conclude that $\lim_{t\to\infty}\nabla_i F\big(x(t)\big) = 0$ for all i. **Q.E.D.**

A similar result is possible for the Gauss–Seidel variant (2.4) of the gradient algorithm, but it is omitted. Notice that in the case of a quadratic cost function of the form $F(x) = \frac{1}{2}x'Ax - x'b$, we have $d_i = D_i = a_{ii}$ and the condition on γ becomes $0 < \gamma < 2$. Assuming that A is symmetric positive definite, F is minimized at the unique solution of the system $Ax = b$ and the Gauss–Seidel algorithm (2.2) coincides with the SOR method for linear equations. We conclude that Prop. 2.2 establishes the convergence of SOR for $0 < \gamma < 2$, which is Prop. 6.10(a) in Section 2.6. Similarly, Prop. 2.1 establishes the convergence of the JOR algorithm and of Richardson's method for solving the system $Ax = b$, when A is symmetric positive definite and γ is positive and small enough. This proves Prop. 6.11 in Section 2.6.

The preceding results say nothing about convergence of the sequence $\{x(t)\}$ and indeed there is nothing in our hypotheses that ensures boundedness of $x(t)$. On the other hand, the convergence of ∇F to zero and the continuity of ∇F imply that if x^* is a limit point of $x(t)$, then $\nabla F(x^*) = 0$.

3.2.3 The Case of a Convex Cost Function

The preceding results can be strengthened when F is a convex function (see Appendix A for a review of convexity notions). In particular, if F is convex and continuously differentiable, then any point x such that $\nabla F(x) = 0$ is guaranteed to be a global minimum of F [Prop. A.39(c) in Appendix A]. We then obtain the following result.

Proposition 2.3. (*Convergence of Descent Methods in Convex Optimization*) Suppose that F is convex and satisfies Assumption 2.1, and that the sequence $\{x(t)\}$ is as in Prop. 2.1 or 2.2. If x^* is a limit point of the sequence $\{x(t)\}$, then x^* minimizes F.

The following result uses a more stringent assumption on the cost function F and leads to a bound on the convergence rate of the algorithms under consideration.

Proposition 2.4. (*Geometric Convergence for Strongly Convex Problems*) Suppose, in addition to Assumption 2.1, that there exists some $\alpha > 0$ such that

$$\big(\nabla F(x) - \nabla F(y)\big)'(x - y) \geq \alpha \|x - y\|_2^2, \qquad \forall x, y \in \Re^n. \tag{2.13}$$

Then there exists a unique vector $x^* \in \Re^n$ that minimizes F. Furthermore, provided that γ is chosen positive and small enough, the sequence $\{x(t)\}$ generated by the gradient algorithm (2.3) converges to x^* geometrically.

Proof. Inequality (2.13) implies that the mapping $T : \Re^n \mapsto \Re^n$ defined by $T(x) = x - \gamma \nabla F(x)$ is a contraction with respect to the Euclidean norm $\|\cdot\|_2$, provided that γ is positive and sufficiently small. (Use Prop. 1.12 of Subsection 3.1.3, specialized to the case of a single block–component.) In particular, the mapping T has a unique fixed point x^* and the sequence generated by the gradient algorithm $x := T(x)$ converges to x^* geometrically. Such a fixed point satisfies $\nabla F(x^*) = 0$. Inequality (2.13) also implies that the function F is strictly convex (Prop. A.41 in Appendix A). It follows that x^* minimizes F [Prop. A.39(c) in Appendix A]. The strict convexity of F also implies that no other minimizing points of F exist [Prop. A.35(g) in Appendix A]. **Q.E.D.**

Any function F satisfying the condition (2.13) is called *strongly convex*. In the case where F is twice continuously differentiable, strong convexity is equivalent to positive definiteness of $\nabla^2 F(x)$, uniformly in x (Prop. A.41 in Appendix A). Intuitively, strong convexity amounts to assuming that the curvature of F is positive and bounded away from zero at every point. It should be also noticed that strong convexity of F is equivalent to strong monotonicity of ∇F (strong monotonicity was defined in Subsection 3.1.3). It turns out that under strong convexity, the Jacobi and Gauss–Seidel algorithms also converge to the optimal solution geometrically, but the proofs are omitted.

3.2.4 Nonlinear Algorithms

The algorithms considered so far are called *linearized* algorithms, because the update is a linear function of $\nabla F(x)$. This is in contrast to *nonlinear* or *coordinate descent* algorithms, which are based on a different idea. In the latter class of algorithms, we fix all of the components of x to some value, except for the ith component, and then we minimize $F(x)$ with respect to x_i. This procedure is repeated, leading to an iterative algorithm. There are two alternative implementations; in the first, called the *nonlinear Jacobi* algorithm, the minimizations with respect to the different components x_i are carried out simultaneously (see Fig. 3.2.2); in the second, called the *nonlinear Gauss–Seidel* algorithm, the minimizations are carried out successively for each component (see Fig. 3.2.3). Notice that each step involves the solution of one–dimensional minimization problems that are in many cases easy to solve with practically adequate precision.

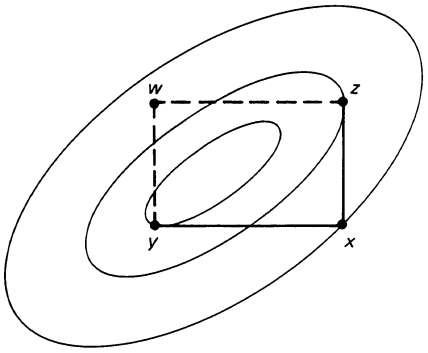

Figure 3.2.2 Illustration of an iteration of the nonlinear Jacobi algorithm. Given an initial vector x, we obtain the vectors y and z by minimizing along the first (respectively, second) coordinate. By combining the updates of both components, we obtain the new vector w.

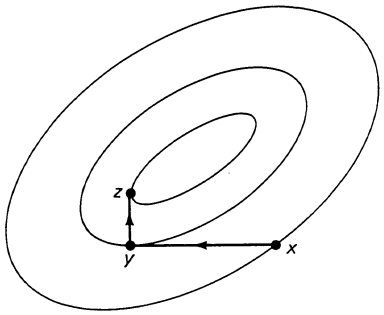

Figure 3.2.3 Illustration of an iteration of the nonlinear Gauss–Seidel algorithm. Given an initial vector x, we minimize with respect to the first coordinate to obtain the vector y, and then along the second coordinate to obtain the vector z.

Mathematically, the nonlinear Jacobi algorithm is described by the equation

$$x_i(t+1) = \arg\min_{x_i} F\Big(x_1(t), \ldots, x_{i-1}(t), x_i, x_{i+1}(t), \ldots, x_n(t)\Big) \tag{2.14}$$

and the nonlinear Gauss–Seidel algorithm by the equation

$$x_i(t+1) = \arg\min_{x_i} F\Big(x_1(t+1), \ldots, x_{i-1}(t+1), x_i, x_{i+1}(t), \ldots, x_n(t)\Big). \tag{2.15}$$

We are assuming here that a minimizing x_i always exists; if several minimizing x_i exist, $x_i(t+1)$ is chosen arbitrarily from the set of minimizing values.

Proposition 2.5. (*Convergence of the Nonlinear Gauss–Seidel Algorithm*) Suppose that $F : \Re^n \mapsto \Re$ is continuously differentiable and convex. Furthermore, suppose that for each i, F is a strictly convex function of x_i, when the values of the other components of x are held constant. Let $\{x(t)\}$ be the sequence generated by the nonlinear Gauss–Seidel algorithm, assumed to be well defined. Then, every limit point of $\{x(t)\}$ minimizes F over $\Re^n$.

The proof of Prop. 2.5 is omitted because it is a special case of a more general result to be proved later (Prop. 3.9 in Section 3.3). Let us just point out that Prop. 2.5

is derived using the descent approach for proving convergence. In particular, successive minimizations cannot increase the value of F. This shows that $F\big(x(t+1)\big) \leq F\big(x(t)\big)$ and implies the convergence of $F\big(x(t)\big)$ provided that F is bounded below. If F is not differentiable, the nonlinear Gauss–Seidel method can fail to converge to the minimum of F because it can stop at a nonoptimal "corner" point at which F is nondifferentiable and from which F cannot be reduced along any coordinate (Exercise 2.2). This difficulty will be encountered in the context of network flow problems in Chapter 5.

The proof just outlined fails altogether in the case of the nonlinear Jacobi algorithm; even though the minimization with respect to each coordinate cannot increase the value of F, the fact that these minimizations are carried out simultaneously allows the possibility that $F\big(x(t+1)\big) > F\big(x(t)\big)$. Convergence of the nonlinear Jacobi algorithm can be established using the results on contraction mappings of Section 3.1, under certain assumptions. We have, for example, the following result which is proved in more generality in the next section (Prop. 3.10).

Proposition 2.6. (*Convergence of Nonlinear Algorithms under Contraction Assumptions*) Let $F : \Re^n \mapsto \Re$ be continuously differentiable, let γ be a positive scalar, and suppose that the mapping $T : \Re^n \mapsto \Re^n$, defined by $T(x) = x - \gamma \nabla F(x)$, is a contraction with respect to a weighted maximum norm. Then, there exists a unique vector x^* which minimizes F over $\Re^n$. Furthermore, the nonlinear Jacobi and Gauss–Seidel algorithms are well defined, that is, a minimizing $\dot{x}_i$ in Eqs. (2.14) and (2.15) always exists. Finally, the sequence $\{x(t)\}$ generated by either of these algorithms converges to x^* geometrically.

The contraction assumption of Prop. 2.6 is satisfied if the matrix $\nabla^2 F(x)$ satisfies a diagonal dominance condition [see Prop. 1.11 with the identification $f(x) = \nabla F(x)$]. A weaker condition is diagonal dominance with respect to some set of weights (see Exercise 1.3).

A different version of the nonlinear Gauss–Seidel algorithm is obtained if instead of minimizing with respect to a single component at a time, we decompose $\Re^n$ as a Cartesian product $\prod_{i=1}^m \Re^{n_i}$, and at each stage, we minimize with respect to the n_i–dimensional subvector x_i. Proposition 2.6 remains valid and such a Gauss–Seidel algorithm also converges, under a block–contraction assumption on the mapping $T(x) = x - \gamma \nabla F(x)$.

The machinery of contraction mappings of Section 3.1 could be also used to establish convergence of linearized algorithms. However, the results thus obtained are not any stronger than the results obtained using the descent approach.

EXERCISES

2.1. Suppose that the Lipschitz condition on ∇F of Assumption 2.1 is replaced by the following two conditions:

(i) For every bounded set $A \subset \Re^n$, there exists some constant K such that $\|\nabla F(x) - \nabla F(y)\|_2 \leq K\|x - y\|_2$ for all $x, y \in A$.

(ii) The set $\{x \mid F(x) \leq c\}$ is bounded for every $c \in \Re$.

(a) Show that condition (i) is always satisfied if F is twice continuously differentiable.

(b) Show that Prop. 2.1 remains valid provided that the stepsize γ is allowed to depend on the choice of the initial vector $x(0)$. *Hint:* Choose a stepsize that guarantees that $x(t)$ stays within the set $\{x \mid F(x) \leq F(x(0))\}$.

2.2. Show by means of an example that if F is continuous but not differentiable, then the nonlinear Jacobi and Gauss–Seidel algorithms can fail to converge to the minimum of F, even if F is strictly convex and has bounded level sets.

2.3. Suppose that F is quadratic of the form $F(x) = \frac{1}{2}x'Ax - b'x$, where A is an $n \times n$ positive definite symmetric matrix and $b \in \Re^n$ is given. Show that the Lipschitz condition $\|\nabla F(x) - \nabla F(y)\|_2 \leq K\|x - y\|_2$ is satisfied with K equal to the maximal eigenvalue of A. Consider also the scaled gradient iteration $x(t+1) = x(t) - \gamma M^{-1}\nabla F(x(t))$, where M is positive definite and symmetric. Show that the method converges to $x^* = A^{-1}b$ if $\gamma \in (0, 2/\overline{K})$, where $\overline{K}$ is the maximum eigenvalue of $M^{-1/2}AM^{-1/2}$.

3.3 CONSTRAINED OPTIMIZATION

We consider in this section the problem of minimizing a cost function $F : \Re^n \mapsto \Re$ over a set $X \subset \Re^n$. Throughout, we assume that F is continuously differentiable and that X is nonempty, closed, and convex.

3.3.1 Optimality Conditions and the Projection Theorem

We start with a set of necessary and sufficient conditions for a vector $x \in X$ to be optimal.

Proposition 3.1. (*Optimality Conditions*)

(a) If a vector $x \in X$ minimizes F over X, then $(y - x)'\nabla F(x) \geq 0$ for every $y \in X$.

(b) If F is also convex on the set X, then the condition of part (a) is also sufficient for x to minimize F over X.

Proof.

(a) Suppose that $(y - x)'\nabla F(x) < 0$ for some $y \in X$. Since this is the directional derivative of F along the direction of $y - x$, it follows that there exists some $\epsilon \in (0, 1)$ such that $F(x + \epsilon(y - x)) < F(x)$. Then, $x + \epsilon(y - x) \in X$, because X is convex, which proves that x does not minimize F over the set X.

(b) Suppose that $(y - x)'\nabla F(x) \geq 0$ holds for every $y \in X$. Then, using the convexity of F (Prop. A.39 in Appendix A), we obtain $F(y) \geq F(x) + (y - x)'\nabla F(x) \geq F(x)$ for every $y \in X$, and, therefore, x minimizes F over X. **Q.E.D.**

The linearized algorithms of Section 3.2 are not applicable to constrained optimization because, even if we start inside the feasible set X, an update can take us outside that set. A simple remedy is to project back to the set X whenever such a situation arises.

We use the notation $[x]^+$ to denote the orthogonal projection (with respect to the Euclidean norm) of a vector x onto the convex set X. In particular, $[x]^+$ is defined by

$$[x]^+ = \arg\min_{z \in X} \|z - x\|_2. \tag{3.1}$$

The following result ensures that $[x]^+$ is well defined and also provides some useful properties of the projection.

Proposition 3.2. *(Projection Theorem)*

(a) For every $x \in \Re^n$, there exists a unique $z \in X$ that minimizes $\|z - x\|_2$ over all $z \in X$, and will be denoted by $[x]^+$.

(b) Given some $x \in \Re^n$, a vector $z \in X$ is equal to $[x]^+$ if and only if $(y - z)'(x - z) \leq 0$ for all $y \in X$.

(c) The mapping $f : \Re^n \mapsto X$ defined by $f(x) = [x]^+$ is continuous and nonexpansive, that is, $\|[x]^+ - [y]^+\|_2 \leq \|x - y\|_2$ for all $x, y \in \Re^n$.

Proof.

(a) Let x be fixed and let w be some element of X. Minimizing $\|x - z\|_2$ over all $z \in X$ is equivalent to minimizing the same function over all $z \in X$ such that $\|x - z\|_2 \leq \|x - w\|_2$, which is a compact set. Furthermore, the function g defined by $g(z) = \|z - x\|_2^2$ is continuous. Existence follows because a continuous function on a compact set always attains its minimum (Prop. A.8 in Appendix A).

To prove uniqueness, notice that the square of the Euclidean norm is a strictly convex function of its argument [Prop. A.40(d) in Appendix A]. Therefore, g is strictly convex and it follows that its minimum is attained at a unique point [Prop. A.35(g) in Appendix A].

(b) The vector $[x]^+$ is the minimizer of $g(z)$ over all $z \in X$. Notice that $\nabla g(z) = 2(z - x)$ and the result follows from the optimality conditions for constrained optimization problems (Prop. 3.1). (See Fig. 3.3.1 for an illustration of this result.)

(c) Let x and y be elements of $\Re^n$. From part (b), we have $\big(v - [x]^+\big)'\big(x - [x]^+\big) \leq 0$ for all $v \in X$. Since $[y]^+ \in X$, we obtain

$$\big([y]^+ - [x]^+\big)'\big(x - [x]^+\big) \leq 0.$$

Similarly,

$$\big([x]^+ - [y]^+\big)'\big(y - [y]^+\big) \leq 0.$$

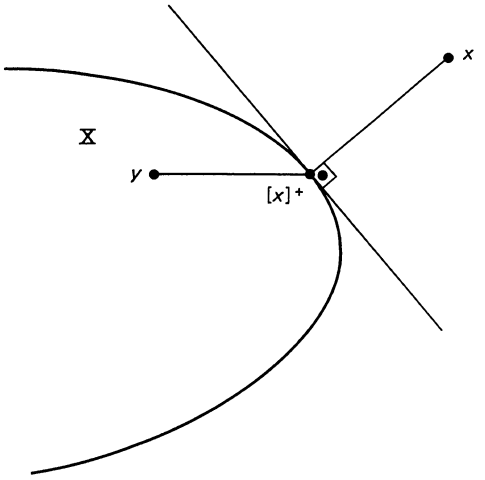

Figure 3.3.1 Illustration of the condition satisfied by the projection $[x]^+$. When the vector x is projected on the set X, the vector $x - [x]^+$ is normal to a plane supporting X at $[x]^+$. Each vector $y \in X$ lies on the other side of that plane, so the vectors $x - [x]^+$ and $y - [x]^+$ form an angle larger than or equal to 90 degrees or, equivalently, $\big(y - [x]^+\big)'\big(x - [x]^+\big) \leq 0$.

Adding these two inequalities and rearranging, we obtain

$$\|[y]^+ - [x]^+\|_2^2 \leq \big([y]^+ - [x]^+\big)'(y - x) \leq \|[y]^+ - [x]^+\|_2 \cdot \|y - x\|_2,$$

which proves that $[\cdot]^+$ is nonexpansive and *a fortiori* continuous. **Q.E.D.**

3.3.2 The Gradient Projection Algorithm

The gradient projection algorithm generalizes the gradient algorithm to the case where there are constraints, and is described by the equation

$$x(t+1) = \Big[x(t) - \gamma \nabla F\big(x(t)\big)\Big]^+, \tag{3.2}$$

where γ is a positive stepsize (see Fig. 3.3.2). Let $T : X \mapsto X$ be the mapping that corresponds to one iteration of this algorithm, that is,

$$T(x) = \Big[x - \gamma \nabla F(x)\Big]^+.$$

For an equivalent definition of the mapping T, notice that $T(x)$ is the unique vector y that minimizes $\|y - x + \gamma \nabla F(x)\|_2^2$ over all $y \in X$. After expanding this quadratic function, discarding the term $\gamma^2 \|\nabla F(x)\|_2^2$, which does not depend on y, and dividing by 2γ, we conclude that $T(x)$ is the unique minimizer of

$$(y - x)' \nabla F(x) + \frac{1}{2\gamma} \|y - x\|_2^2, \tag{3.3}$$

over all $y \in X$.

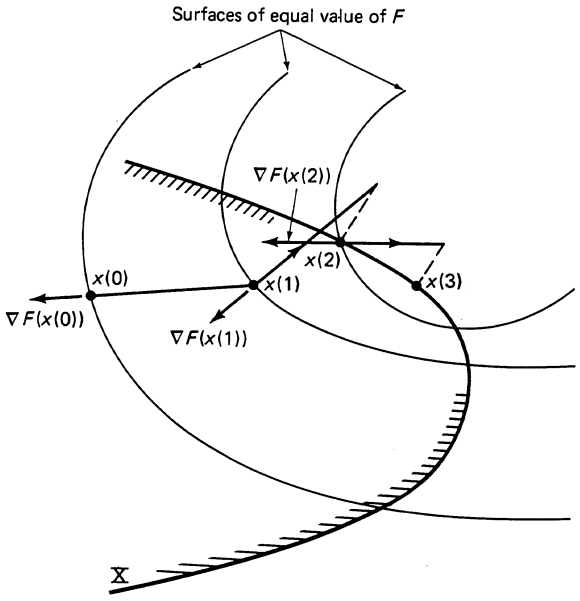

Figure 3.3.2 Illustration of a few iterations of the gradient projection method. Here $x(1) - \gamma \nabla F\big(x(1)\big)$ and $x(2) - \gamma \nabla F\big(x(2)\big)$ lie outside the feasible set X. These vectors are being projected on X in order to obtain $x(2)$ and $x(3)$, respectively.

We now study the convergence of the gradient projection algorithm under the same assumptions as in unconstrained optimization.

Assumption 3.1.

(a) There holds $F(x) \geq 0$ for all $x \in X$.

(b) (*Lipschitz Continuity of* ∇F) The function F is continuously differentiable and there exists a constant K such that

$$\|\nabla F(x) - \nabla F(y)\|_2 \leq K\|x - y\|_2, \qquad \forall x, y \in X. \tag{3.4}$$

The following result shows that for γ sufficiently small, each iteration of the gradient projection algorithm decreases the value of the cost function, unless a fixed point of the iteration mapping T has been reached.

Proposition 3.3. (*Properties of the Gradient Projection Mapping*) If F satisfies the Lipschitz condition of Assumption 3.1(b), γ is positive, and $x \in X$, then:

(a) $F\big(T(x)\big) \leq F(x) - (1/\gamma - K/2)\|T(x) - x\|_2^2$.

(b) We have $T(x) = x$ if and only if $(y - x)'\nabla F(x) \geq 0$ for all $y \in X$. In particular, if F is convex on the set X, we have $T(x) = x$ if and only if x minimizes F over the set X.

(c) The mapping T is continuous.

$$\big(y - T(x)\big)'\big(x - \gamma\nabla F(x) - T(x)\big) \leq 0, \qquad \forall y \in X. \tag{3.5}$$

In particular, letting $y = x$, we obtain

$$\big(x - T(x)\big)'\big(x - \gamma\nabla F(x) - T(x)\big) \leq 0,$$

which yields $\gamma\big(T(x) - x\big)'\nabla F(x) \leq -\|T(x) - x\|_2^2$. Using the Descent Lemma (Lemma 2.1), we obtain

$$\begin{aligned} F\big(T(x)\big) &\leq F(x) + \big(T(x) - x\big)'\nabla F(x) + \frac{K}{2}\|T(x) - x\|_2^2 \\ &\leq F(x) - \left(\frac{1}{\gamma} - \frac{K}{2}\right)\|T(x) - x\|_2^2, \end{aligned}$$

which proves part (a).

(b) By the Projection Theorem, the relation (3.5) can be used as the definition of $T(x)$. Thus, if $T(x) = x$, then $(y - x)'\gamma\nabla F(x) \geq 0$ for all $y \in X$. Conversely, if $(y - x)'\gamma\nabla F(x) \geq 0$ for every $y \in X$, then $(y - x)'(x - \gamma\nabla F(x) - x) \leq 0$, and we conclude that $x = T(x)$. In the convex case, the result follows from the optimality conditions for constrained optimization (Prop. 3.1).

(c) Since F is continuously differentiable, the mapping $x \mapsto x - \gamma\nabla F(x)$ is continuous. Given that the projection mapping is also continuous [Prop. 3.2(c)], T is the composition of two continuous mappings and is therefore continuous. **Q.E.D.**

From Prop. 3.3, the convergence of the gradient projection algorithm is straightforward to establish. Let $\{x(t)\}$ be the sequence of vectors generated by the algorithm. Assuming that $0 < \gamma < 2/K$, Prop. 3.3(a) shows that the sequence $\big\{F\big(x(t)\big)\big\}$ is nonincreasing, and if F is bounded below, this sequence converges, while $T\big(x(t)\big) - x(t)$ converges to zero. Let x^* be a limit point of the sequence $\{x(t)\}$ and let $\{x(t_k)\}$ be a subsequence converging to x^*. Then, $T\big(x(t_k)\big)$ also converges to x^* and the continuity of T implies that $T(x^*) = x^*$. Then, Prop. 3.3(b) shows that $(y - x^*)'\nabla F(x^*) \geq 0$, for all $y \in X$. We have thus proved the following.

Proposition 3.4. (*Convergence of the Gradient Projection Algorithm*) Suppose that F satisfies Assumption 3.1. If $0 < \gamma < 2/K$ and if x^* is a limit point of the sequence $\{x(t)\}$ generated by the gradient projection algorithm (3.2), then $(y - x^*)'\nabla F(x^*) \geq 0$ for all $y \in X$. In particular, if F is convex on the set X, then x^* minimizes F over the set X.

Proposition 3.4 was proved by means of a descent argument. Using the general convergence properties of contracting iterations (Section 3.1), we can prove the following result, which provides us with a convergence rate estimate. The proof is omitted because it is almost identical to the proof of Prop. 2.4.

Proposition 3.5. (*Geometric Convergence for Strongly Convex Problems*) Suppose, in addition to Assumption 3.1, that there exists some $\alpha > 0$ such that

$$\big(\nabla F(x) - \nabla F(y)\big)'(x - y) \geq \alpha \|x - y\|_2^2, \qquad \forall x, y \in X.$$

Then, there exists a unique vector x^* that minimizes F over the set X. Furthermore, provided that γ is chosen positive and small enough, the sequence $\{x(t)\}$ generated by the gradient projection algorithm (3.2) converges to x^* geometrically.

3.3.3 Scaled Gradient Projection Algorithms

As in the case of unconstrained optimization, we may wish to scale the update direction $\nabla F\big(x(t)\big)$. We thus generalize the gradient projection algorithm (3.2) by letting

$$x(t+1) = \Big[x(t) - \gamma\big(M(t)\big)^{-1}\nabla F\big(x(t)\big)\Big]^+, \tag{3.6}$$

where $M(t)$ is an invertible scaling matrix. Typically, $M(t)$ would be chosen to approximate the Hessian matrix $\nabla^2 F\big(x(t)\big)$. For example, in a projected Jacobi method, $M(t)$ would be diagonal, with its diagonal entries being equal to the diagonal entries of $\nabla^2 F\big(x(t)\big)$, thus generalizing the linearized Jacobi algorithm (2.1) of Section 3.2. However, the algorithm (3.6) fails, in general, to converge to a minimizing point, as illustrated in Fig. 3.3.3. For convergence to be obtained, the projection should be carried out with respect to a different coordinate system (equivalently, with respect to a different norm) determined by $M(t)$. (An alternative approach is discussed in Exercise 3.3.)

Let us temporarily assume that $M(t)$ is symmetric and positive definite. We consider the norm $\|\cdot\|_{M(t)}$ defined by

$$\|x\|_{M(t)} = \big(x'M(t)x\big)^{1/2}.$$

We then define $[x]^+_{M(t)}$ as the vector y that minimizes $\|y - x\|_{M(t)}$ over all $y \in X$, and we replace Eq. (3.6) by the iteration

$$x(t+1) = \Big[x(t) - \gamma\big(M(t)\big)^{-1}\nabla F\big(x(t)\big)\Big]^+_{M(t)}. \tag{3.7}$$

As in the case of the unscaled gradient projection method, we can define $x(t+1)$ as the solution of a quadratic programming problem [cf. the expression (3.3)]. In particular, it can be seen that

$$x(t+1) = \arg\min_{y \in X}\left\{\frac{1}{2\gamma}\big(y - x(t)\big)'M(t)\big(y - x(t)\big) + \big(y - x(t)\big)'\nabla F\big(x(t)\big)\right\}. \tag{3.8}$$

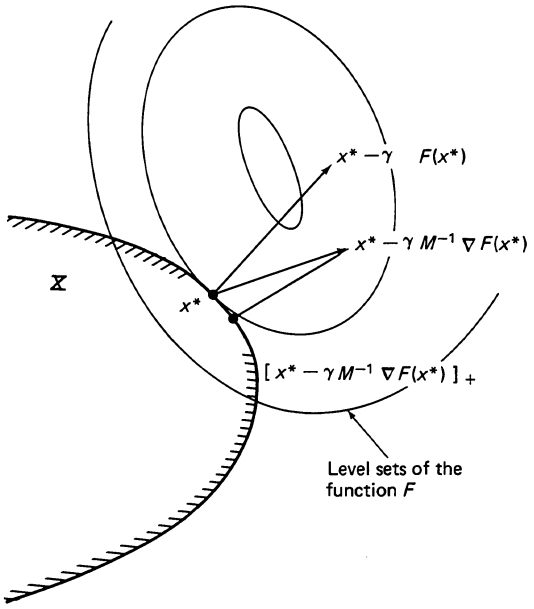

Figure 3.3.3 Illustration of failure of the algorithm $x(t+1) = \left[x(t) - \gamma M^{-1}\nabla F\big(x(t)\big)\right]^+$. Here, the point x^* minimizes the convex function F over the set X. However, the iteration $x := [x - \gamma M^{-1}\nabla F(x)]^+$ does not have x^* as a fixed point.

It is actually preferable to define $x(t+1)$ by means of the quadratic optimization in Eq. (3.8) rather than as a projection [cf. Eq. (3.7)], because the quadratic expression in Eq. (3.8) is well defined even if $M(t)$ is not an invertible matrix. (This provides some additional flexibility, which is sometimes useful. The algorithm that generates $x(t+1)$ according to Eq. (3.8) will be called the *scaled gradient projection* algorithm. For this algorithm to be well defined, we need to ensure that the minimum in Eq. (3.8) is attained at a unique element of X. The following auxiliary result provides sufficient conditions for this to be the case.

Proposition 3.6. Suppose that a matrix $M(t)$ is symmetric and satisfies the positivity condition

$$(x-y)'M(t)(x-y) \geq \alpha \|x-y\|_2^2, \qquad \forall x, y \in X, \tag{3.9}$$

where α is some positive constant. Then, the minimum in the quadratic programming problem of Eq. (3.8) is attained at a unique vector $y \in X$.

Proof. From inequality (3.9), it can be verified that the expression minimized in Eq. (3.8), viewed as a function of y, is a strictly convex function on the set X, which proves uniqueness. Furthermore, by inequality (3.9), this expression goes to infinity when $\|y\|_2$ goes to infinity. Therefore, the minimization can be restricted to a compact subset of X. Existence of a minimizing vector follows because a continuous function on a compact set attains its minimum (Weierstrass' theorem given as Prop. A.8 in Appendix A). **Q.E.D.**

The positivity condition (3.9) is satisfied if $M(t) - \alpha I$ is symmetric nonnegative definite. In that case, α is a lower bound for the smallest eigenvalue of $M(t)$. However, this is stronger than necessary. For example, if the set $\{x - y \mid x \in X, y \in X\}$ is contained in a proper subspace of $\Re^n$, then it is only the action of $M(t)$ on vectors in that subspace that matters. Roughly speaking, condition (3.9) states that the restriction of $M(t)$ on such a subspace is positive definite.

The following result generalizes Props. 3.2–3.4 to the case where scaling is used. The proof is similar to the proof of Props. 3.2–3.4 and is outlined in Exercise 3.2.

Proposition 3.7. (*Properties and Convergence of the Scaled Projection Algorithm*) Let $\{M(t) \mid t = 0, 1, \ldots\}$ be a bounded sequence of $n \times n$ symmetric matrices and assume that for some $\alpha > 0$, each $M(t)$ satisfies the positivity condition (3.9). Let $F : \Re^n \mapsto \Re$ satisfy Assumption 3.1.

(a) For every $x \in \Re^n$, there exists a unique $y \in X$ that minimizes $(x-y)'M(t)(x-y)$ over the set X and will be denoted by $[x]^+_{M(t)}$, or $[x]^+_t$ for short.

(b) (*Scaled Projection Theorem*) Given some $x \in \Re^n$, a vector $z \in X$ is equal to $[x]^+_t$ if and only if $(y - z)'M(t)(x - z) \leq 0$ for all $y \in X$.

(c) There exists a constant A_1 such that

$$\left\| [x]^+_t - [y]^+_t \right\|_2 \leq A_1 \|x - y\|_2,$$

for every t and every $x, y \in \Re^n$.

(d) If $M(t)$ is also positive definite, then

$$\big([x]^+_t - [y]^+_t\big)'M(t)\big([x]^+_t - [y]^+_t\big) \leq (x - y)'M(t)(x - y), \qquad \forall x, y \in \Re^n.$$

Let $T_t : X \mapsto X$ be the mapping that corresponds to the tth iteration of the scaled gradient projection algorithm. That is, $x(t+1) = T_t\big(x(t)\big)$, where $x(t+1)$ is defined by the quadratic minimization in Eq. (3.8). We assume that γ is positive.

(e) We have $T_t(x) = x$ if and only if $(y-x)'\nabla F(x) \geq 0$ for every $y \in X$. In particular, if F is convex on the set X, we have $T_t(x) = x$ if and only if x minimizes F over the set X.

(f) There exists a constant A_2 such that

$$\|T_t(x) - T_t(y)\|_2 \leq A_2 \|x - y\|_2, \qquad \forall x, y \in X, \ \forall t.$$

(g) If γ is small enough, then there exists a positive constant A_3 such that $F\big(T_t(x)\big) \leq F(x) - A_3 \|T_t(x) - x\|_2^2$ for every $x \in X$ and every t.

(h) If γ is small enough, then any limit point x^* of the sequence $\{x(t)\}$ generated by the scaled gradient projection algorithm satisfies $(y - x^*)'\nabla F(x^*) \geq 0$ for every $y \in X$. If F is also convex on the set X then x^* minimizes F over the set X.

3.3.4 The Case of a Product Constraint Set: Parallel Implementations

The gradient projection algorithm is not, in general, amenable to parallel implementation. Even though the computation of $x - \gamma \nabla F(x)$ can be parallelized in the obvious manner, the computation of the projection is a nontrivial optimization problem involving all components of x. However, in the important special case where the set X is a box (i.e., $X = \prod_{i=1}^{n} [a_i, b_i]$ for some real numbers a_i, b_i), the projection of x on X is obtained by projecting the ith component of x on the interval $[a_i, b_i]$, which is straightforward and can be done independently for each component (see Fig. 3.3.4).

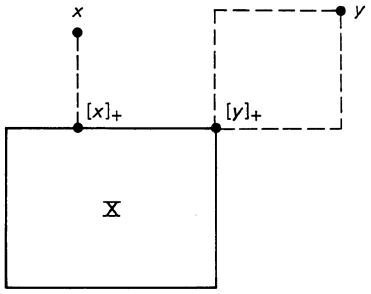

Figure 3.3.4 Illustration of the projection on a box. The ith component of the projection of a vector is the projection of its ith component.

More generally, suppose that the space R^n is represented as the Cartesian product of spaces $\Re^{n_i}$, where $n_1 + \cdots + n_m = n$, and that the constraint set X is a Cartesian product of sets X_i, where each X_i is a closed convex subset of $\Re^{n_i}$. Accordingly, we represent any vector $x \in \Re^n$ in the form $x = (x_1, \ldots, x_m)$, where each x_i is an element of R^{n_i}. It is easily seen that the projection of x on X is equal to the vector $([x_1]_1^+, \ldots, [x_m]_m^+)$, where $[x_i]_i^+$ is the projection of x_i onto X_i. The same discussion applies to the scaled gradient projection algorithm, provided that the scaling matrices $M(t)$ are block–diagonal. To see this, suppose that $M(t)$ is block–diagonal, the ith diagonal block $M_i(t)$ being of dimension $n_i \times n_i$. The quadratic expression minimized in Eq. (3.8) can be rewritten as

$$\sum_{i=1}^{m} \left[\frac{1}{2\gamma} \big(y_i - x_i(t)\big)' M_i(t) \big(y_i - x_i(t)\big) + \big(y_i - x_i(t)\big)' \nabla_i F\big(x(t)\big) \right]. \tag{3.10}$$

Evidently, when X is a Cartesian product, minimizing the above quadratic expression over all $y \in X$ is equivalent to minimizing the ith summand over all $y_i \in X_i$ for each i. A parallel algorithm is then obtained because these minimizations can be carried out independently by different processors.

The assumption that X is a Cartesian product opens up the possibility for a Gauss–Seidel version of the gradient projection algorithm. We only discuss the case of identity scaling. The results are similar for the case of block–diagonal scaling, as long as the scaling matrices $M(t)$ satisfy the positivity condition (3.9).

The Gauss–Seidel algorithm (with identity scaling) is defined by the iteration

$$x_i(t+1) = \Big[x_i(t) - \gamma \nabla_i F\big(z(i,t)\big)\Big]_i^+, \tag{3.11}$$

where $z(i,t) = \big(x_1(t+1), \ldots, x_{i-1}(t+1), x_i(t), \ldots, x_m(t)\big)$, $1 \leq i \leq m$. To simplify notation in the following, we also let $z(m+1,t) = x(t+1)$.

Proposition 3.8. (*Convergence of the Gauss–Seidel Gradient Projection Algorithm*) If $F : \Re^n \mapsto \Re$ satisfies Assumption 3.1 and if γ is chosen positive and small enough, then any limit point x^* of the sequence $\{x(t)\}$ generated by the Gauss–Seidel algorithm (3.11) satisfies $(y - x^*)'\nabla F(x^*) \geq 0$ for all $y \in X$.

Proof. We apply Prop. 3.3(a) on the function F, viewed as a function of x_i alone, to conclude that if γ is sufficiently small, there exists some $A > 0$ such that

$$F\big(z(i+1,t)\big) \leq F\big(z(i,t)\big) - A\big\|z(i+1,t) - z(i,t)\big\|_2^2, \qquad \forall t.$$

It follows that $F\big(x(t)\big)$ is nonincreasing and therefore converges. This implies that $z(i+1,t) - z(i,t)$ converges to zero for each i. In particular, $x(t+1) - x(t)$ converges to zero. Let x^* be a limit point of the sequence $\{x(t)\}$. Taking the limit in Eq. (3.11), along a sequence of times such that $x(t)$ converges to x^*, and using the continuity of ∇F and of the projection, we obtain $x_i^* = \big[x_i^* - \gamma \nabla_i F(x^*)\big]_i^+$ for all i. Thus, $x^* = \big[x^* - \gamma \nabla F(x^*)\big]^+$ and the result follows from Prop. 3.3(b). **Q.E.D.**

3.3.5 Nonlinear Algorithms

Assuming that X is a Cartesian product, it is meaningful to consider the nonlinear Jacobi and Gauss–Seidel algorithms that are the natural extensions to the constrained case of the nonlinear algorithms introduced in Subsection 3.2.4. The *nonlinear Jacobi* algorithm is defined by

$$x_i(t+1) = \arg\min_{x_i \in X_i} F\Big(x_1(t), \ldots, x_{i-1}(t), x_i, x_{i+1}(t), \ldots, x_m(t)\Big), \tag{3.12}$$

and the *nonlinear Gauss–Seidel* algorithm is defined by

$$x_i(t+1) = \arg\min_{x_i \in X_i} F\Big(x_1(t+1), \ldots, x_{i-1}(t+1), x_i, x_{i+1}(t), \ldots, x_m(t)\Big). \tag{3.13}$$

Convergence of the nonlinear Gauss–Seidel algorithm can be established using the descent approach.

Proposition 3.9. (*Convergence of the Nonlinear Gauss–Seidel Algorithm*) Suppose that $F : \Re^n \mapsto \Re$ is continuously differentiable and convex on the set X. Furthermore, suppose that for each i, F is a strictly convex function of x_i, when the values of the other components of x are held constant. Let $\{x(t)\}$ be the sequence generated

by the nonlinear Gauss–Seidel algorithm, assumed to be well defined. Then, every limit point of $\{x(t)\}$ minimizes F over X.

Proof. Let

$$z^i(t) = \big(x_1(t+1), \ldots, x_i(t+1), x_{i+1}(t), \ldots, x_m(t)\big).$$

Using the definition (3.13) of the Gauss–Seidel iteration, we obtain

$$F\big(x(t)\big) \geq F\big(z^1(t)\big) \geq F\big(z^2(t)\big) \geq \cdots \geq F\big(z^{m-1}(t)\big) \geq F\big(x(t+1)\big), \qquad \forall t. \quad (3.14)$$

Let $x^* = \big(x_1^*, \ldots, x_m^*\big)$ be a limit point of the sequence $\{x(t)\}$. Notice that $x^* \in X$ because X is closed. Let $\{x(t_k)\}$ be a subsequence of $\{x(t)\}$ that converges to x^*. We notice from Eq. (3.14) that the sequence $\big\{F\big(x(t)\big)\big\}$ converges to either $-\infty$ or a finite real number. Using the convergence of $x(t_k)$ to x^* and the continuity of F, we see that $F\big(x(t_k)\big)$ converges to $F(x^*)$, and this implies that the entire sequence $\big\{F\big(x(t)\big)\big\}$ converges to $F(x^*)$. It now remains to show that x^* minimizes F over the set X.

We first show that $x_1(t_k+1) - x_1(t_k)$ converges to zero. Assume the contrary or, equivalently, that $z^1(t_k) - x(t_k)$ does not converge to zero. Let $\gamma(t_k) = \|z^1(t_k) - x(t_k)\|_2$. By possibly restricting to a subsequence of $\{t_k\}$, we may assume that there exists some $\gamma_0 > 0$ such that $\gamma(t_k) \geq \gamma_0$ for all k. Let $s^1(t_k) = \big(z^1(t_k) - x(t_k)\big)/\gamma(t_k)$. Thus, $z^1(t_k) = x(t_k) + \gamma(t_k)s^1(t_k)$, $\|s^1(t_k)\|_2 = 1$, and $s^1(t_k)$ differs from zero only along the first block–component. Notice that $s^1(t_k)$ belongs to a compact set and therefore has a limit point $\bar{s}^1$. By restricting to a further subsequence of $\{t_k\}$, we assume that $s^1(t_k)$ converges to $\bar{s}^1$.

Let us fix some $\epsilon \in [0,1]$. Notice that $0 \leq \epsilon\gamma_0 \leq \gamma(t_k)$. Therefore, $x(t_k) + \epsilon\gamma_0 s^1(t_k)$ lies on the segment joining $x(t_k)$ and $x(t_k) + \gamma(t_k)s^1(t_k) = z^1(t_k)$ and belongs to X because X is convex. Using the convexity of F, and the fact that $z^1(t_k)$ minimizes F over all x that differ from $x(t_k)$ along the first block–component, we obtain

$$F\big(z^1(t_k)\big) = F\big(x(t_k) + \gamma(t_k)s^1(t_k)\big) \leq F\big(x(t_k) + \epsilon\gamma_0 s^1(t_k)\big) \leq F\big(x(t_k)\big).$$

Since $F\big(x(t)\big)$ converges to $F(x^*)$, Eq. (3.14) shows that $F\big(z^1(t)\big)$ also converges to $F(x^*)$. We now take the limit as k tends to infinity, to obtain $F(x^*) \leq F\big(x^* + \epsilon\gamma_0\bar{s}^1\big) \leq F(x^*)$. We conclude that $F(x^*) = F\big(x^* + \epsilon\gamma_0\bar{s}^1\big)$, for every $\epsilon \in [0,1]$. Since $\gamma_0\bar{s}^1 \neq 0$, this contradicts the strict convexity of F as a function of the first block–component. This contradiction establishes that $x_1(t_{k+1}) - x_1(t_k)$ converges to zero. In particular, $z^1(t_k)$ converges to x^*.

From the definition (3.13) of the algorithm, we have

$$F\big(z^1(t_k)\big) \leq F\big(x_1, x_2(t_k), \ldots, x_m(t_k)\big), \qquad \forall x_1 \in X_1.$$

Taking the limit as k tends to infinity, we obtain

$$F(x^*) \leq F(x_1, x_2^*, \ldots, x_m^*), \qquad \forall x_1 \in X_1.$$

Using the optimality conditions for constrained optimization (Prop. 3.1), we conclude that

$$\nabla_1 F(x^*)'(x_1 - x_1^*) \geq 0, \qquad \forall x_1 \in X_1.$$

Let us now consider the sequence $\{z^1(t_k)\}$. We have already shown that $z^1(t_k)$ converges to x^*. A verbatim repetition of the preceding argument shows that $x_2(t_{k+1}) - x_2(t_k)$ converges to zero and $\nabla_2 F(x^*)'(x_2 - x_2^*) \geq 0$ for every $x_2 \in X_2$. Continuing inductively, we obtain $\nabla_i F(x^*)'(x_i - x_i^*) \geq 0$ for every $x_i \in X_i$ and for every i. Adding these inequalities, and using the Cartesian product structure of the set X, we conclude that $\nabla F(x^*)'(x - x^*) \geq 0$ for every $x \in X$. In view of the convexity of F, this shows that x^* minimizes F over the set X (Prop. 3.1). **Q.E.D.**

Notice that by letting $X_i = \Re^{n_i}$ in this proposition, we have also established the convergence of the nonlinear Gauss–Seidel algorithm for unconstrained optimization (Prop. 2.5).

The convergence of the nonlinear Jacobi algorithm can be established under suitable contraction assumptions on the mapping $x := x - \gamma \nabla F(x)$. (Sufficient conditions for this to be a contraction mapping have been furnished in Subsection 3.1.3.) In particular, the following result extends and provides a proof for the corresponding unconstrained optimization result (Prop. 2.6 in Section 3.2).

Proposition 3.10. (*Convergence of Nonlinear Algorithms under Contraction Assumptions*) Let $F : \Re^n \mapsto \Re$ be continuously differentiable, let γ be a positive scalar, and suppose that the mapping $R : X \mapsto \Re^n$, defined by $R(x) = x - \gamma \nabla F(x)$, is a contraction with respect to the block–maximum norm $\|x\| = \|(x_1, \ldots, x_m)\| = \max_i \|x_i\|_i / w_i$, where each $\|\cdot\|_i$ is the Euclidean norm on $\Re^{n_i}$ and each w_i is a positive scalar. Then, there exists a unique vector x^* which minimizes F over X. Furthermore, the nonlinear Jacobi and Gauss–Seidel algorithms are well defined, that is, a minimizing x_i in Eqs. (3.12) and (3.13) always exists. Finally, the sequence $\{x(t)\}$ generated by either of these algorithms converges to x^* geometrically.

Proof. The contraction assumption on R and the nonexpansive property of the projection [Prop. 3.2(c)] imply that the mapping $T : X \mapsto X$ defined by $T(x) = [x - \gamma \nabla F(x)]^+$, is also a contraction. In particular, T has a unique fixed point $x^* \in X$, and the iteration $x := T(x)$ converges to x^* geometrically. Our first task is to show that x^* minimizes F over the set X.

Since R is a contraction, we have

$$\gamma \|\nabla F(x) - \nabla F(y)\| = \big\|(x-y) - \big(R(x) - R(y)\big)\big\| \leq \|x-y\| + \|R(x) - R(y)\| \leq 2\|x-y\|.$$

This proves that ∇F satisfies the Lipschitz Continuity Assumption 3.1(b).

We now introduce the notation R^δ and T^δ to denote the mappings obtained from R and T, respectively, when the stepsize parameter γ is replaced by δ. For every $\delta \in (0, \gamma]$, we have $R^\delta(x) = (1 - \delta/\gamma)x + (\delta/\gamma)R(x)$. Thus,

$$\|R^\delta(x) - R^\delta(y)\| \leq \Big(1 - \frac{\delta}{\gamma}\Big)\|x - y\| + \frac{\delta}{\gamma}\|R(x) - R(y)\| \leq \Big(1 - \frac{\delta}{\gamma} + \frac{\alpha\delta}{\gamma}\Big)\|x - y\|,$$

where α is the contraction modulus of R. Since $\alpha < 1$, it is seen that R^δ is a contraction. It follows that T^δ is also a contraction for every $\delta \in (0, \gamma]$. By Prop. 3.3(b), we see that $x \in X$ is a fixed point of T^δ if and only if $(y - x)'\nabla F(x) \geq 0$ for all $y \in X$. Since this condition is independent of the value of δ, we conclude that the fixed point x^* of T is also the unique fixed point of T^δ for every $\delta \in (0, \gamma]$.

Let us now consider some $\delta \in (0, \gamma]$ which is sufficiently small so that the iteration $x := T^\delta(x)$ has the property $F\big(T^\delta(x)\big) \leq F(x) - A\|T^\delta(x) - x\|^2$ for every $x \in X$, where A is some positive constant. [Such a δ exists because of the descent properties of gradient projection iterations; see Prop. 3.3(a).] Consider some $x(0) \in X$ different from x^* and let $x(t+1) = T^\delta\big(x(t)\big)$, for $t = 0, 1, \ldots$. Since $x(0) \neq x^*$, we have $T^\delta\big(x(0)\big) \neq x(0)$, and the preceding remarks imply that $F\big(x(1)\big) < F\big(x(0)\big)$. Furthermore, $x(t)$ converges to x^*, and the sequence $\big\{F\big(x(t)\big)\big\}$ is monotonically nonincreasing and converges to $F(x^*)$. Therefore, $F(x^*) \leq F\big(x(1)\big) < F\big(x(0)\big)$. Since this inequality is true whenever $x(0) \in X$ and $x(0) \neq x^*$, it follows that x^* is the unique minimizing point of the function F.

Let us now fix some index i and represent the vector x in the form $x = (x_i, \bar{x})$ where $x_i \in X_i$ and $\bar{x}$ is the vector with the remaining block–components of x. We fix $\bar{x}$ and view $F(x) = F(x_i, \bar{x})$ as a function of x_i alone. The mapping $R_i : X_i \mapsto \Re^{n_i}$, defined by $R_i(x_i) = x_i - \gamma\nabla_i F(x_i, \bar{x})$ inherits the contraction property of R. (This is because a block–maximum norm is employed.) Our previous arguments can be repeated to establish that for any fixed $\bar{x}$, there exists a unique $x_i \in X_i$ which satisfies $x_i = \big[x_i - \gamma\nabla_i F(x_i, \bar{x})\big]_i^+$, and such an x_i is the unique minimizer of $F(x_i, \bar{x})$ over the set X_i. This is exactly the type of minimization carried out in Eqs. (3.12) and (3.13). Thus, the nonlinear Jacobi and Gauss–Seidel algorithms are well defined.

We now consider the component solution method for solving the fixed point problem $T(x) = x$. We recall from Subsection 3.1.2 that given some current vector $x(t)$, the component solution method determines $x_i(t+1)$ by solving the equation $x_i = T_i(x) = \big[x_i - \gamma\nabla_i F(x)\big]_i^+$ for x_i while fixing the remaining block–components of x at the values determined by $x(t)$. Given the discussion of the preceding paragraph, this is equivalent to determining x_i by minimizing F over all $x_i \in X_i$ while keeping the values of the remaining block–components fixed. But this is precisely the nonlinear Jacobi algorithm. Similarly, the nonlinear Gauss–Seidel algorithm coincides with the Gauss–Seidel version of the component solution method. We now apply our earlier convergence results for component solution methods for solving fixed point problems involving block–contractions (Prop. 1.7 in Subsection 3.1.2) to conclude that the sequences generated by the algorithms under consideration converge geometrically to the unique fixed point of T which as we have already established, is the unique minimizer of F over the set X. **Q.E.D.**

The nonlinear Jacobi algorithm can be parallelized by assigning a separate processor to each block–component x_i. The nonlinear Gauss–Seidel algorithm can be also parallelized, provided that a coloring scheme can be applied (see Subsection 1.2.4).

We may also consider hybrid methods which combine certain features of the Jacobi and Gauss–Seidel methods. For example, we could split the m block–components of x into two groups: $(x_1, \ldots, x_k)$ and $(x_{k+1}, \ldots, x_m)$ and use the update equations

$$x_i(t+1) = \arg\min_{x_i \in X_i} F\Big(x_1(t), \ldots, x_{i-1}(t), x_i, x_{i+1}(t), \ldots, x_m(t)\Big),$$

if $1 \leq i \leq k$, and

$$x_i(t+1) = \arg\min_{x_i \in X_i} F\Big(x_1(t+1), \ldots, x_k(t+1), x_{k+1}(t), \ldots, x_{i-1}(t), x_i, x_{i+1}(t), \ldots, x_m(t)\Big),$$

if $k+1 \leq i \leq m$. Thus each group of components is updated in Jacobi fashion but the updates of the second group incorporate the results of the update for the first group, as in Gauss–Seidel iterations. (Generalizations to more than two groups are clearly possible.) As long as the number of processors is smaller than the number of components in each group, we obtain the same parallelism as for the nonlinear Jacobi method, while convergence could be faster due to the Gauss–Seidel element. The convergence result of Prop. 3.10 remains true for such hybrid methods as well.

Notice that a nonlinear method can be viewed as a procedure whereby at each stage, an infinite number of iterations of a linearized algorithm is performed on the same component, until the cost function is minimized with respect to that component. We could have also considered intermediate methods whereby a limited number of linearized iterations on the same component is performed at each stage. Such methods are also convergent under the block–contraction assumption of Prop. 3.10; their convergence is most easily established by viewing them as special cases of asynchronous iterations of the type studied in Chapter 6.

EXERCISES

3.1. (**Projection on a Subspace.**) Let X be a subspace of $\Re^n$ and consider the mapping $f : \Re^n \mapsto \Re^n$, defined by $f(x) = [x]^+$.

(a) Show that $y = f(x)$ if and only if $y \in X$ and $(y - x)'z = 0$ for every $z \in X$.

(b) Show that $f(ax + by) = af(x) + bf(y)$ for every x, $y \in \Re^n$ and every a, $b \in \Re$. This establishes that the mapping f is of the form $f(x) = Px$, where P is an $n \times n$ matrix.

(c) Show that the matrix P defined in (b) has the following properties:

(i) $Px = x$ for every $x \in X$.

(ii) $P^2 = P$.

(iii) $\|x\|_2^2 = \|Px\|_2^2 + \|(I - P)x\|_2^2$ for every $x \in \Re^n$.

(iv) P is symmetric.

(d) Suppose that the subspace X is of the form $X = \{x \mid Ax = 0\}$, where A is an $m \times n$ matrix with the property that AA' is nonsingular. Find a formula for the matrix P of part (b) in terms of A.

Hint: To establish symmetry of P, show that $x'Py = x'P'y$ for all $x, y \in \Re^n$. In part (d), formulate the problem defining the projection of a vector on X and apply the optimality conditions.

3.2. Prove Prop. 3.7.

Hints: For part (a), show that $(x-y)'M(t)(x-y)$ is a strictly convex function of y when y is restricted to X. For part (b), use the optimality conditions of Prop. 3.1. For part (c), mimic the proof of Prop. 3.2 to show that

$$\big([y]_t^+ - [x]_t^+\big)'M(t)\big([y]_t^+ - [x]_t^+\big) \leq \big([y]_t^+ - [x]_t^+\big)'M(t)(x-y). \tag{3.15}$$

Then use inequality (3.9) for the left hand side of inequality (3.15), and the Schwartz inequality for the right hand side. For part (d), continue as in Prop. 3.2, using the norm $\|x\| = \big\|\big(M(t)\big)^{1/2}x\big\|_2$. For part (e), apply Prop. 3.1 to the minimization problem of Eq. (3.8). For part (f), use the optimality conditions for the problem in Eq. (3.8) and then proceed as in part (c). For part (g), proceed as in the proof of Prop. 3.3 and use inequality (3.9). Finally, for part (h), show that

$$\big(y - x(t+1)\big)'\Big(M(t)\big(x(t+1)-x(t)\big) + \gamma\nabla F\big(x(t)\big)\Big) \geq 0, \qquad \forall y \in X,$$

and take the limit along a sequence $\{t_k\}$ such that $x(t_k)$ and $x(t_k+1)$ converge to x^*.

3.3. [Ber82b] As illustrated in Fig. 3.3.3, the iteration (3.6) given by

$$x(t+1) = \Big[x(t) - \gamma\big(M(t)\big)^{-1}\nabla F\big(x(t)\big)\Big]^+ \tag{3.16}$$

may increase the value of F, no matter how $\gamma > 0$ is chosen. For this reason, the iteration was modified as in Eq. (3.7), so that the projection on X is carried out with respect to the norm corresponding to $M(t)$. An alternative for the case where X is the positive orthant is to suitably restrict the form of the matrix $M(t)$ in Eq. (3.16) so that for γ sufficiently small, a cost improvement is obtained.

Let $X = \{x \mid x_i \geq 0,\ i = 1,\ldots,n\}$. Suppose that $M(t)$ is symmetric positive definite and its elements satisfy

$$[M(t)]_{ij} = 0, \qquad \text{if } i \in I(t) \text{ and } i \neq j,$$

where

$$I(t) = \left\{ i \;\middle|\; x_i(t) = 0 \text{ and } \frac{\partial F}{\partial x_i}\big(x(t)\big) < 0 \right\}.$$

Show that if $x(t)$ is not optimal, there exists some $\bar{\gamma} > 0$ such that $F\big(x(t+1)\big) < F\big(x(t)\big)$ for every choice of γ in $(0, \bar{\gamma}]$. Modify this result for the case where $X = \{x \mid a_i \leq x_i \leq b_i,\ i = 1,\ldots,n\}$, for given scalars a_i and b_i.

3.4 PARALLELIZATION AND DECOMPOSITION OF OPTIMIZATION PROBLEMS

In the last two sections, we analyzed several optimization methods that are well suited for parallelization, such as the Jacobi, Gauss–Seidel, gradient–like, and approximate Newton

algorithms. These methods are not always applicable, e.g., when there is a constraint set that is not the Cartesian product of simpler sets. In this section, we show how to exploit structural problem features and enhance parallelization by means of suitable problem transformations. We thus switch our focus from parallelization based on method structure to parallelization based on problem structure.

Our approach is based on the duality theory of Appendix C. The idea here is to consider a dual optimization problem that may be more suitable for parallel solution than the original. Related approaches, known as *decomposition methods*, have been applied for many years to large problems with special structure (see e.g., [Las70]). These methods involve the solution of many simple optimization subproblems of small dimension in place of the original problem. When a parallel computing system is available, decomposition methods typically become even more attractive because the simple subproblems can be solved in parallel.

We begin in Subsection 3.4.1 with a strictly convex quadratic programming problem. This problem arises often in applications, or as a subroutine in more complex calculations (e.g. the gradient projection method). The dual cost here is also quadratic and has a gradient that can be conveniently calculated.

In Subsection 3.4.2, we consider another class of problems with special structure. Here the (primal) cost function is separable and strictly convex. The strict convexity property is important because it implies differentiability of the dual cost function (see the Differentiability Theorem in Appendix C). The separability property is important because it facilitates parallelization.

The remainder of the section is devoted to methods for dealing with lack of strict convexity of the primal cost, and the attendant lack of differentiability of the dual cost. This difficulty arises, for example, in the important special case of a linear programming problem. In Subsections 3.4.3 and 3.4.4, we show how the dual problem can be converted into a differentiable optimization problem, and can be solved by methods similar to those used for the separable strictly convex problem of Subsection 3.4.2. An alternative possibility, which we will not consider in this text, is to solve the dual problem by a nondifferentiable optimization method (see [Sha79] and [Pol87]). An example of such a method for linear network flow problems will be developed in Chapter 5.

Throughout this section, we use the duality framework of Appendix C, which is restricted for simplicity to optimization problems with convex cost functions and linear constraints. The reader who is familiar with duality theory will have no difficulty applying the parallelization approaches of this section to more general duality frameworks.

3.4.1 Quadratic Programming

Consider the quadratic programming problem

$$\begin{aligned} &\text{minimize} && \tfrac{1}{2}x'Qx - b'x \\ &\text{subject to} && Ax \leq c, \end{aligned} \tag{4.1}$$

where Q is a given $n \times n$ positive definite symmetric matrix, A is a given $m \times n$ matrix, and $b \in \Re^n$ and $c \in \Re^m$ are given vectors. This is an important problem that arises naturally in many contexts, and also provides a convenient vehicle for reformulation of other problems. For example, the feasibility problem of finding a point in the set $\{x \mid Ax \leq c\}$ can be formulated as the quadratic programming problem of projecting any given point on that set. As another example, a solution of a linear programming problem can be obtained by solving a finite number of quadratic programming problems (see Subsection 3.4.3).

We use the duality theory developed in Appendix C. The dual of the quadratic programming problem (4.1) is given by

$$\begin{aligned} &\text{minimize} \quad \tfrac{1}{2}u'Pu + r'u, \\ &\text{subject to} \quad u \geq 0, \end{aligned} \tag{4.2}$$

where

$$P = AQ^{-1}A', \qquad r = c - AQ^{-1}b. \tag{4.3}$$

It is shown in Appendix C that if u^* solves the dual problem, then $x^* = Q^{-1}(b - A'u^*)$ solves the primal problem (4.1). The dual problem has a simple constraint set, so it is amenable to the use of parallel algorithms.

Let a_j denote the jth column of A'. We assume that a_j is nonzero for all j (if $a_j = 0$, then the corresponding constraint $a_j'x \leq c_j$ is meaningless and can be eliminated). Since Q is symmetric and positive definite, the jth diagonal element of P, given by $p_{jj} = a_j'Q^{-1}a_j$, is positive. This means that for every j, the dual cost function is strictly convex along the jth coordinate. Therefore, the strict convexity assumption of Prop. 3.9 in Section 3.3 is satisfied and it is possible to use the nonlinear Gauss–Seidel algorithm. Because the dual cost is quadratic, the minimization with respect to u can be done analytically, and the iteration can be written explicitly as we proceed to show.

The first partial derivative of the dual cost function with respect to u_j is given by

$$r_j + \sum_{k=1}^{m} p_{jk}u_k, \tag{4.4}$$

where p_{jk} and r_j are the corresponding elements of the matrix P and the vector r, respectively. Setting the derivative to zero, we see that the unconstrained minimum of the dual cost along the jth coordinate starting from u is attained at $\tilde{u}_j$ given by

$$\tilde{u}_j = -\frac{1}{p_{jj}}\Big(r_j + \sum_{k \neq j} p_{jk}u_k\Big) = u_j - \frac{1}{p_{jj}}\Big(r_j + \sum_{k=1}^{m} p_{jk}u_k\Big).$$

Taking into account the nonnegativity constraint $u_j \geq 0$, we see that the Gauss–Seidel iteration, when the jth coordinate is updated, has the form

$$u_j := \max\{0, \tilde{u}_j\} = \max\Big\{0, u_j - \frac{1}{p_{jj}}\Big(r_j + \sum_{k=1}^{m} p_{jk}u_k\Big)\Big\},$$
$$u_i := u_i, \qquad \forall\, i \neq j. \tag{4.5}$$

We can also consider a linearized projected Jacobi method [cf. the discussion following Eq. (3.6) in Section 3.3]. This is a special case of the scaled gradient projection method. In particular, the scaling matrix $M(t)$ is diagonal and its jth diagonal entry is equal to p_{jj}. Taking into account the form of the first partial derivative of the dual cost with respect to u_j given by Eq. (4.4), we see that the method is given by

$$u_j(t+1) = \max\Big\{0, u_j(t) - \frac{\gamma}{p_{jj}}\Big(r_j + \sum_{k=1}^{m} p_{jk}u_k(t)\Big)\Big\}, \qquad j = 1, \ldots, m, \tag{4.6}$$

where $\gamma > 0$ is the stepsize parameter. This iteration is more suitable for parallelization than the Gauss–Seidel iteration (4.5). On the other hand, for convergence, the stepsize γ should be chosen sufficiently small, and some experimentation may be needed to obtain the appropriate range for γ. Convergence can be shown when $\gamma = 1/m$ (Exercise 4.1) but this value may be too small for some problems, and can lead to an unnecessarily slow rate of convergence. A frequently more practical scheme is a hybrid Gauss–Seidel and Jacobi method, whereby the index set $\{1, \ldots, n\}$ is partitioned in subsets and at each iteration, the coordinates of only one of the subsets are updated according to Eq. (4.6). In this way, one may enlarge the range of stepsizes γ for which convergence is obtained.

The matrix A often has a sparse structure in practice, and one would like to take advantage of this structure. Unfortunately, the matrix $P = AQ^{-1}A'$ typically has a less advantageous sparsity structure than A. Furthermore, it may be undesirable to calculate and store the elements of P, particularly when m is large. It turns out that the Gauss–Seidel iteration (4.5) can be performed without explicit knowledge of the elements p_{jk} of the matrix P; only the elements of the matrix AQ^{-1} are needed instead. To see how this can be done, consider the vector

$$y = -A'u. \tag{4.7}$$

We have

$$Pu = AQ^{-1}A'u = -AQ^{-1}y,$$

and the jth component of this vector equation yields

$$\sum_{k=1}^{m} p_{jk}u_k = -w_j'y, \tag{4.8}$$

where w_j' is the jth row of AQ^{-1}. We also have

$$p_{jj} = w_j' a_j, \tag{4.9}$$

where a_j is the jth column of A'. The Gauss–Seidel iteration (4.5) can now be written, using Eqs. (4.8) and (4.9), as

$$u_j := \max\left\{0, u_j - \frac{1}{w_j' a_j}(r_j - w_j' y)\right\},$$

$$u_i := u_i, \qquad \forall\, i \neq j,$$

or, equivalently,

$$u := u - \min\left\{u_j, \frac{1}{w_j' a_j}(r_j - w_j' y)\right\} e_j, \tag{4.10}$$

where e_j is the jth unit vector (all its elements are zero except for the jth, which is unity). The corresponding iteration for the vector y of Eq. (4.7) is obtained by multiplication of Eq. (4.10) with $-A'$ yielding

$$y := y + \min\left\{u_j, \frac{1}{w_j' a_j}(r_j - w_j' y)\right\} a_j. \tag{4.11}$$

The (nonlinear) Gauss–Seidel method can now be summarized as follows. Initially, u is any vector in the nonnegative orthant and $y = -A'u$. At each iteration, a coordinate index j is chosen and u and y are iterated simultaneously using Eqs. (4.10) and (4.11). For problems with special structure, it is possible to parallelize the Gauss–Seidel method by observing that the iterations corresponding to any indices j_1 and j_2 are decoupled and can be carried out in parallel if there is no coordinate which is nonzero for both a_{j_1} and w_{j_2}. To see this, suppose that starting with the vectors u and y, the iteration corresponding to index j_1 yields u_1 and y_1. Let also u_2 and y_2 be the vectors obtained by an iteration corresponding to index j_2 starting with the vectors u_1 and y_1. It is seen from Eq. (4.11) that y and y_1 differ in a given coordinate only if the corresponding coordinate of a_{j_1} is nonzero. Thus, if there is no coordinate which is nonzero for both a_{j_1} and w_{j_2}, we have $w_{j_2}' y = w_{j_2}' y_1$. It follows from Eqs. (4.10) and (4.11) that the values of u_2 and y_2 will be the same, regardless of whether the iteration corresponding to j_1 precedes or is carried out simultaneously with the iteration corresponding to j_2.

When Q is the identity matrix and $b = 0$, the problem is equivalent to projecting the origin on the constraint set. In this case, iteration (4.11) has a nice interpretation as a "modified projection" of y on the halfspace $H_j = \{x \mid a_j' x \leq c_j\}$, as illustrated in Fig. 3.4.1. The Gauss–Seidel algorithm therefore involves, at each iteration, a sequence of modified projections on a halfspace, to update y, while simultaneously updating the corresponding coordinates of u. The Jacobi algorithm involves simultaneous projections on all halfspaces of this type. These and related algorithms have proved successful on

very large problems arising, for example, in image reconstruction (see e.g., [CeH87] for a survey).

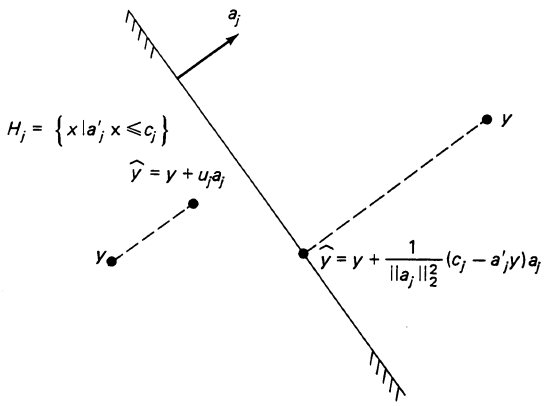

Figure 3.4.1 Interpretation of iteration (4.11) as a projection when $Q = I$ and $b = 0$. Then $w_j = a_j$, $r_j = c_j$, and the iteration takes the form

$$y := y + \min\left\{u_j, \frac{1}{\|a_j\|_2^2}(c_j - a_j' y)\right\} a_j.$$

When $y \notin H_j$, we have $c_j - a_j' y < 0$ and, because $u_j \geq 0$, y is set to the vector

$$\hat{y} = y + \frac{1}{\|a_j\|_2^2}(c_j - a_j' y) a_j,$$

which is the orthogonal projection of y on H_j. When $y \in H_j$, then y is set to the projection of y on the boundary of H_j if $u_j \geq \left(1/\|a_j\|_2^2\right)(c_j - a_j' y)$, and, otherwise, is set to $\hat{y} = y + u_j a_j$, which lies between y and the boundary of H_j.

3.4.2 Separable Strictly Convex Programming

Suppose that the space $\Re^n$ is represented as the Cartesian product of spaces $\Re^{n_i}$, where $n_1 + \cdots + n_m = n$, and consider the problem

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} F_i(x_i) \\ &\text{subject to } e_j' x = s_j, \qquad j = 1, \ldots, r, \\ &\qquad\qquad x_i \in P_i, \qquad i = 1, \ldots, m, \end{aligned} \tag{4.12}$$

where $F_i : \Re^{n_i} \mapsto \Re$ are strictly convex functions, x_i are the components of x, e_j are given vectors in $\Re^n$, s_j are given scalars, and P_i are given bounded polyhedral subsets of $\Re^{n_i}$. We note that if the constraints $e_j' x = s_j$ were not present, it would be possible to decompose this problem into independent subproblems. This motivates us to consider a dual problem that involves Lagrange multipliers for these constraints. In accordance with the theory of Appendix C, this dual problem has the form

$$\begin{aligned} &\text{maximize} \quad q(p) \\ &\text{subject to} \quad p \in \Re^r. \end{aligned} \tag{4.13}$$

The dual function is given by

$$q(p) = \min_{x_i \in P_i} \left\{ \sum_{i=1}^{m} F_i(x_i) + \sum_{j=1}^{r} p_j(e_j' x - s_j) \right\} = \sum_{i=1}^{m} q_i(p) - p's, \tag{4.14}$$

where $p's = \sum_{j=1}^{r} p_j s_j$, and

$$q_i(p) = \min_{x_i \in P_i} \left\{ F_i(x_i) + \sum_{j=1}^{r} p_j e_{ji}' x_i \right\}, \qquad i = 1, \ldots, m, \tag{4.15}$$

with e_{ji} denoting the appropriate subvector of e_j that corresponds to x_i. An important observation is that due to the separable structure of the problem, the evaluation of the dual function is amenable to parallel computation with a separate processor calculating each component $q_i(p)$ of $q(p)$.

By applying the Differentiability Theorem of Appendix C, we see that strict convexity of F_i implies that the dual function is continuously differentiable, and that if the minimum in Eq. (4.15) is attained at the point $x_i(p)$, the partial derivative of q with respect to p_j is given by

$$\frac{\partial q(p)}{\partial p_j} = e_j' x(p) - s_j, \tag{4.16}$$

where $x(p) = \big(x_1(p), \ldots, x_m(p)\big)$. Since the dual function is differentiable, we can apply methods considered earlier in this chapter, such as Gauss–Seidel, Jacobi, and gradient methods, that are amenable to parallel implementation. This is possible because, in contrast with the primal problem (4.12), the dual problem (4.13) is unconstrained. [If the primal problem (4.12) had inequality constraints in place of the equality constraints, the dual problem would have nonnegativity constraints but the parallelizable Gauss–Seidel, Jacobi, and gradient projection methods would still be applicable.] Note also that the calculation of the ith component $q_i(p)$ of the dual cost [Eq. (4.15)] yields $x_i(p)$ and therefore also the ith components $e_{ji}' x_i(p)$ of the dual cost derivatives $\partial q(p)/\partial p_j$ of Eq. (4.16). In a message–passing parallel computing system, where there is a separate processor assigned to the ith component, the calculation of the dual cost gradient via Eq. (4.16) requires a single or multinode accumulation (cf. Subsection 1.3.4). The gradient $\nabla q(p)$ can then be distributed to all processors, if necessary, by means of a single or multinode broadcast [depending on whether all coordinates of $\nabla q(p)$ are accumulated at a single node or not].

In some problems where the separability structure of problem (4.12) is not present, it may be desirable to create this structure through some transformation in order to make the application of parallel methods possible. We provide an example:

Example 4.1. *Minimizing the Sum of Strictly Convex Functions*

Consider the problem

$$\begin{aligned} &\text{minimize} \quad \sum_{i=0}^{m} F_i(x) \\ &\text{subject to } x \in P_i, \qquad i = 0, 1, \ldots, m, \end{aligned} \tag{4.17}$$

where $F_i : \Re^n \mapsto \Re$, $i = 0, 1, \ldots, m$ are strictly convex functions, and P_i are bounded polyhedral subsets of $\Re^n$. An interesting special case of this problem arises when minimizing an expected cost $E\big[F(x, w)\big]$, subject to $x \in P(w)$, where w is a random variable taking a finite number of values, each with a given probability, $F(x, w)$ is strictly convex for each w, and $P(w)$ is a bounded polyhedral set for each w.

We consider the equivalent separable problem

$$\begin{aligned} &\text{minimize} \quad F_0(x) + \sum_{i=1}^{m} F_i(x_i) \\ &\text{subject to } x_i = x, \qquad i = 1, \ldots, m, \\ &\qquad\qquad x \in P_0, \qquad x_i \in P_i, \qquad i = 1, \ldots, m, \end{aligned} \tag{4.18}$$

where $x_i \in \Re^n$, $i = 1, \ldots m$, are additional (artificial) variables. Based on the theory of Appendix C, the corresponding dual problem is given by

$$\begin{aligned} &\text{maximize} \quad q(p) = q_0(p_1 + p_2 + \cdots + p_m) + \sum_{i=1}^{m} q_i(p_i) \\ &\text{subject to } p_i \in \Re^n, \qquad i = 1, 2, \ldots, m, \end{aligned} \tag{4.19}$$

where

$$q_0(p_1 + p_2 + \cdots + p_m) = \min_{x \in P_0} \{F_0(x) - (p_1 + p_2 + \cdots + p_m)'x\}, \tag{4.20}$$

$$q_i(p_i) = \min_{x_i \in P_i} \left\{F_i(x_i) + p_i' x_i\right\}. \tag{4.21}$$

By the Differentiability Theorem of Appendix C, the dual cost is continuously differentiable, and its gradient is given by

$$\frac{\partial q(p)}{\partial p_i} = x_i(p) - x(p), \qquad i = 1, 2, \ldots, m,$$

where $x(p)$ and $x_i(p)$ are the unique minimizing vectors in Eqs. (4.20) and (4.21), respectively. Note that $x(p)$ and $x_i(p)$ can be computed in parallel, and that the dual problem is well suited for solution using parallel gradient methods. An alternative gradient–like dual method for this problem that does not require strict convexity of the functions F_i will be given in Subsection 3.4.4.

3.4.3 The Proximal Minimization Algorithm

We mentioned earlier that strict convexity of the primal cost function is an important property, since it implies differentiability of the dual cost function. When the dual cost function is not differentiable, one might try to solve the dual problem using a method that can handle nondifferentiabilities (see [BaW75], [Sha79], and [Pol87]). A different approach, considered in this subsection, is to make the primal cost function strictly convex by adding a quadratic term to it. We use this approach to develop an algorithm, called the *proximal minimization algorithm.* We show how this algorithm allows us to transform a linear programming problem into a strictly convex quadratic programming problem that can be solved, for example, using the dual methods of Subsection 3.4.1. In the next subsection we look at the proximal minimization algorithm in a dual setting, thereby obtaining decomposition methods for separable problems that are not strictly convex.

Consider the problem

$$\begin{aligned} &\text{minimize} \quad F(x) \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{4.22}$$

where $F : \Re^n \mapsto \Re$ is a given convex function, and X is a nonempty closed convex set. We introduce an additional vector $y \in \Re^n$, and consider the following equivalent optimization problem

$$\begin{aligned} &\text{minimize} \quad F(x) + \frac{1}{2c}\|x - y\|_2^2 \\ &\text{subject to} \quad x \in X,\ y \in \Re^n, \end{aligned}$$

where c is a positive scalar parameter, and $\|\cdot\|_2$ is the standard Euclidean norm. This problem can be solved by the nonlinear Gauss–Seidel method of Subsection 3.3.5, which alternately minimizes the cost over $x \in X$ while keeping y fixed, then minimizes the cost over $y \in \Re^n$ while keeping x fixed, and repeats. The method is given by

$$\begin{aligned} x(t+1) &= \arg\min_{x \in X} \left\{ F(x) + \frac{1}{2c}\|x - y(t)\|_2^2 \right\}, \\ y(t+1) &= x(t+1), \end{aligned}$$

or, equivalently,

$$x(t+1) = \arg\min_{x \in X} \left\{ F(x) + \frac{1}{2c}\|x - x(t)\|_2^2 \right\}. \tag{4.23}$$

It will be shown as part of the following Prop. 4.1 that the minimum in Eq. (4.23) is uniquely attained for any given $x(t) \in \Re^n$. As a result the method is well defined.

Figures 3.4.2 through 3.4.4 illustrate how the method converges, and how the parameter c affects the rate of convergence. Note that when the cost function F is linear, then a straightforward calculation shows that the iteration (4.23) can be written as

$$x(t+1) = \left[x(t) - c\nabla F\big(x(t)\big)\right]^+,$$

where $[\cdot]^+$ denotes projection on the set X, so for this case, iteration (4.23) can be viewed as a gradient projection iteration.

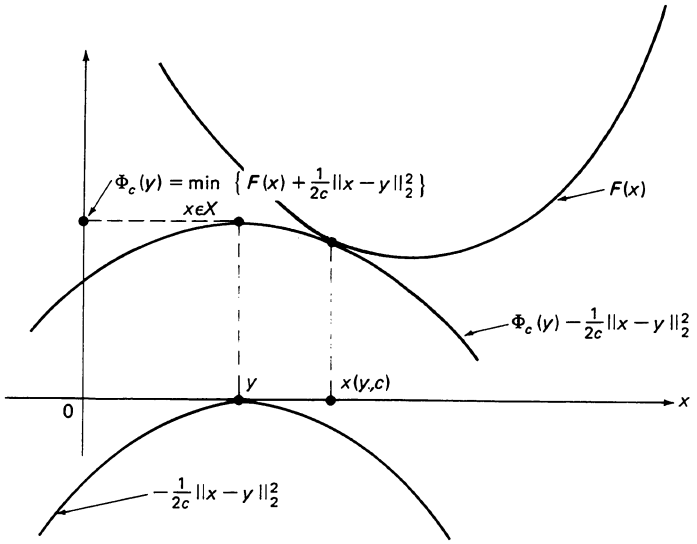

Figure 3.4.2 Finding the minimum of $F(x) + (1/2c)\|x-y\|_2^2$ over X for a given y and c. The minimum is attained at the unique point $x(y,c)$ at which the graph of the quadratic function $-(1/2c)\|x-y\|_2^2$ raised up or down just touches the graph of $F(x)$.

Since the cost function $F(x) + (1/2c)\|x-y\|_2^2$ is strictly convex with respect to x for fixed y, and strictly convex with respect to y for fixed x, our nonlinear Gauss–Seidel convergence result (Prop. 3.9 in Section 3.3) can be applied assuming that F is continuously differentiable. Figure 3.4.3 suggests that convergence occurs even if F is not differentiable, and even if c increases from one iteration to the next. Figure 3.4.5 suggests that convergence is finite under certain circumstances. We show these facts in the following proposition:

Proposition 4.1. Let $F : \Re^n \mapsto \Re$ be a convex function, and X be a nonempty closed convex set. Denote also by X^* the set of points that minimize $F(x)$ over $x \in X$

$$X^* = \{x^* \in X \mid F(x^*) \leq F(x),\ \forall\, x \in X\}. \tag{4.24}$$

(a) For every $y \in \Re^n$ and $c > 0$, the minimum of $F(x) + (1/2c)\|x - y\|_2^2$ over $x \in X$ is attained at a unique point denoted by $x(y, c)$.

(b) The function $\Phi_c : \Re^n \mapsto \Re$ defined by

$$\Phi_c(y) = \min_{x \in X} \left\{ F(x) + \frac{1}{2c}\|x - y\|_2^2 \right\} \tag{4.25}$$

is convex and continuously differentiable, and its gradient is given by

$$\nabla \Phi_c(y) = \frac{y - x(y, c)}{c}. \tag{4.26}$$

Furthermore, x^* minimizes $\Phi_c(y)$ over $y \in \Re^n$ if and only if x^* minimizes $F(x)$ over $x \in X$, that is,

$$X^* = \left\{ x^* \,\middle|\, \Phi_c(x^*) = \min_{y \in \Re^n} \Phi_c(y) \right\}, \qquad \forall\, c > 0. \tag{4.27}$$

(c) Assume that X^* is nonempty. A sequence generated by the iteration

$$x(t+1) = \arg\min_{x \in X} \left\{ F(x) + \frac{1}{2c(t)}\|x - x(t)\|_2^2 \right\}, \tag{4.28}$$

where $\{c(t)\}$ is a sequence of positive numbers with $\liminf_{t \to \infty} c(t) > 0$, converges to an element of X^*.

(d) Assume that X^* is nonempty, and that there exists a scalar $\beta > 0$ such that

$$F(x) \geq F^* + \beta \rho(x; X^*), \qquad \forall\, x \in X, \tag{4.29}$$

where

$$F^* = \min_{x \in X} F(x), \qquad \text{and} \qquad \rho(x; X^*) = \min_{x^* \in X^*} \|x - x^*\|_2. \tag{4.30}$$

Then

$$x(y, c) = \arg\min_{x \in X^*} \|x - y\|_2, \qquad \text{if } \rho(y; X^*) \leq c\beta. \tag{4.31}$$

In particular, the algorithm (4.28) converges finitely [i.e., there exists $\bar{t} > 0$, depending on $x(0)$, such that $x(t) \in X^*$ for all $t \geq \bar{t}$], and, for a given $x(0)$, it converges in a single iteration if $c(0)$ is sufficiently large.

Note: The condition (4.29) is illustrated in Fig. 3.4.6. It can be shown to hold in the case of a linear programming problem, that is, when F is a linear function and X is a polyhedral set (see Exercise 4.3).

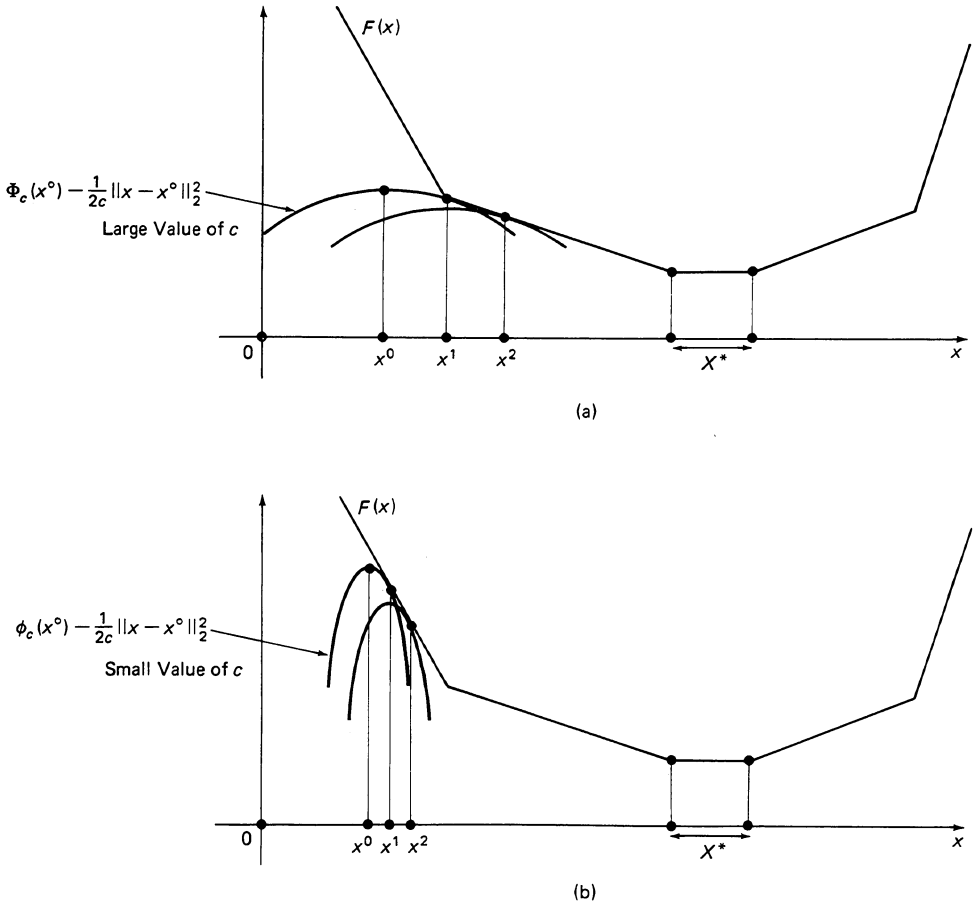

Figure 3.4.3 Illustration of the role of the parameter c in the convergence process of the proximal minimization algorithm. (a) Case of large value of c. The graph of the quadratic term is "blunt" and the method makes fast progress toward the optimal solution set X^*. (b) Case of a small value of c. The graph of the quadratic term is "pointed" and the method makes slow progress.

Proof.

(a) It will suffice to show that for all $c > 0$ and $y \in \Re^n$, the level sets

$$\left\{ x \in X \,\middle|\, F(x) + \frac{1}{2c}\|x - y\|_2^2 \leq \alpha \right\}, \qquad \alpha \in \Re, \tag{4.32}$$

are bounded. It will follow then that we can equivalently search for the minimum of $F(x) + (1/2c)\|x - y\|_2^2$ over a compact subset of X instead of X. Weierstrass' theorem (Prop. A.8 in Appendix A) can then be used to show that the minimum of $F(x) + (1/2c)\|x - y\|_2^2$ over X is attained, necessarily at a unique point in view of

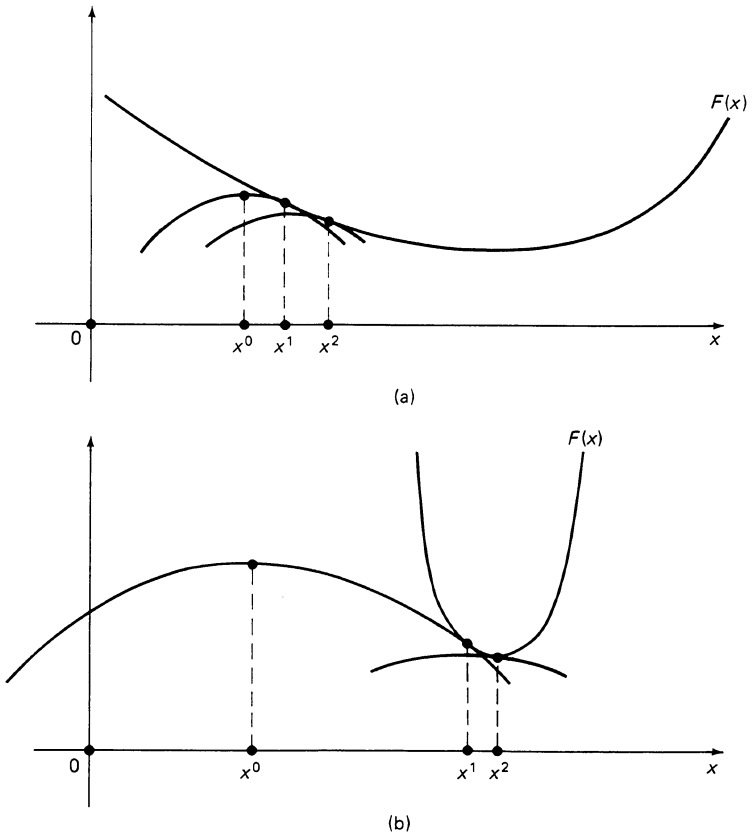

Figure 3.4.4 Illustration of the role of the growth properties of the function F in the convergence process of the proximal minimization algorithm (see Exercise 4.2). (a) Case where $F(x)$ grows slowly, and the convergence is slow. (b) Case where $F(x)$ grows fast, and the convergence is fast.

the strict convexity of the quadratic term $(1/2c)\|x-y\|_2^2$. [Proving boundedness of the set of Eq. (4.32) is very simple if X^* is nonempty or more generally if $F(x)$ is bounded below over X; the following argument primarily addresses the case where $\inf_{x\in X} F(x) = -\infty$, and is based on the idea that a convex function cannot decrease along any one direction at faster than linear rate while the term $(1/2c)\|x-y\|_2^2$ increases at a quadratic rate.]

We argue by contradiction. Suppose that for some $c > 0$ and $y \in \Re^n$, there exists a sequence $\{x^k\}$ such that

$$\|x^k - y\|_2 \to \infty, \qquad F(x^k) + \frac{1}{2c}\|x^k - y\|_2^2 \le \alpha, \qquad \forall\ k. \tag{4.33}$$

Denote $\beta_k = \|x^k - y\|_2$, and assume without loss of generality that $\beta_k \ge 1$ for all k. Define also $z^k = (x^k - y)/\beta_k$, and consider the convex function $\hat{F}(x) = F(x+y)$.

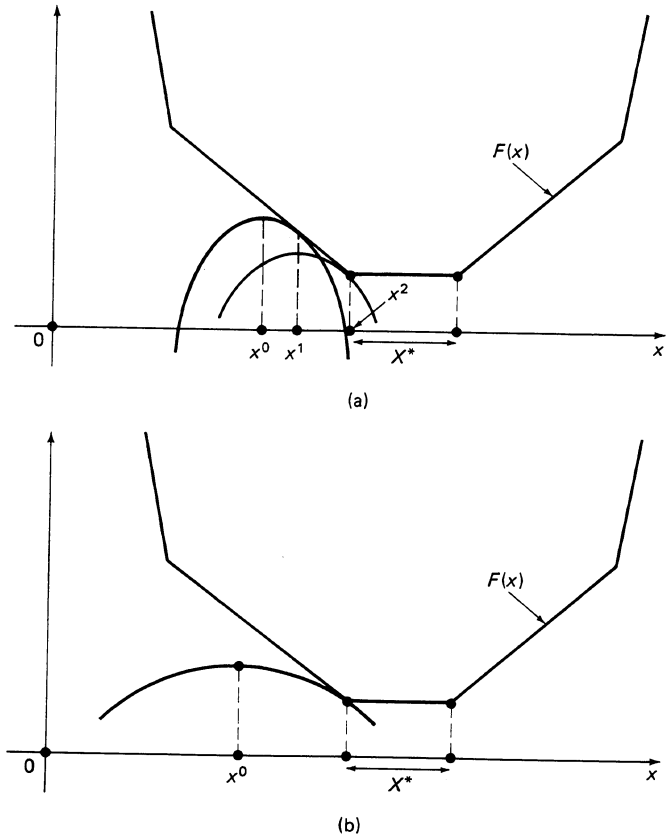

Figure 3.4.5 Finite convergence of the proximal minimization algorithm when $F(x)$ grows at a linear rate near the optimal solution set X^*. (a) Finite convergence for a small value of c. (b) Convergence in a single iteration for a large enough value of c.

From Eq. (4.33) we obtain

$$\hat{F}(\beta_k z^k) + \frac{(\beta_k)^2}{2c} = \hat{F}(x^k - y) + \frac{1}{2c}\|x^k - y\|_2^2 \leq \alpha, \qquad \forall\, k. \tag{4.34}$$

By convexity of $\hat{F}$ we have

$$\min_{\|z\|_2=1} \hat{F}(z) \leq \hat{F}(z^k) \leq \frac{1}{\beta_k}\hat{F}\left(\beta_k z^k\right) + \left(1 - \frac{1}{\beta_k}\right)\hat{F}(0),$$

from which we obtain

$$(1 - \beta_k)\hat{F}(0) + \beta_k \min_{\|z\|_2=1} \hat{F}(z) \leq \hat{F}\left(\beta_k z^k\right).$$

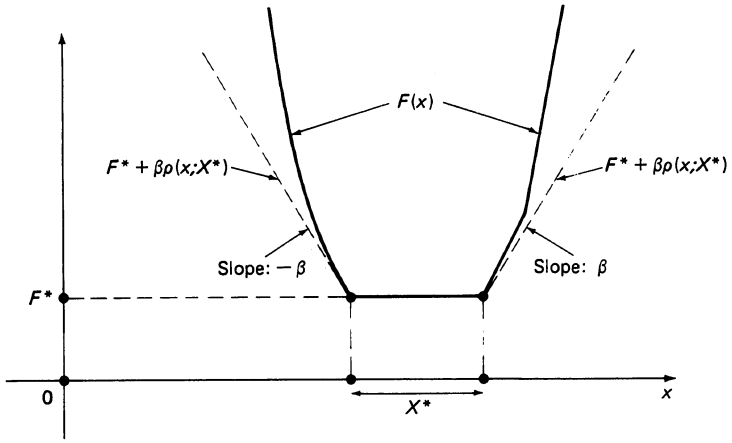

Figure 3.4.6 Illustration of the condition (4.29). Here $F(x)$ majorizes the function $F^* + \beta\rho(x; X^*)$ which grows at a rate $\beta > 0$ with the distance of x from the optimal solution set X^*.

Combining this relation with Eq. (4.34), we obtain

$$\hat{F}(0) + \beta_k \left(\min_{\|z\|_2=1} \hat{F}(z) - \hat{F}(0) \right) + \frac{(\beta_k)^2}{2c} \leq \alpha, \qquad \forall\, k.$$

Since $\beta_k \to \infty$, we reach a contradiction.

(b) To show convexity of Φ_c, let y_1 and y_2 be any vectors in $\Re^n$ and let α be a scalar in $[0, 1]$. Denote $x_1 = x(y_1, c)$ and $x_2 = x(y_2, c)$. We have, using the convexity of F and of the norm function $\|\cdot\|_2$,

$$\begin{aligned}
\alpha\Phi_c(y_1) + (1-\alpha)\Phi_c(y_2) =& \alpha \left[F(x_1) + \frac{1}{2c}\|x_1 - y_1\|_2^2 \right] \\
&+ (1-\alpha) \left[F(x_2) + \frac{1}{2c}\|x_2 - y_2\|_2^2 \right] \\
\geq & F\big(\alpha x_1 + (1-\alpha)x_2\big) \\
&+ \frac{1}{2c}\|\alpha x_1 + (1-\alpha)x_2 - \alpha y_1 - (1-\alpha)y_2\|_2^2 \\
\geq & \min_{x \in X} \left\{ F(x) + \frac{1}{2c}\|x - \alpha y_1 - (1-\alpha)y_2\|_2^2 \right\} \\
=& \Phi_c\big(\alpha y_1 + (1-\alpha)y_2\big).
\end{aligned}$$

This proves the convexity of Φ_c.

To show differentiability of Φ_c, let us consider any $y \in \Re^n$, $d \in \Re^n$, and $\alpha > 0$. We have

$$F\big(x(y,c)\big) + \frac{1}{2c}\|x(y,c) - (y+\alpha d)\|_2^2 \geq \Phi_c(y+\alpha d) \geq \Phi_c(y) + \alpha\Phi_c'(y;d)$$

$$= F\big(x(y,c)\big) + \frac{1}{2c}\|x(y,c) - y\|_2^2 + \alpha\Phi_c'(y;d), \tag{4.35}$$

where the second inequality follows from the convexity of Φ_c and the definition of the directional derivative $\Phi_c'(y;d)$. By expanding the quadratic form in the left–hand side of Eq. (4.35), by collecting terms, and then by dividing by α, we obtain

$$\left[\frac{y - x(y,c)}{c}\right]' d + \frac{\alpha}{2c}\|d\|_2^2 \geq \Phi_c'(y;d), \qquad \forall\ \alpha > 0,\ d \in \Re^n.$$

Taking the limit as $\alpha \to 0$, it follows that

$$\left[\frac{y - x(y,c)}{c}\right]' d \geq \Phi_c'(y;d), \qquad \forall\ d \in \Re^n.$$

By replacing d with $-d$ in the preceding relation, we obtain

$$-\left[\frac{y - x(y,c)}{c}\right]' d \geq \Phi_c'(y;-d) \geq -\Phi_c'(y;d),$$

where the second inequality follows from Eq. (A.16) in Appendix A. The last two relations imply that

$$\left[\frac{y - x(y,c)}{c}\right]' d = \Phi_c'(y;d), \qquad \forall\ d \in \Re^n,$$

or equivalently, that Φ_c is differentiable with gradient equal to $\big[y - x(y,c)\big]/c$. We note also that since Φ_c is a convex function, its gradient is continuous (Prop. A.42 in Appendix A).

We finally show that the minimizing points of $\Phi_c(y)$ over $\Re^n$ and of $F(x)$ over X coincide. We first note that the function $F(x) + (1/2c)\|x - y\|_2^2$ takes the value $F(y)$ for $x = y$, from which it follows that

$$\Phi_c(y) \leq F(y), \qquad \forall\ y \in X. \tag{4.36}$$

If y^* minimizes $F(x)$ over $x \in X$ then using Eq. (4.36) we have

$$\Phi_c(y^*) \leq F(y^*) \leq F\big(x(y,c)\big) \leq F\big(x(y,c)\big) + \frac{1}{2c}\|x(y,c) - y^*\|_2^2 = \Phi_c(y), \quad \forall y \in \Re^n,$$

which implies that y^* minimizes $\Phi_c(y)$ over $\Re^n$. Conversely, if y^* minimizes $\Phi_c(y)$ over $\Re^n$, then $c\nabla\Phi_c(y^*) = y^* - x(y^*, c) = 0$. This implies that $y^* \in X$ and, using also Eq. (4.36), it is seen that

$$F(y^*) = \Phi_c(y^*) \leq \Phi_c(y) \leq F(y), \qquad \forall\, y \in X.$$

Therefore y^* minimizes $F(y)$ over $y \in X$.

(c) The proof proceeds in two stages. We first show that all limit points of $\{x(t)\}$ belong to X^*, and then we show that $\{x(t)\}$ is bounded and has a unique limit point.

We have, using Eq. (4.28),

$$F\big(x(t+1)\big) + \frac{1}{2c(t)}\|x(t+1) - x(t)\|_2^2 \leq F(x) + \frac{1}{2c(t)}\|x - x(t)\|_2^2, \qquad \forall\, x \in X, \tag{4.37}$$

from which, by setting $x = x(t)$ we obtain

$$F\big(x(t+1)\big) + \frac{1}{2c(t)}\|x(t+1) - x(t)\|_2^2 \leq F\big(x(t)\big), \qquad \forall\, t. \tag{4.38}$$

Let $\{x(t)\}_{t\in T}$ be a subsequence converging to a vector $x_\infty \in X$. From Eq. (4.38), it follows that $F\big(x(t)\big)$ decreases monotonically to $F(x_\infty)$ and that

$$\lim_{t\to\infty} \|x(t+1) - x(t)\|_2 = 0. \tag{4.39}$$

Furthermore, Eq. (4.39) implies that the subsequence $\{x(t+1)\}_{t\in T}$ also converges to x_∞. Let $x^* \in X^*$, and $\alpha \in (0,1)$. By setting $x = \alpha x^* + (1-\alpha)x(t+1)$ in Eq. (4.37) and using the convexity of F, we obtain

$$\begin{aligned} &F\big(x(t+1)\big) + \frac{1}{2c(t)}\|x(t+1) - x(t)\|_2^2 \\ &\leq F\big(\alpha x^* + (1-\alpha)x(t+1)\big) + \frac{1}{2c(t)}\|\alpha x^* + (1-\alpha)x(t+1) - x(t)\|_2^2 \\ &\leq \alpha F(x^*) + (1-\alpha)F\big(x(t+1)\big) + \frac{1}{2c(t)}\|\alpha\big(x^* - x(t+1)\big) + x(t+1) - x(t)\|_2^2. \end{aligned}$$

Taking the limit as $t \to \infty$, $t \in T$, and using Eq. (4.39), we obtain

$$F(x_\infty) - F(x^*) \leq \frac{\alpha\|x^* - x_\infty\|_2^2}{2\liminf_{t\to\infty} c(t)}, \qquad \forall\, \alpha \in (0,1). \tag{4.40}$$

Since this relation holds for all $\alpha \in (0, 1)$, it follows that $F(x_\infty) = F(x^*)$, implying that $x_\infty \in X^*$. We have thus proved that every limit point of $\{x(t)\}$ is an optimal solution.

There remains to show that $\{x(t)\}$ converges. From Eq. (4.37) we obtain

$$\|x(t+1) - x(t)\|_2 \leq \|x - x(t)\|_2, \qquad \forall\, x \in X \text{ with } F(x) \leq F\big(x(t+1)\big), \tag{4.41}$$

from which it follows that $x(t+1)$ is the unique projection of $x(t)$ on the convex set $\{x \in X \mid F(x) \leq F\big(x(t+1)\big)\}$. From the Projection Theorem (Prop. 3.2 in Section 3.3) we obtain

$$\big(x(t+1) - x(t)\big)'\big(x - x(t+1)\big) \geq 0, \qquad \forall\, x \in X \text{ with } F(x) \leq F\big(x(t+1)\big). \tag{4.42}$$

For every optimal solution $x^* \in X^*$, we have

$$\|x^* - x(t)\|_2^2 = \|x^* - x(t+1)\|_2^2 + 2\big(x(t+1) - x(t)\big)'\big(x^* - x(t+1)\big) + \|x(t+1) - x(t)\|_2^2,$$

and by using Eq. (4.42) in this relation we obtain

$$\|x^* - x(t+1)\|_2 \leq \|x^* - x(t)\|_2, \qquad \forall\, x^* \in X^*. \tag{4.43}$$

From Eq. (4.43) we see that $\{x(t)\}$ is bounded, so it must have one or more limit points. We have already proved that all limit points of $\{x(t)\}$ belong to X^*. If x^* is a limit point, then Eq. (4.43) implies that the distance of $x(t)$ from x^* cannot increase at any iteration. Therefore, $\{x(t)\}$ cannot have a second limit point, and must converge to x^*.

(d) See Fig. 3.4.7. **Q.E.D.**

The operator that assigns to y the unique minimizing point $x(y, c)$ in the definition (4.25) of Φ_c is known as the *prox operator* [Mor65]; this explains the name proximal minimization algorithm for the iteration (4.28).

Note that when $c(t)$ is constant [say $c(t) = c$ for all t], the proximal minimization algorithm (4.28) can also be written, based on the gradient expression (4.26), as

$$x(t+1) = x(t) - c\nabla\Phi_c\big(x(t)\big),$$

so it can be viewed as a gradient method for minimizing Φ_c. The rate of convergence of the iteration depends on c and improves as c becomes larger, as indicated in Fig. 3.4.3 and in Exercise 4.2. Computational experience has shown that using an increasing sequence $\{c(t)\}$ often works considerably better than using a constant value of c. Note that one may not wish to use a very high value of c because this can lead to numerical difficulties in minimizing $F(x) + (1/2c)\|x - y\|_2^2$.

The gradient interpretation also suggests the variation

$$x(t+1) = x(t) - \gamma(t)\nabla\Phi_c\big(x(t)\big),$$

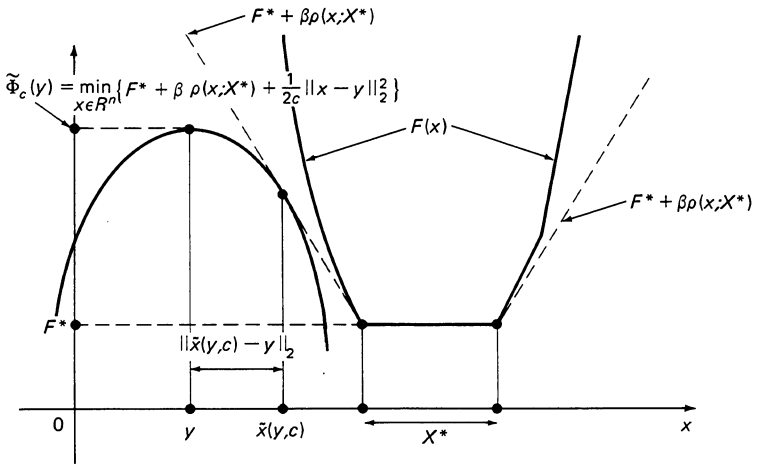

Figure 3.4.7 Proof of part (d) of Prop. 4.1. We first note that the function $\rho(x;X^*) = \min_{x^*\in X^*}\|x-x^*\|_2$ is convex, and that

$$\nabla\rho(x;X^*) = \frac{x-\hat{x}}{\rho(x;X^*)}, \qquad \forall\, x \notin X^*,$$

where $\hat{x}$ denotes the unique projection of x on X^*. The verification of convexity is left for the reader. The formula for $\nabla\rho(x;X^*)$ is obtained by differentiating in the equation $\rho(x;X^*) = \sqrt{2c\hat{\Phi}_c(x)}$ where $\hat{\Phi}_c(x) = \min_{x^*\in X^*}(1/2c)\|x-x^*\|_2^2$, and by using the fact $\nabla\hat{\Phi}_c(x) = (x-\hat{x})/c$ [cf. Eq. (4.26)].

Consider the function $\tilde{\Phi}_c : \Re^n \mapsto \Re$ defined by

$$\tilde{\Phi}_c(y) = \min_{x\in\Re^n}\left\{F^* + \beta\rho(x;X^*) + \frac{1}{2c}\|x-y\|_2^2\right\},$$

(compare with the figure). From part (a) of Prop. 4.1, the minimum in the definition of $\tilde{\Phi}_c(y)$ is attained at a unique point denoted $\tilde{x}(y,c)$. Suppose that $\tilde{x}(y,c) \notin X^*$. By setting to zero the gradient of $F^* + \beta\rho(x;X^*) + (1/2c)\|x-y\|_2^2$ at $\tilde{x}(y,c)$ we obtain

$$\frac{\beta\big(\tilde{x}(y,c) - \hat{x}(y,c)\big)}{\rho\big(\tilde{x}(y,c);X^*\big)} + \frac{\tilde{x}(y,c)-y}{c} = 0,$$

where $\hat{x}(y,c)$ is the projection of $\tilde{x}(y,c)$ on X^*. It follows that $\|\tilde{x}(y,c)-y\|_2 = c\beta$. Using the analog of Eq. (4.41) with $F(x)$ replaced by $F^* + \beta\rho(x;X^*)$, we have that $\tilde{x}(y,c)$ is the projection of y on the set $\big\{\tilde{x}\in X \mid \rho(x;X^*) \le \rho\big(\tilde{x}(y,c);X^*\big)\big\}$, which contains X^*. Therefore, $\rho(y;X^*) > \|\tilde{x}(y,c)-y\|_2 = c\beta$. We have thus shown that $\tilde{x}(y,c) \notin X^*$ implies that $\rho(y;X^*) > c\beta$. It follows that if $\rho(y;X^*) \le c\beta$, then $\tilde{x}(y,c) \in X^*$.

Suppose now that the condition (4.29) holds. We will show that if $\tilde{x}(y,c) \in X^*$, then $x(y,c) = \tilde{x}(y,c)$ (as can be seen from the figure). Indeed condition (4.29) implies that $\tilde{\Phi}_c(y) \le \Phi_c(y) \le F(x) + (1/2c)\|x-y\|_2^2$ for all $x \in X$. On the other hand if $\tilde{x}(y,c) \in X^*$, then

$$\tilde{\Phi}_c(y) = F^* + \beta\rho\big(\tilde{x}(y,c);X^*\big) + \frac{1}{2c}\|\tilde{x}(y,c)-y\|_2^2 = F\big(\tilde{x}(y,c)\big) + \frac{1}{2c}\|\tilde{x}(y,c)-y\|_2^2,$$

showing that the minimum of $F(x) + (1/2c)\|x-y\|_2^2$ is attained at $\tilde{x}(y,c)$. Therefore $\tilde{x}(y,c) = x(y,c)$. It follows that $x(y,c) \in X^*$ if $\rho(y;X^*) \le c\beta$. Also since $x(y,c)$ minimizes $F(x) + (1/2c)\|x-y\|_2^2$ over X, we obtain that $\|x(y,c)-y\|_2 \le \|x^*-y\|_2$ for all $x^* \in X^*$, so $x(y,c)$ is the projection of y on X^*.

where $\gamma(t)$ is a stepsize parameter, possibly different than c. Exercise 4.4 develops a related convergence result for the choice $\gamma(t) \in (0, 2c)$.

We finally note that the cost function $F(x)+(1/2c)\|x-y\|_2^2$ is strictly convex with respect to x, so when F has the separable form $F(x) = \sum_{i=1}^m F_i(x_i)$, the dual methods discussed in Subsection 3.4.2 are applicable. The price for this is that we must solve a sequence of separable (strictly convex) problems instead of a single problem (which, however, may not be strictly convex, and may involve a nondifferentiable dual problem). An interesting alternative will be explored in the next subsection.

3.4.4 Augmented Lagrangian Methods

We now consider a dual approach for overcoming the lack of strict convexity of the primal cost, which is based again on adding a quadratic term to the cost function. The resulting algorithm, with proper interpretation, turns out to be equivalent to the proximal minimization algorithm of the previous subsection.

Consider the constrained optimization problem

$$\begin{aligned} &\text{minimize} && F(x) \\ &\text{subject to} && e_j'x = s_j, \qquad j = 1, \ldots, r \\ &&& x \in P, \end{aligned} \tag{4.44}$$

where $F : \Re^n \mapsto \Re$ is a convex function, e_j are given vectors in $\Re^n$, s_j are given scalars, and P is a nonempty polyhedral subset of $\Re^n$. This is the optimization problem that we used to develop the duality theory of Appendix C, except that here we have disregarded linear inequality constraints of the form $a_j'x \leq t_j$. It turns out that this does not involve a loss of generality (see Exercise 4.5). We will also assume for simplicity that P is bounded; the subsequent analysis, however, can be generalized considerably (see [Ber82a], [Roc76b], and [Roc76c]).

We can consider in place of the original problem (4.44), the equivalent problem

$$\begin{aligned} &\text{minimize} && F(x) + \frac{c}{2}\|Ex - s\|_2^2 \\ &\text{subject to} && Ex = s, \\ &&& x \in P, \end{aligned}$$

where c is a positive scalar parameter, and $Ex = s$ is a compact notation for the constraints $e_j'x = s_j$, that is, E is the matrix with rows e_j', and s is the vector with coordinates s_j. The dual problem is

$$\begin{aligned} &\text{maximize} && q_c(p) = \inf_{x \in P} L_c(x, p) \\ &\text{subject to} && p \in \Re^m, \end{aligned}$$

where $L_c(x,p)$ is the *Augmented Lagrangian* function

$$L_c(x,p) = F(x) + p'(Ex - s) + \frac{c}{2}\|Ex - s\|_2^2.$$

An important method using the Augmented Lagrangian function, called the *method of multipliers* ([HaB70], [Hes69], and [Pow69]), consists of successive minimizations of the form

$$x(t+1) = \arg\min_{x \in P} L_{c(t)}\big(x, p(t)\big), \tag{4.45}$$

followed by updates of the vector $p(t)$ according to

$$p(t+1) = p(t) + c(t)\big(Ex(t+1) - s\big). \tag{4.46}$$

The initial vector $p(0)$ is arbitrary, and $\{c(t)\}$ is a nondecreasing sequence of positive numbers. Note that the minimum of the Augmented Lagrangian in Eq. (4.45) is attained based on our earlier assumption that P is bounded and the Weierstrass theorem (Prop. A.8 in Appendix A). In the case where this minimum is not uniquely attained, the vector $x(t+1)$ in Eq. (4.45) is chosen arbitrarily from the set of minimizing points of $L_{c(t)}\big(\cdot, p(t)\big)$.

It turns out that the iteration (4.45)–(4.46) is in reality the proximal minimization algorithm (4.28) in disguise. To see this, we introduce an auxiliary vector $z \in \Re^m$, and write

$$\begin{aligned}\min_{x \in P} L_{c(t)}\big(x, p(t)\big) &= \min_{x \in P}\left\{F(x) + p(t)'(Ex - s) + \frac{c(t)}{2}\|Ex - s\|_2^2\right\} \\ &= \min_{Ex - s = z,\ x \in P,\ z \in \Re^m}\left\{F(x) + p(t)'z + \frac{c(t)}{2}\|z\|_2^2\right\}.\end{aligned} \tag{4.47}$$

We view the problem on the right–hand side in Eq. (4.47) as a constrained optimization problem in the variables x and z. The vector pair $\big(x(t+1), z(t+1)\big)$, where

$$z(t+1) = Ex(t+1) - s, \tag{4.48}$$

is an optimal solution to this problem. Let $\bar{y}$ be a corresponding optimal dual solution, that is,

$$\begin{aligned}\bar{y} &= \arg\max_{y \in \Re^m}\left\{\min_{x \in P,\ z \in \Re^m}\left\{F(x) + y'(Ex - s - z) + p(t)'z + \frac{c(t)}{2}\|z\|_2^2\right\}\right\} \\ &= \arg\max_{y \in \Re^m}\left\{\min_{x \in P}\{F(x) + y'(Ex - s)\} + \min_{z \in \Re^m}\left\{(p(t) - y)'z + \frac{c(t)}{2}\|z\|_2^2\right\}\right\}.\end{aligned} \tag{4.49}$$

(An optimal dual solution is guaranteed to exist by the Duality Theorem of Appendix C.) Then $z(t+1)$ attains the minimum in the right–hand side of the above equation when $y = \bar{y}$, which implies that

$$z(t+1) = \frac{\bar{y} - p(t)}{c(t)}$$

or equivalently using Eqs. (4.46) and (4.48),

$$\bar{y} = p(t+1). \tag{4.50}$$

A straightforward calculation shows that

$$\min_{z \in \Re^m} \left\{ \big(p(t) - y\big)'z + \frac{c(t)}{2}\|z\|_2^2 \right\} = -\frac{1}{2c(t)}\|y - p(t)\|_2^2,$$

so from Eqs. (4.49) and (4.50) we obtain

$$p(t+1) = \arg\max_{y \in \Re^m} \left\{ q(y) - \frac{1}{2c(t)}\|y - p(t)\|_2^2 \right\}, \tag{4.51}$$

where $q(y)$ is the dual functional of the original problem (4.44)

$$q(y) = \min_{x \in P} \big\{ F(x) + y'(Ex - s) \big\}.$$

Thus, from Eq. (4.51) we see that the multiplier iteration (4.45)–(4.46) is equivalent to the proximal minimization algorithm applied to the problem of minimizing the real–valued convex function $-q$ or equivalently to the dual problem of maximizing q.

By applying now the convergence result of Prop. 4.1(c), we see that the sequence $\{p(t)\}$ generated by the method of multipliers converges to some dual optimal solution. Furthermore, convergence in a finite number of iterations is obtained in the case of a linear programming problem [cf. Prop. 4.1(d) and Exercise 4.3 applied to the dual problem, which is also a linear programming problem]. We also claim that every limit point of the generated sequence $\{x(t)\}$ is an optimal solution of the primal problem (4.44). To see this, note that from the multiplier update formula (4.46) we obtain

$$Ex(t+1) - s \to 0, \qquad c(t)\big(Ex(t+1) - s\big) \to 0.$$

We also have

$$L_{c(t)}\big(x(t+1), p(t)\big) = \min_{x \in P} \left\{ F(x) + p(t)'(Ex - s) + \frac{c(t)}{2}\|Ex - s\|_2^2 \right\}.$$

The last two relations yield

$$\limsup_{t \to \infty} F\big(x(t+1)\big) = \limsup_{t \to \infty} L_{c(t)}\big(x(t+1), p(t)\big) \leq F(x), \qquad \forall\ x \in P, \text{ with } Ex = s,$$

so if $x^* \in P$ is a limit point of $\{x(t)\}$, we obtain

$$F(x^*) \leq F(x), \qquad \forall\, x \in P, \text{ with } Ex = s,$$

as well as $Ex^* = s$ [in view of $Ex(t+1) - s \to 0$]. Therefore any limit point x^* of the generated sequence $\{x(t)\}$ is an optimal solution of the primal problem (4.44).

The method of multipliers of Eqs. (4.45) and (4.46) is an excellent general purpose method for constrained optimization, and applies to considerably more general problems than the one treated here. For example, it can be used for problems involving nonconvex cost functions and constraint equations. It involves a sequence of minimizations of $L_{c(t)}\big(x, p(t)\big)$, but each of these minimizations is subject to fewer constraints and is presumably easier than solving the original problem (4.44). For this, it is necessary that the parameter $c(t)$ is not too large in order to avoid "ill–conditioning" the minimization of the Augmented Lagrangian. Practical experience has shown that it is best to start with a moderate value of c (perhaps obtained through some preliminary experimentation), and either to keep c constant, or to increase c by some factor (say, 2 to 10) with each minimization of the Augmented Lagrangian. There are a number of practical ways to use the results of one minimization in the next minimization (see [Ber82a]).

One difficulty with the method of multipliers is that even if the cost function $F(x)$ is separable, the Augmented Lagrangian $L_c\big(\cdot, p(t)\big)$ is typically nonseparable because it involves the quadratic term $\|Ex - s\|_2^2$. With some reformulation, however, it is possible to preserve a good deal of the separable structure, as shown in the following examples.

Example 4.2. *Minimizing the Sum of Convex Functions*

Consider the problem

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} F_i(x) \\ &\text{subject to} \quad x \in P_i, \qquad i = 1, \ldots, m, \end{aligned} \tag{4.52}$$

where $F_i : \Re^n \mapsto \Re$, $i = 0, 1, \ldots, m$, are convex functions, and P_i are bounded polyhedral subsets of $\Re^n$. Note the difference with the related Example 4.1; here the functions F_i are not necessarily strictly convex.

We consider the equivalent separable problem

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} F_i(x_i) \\ &\text{subject to} \quad x_i = x, \qquad i = 1, \ldots, m, \\ &\qquad\qquad\quad x_i \in P_i, \qquad i = 1, \ldots, m, \end{aligned}$$

where $x_i \in \Re^n$, $i = 1, \ldots m$, are additional (artificial) variables. We apply the method of multipliers to this problem. It takes the form

$$p_i(t+1) = p_i(t) + c(t)\big(x(t+1) - x_i(t+1)\big), \qquad i = 1, \ldots, m, \tag{4.53}$$

where $x_i(t+1)$ and $x(t+1)$ solve the problem

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} \left\{ F_i(x_i) + p_i(t)'(x - x_i) + \frac{c(t)}{2}\|x - x_i\|_2^2 \right\} \\ &\text{subject to} \quad x \in \Re^n, \quad x_i \in P_i, \qquad i = 1, \ldots, m. \end{aligned} \tag{4.54}$$

Note that there is coupling in this problem between x and the vectors x_i, so this problem cannot be decomposed into separate minimizations with respect to some of the variables. On the other hand, the problem (4.54) has a Cartesian product constraint set, and a structure that is suitable for the application of the nonlinear Gauss–Seidel method. In particular, we can consider a method that minimizes the Augmented Lagrangian with respect to x with the iteration

$$x := \frac{\sum_{i=1}^{m} x_i}{m} - \frac{\sum_{i=1}^{m} p_i(t)}{mc(t)}, \tag{4.55}$$

then minimizes the Augmented Lagrangian with respect to x_i with the iteration

$$x_i := \arg\min_{x_i \in P_i} \left\{ F_i(x_i) - p_i(t)x_i + \frac{c(t)}{2}\|x - x_i\|_2^2 \right\}, \qquad \forall\, i = 1, \ldots, m, \tag{4.56}$$

and repeats until convergence to a minimum of the Augmented Lagrangian. Note that the method can be parallelized to a great extent because the minimizations in Eq. (4.56) can be done in parallel. In a message–passing system, the "averaging" step of Eq. (4.55) used to update x can be performed by means of a single node accumulation algorithm at some processor (cf. Subsection 1.3.4). The resulting vector x can then be distributed to all processors by using a single node broadcast.

The next example is essentially a special case of the preceding one. It has the property that a single minimization of the Augmented Lagrangian is needed because the primal cost function is identically zero and the corresponding dual function has the property of Eq. (4.29) with β arbitrarily large [cf. part (d) of Prop. 4.1].

Example 4.3. *Finding a Point in a Set Intersection by Parallel Projections*

We are given m closed convex sets $C_1, C_2, \ldots, C_m$ in $\Re^n$, and we want to find a point in their intersection. An equivalent problem is

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\sum_{i=1}^{m} \|x_i - x\|_2^2 \\ &\text{subject to} \quad x \in \Re^n,\ x_i \in C_i,\ i = 1, \ldots, m. \end{aligned} \tag{4.57}$$

Here the variables of the optimization are $x, x_1, \ldots, x_m$ and if the intersection $C_1 \cap \cdots \cap C_m$ is nonempty, an optimal solution $(x^*, x_1^*, \ldots, x_m^*)$ of the above problem is such that $x^* = x_i^*$ for all i, and x^* belongs to the intersection. The problem (4.57) may also be viewed as a minimization of the Augmented Lagrangian function (4.54) of the preceding example, where $c(t) = 1$, $F_i(x_i) = 0$, and $p_i(t) = 0$.

Let us now apply the nonlinear Gauss–Seidel method to problem (4.57). The order of variable updating is $x, x_1, \ldots, x_m$, repeated cyclically. Minimization of the cost with respect to each one of $x, x_1, \ldots, x_m$, while all the other variables are fixed, yields the algorithm

$$x(t+1) = \frac{1}{m}\sum_{i=1}^{m} x_i(t), \tag{4.58}$$

$$x_i(t+1) = P_i\big(x(t+1)\big), \qquad i = 1, \ldots, m, \tag{4.59}$$

where $P_i(\cdot)$ denotes projection on C_i. Note that the strict convexity assumption of Prop. 3.9 in Section 3.3 is satisfied in problem (4.57), so the convergence result of that proposition applies. Exercise 4.6 refines this result, showing convergence to an element of the intersection.

Figure 3.4.8 illustrates the parallelizable character of the method and shows that its convergence rate can be slow under some circumstances.

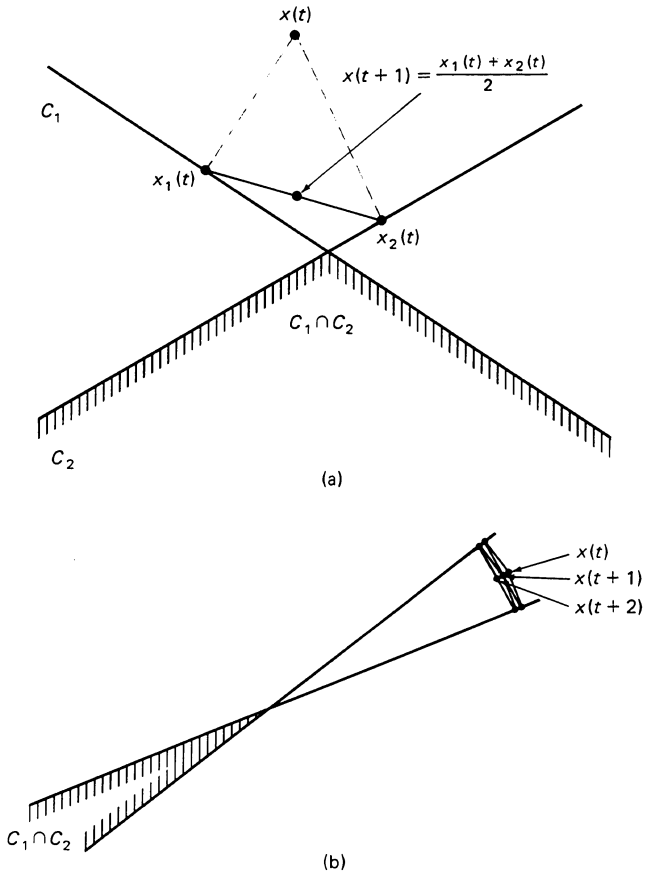

Figure 3.4.8 (a) Illustration of the parallel character of the method of Eqs. (4.58)–(4.59) for the feasibility problem of finding a point in the intersection $C_1 \cap \cdots \cap C_m$. (b) Example illustrating how the rate of convergence of the method can be poor.

Example 4.4. *Separable Problems*

Consider the separable problem of Subsection 3.4.2

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} F_i(x_i) \\ &\text{subject to} \quad e_j' x = s_j, \qquad j = 1, \ldots, r, \\ &\qquad\qquad x_i \in P_i, \qquad i = 1, \ldots, m, \end{aligned} \tag{4.60}$$

with the difference that we assume that the functions $F_i : \Re^{n_i} \mapsto \Re$ are convex but not necessarily strictly convex. Recall here that $x = (x_1 \ldots, x_m)$, where x_i is a subvector of dimension n_i, and P_i is a bounded polyhedral set.

Let e_{ji} denote the subvector of e_j that corresponds to x_i, and for a given j, let $I(j)$ be the set of indices i of subvectors x_i that appear in the jth constraint $e_j' x = s_j$, that is

$$I(j) = \{i \mid e_{ji} \neq 0\}, \qquad j = 1, \ldots, r.$$

We transform the problem by introducing additional variables z_{ji}, $i \in I(j)$, as follows

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} F_i(x_i) \\ &\text{subject to} \quad e_{ji}' x_i = z_{ji}, \qquad j = 1, \ldots, r, \ i \in I(j), \\ &\qquad\qquad \sum_{i \in I(j)} z_{ji} = s_j, \qquad j = 1, \ldots, r, \\ &\qquad\qquad x_i \in P_i, \qquad i = 1, \ldots, m. \end{aligned} \tag{4.61}$$

For each $j = 1, \ldots, r$, we consider Lagrange multipliers p_{ji} for the equality constraints $e_{ji}' x_i = z_{ji}$, $i \in I(j)$. The method of multipliers consists of

$$p_{ji}(t+1) = p_{ji}(t) + c(t) \left(e_{ji}' x_i(t+1) - z_{ji}(t+1) \right), \qquad j = 1, \ldots, r, \ i \in I(j), \tag{4.62}$$

where $x_i(t+1)$ and $z_{ji}(t+1)$ minimize the Augmented Lagrangian

$$\sum_{i=1}^{m} F_i(x_i) + \sum_{j=1}^{r} \sum_{i \in I(j)} p_{ji}(t) \left(e_{ji}' x_i - z_{ji} \right) + \frac{c(t)}{2} \sum_{j=1}^{r} \sum_{i \in I(j)} \left(e_{ji}' x_i - z_{ji} \right)^2,$$

subject to $\sum_{i \in I(j)} z_{ji} = s_j, j = 1, \ldots, r$, and $x_i \in P_i, i = 1, \ldots, m$. Similarly as in Example 4.2 [cf. Eqs. (4.55) and (4.56)], this minimization can be done iteratively by alternate minimizations with respect to the vectors x_i, and the vectors z_{ji}.

The iteration has the form

$$x_i := \arg\min_{\xi_i \in P_i} \left\{ F_i(\xi_i) + \sum_{\{j \mid i \in I(j)\}} \left\{ p_{ji}(t) e'_{ji} \xi_i + \frac{c(t)}{2} \left(e'_{ji} \xi_i - z_{ji} \right)^2 \right\} \right\},$$

$$\forall\, i = 1, \ldots, m, \tag{4.63}$$

$$\{z_{ji} \mid i \in I(j)\}$$

$$:= \arg\min_{\left\{\zeta_{ji} \mid i \in I(j),\ \sum_{i \in I(j)} \zeta_{ji} = s_j\right\}} \left\{ - \sum_{i \in I(j)} p_{ji}(t) \zeta_{ji} + \frac{c(t)}{2} \sum_{i \in I(j)} \left(e'_{ji} x_i - \zeta_{ji} \right)^2 \right\}$$

$$\forall\, j = 1, \ldots, r. \tag{4.64}$$

Note that the minimization with respect to $\{\zeta_{ji} \mid i \in I(j)\}$ in the above equation involves a separable quadratic cost and a single equality constraint, and can be carried out analytically. In particular, the minimum is attained for

$$\zeta_{ji} = e'_{ji} x_i + \frac{p_{ji}(t) - \lambda_j}{c(t)}, \qquad j = 1, \ldots, r,\ i \in I(j), \tag{4.65}$$

where λ_j is a scalar Lagrange multiplier, chosen so that the constraint $\sum_{i \in I(j)} \zeta_{ji} = s_j$ is satisfied or equivalently

$$\lambda_j = \frac{1}{m_j} \sum_{i \in I(j)} p_{ji}(t) + \frac{c(t)}{m_j} \left[\sum_{i \in I(j)} e'_{ji} x_i - s_j \right], \qquad j = 1, \ldots, r, \tag{4.66}$$

where m_j is the number of elements of $I(j)$, that is

$$m_j = |I(j)|.$$

Using the preceding equations we can simplify the update formula (4.62) for p_{ji}. Suppose we have found the optimal values $x_i(t+1)$. Then from Eq. (4.65), the optimal values $z_{ij}(t+1)$ are given by

$$z_{ji}(t+1) = e'_{ji} x_i(t+1) + \frac{p_{ji}(t) - \lambda_j(t+1)}{c(t)}, \qquad j = 1, \ldots, r,\ i \in I(j),$$

where $\lambda_j(t+1)$ is given by Eq. (4.66) after x_i is replaced by $x_i(t+1)$. By comparing this equation with Eq. (4.62) we see that

$$p_{ji}(t+1) = \lambda_j(t+1), \qquad \forall\, j = 1, \ldots, r,\ i \in I(j).$$

Thus the single multiplier variable λ_j can be used in place of the m_j variables p_{ji}, $i \in I(j)$. By writing the multiplier update formula (4.62) for $i \in I(j)$ and by adding we obtain

$$\lambda_j(t+1) = \lambda_j(t) + \frac{c(t)}{m_j}\left[\sum_{i\in I(j)}\left(e'_{ji}x_i(t+1) - z_{ji}(t+1)\right)\right], \qquad j = 1,\ldots,r,$$

or equivalently,

$$\lambda_j(t+1) = \lambda_j(t) + \frac{c(t)}{m_j}\left(e'_j x(t+1) - s_j\right), \qquad j = 1,\ldots,r. \tag{4.67}$$

By replacing $p_{ji}(t)$ by $\lambda_j(t)$ in Eqs. (4.65) and (4.66), we obtain the following updating formula for z_{ji}

$$z_{ji} := e'_{ji}x_i + \frac{\lambda_j(t) - \lambda_j}{c(t)}, \qquad j = 1,\ldots,r,\ i \in I(j),$$

where λ_j is given by

$$\lambda_j = \lambda_j(t) + \frac{c(t)}{m_j}\left[\sum_{i\in I(j)} e'_{ji}x_i - s_j\right], \qquad j = 1,\ldots,r.$$

By combining these two equations, the iteration for z_{ji} becomes

$$z_{ji} := e'_{ji}x_i - \frac{1}{m_j}(e'_j x - s_j), \qquad j = 1,\ldots,r,\ i \in I(j).$$

This relation can be used to eliminate z_{ji} from Eq. (4.63), thereby obtaining the following highly parallelizable iteration for minimizing the Augmented Lagrangian:

$$x_i := \arg\min_{\xi_i\in P_i}\left\{F_i(\xi_i) + \sum_{\{j|i\in I(j)\}}\left\{\lambda_j(t)e'_{ji}\xi_i + \frac{c(t)}{2}\left(e'_{ji}(\xi_i - x_i) + w_j\right)^2\right\}\right\},$$
$$\forall\ i = 1,\ldots,m, \tag{4.68}$$

where w_j is given in terms of x by

$$w_j = \frac{1}{m_j}(e'_j x - s_j), \qquad j = 1,\ldots,r. \tag{4.69}$$

Example 4.5. *Multiplier Methods for Linear Programming*

Consider the linear program

$$\text{minimize } a'x$$
$$\text{subject to } Ex = s, \qquad 0 \le x \le b,$$

where $a \in \Re^m$, $b \in \Re^m$, $s \in \Re^r$, and E is a given $r \times m$ matrix. The method of multipliers is given by

$$x(t+1) = \arg\min_{0 \le x \le b} \left\{ a'x + p(t)'(Ex - s) + \frac{c(t)}{2} \|Ex - s\|_2^2 \right\},$$

$$p(t+1) = p(t) + c(t)\big(Ex(t+1) - s\big).$$

By expanding the quadratic form $\|Ex - s\|_2^2$, collecting terms, and neglecting those terms that do not depend on x, we can write the minimization of the Augmented Lagrangian as

$$\text{minimize} \quad \frac{c(t)}{2} x'E'Ex + \big[a + E'\big(p(t) - c(t)s\big)\big]'x$$

$$\text{subject to } 0 \le x \le b.$$

This quadratic program can be solved using the Gauss–Seidel method of Eqs. (4.10) and (4.11), suitably modified to take into account the additional upper bound constraint $x \le b$. In this modification, the unconstrained minimum of the cost function along each coordinate i is projected on the interval $[0, b_i]$ instead of being projected on the interval $[0, \infty)$ as in Eq. (4.5).

Let $e_i \in \Re^m$ be the ith column of E. Initially we choose $x \ge 0$ and we let $y = -Ex$. At each iteration, we select an index $i \in \{1, \ldots, n\}$ and we update x and y according to

$$x := \left[x_i - \frac{1}{c(t)\|e_i\|_2^2} \big[a_i + e_i'\big(p(t) - c(t)(s + y)\big)\big] \right]^+ v_i, \qquad (4.70a)$$

where $[\cdot]^+$ denotes projection on the interval $[0, b_i]$, v_i is the ith unit vector, and

$$y := y + \left\{ x_i - \left[x_i - \frac{1}{c(t)\|e_i\|_2^2} \big[a_i + e_i'\big(p(t) - c(t)(s + y)\big)\big] \right]^+ \right\} e_i. \qquad (4.70b)$$

Note that similarly as for the Gauss–Seidel algorithm of Eqs. (4.10)–(4.11), the iterations of any two indices i_1 and i_2 are decoupled if there is no coordinate that is nonzero for both e_{i_1} and e_{i_2}. Thus the method is highly parallelizable for favorable sparsity structures of the matrix E. There are a number of variations of this method including hybrid Gauss–Seidel and Jacobi schemes as discussed in Subsection 3.4.1.

An alternative to the preceding method is obtained by viewing the linear program as a separable problem and by applying the corresponding multiplier method given by Eqs. (4.67)–(4.69). By using the identifications $n_i = 1$, $F_i(x_i) = a_i x_i$, $P_i = \{x_i \mid 0 \le x_i \le b_i\}$, for all i, it is straightforward to verify that this method is given by the multiplier iteration [cf. Eq. (4.67)]

$$\lambda_j(t+1) = \lambda_j(t) + \frac{c(t)}{m_j}\big(e_j' x_i(t+1) - s_j\big), \qquad j = 1, \ldots, r,$$

where $x(t+1)$ minimizes the corresponding Augmented Lagrangian and is obtained via the iteration [cf. Eqs. (4.68) and (4.69)]

$$x_i := \arg\min_{0\le\xi_i\le b_i}\left\{\left(a_i+\sum_{\{j\mid i\in I(j)\}}\lambda_j(t)e_{ji}\right)\xi_i+\frac{c(t)}{2}\sum_{\{j\mid i\in I(j)\}}\big(e_{ji}(\xi_i-x_i)+w_j\big)^2\right\},$$
$$i=1,\ldots,m,$$

where

$$w_j=\frac{1}{m_j}(e_j'x-s_j),\qquad j=1,\ldots,r.$$

The above one–dimensional quadratic minimization can be carried out analytically yielding, after some calculation, the iteration

$$x_i:=\left[x_i-\frac{1}{c(t)\|e_i\|_2^2}\big[a_i+e_i'\big(\lambda(t)+c(t)w\big)\big]\right]^+, \tag{4.71}$$

where $[\cdot]^+$ denotes projection on the interval $[0,b_i]$, e_i is the ith column of E, and $\lambda(t)$ and w are the vectors with coordinates $\lambda_j(t)$ and w_j, respectively. This iteration bears considerable resemblance with the alternative iteration (4.70), but it is of the Jacobi type, that is, it can be executed simultaneously for all i.

The main potential difficulty with the preceding methods of multipliers is that the Gauss–Seidel iterations used to minimize the Augmented Lagrangian may converge very slowly. In such cases, it may be useful to try to accelerate convergence by using Newton–like methods specially designed for minimizing quadratic functions subject to upper and lower bounds on the variables (see [Ber82a], [Ber82b], and [Tho87]).

A number of modifications to the method of multipliers have been suggested in order to make it more suitable for decomposition techniques. One such modification is discussed in the following.

The Alternating Direction Method of Multipliers

We draw motivation for this method from Example 4.2 which involves the problem

$$\text{minimize}\quad\sum_{i=1}^m F_i(x)$$
$$\text{subject to}\quad x\in P_i,\qquad i=1,\ldots,m.$$

We saw that one implementation of the method of multipliers for this problem alternately updates x and x_i, and changes the multipliers $p_i(t)$ only after (typically) many updates of x and x_i (enough to minimize the Augmented Lagrangian within adequate precision). An interesting variation is to perform only a small number, k, of minimizations with

respect to x and x_i before changing the multipliers. In the extreme case where $k = 1$, the method takes the form

$$x(t+1) = \frac{\sum_{i=1}^m x_i(t)}{m} - \frac{\sum_{i=1}^m p_i(t)}{mc}, \tag{4.72}$$

$$x_i(t+1) = \arg\min_{x_i \in P_i} \left\{ F_i(x_i) - p_i(t)x_i + \frac{c}{2}\|x(t+1) - x_i\|_2^2 \right\}, \qquad \forall\, i = 1, \ldots, m, \tag{4.73}$$

$$p_i(t+1) = p_i(t) + c\big(x(t+1) - x_i(t+1)\big), \qquad \forall\, i = 1, \ldots, m. \tag{4.74}$$

Thus, this method operates in cycles, where in each cycle we minimize the Augmented Lagrangian with respect to one set of variables, then minimize it with respect to the remaining variables, and then carry out a multiplier update. We use the name *alternating direction multiplier method* to refer to this type of algorithm. The name comes from its similarity with some methods for solving differential equations, known as alternating direction methods (see [FoG83] and [GlL87] for detailed explanations).

Consider next the separable problem of Example 4.4:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^m F_i(x_i) \\ \text{subject to} \quad & e_j' x = s_j, \qquad j = 1, \ldots, r, \\ & x_i \in P_i, \qquad i = 1, \ldots, m. \end{aligned}$$

The natural alternating direction multiplier method is given by [cf. Eqs. (4.67)–(4.69)]

$$x_i(t+1) = \arg\min_{x_i \in P_i} \left\{ F_i(x_i) + \sum_{\{j \mid i \in I(j)\}} \left\{ \lambda_j(t) e_{ji}' x_i + \frac{c}{2}\big(e_{ji}'(x_i - x_i(t)) + w_j(t)\big)^2 \right\} \right\}, \qquad i = 1, \ldots, m, \tag{4.75a}$$

$$\lambda_j(t+1) = \lambda_j(t) + c w_j(t+1), \qquad j = 1, \ldots, r, \tag{4.75b}$$

where

$$w_j(t) = \frac{1}{m_j}\left(e_j' x(t) - s_j\right), \qquad j = 1, \ldots, r, \tag{4.75c}$$

and the initial vectors $x(0)$ and $\lambda(0)$ are arbitrary. It is seen that this is a highly parallelizable method, which applies to convex separable problems that are not necessarily strictly convex, including general linear programs.

We now formulate more precisely the alternating direction method of multipliers and prove its convergence. The starting point is the optimization problem

$$\begin{aligned} &\text{minimize} \quad G_1(x) + G_2(Ax) \\ &\text{subject to} \quad x \in C_1, \ Ax \in C_2. \end{aligned} \tag{4.76}$$

Here, $G_1 : \Re^n \mapsto \Re$ and $G_2 : \Re^m \mapsto \Re$ are convex functions, A is an $m \times n$ matrix, and $C_1 \subset \Re^n$ and $C_2 \subset \Re^m$ are nonempty polyhedral sets. As in our earlier development, we are assuming polyhedral constraint sets to be able to use the duality theory developed in Appendix C. The subsequent algorithm and convergence result can be formulated for more general convex constraint sets.

We will make the following assumption:

Assumption 4.1. The optimal solution set X^* of problem (4.76) is nonempty. Furthermore, either C_1 is bounded or else the matrix $A'A$ is invertible.

A slightly more general version of Assumption 4.1 requires that the level sets $\{x \in C_1 \mid G_1(x) \leq \alpha\}$ be compact for all $\alpha \in \Re$ in place of the condition that C_1 be compact. The subsequent convergence result can also be proved under this version of Assumption 4.1 using a somewhat more complicated analysis.

We introduce an additional vector $z \in \Re^m$ and reformulate the problem as

$$\begin{aligned} &\text{minimize} \quad G_1(x) + G_2(z) \\ &\text{subject to} \quad x \in C_1, \ z \in C_2, \ Ax = z. \end{aligned} \tag{4.77}$$

We assign a Lagrange multiplier vector $p \in \Re^m$ to the equality constraint $Ax = z$, and we consider the Augmented Lagrangian function

$$L_c(x, z, p) = G_1(x) + G_2(z) + p'(Ax - z) + \frac{c}{2}\|Ax - z\|_2^2. \tag{4.78}$$

The alternating direction method of multipliers is given by

$$x(t+1) = \arg\min_{x \in C_1} \left\{ G_1(x) + p(t)'Ax + \frac{c}{2}\|Ax - z(t)\|_2^2 \right\}, \tag{4.79}$$

$$z(t+1) = \arg\min_{z \in C_2} \left\{ G_2(z) - p(t)'z + \frac{c}{2}\|Ax(t+1) - z\|_2^2 \right\}, \tag{4.80}$$

$$p(t+1) = p(t) + c\big(Ax(t+1) - z(t+1)\big). \tag{4.81}$$

The parameter c is any positive number, and the initial vectors $p(0)$ and $z(0)$ are arbitrary. Note that the functions G_1 and G_2, and the constraint sets C_1 and C_2 have been decoupled in the minimization problems of Eqs. (4.79) and (4.80); this turns out to be very useful in some problems.

Prop. 4.1(a) shows that the minimum with respect to z in Eq. (4.80) is attained. The minimum with respect to x in Eq. (4.79) is attained if C_1 is compact by the Weierstrass theorem (Prop. A.8 in Appendix A) or if the matrix $A'A$ is invertible, in which case the

quadratic term in Eq. (4.79) is positive definite, and a slight modification of the proof of Prop. 4.1(a) applies. Therefore, under Assumption 4.1, the minima in Eqs. (4.79) and (4.80) are attained, and the algorithm is well defined.

Note that we can consider changing c from one iteration of the algorithm to the next, but there is no clear reason why we would want to do so in the alternating direction method. (This is in contrast with the method of multipliers, where increasing c is often useful in practice.) Furthermore, practical experience shows that the proper choice of c may require considerably more experimentation in this method than in the method of multipliers.

It can be seen that both algorithms of Eqs. (4.72)–(4.74) and Eq. (4.75) are special cases of the general alternating direction multiplier method of Eqs. (4.79)–(4.81). Indeed, the algorithm of Eqs. (4.72)–(4.74) for minimizing the sum of convex functions $\sum_{i=1}^m F_i(x)$ over $x \in \cap_{i=1}^m P_i$ is obtained with the identifications

$$G_1(x) = 0, \qquad C_1 = \Re^n,$$

$$A = \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix}, \qquad (I \text{ is the } n \times n \text{ identity matrix}),$$

$$G_2(z_1, \ldots, z_m) = \sum_{i=1}^m F_i(z_i), \qquad C_2 = P_1 \times P_2 \times \ldots \times P_m.$$

The algorithm of Eq. (4.75) for minimizing the separable function $\sum_{i=1}^m F_i(x_i)$ subject to the constraints $Ex = s$ and $x_i \in P_i$ is obtained with the identifications

$$G_1(x) = \sum_{i=1}^m F_i(x_i), \qquad C_1 = P_1 \times P_2 \times \ldots \times P_m,$$

$$G_2(z) = 0, \qquad C_2 = \left\{ z \,\middle|\, \sum_{i \in I(j)} z_{ji} = s_j,\ j = 1, \ldots, r \right\},$$

and with A being the matrix that maps x into the vector having coordinates $e'_{ji} x_i$, $j = 1, \ldots, r$, $i \in I(j)$.

The following proposition gives the main convergence properties of the alternating direction method.

Proposition 4.2. Let Assumption 4.1 hold. A sequence $\{x(t), z(t), p(t)\}$ generated by the algorithm of Eqs. (4.79)–(4.81) is bounded, and every limit point of $\{x(t)\}$ is an optimal solution of the original problem (4.76). Furthermore $\{p(t)\}$ converges to an optimal solution p^* of the dual problem [cf. Eq. (4.77)]

$$\begin{aligned} &\text{maximize} \quad H_1(p) + H_2(p) \\ &\text{subject to} \quad p \in \Re^m, \end{aligned} \tag{4.82}$$

where for all $p \in \Re^n$,

$$H_1(p) = \inf_{x \in C_1} \{G_1(x) + p'Ax\}, \qquad H_2(p) = \inf_{z \in C_2} \{G_2(z) - p'z\}. \tag{4.83}$$

The following lemma will be useful for proving Prop. 4.2.

Lemma 4.1. If $y^* = \arg\min_{y \in Y}\{J_1(y) + J_2(y)\}$, where $J_1 : \Re^n \mapsto \Re$ and $J_2 : \Re^n \mapsto \Re$ are convex functions, Y is a polyhedral subset of $\Re^n$, and J_2 is continuously differentiable, then

$$y^* = \arg\min_{y \in Y} \{J_1(y) + \nabla J_2(y^*)'y\}. \tag{4.84}$$

Proof. We have that

$$(y^*, y^*) = \arg \min_{y \in Y,\ z \in \Re^n,\ z = y} \{J_1(y) + J_2(z)\}.$$

By the Lagrange Multiplier Theorem of Appendix C, there exists $\lambda \in \Re^n$ such that

$$y^* = \arg\min_{y \in Y} \{J_1(y) + \lambda'y\}, \tag{4.85}$$

$$y^* = \arg\min_{z \in \Re^n} \{J_2(z) - \lambda'z\}. \tag{4.86}$$

From Eq. (4.86) we obtain $\lambda = \nabla J_2(y^*)$, which together with Eq. (4.85) proves the result. **Q.E.D.**

Proof of Prop. 4.2. By applying Lemma 4.1 with the identifications $Y = C_1$, $J_1(x) = G_1(x)$, $J_2(x) = p(t)'Ax + (c/2)\|Ax - z(t)\|_2^2$ [cf. Eq. (4.79)] we obtain

$$\begin{aligned} &G_1\big(x(t+1)\big) + \big[p(t) + c\big(Ax(t+1) - z(t)\big)\big]' Ax(t+1) \\ &\quad \le G_1(x) + \big[p(t) + c\big(Ax(t+1) - z(t)\big)\big]' Ax, \qquad \forall\, x \in C_1. \end{aligned} \tag{4.87}$$

Similarly we obtain [cf. Eq. (4.80)]

$$\begin{aligned} &G_2\big(z(t+1)\big) - \big[p(t) + c\big(Ax(t+1) - z(t+1)\big)\big]' z(t+1) \\ &\quad \le G_2(z) - \big[p(t) + c\big(Ax(t+1) - z(t+1)\big)\big]' z, \qquad \forall\, z \in C_2. \end{aligned} \tag{4.88}$$

Let (x^*, z^*) be an optimal solution of problem (4.77) and let p^* be an optimal solution of its dual problem (4.82). By applying Eq. (4.87) with $x = x^*$, and Eq. (4.88) with $z = z^*$, and by using also the multiplier update formula (4.81) we have

$$G_1\big(x(t+1)\big) + p(t+1)'Ax(t+1) + c\big(z(t+1) - z(t)\big)'Ax(t+1) \leq G_1(x^*) + p(t+1)'Ax^* + c\big(z(t+1) - z(t)\big)'Ax^*,$$

$$G_2\big(z(t+1)\big) - p(t+1)'z(t+1) \leq G_2(z^*) - p(t+1)'z^*.$$

By adding these two relations and using also the fact $Ax^* = z^*$, we obtain

$$G_1\big(x(t+1)\big)+G_2\big(z(t+1)\big)+p(t+1)'\big(Ax(t+1)-z(t+1)\big)+c\big(z(t+1)-z(t)\big)'A\big(x(t+1)-x^*\big) \leq G_1(x^*) + G_2(z^*). \tag{4.89}$$

By the Saddle Point Theorem of Appendix C, we must have

$$G_1(x^*)+G_2(z^*) \leq G_1\big(x(t+1)\big)+G_2\big(z(t+1)\big)+p^{*\prime}\big(Ax(t+1)-z(t+1)\big), \qquad \forall\, t. \tag{4.90}$$

By adding Eqs. (4.89) and (4.90) we obtain

$$\big(p(t+1)-p^*\big)'\big(Ax(t+1)-z(t+1)\big)+c\big(z(t+1)-z(t)\big)'A\big(x(t+1)-x^*\big) \leq 0. \tag{4.91}$$

We now denote for all t

$$\bar{x}(t) = x(t) - x^*, \qquad \bar{z}(t) = z(t) - z^*, \qquad \bar{p}(t) = p(t) - p^*,$$

and we observe that, since $Ax^* = z^*$, we can write the multiplier update formula (4.81) as

$$\bar{p}(t+1) = \bar{p}(t) + c\big(A\bar{x}(t+1) - \bar{z}(t+1)\big),$$

and

$$\bar{p}(t+1) = \bar{p}(t) + c\big(Ax(t+1) - z(t+1)\big).$$

By using the preceding relations in Eq. (4.91), we obtain after collecting terms

$$\frac{1}{c}\bar{p}(t+1)'\big(\bar{p}(t+1)-\bar{p}(t)\big)+c\big(\bar{z}(t+1)-\bar{z}(t)\big)'\bar{z}(t+1)+\big(\bar{z}(t+1)-\bar{z}(t)\big)'\big(\bar{p}(t+1)-\bar{p}(t)\big) \leq 0. \tag{4.92}$$

We estimate each of the three terms in the preceding relation. We have

$$\bar{p}(t+1)'\big(\bar{p}(t+1) - \bar{p}(t)\big) = \tfrac{1}{2}\|\bar{p}(t+1) - \bar{p}(t)\|_2^2 + \tfrac{1}{2}\|\bar{p}(t+1)\|_2^2 - \tfrac{1}{2}\|\bar{p}(t)\|_2^2, \tag{4.93}$$

$$\big(\bar{z}(t+1) - \bar{z}(t)\big)'\bar{z}(t+1) = \tfrac{1}{2}\|\bar{z}(t+1) - \bar{z}(t)\|_2^2 + \tfrac{1}{2}\|\bar{z}(t+1)\|_2^2 - \tfrac{1}{2}\|\bar{z}(t)\|_2^2. \tag{4.94}$$

To estimate the third term in Eq. (4.92), we consider the optimality relation (4.88) with $z = z(t)$, that is,

$$G_2\big(z(t+1)\big) - p(t+1)'z(t+1) \leq G_2\big(z(t)\big) - p(t+1)'z(t), \tag{4.95}$$

and we consider also Eq. (4.88) at iteration t with $z = z(t+1)$, that is,

$$G_2\big(z(t)\big) - p(t)'z(t) \leq G_2\big(z(t+1)\big) - p(t)'z(t+1). \tag{4.96}$$

Adding Eqs. (4.95) and (4.96) we obtain $0 \leq \big(z(t+1) - z(t)\big)'\big(p(t+1) - p(t)\big)$ or equivalently

$$0 \leq \big(\bar{z}(t+1) - \bar{z}(t)\big)'\big(\bar{p}(t+1) - \bar{p}(t)\big). \tag{4.97}$$

We now use Eqs. (4.93), (4.94), and (4.97) in inequality (4.92). We obtain

$$\|\bar{p}(t+1)-\bar{p}(t)\|_2^2+c^2\|\bar{z}(t+1)-\bar{z}(t)\|_2^2 \leq \big(\|\bar{p}(t)\|_2^2+c^2\|\bar{z}(t)\|_2^2\big)-\big(\|\bar{p}(t+1)\|_2^2+c^2\|\bar{z}(t+1)\|_2^2\big). \tag{4.98}$$

It follows that

$$\bar{p}(t+1) - \bar{p}(t) \to 0, \qquad \bar{z}(t+1) - \bar{z}(t) \to 0. \tag{4.99}$$

Since $\bar{p}(t+1) - \bar{p}(t) = c\big(Ax(t+1) - z(t+1)\big)$, we obtain from Eqs. (4.89) and (4.90)

$$\lim_{t\to\infty}\big[G_1\big(x(t+1)\big)+G_2\big(z(t+1)\big)\big] = G_1(x^*)+G_2(z^*) = \min_{x\in C_1,\ z\in C_2,\ Ax=z}\{G_1(x)+G_2(z)\}. \tag{4.100}$$

Furthermore, for every limit point $(\tilde{x}, \tilde{z})$ of $\big\{\big(x(t), z(t)\big)\big\}$ we have that $A\tilde{x} = \tilde{z}$, and that $\tilde{x}$ is an optimal solution of the original problem (4.76).

From Eq. (4.98) we obtain that $\{p(t)\}$ and $\{z(t)\}$ are bounded, and from Eqs. (4.81) and (4.99) we see that $\|Ax(t) - z(t)\|_2^2 \to 0$. In view of Assumption 4.1 it follows that $\{x(t)\}$ is also bounded. Consider a convergent subsequence $\big\{\big(x(t), z(t), p(t)\big) \mid t \in T\big\}$, and let $(\tilde{x}, \tilde{z}, \tilde{p})$ be its limit. Then, as shown earlier, $\tilde{x}$ is an optimal solution of the original problem (4.76). To show that $\tilde{p}$ is a solution of the dual problem (4.82), define $\hat{p}(t+1) = p(t) + c\big(Ax(t+1) - z(t)\big)$. From the definitions (4.83), and Eqs. (4.87) and (4.88) we see that

$$H_1\big(\hat{p}(t+1)\big) = G_1\big(x(t+1)\big) + \hat{p}(t+1)'Ax(t+1) \leq G_1(x) + \hat{p}(t+1)'Ax, \qquad \forall\ x \in C_1, \tag{4.101}$$

$$H_2\big(p(t+1)\big) = G_2\big(z(t+1)\big) - p(t+1)'z(t+1) \leq G_2(z) - p(t+1)'z, \qquad \forall\ z \in C_2. \tag{4.102}$$

By taking limits in these relations and using the fact that $\tilde{p}$ is also the limit of the subsequence $\{p(t+1) \mid t \in T\}$ [cf. Eq. (4.99)], we obtain

$$\limsup_{t\to\infty,\ t\in T} H_1\big(\hat{p}(t+1)\big) \le G_1(x) + \tilde{p}'Ax, \qquad \forall\ x \in C_1,$$

$$\limsup_{t\to\infty,\ t\in T} H_2\big(p(t+1)\big) \le G_2(z) - \tilde{p}'z, \qquad \forall\ z \in C_2,$$

so

$$\limsup_{t\to\infty,\ t\in T} H_1\big(\hat{p}(t+1)\big) \le H_1(\tilde{p}), \qquad \limsup_{t\to\infty,\ t\in T} H_2\big(p(t+1)\big) \le H_2(\tilde{p}). \tag{4.103}$$

On the other hand by adding Eqs. (4.101) and (4.102), and using the fact $A\tilde{x} = \tilde{z}$, we obtain

$$\begin{aligned}\lim_{t\to\infty,\ t\in T} \big[H_1\big(\hat{p}(t+1)\big) + H_2\big(p(t+1)\big)\big] &= G_1(\tilde{x}) + G_2(\tilde{z}) \\ &= \min_{x\in C_1,\ z\in C_2,\ Ax=z} \{G_1(x) + G_2(z)\}.\end{aligned} \tag{4.104}$$

Since by the Duality Theorem of Appendix C, we have

$$\max_{p\in\Re^n} \{H_1(p) + H_2(p)\} = \min_{x\in C_1,\ z\in C_2,\ Ax=z} \{G_1(x) + G_2(z)\},$$

we obtain from Eqs. (4.103) and (4.104) that $\tilde{p}$ is an optimal solution of the dual problem (4.82).

We now show that $\big\{\big(z(t), p(t)\big)\big\}$ has a unique limit point. Indeed Eq. (4.98) shows that

$$\|p(t) - p^*\|_2^2 + c^2\|z(t) - z^*\|_2^2$$

is a nondecreasing sequence for *every* choice of optimal solutions (x^*, z^*) and p^* of the primal problem (4.77) and the dual problem (4.82), respectively. In particular, any limit point $(\tilde{z}, \tilde{p})$ of $\big\{\big(z(t), p(t)\big)\big\}$ can be used in place of (z^*, p^*) in Eq. (4.98). It follows that $\big\{\big(z(t), p(t)\big)\big\}$ cannot have more than one limit point. **Q.E.D.**

Note that in the course of the preceding proof, we showed that the sequence $\{z(t)\}$ converges, and that $Ax(t) - z(t) \to 0$. It follows that if the matrix $A'A$ is invertible, then $\{x(t)\}$ must also converge, necessarily to an optimal solution of the original problem.

To see what can happen when $A'A$ is not invertible, consider the case where $n = 1$, $C_1 = [0, 1]$, $C_2 = \Re$, $A = 0$, and $G_1(x) = 0$, $G_2(x) = 0$ for all x. Here, the optimal solution set X^* is $[0, 1]$ and Assumption 4.1 is satisfied because C_1 is compact. It can be verified that the sequence $\big\{\big(z(t), p(t)\big)\big\}$ generated by the algorithm converges to $(0, 0)$ in one iteration, but the sequence $\{x(t)\}$ need not converge; it can be any sequence in $[0, 1]$. By changing C_1 to be equal to $\Re$, we obtain an example where X^* is nonempty, but Assumption 4.1 is violated and the generated sequence $\{x(t)\}$ can be unbounded.

We finally mention that there are several variations of the alternating direction method of multipliers. An example is the iteration

$$x(t+1) = \arg\min_{x \in C_1} \left\{ G_1(x) + p(t)'Ax + \frac{c}{2}\|Ax - z(t)\|_2^2 \right\}, \tag{4.105}$$

$$\hat{p}(t) = p(t) + c\big(Ax(t+1) - z(t)\big). \tag{4.106}$$

$$z(t+1) = \arg\min_{z \in C_2} \left\{ G_2(z) - \hat{p}(t)'z + \frac{c}{2}\|Ax(t+1) - z\|_2^2 \right\}, \tag{4.107}$$

$$p(t+1) = \hat{p}(t) + c\big(Ax(t+1) - z(t+1)\big), \tag{4.108}$$

which is the same as the method of Eqs. (4.79)–(4.81) given earlier except for the additional multiplier update (4.106) executed between the updates of x and z. A convergence analysis and a discussion of this and other related methods is given in [GIL87].

EXERCISES

4.1. Consider the problem

$$\begin{aligned} &\text{minimize} && F(x) = \frac{1}{2}x'Px + r'x \\ &\text{subject to} && x \geq 0, \end{aligned}$$

where P is a nonnegative definite symmetric $n \times n$ matrix with positive diagonal elements and $r \in \Re^n$ is given. Let K be the largest eigenvalue of P and assume that $K > 0$.

(a) Show that all the limit points of the sequence generated by the gradient projection method

$$x(t+1) = \Big[x(t) - \gamma \nabla F\big(x(t)\big)\Big]^+,$$

are optimal solutions provided that $\gamma \in (0, 2/K)$. *Hint:* Do Exercise 2.3 in Section 3.2.

(b) Show that the sum of the diagonal elements of P is an upper bound for K.

(c) Consider the linearized Jacobi method

$$x(t+1) = \Big[x(t) - \gamma M^{-1} \nabla F\big(x(t)\big)\Big]^+,$$

where M is the diagonal matrix with diagonal elements equal to the corresponding diagonal elements of P. Show that if $\gamma \in (0, 2/n)$, all limit points of the sequence $\{x(t)\}$ are optimal solutions. *Hint:* Consider the transformation of variables $y(t) = M^{1/2}x(t)$ and use part (b).

4.2. (Convergence Rate of the Proximal Minimization Algorithm [KoB76].) Assume that there exist $\beta > 0$, $\delta > 0$, and $\alpha > 1$ such that

$$F^* + \beta\big(\rho(x; X^*)\big)^\alpha \leq F(x), \qquad \forall\, x \in X \text{ with } \rho(x; X^*) \leq \delta.$$

Let $\{x(t)\}$ be a sequence generated by the proximal minimization algorithm and assume that $\liminf_{t\to\infty} c(t) > 0$. Show that:

(a) If $\alpha < 2$, then

$$\limsup_{t\to\infty} \frac{\rho\big(x(t+1);X^*\big)}{\big(\rho\big(x(t);X^*\big)\big)^{1/(\alpha-1)}} < \infty.$$

This is known as superlinear convergence of order $1/(\alpha-1)$.

Hint: Part (a) and the following parts (b) and (c) are based on the relation

$$\rho\big(x(t+1);X^*\big) + \beta c(t)\big(\rho\big(x(t+1);X^*\big)\big)^{\alpha-1} \le \rho\big(x(t);X^*\big).$$

To show this relation, let $\hat{x}$ denote the projection of any x on X^* and let $d = \hat{x}(t+1) - x(t+1)$. Consider the scalar convex function

$$H(\gamma) = F\big(x(t+1)+\gamma d\big) + \frac{1}{2c(t)}\|x(t+1)+\gamma d - x(t)\|_2^2.$$

Since H is minimized at $\gamma = 0$, its right derivative $H^+(0)$ is nonnegative, from which we obtain

$$\begin{aligned} 0 \le H^+(0) &= F'\big(x(t+1);d\big) + \frac{1}{c(t)}\big(x(t+1)-x(t)\big)'\big(\hat{x}(t+1)-x(t+1)\big) \\ &\le F^* - F\big(x(t+1)\big) + \frac{1}{c(t)}\big(x(t+1)-x(t)\big)'\big(\hat{x}(t+1)-x(t+1)\big). \end{aligned}$$

Using the hypothesis, it follows that

$$\beta c(t)\big(\rho\big(x(t+1);X^*\big)\big)^{\alpha} \le \big(x(t+1)-x(t)\big)'\big(\hat{x}(t+1)-x(t+1)\big),$$

for t sufficiently large. We now add to both sides $\big(x(t+1)-\hat{x}(t)\big)'\big(x(t+1)-\hat{x}(t+1)\big)$ and we use the fact

$$\|x(t+1)-\hat{x}(t+1)\|_2^2 \le \big(x(t+1)-\hat{x}(t)\big)'\big(x(t+1)-\hat{x}(t+1)\big),$$

(which follows from the Projection Theorem) to obtain

$$\|x(t+1)-\hat{x}(t+1)\|_2^2 + \beta c(t)\big(\rho\big(x(t+1);X^*\big)\big)^{\alpha} \le \|x(t)-\hat{x}(t)\|_2\|x(t+1)-\hat{x}(t+1)\|_2,$$

from which the desired relation follows.

(b) If $\alpha = 2$ and $\lim_{t\to\infty} c(t) = \bar{c} < \infty$ then

$$\limsup_{t\to\infty} \frac{\rho\big(x(t+1);X^*\big)}{\rho\big(x(t);X^*\big)} \le \frac{1}{1+\beta\bar{c}}.$$

(c) If $\alpha = 2$ and $\lim_{t\to\infty} c(t) = \infty$, then

$$\limsup_{t\to\infty} \frac{\rho\big(x(t+1);X^*\big)}{\rho\big(x(t);X^*\big)} = 0.$$

This is known as superlinear convergence.

(d) If F is a positive definite quadratic function, $X = \Re^n$, $\alpha = 2$ and $\lim_{t\to\infty} c(t) = \bar{c} < \infty$, then

$$\limsup_{t\to\infty} \frac{\rho\big(x(t+1); X^*\big)}{\rho\big(x(t); X^*\big)} \leq \frac{1}{1+2\beta\bar{c}}.$$

Show by example that this estimate is tight. *Hint:* Let x^* be the minimizing point of F over $x \in \Re^n$, and let $\tilde{y}$ denote the unique vector that minimizes $\beta\|x - x^*\|_2^2 + (1/2c)\|x - y\|_2^2$ over $x \in \Re^n$. Show that $y - x^* = (1 + 2\beta c)(\tilde{y} - x^*)$, and that $\|x(y,c) - x^*\|_2 \leq \|\tilde{y} - x^*\|_2$.

(e) Prove that

$$F\big(x(t+1)\big) - F^* \leq \frac{\rho\big(x(t); X^*\big)^2}{2c(t)},$$

and use this relation to show that

$$\limsup_{t\to\infty} \frac{\rho\big(x(t+1); X^*\big)}{\rho\big(x(t); X^*\big)^{2/\alpha}} < \infty.$$

For $\alpha > 2$, this is known as sublinear convergence.

4.3. Show that the condition (4.29) holds when F is a linear function, X is a polyhedral set, and X^* is nonempty. *Hint:* Suppose that X has the form $\{x \mid a_j'x \leq t_j,\ j = 1,\ldots,m\}$ for some vectors a_j and scalars t_j, and that $F(x) = c'x$ for some vector c. For $x \in X$, let $p(x)$ be the projection of x on X^*, and consider the cone of $\Re^{n+1}$

$$C_x = \{(z,\mu) \mid c'z \leq \mu,\ a_j'z \leq 0 \text{ for all } j \text{ such that } a_j'p(x) = t_j\},$$

and the cones

$$M_x = \{(z,\mu) \in C_x \mid \mu = 0\},$$

$$Z_x = \{(z,\mu) \in C_x \mid \text{the projection of } (z,\mu) \text{ on } M_x \text{ is the origin } (0,0)\}.$$

Show that the collection of distinct sets C_x is finite, and that for each such set there exists $\theta_x > 0$ such that

$$|\mu| \geq \theta_x\|z\|_2, \qquad \forall\ (z,\mu) \in Z_x.$$

Take $\beta = \min_{x\in X} \theta_x$ in condition (4.29).

4.4. Consider the variation of the proximal minimization algorithm given by

$$x(t+1) = x(t) - \gamma(t)\nabla\Phi_c\big(x(t)\big),$$

where $\gamma(t)$ is a stepsize parameter satisfying $\gamma(t) \in [\delta, 2c - \delta]$ for all t and some $\delta \in (0, c]$. Show that all limit points of the sequences that the algorithm generates are optimal solutions. *Hint:* Modify the proof of part (c) of Prop. 4.1.

4.5. (The Method of Multipliers for Inequality Constraints [Roc71].) Consider the problem of Subsection 3.4.4 for the case where we have the inequality constraints $a_j'x - t_j \leq 0$ instead of the equality constraints $e_j'x - s_j = 0$. Replace these inequality constraints by the equality constraints $a_j'x - t_j + w_j = 0$, where w_j is constrained to be nonnegative, and show that the method of multipliers takes the form

$$x(t+1) = \arg\min_{x \in P} \left\{ F(x) + \frac{1}{2c(t)} \sum_{j=1}^{r} \left[\max\{0, p_j(t) + c(t)(a_j'x - t_j)\} \right]^2 \right\},$$

$$p_j(t+1) = \max\left\{0, p_j(t) + c(t)\big(a_j'x(t+1) - t_j\big)\right\}, \qquad \forall\, j = 1, \ldots, r.$$

4.6. Consider the set intersection problem of Example 4.3 and assume that the intersection is nonempty. Show that the parallel projection method of Eqs. (4.58)–(4.59) converges to an element of the intersection. *Hint:* Show that for all $x^* \in C_1 \cap \cdots \cap C_m$ we have $\|x(t+1) - x^*\|_2 \leq \|x(t) - x^*\|_2$.

4.7. Show convergence of the method of Eqs. (4.58)–(4.59), if Eq. (4.58) is replaced by

$$x(t+1) = \sum_{i=1}^{m} \lambda_i x_i(t),$$

where $\lambda_1, \ldots, \lambda_m$ are positive scalars summing to unity.

3.5 VARIATIONAL INEQUALITIES

The variational inequality problem is as follows. We are given a set $X \subset \Re^n$ and a function $f : \Re^n \mapsto \Re^n$, and our objective is to find a vector $x^* \in X$ such that

$$(x - x^*)'f(x^*) \geq 0, \qquad \forall x \in X. \tag{5.1}$$

As a shorthand notation, we will refer to this problem as VI(X, f). It will be assumed throughout that X is nonempty, closed, and convex.

3.5.1 Examples of Variational Inequality Problems

Several interesting problems can be formulated as variational inequality problems and some examples follow.

(a) *Solution of Systems of Equations.* Let $X = \Re^n$ and let $f : \Re^n \mapsto \Re^n$ be a given function. It is easy to see that a vector $x^* \in \Re^n$ solves the problem VI($\Re^n, f$) if and only if $f(x^*) = 0$. Indeed, if $f(x^*) = 0$ then inequality (5.1) holds with equality. Conversely,

if x^* satisfies Eq. (5.1), let $x = x^* - f(x^*)$. By Eq. (5.1), we have $-\|f(x^*)\|_2^2 \geq 0$, which implies that $f(x^*) = 0$.

(b) *Constrained and Unconstrained Optimization.* Let X be nonempty, closed, and convex and let $F : \Re^n \mapsto \Re$ be a continuously differentiable function that is convex on the set X. Using the optimality conditions for convex optimization (Prop. 3.1), a vector $x^* \in X$ minimizes F over the set X if and only if $(x - x^*)'\nabla F(x^*) \geq 0$ for all $x \in X$, that is, if and only if x^* solves the variational inequality problem $\mathrm{VI}(X, \nabla F)$. In particular, if we let $X = \Re^n$, we see that unconstrained convex optimization is also a variational inequality problem. In the optimization context, the function f of Eq. (5.1) has a special structure because it is the gradient of a scalar function F. In particular, the line integral of f depends only on the end points of the path of integration and not on the path itself. In more general variational inequality problems, this path independence property is absent and such problems cannot be formulated as optimization problems; this restricts the tools available for establishing convergence of an algorithm. In particular, the descent approach cannot be applied.

(c) *Traffic Assignment.* We are given a directed graph $G = (N, A)$, which is viewed as a model of a transportation network. The arcs of the graph represent transportation links such as highways, rail lines, etc. The nodes of the graph represent junction points where traffic can exit from one transportation link and enter another. We are also given a set W of node pairs, referred to as origin–destination (OD) pairs. For OD pair $w = (i, j)$, there is a known input $r_w > 0$ representing traffic entering the network at the origin node i of w and exiting the network at the destination node j of w. For each $w \in W$, the input r_w is to be divided among a given collection P_w of simple positive paths starting at the origin node of w and ending at the destination node of w (i.e., these paths have no cycles and their arcs are oriented as in the graph G). We denote by x_p the portion of r_w carried by path p (also called the flow of path p). Let x be the vector having as coordinates all the path flows x_p, $p \in P_w$, $w \in W$. Thus, x must belong to the set

$$X = \left\{ x \;\middle|\; \sum_{p \in P_w} x_p = r_w, \ \forall w \in W, \text{ and } x_p \geq 0, \ \forall p \in P_w, \ \forall w \in W \right\}.$$

For each path p, we are given a function $t_p(x)$, called the travel time of path p. This function models the time required for traffic to travel from the start node to the end node of path p as a function of the path flow vector x. The problem is to find a path flow vector $x^* \in X$ that consists of path flows that are positive only on paths of minimum travel time. That is, for all $w \in W$ and paths $p \in P_w$, we require that

$$x_p^* > 0 \quad \Longrightarrow \quad t_p(x^*) \leq t_{p'}(x^*), \qquad \forall p' \in P_w. \tag{5.2}$$

This problem is based on a transportation hypothesis called the *user–optimization principle*, which asserts that traffic network equilibrium is established when each user of the network chooses, among all available paths, a path requiring minimum travel time.

We claim that a vector $x^* \in X$ satisfies the user–optimization condition (5.2) if and only if x^* is a solution of the variational inequality

$$\sum_{w \in W} \sum_{p \in P_w} (x_p - x_p^*) t_p(x^*) \geq 0, \qquad \forall x \in X, \tag{5.3}$$

which is the variational inequality problem VI(X, f), with $f(x)$ being the function with components $t_p(x)$. To see this, assume that $x^* \in X$ satisfies the condition (5.2), and let

$$T_w^* = \min_{p \in P_w} t_p(x^*).$$

We have for every $x \in X$, $\sum_{p \in P_w} (x_p - x_p^*) = 0$, so that

$$0 = \sum_{p \in P_w} (x_p - x_p^*) T_w^* \leq \sum_{\{p \in P_w \mid x_p > x_p^*\}} (x_p - x_p^*) t_p(x^*) + \sum_{\{p \in P_w \mid x_p < x_p^*\}} (x_p - x_p^*) T_w^*.$$

Condition (5.2) implies that $t_p(x^*) = T_w^*$ if $x_p^* > 0$, so T_w^* can be replaced by $t_p(x^*)$ in the right–hand side of the previous inequality, thereby yielding

$$0 \leq \sum_{p \in P_w} (x_p - x_p^*) t_p(x^*), \qquad \forall x \in X, \ \forall w \in W.$$

By adding this inequality over all OD pairs $w \in W$, we see that x^* satisfies the variational inequality (5.3).

Conversely, assume that x^* satisfies the variational inequality (5.3). Let $p \in P_w$ be a path of some OD pair w with $x_p^* > 0$, and let $\bar{p} \in P_w$ be a path of the same OD pair with $t_{\bar{p}}(x^*) = T_w^*$. Then, either $p = \bar{p}$, in which case the condition (5.2) holds, or $p \neq \bar{p}$, in which case by taking $x_p = 0$, $x_{\bar{p}} = x_{\bar{p}}^* + x_p^*$, and $x_{p'} = x_{p'}^*$ for all other paths $p' \neq p$, $\bar{p}$, we obtain from Eq. (5.3) $x_p^* \big(T_w^* - t_p(x^*)\big) \geq 0$. Since $x_p^* > 0$, we obtain $t_p(x^*) \leq T_w^*$, thereby showing that the condition (5.2) holds.

(d) *Game Theory and Saddle Point Problems.* A *Nash game* is defined as follows. There are m players. Each player i chooses a strategy x_i belonging to a closed convex set $X_i \subset \Re^{n_i}$. Then, the ith player is penalized by an amount equal to $F_i\big(x_1, \ldots, x_m\big)$, where each $F_i : \Re^{n_i} \mapsto \Re$ is a continuously differentiable function. A set $x^* = (x_1^*, \ldots, x_m^*) \in \prod_{i=1}^m X_i$ of strategies is said to be *in equilibrium* if no player is able to reduce the incurred penalty by unilaterally modifying the chosen strategy. That is,

$$F_i\Big(x_1^*, \ldots, x_{i-1}^*, x_i^*, x_{i+1}^*, \ldots, x_m^*\Big) \leq F_i\Big(x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*\Big), \quad \forall x_i \in X_i, \ \forall i.$$

Let us assume that each one of the functions F_i is convex on the set X_i when viewed as a function of x_i alone and the other components are fixed. Using the optimality

conditions for convex optimization (Prop. 3.1), we see that a set of strategies x^* is in equilibrium if and only if $(x_i - x_i^*)'\nabla_i F_i(x^*) \geq 0$ for every $x_i \in X_i$ and every i. Adding these conditions, we conclude that x^* must be a solution of the variational inequality $(x - x^*)'f(x^*) \geq 0$, where $f : \prod_{i=1}^m \Re^{n_i} \mapsto \prod_{i=1}^m \Re^{n_i}$ is given by $f(x) = \big(\nabla_1 F_1(x), \ldots, \nabla_m F_m(x)\big)$. In fact, under our convexity assumptions, the reverse is also true: any solution of the above defined variational inequality provides a set of strategies in equilibrium (this can be seen using Prop. 5.7 to be proved later in this section).

A related problem is the *saddle point problem*, in which we are given a function $F : X \times Y \mapsto \Re$ and our objective is to find a pair $(x^*, y^*) \in X \times Y$ such that

$$F(x^*, y) \leq F(x^*, y^*) \leq F(x, y^*), \qquad \forall x \in X, \ \forall y \in Y.$$

The saddle point problem is seen to be a special case of a Nash game, provided that we let $F_1 = F$ and $F_2 = -F$. Our convexity assumptions for the Nash game translate to a requirement that the function F is convex in x for each fixed y, and concave in y for each fixed x.

An important application of saddle point problems arises in duality theory for constrained convex optimization. The Saddle Point Theorem of Appendix C shows that an optimal primal solution x^* of a primal optimization problem, and an optimal dual solution $y^* = (p^*, u^*)$ of the corresponding dual optimization problem can be found as a saddle point of the Lagrangian function. The latter function has the form $F(x, y)$, is convex in x for each fixed y, and concave in y for each fixed x. It is thus possible to approach the solution of a constrained optimization problem by considering the associated saddle point problem, and by subsequently applying variational inequality algorithms presented in this section. In some situations, where the optimization problem has separable structure (e.g., the problem considered in Subsection 3.4.2), the saddle point problem can be amenable to decomposition and parallelization (see Exercise 5.2).

3.5.2 Preliminaries

A useful necessary and sufficient condition for x^* to be a solution of VI(X, f) is given by the following result, illustrated in Fig. 3.5.1.

Proposition 5.1. (*Fixed Point Characterization of Solutions*) Let γ be a positive scalar and let G be a symmetric positive definite matrix. A vector x^* is a solution of VI(X, f) if and only if $[x^* - \gamma G^{-1} f(x^*)]_G^+ = x^*$, where $[\cdot]_G^+$ is the projection on X with respect to norm $\|x\|_G = (x'Gx)^{1/2}$.

Proof. Suppose that $x^* = [x^* - \gamma G^{-1} f(x^*)]_G^+$. Then, the Scaled Projection Theorem [Prop. 3.7(b)] yields $(x - x^*)'\big(-\gamma f(x^*)\big) \leq 0$ for all $x \in X$, and since γ is positive, it follows that x^* solves VI(X, f). Conversely, suppose that x^* solves VI(X, f). Then, Eq. (5.1) yields

$$(x - x^*)'G\Big(x^* - \big(x^* - G^{-1}\gamma f(x^*)\big)\Big) \geq 0$$

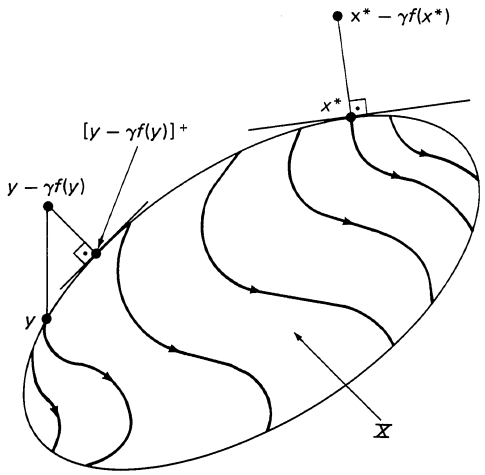

Figure 3.5.1 Illustration of the necessary and sufficient condition for x^* to be a solution of VI(X, f). The function f can be thought of as a vector field on the set X. At the point x^* that solves the variational inequality $(x - x^*)'f(x^*) \geq 0$, the vector field is normal to the boundary and points inwards. For this reason, the projection of $x^* - \gamma f(x^*)$ is equal to x^*, whereas this property is false for other points, such as y.

for all $x \in X$, and the Scaled Projection Theorem implies that $x^* = [x^* - G^{-1}\gamma f(x^*)]_G^+$. **Q.E.D.**

For a fixed positive scalar γ and a symmetric positive definite matrix G, let $R_G : X \mapsto \Re^n$ and $T_G : X \mapsto X$ be the mappings defined by

$$R_G(x) = x - \gamma G^{-1} f(x)$$

and

$$T_G(x) = \left[x - \gamma G^{-1} f(x)\right]_G^+ = \left[R_G(x)\right]_G^+.$$

According to Prop. 5.1, solving the variational inequality VI(X, f) is equivalent to finding a fixed point of the mapping T_G. This allows us to use all of the results on fixed point problems developed in Section 3.1. For instance, we obtain the following existence and uniqueness results.

Proposition 5.2. (*Existence*) Suppose that X is compact and that $f : \Re^n \mapsto \Re^n$ is continuous. Then, there exists a solution to the variational inequality VI(X, f).

Proof. Fix a positive scalar γ and a symmetric positive definite matrix G. If f is continuous then R_G is continuous. Since the projection is a continuous operation (Prop. 3.2), it follows that T_G is continuous as well, and the Leray–Schauder–Tychonoff Fixed Point Theorem (Prop. 1.3) shows that T_G has a fixed point which, by Prop. 5.1, is a solution of VI(X, f). **Q.E.D.**

Figure 3.5.2 shows that if X is not convex, VI(X, f) could have no solutions.

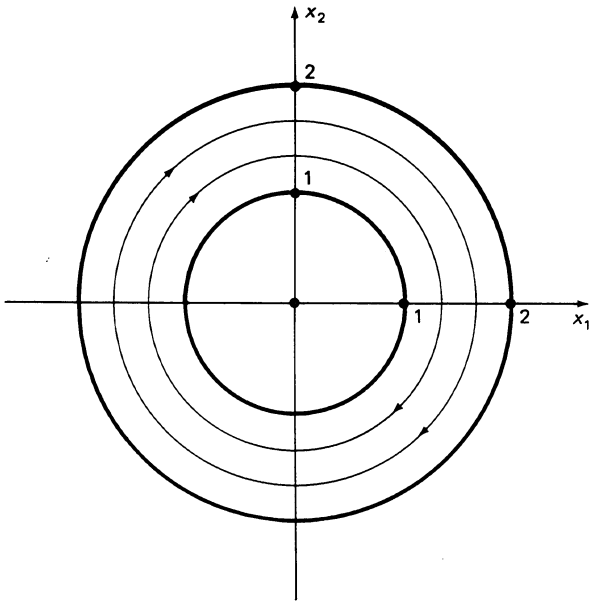

Figure 3.5.2 Illustration of a variational inequality that has no solution. Let $X = \{x \mid 1 \leq \|x\|_2 \leq 2\}$, which is closed but nonconvex. Let $f(x) = (x_2, -x_1)$. The figure shows the corresponding vector field and it is seen that the variational inequality $(x - x^*)'f(x^*) \geq 0$ has no solution.

Proposition 5.3. (*Existence and Uniqueness*) Suppose that there exists some $\gamma > 0$, some symmetric positive definite matrix G, and some $\alpha \in [0, 1)$ such that the mapping R_G satisfies

$$\|R_G(x) - R_G(y)\|_G \leq \alpha\|x - y\|_G, \qquad \forall x, y \in X. \tag{5.4}$$

Then, the problem VI(X, f) has a unique solution.

Proof. Proposition 3.7(d) states that the projection $[\cdot]_G^+$ is nonexpansive with respect to the norm $\|x\|_G = (x'Gx)^{1/2}$. Therefore,

$$\|T_G(x) - T_G(y)\|_G \leq \|R_G(x) - R_G(y)\|_G \leq \alpha\|x - y\|_G.$$

Thus, T_G is a contraction with respect to the norm $\|\cdot\|_G$ and has a unique fixed point x^*. By Prop. 5.1, x^* is the unique solution of VI(X, f). **Q.E.D.**

Recall that some sufficient conditions for the mapping R_G to be a contraction mapping have been furnished in Subsection 3.1.3.

3.5.3 The Projection Algorithm

Since our objective is to find a fixed point of the mapping T_G, it is natural to employ the iteration

$$x(t+1) = T_G(x(t)) = \left[x(t) - \gamma G^{-1} f(x(t))\right]_G^+, \qquad t = 0, 1, \ldots, \tag{5.5}$$

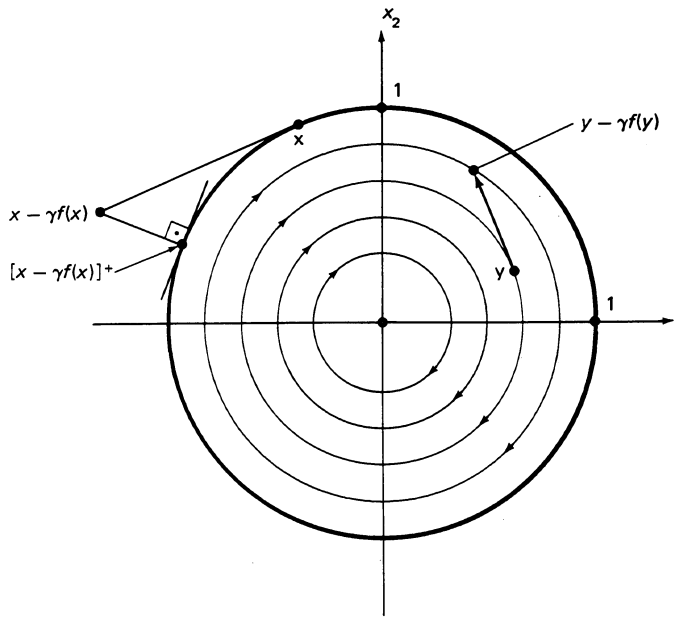

Figure 3.5.3 Illustration of failure of the projection method. Let $X = \{x \mid \|x\|_2 \leq 1\}$ and let $f(x) = (x_2, -x_1)$. The set X is convex and the variational inequality $(x - x^*)'f(x^*) \geq 0$ has the unique solution $x^* = 0$. (Compare with Fig. 3.5.2.) On the other hand, if the projection method is initialized on the boundary, it always stays on the boundary and does not converge. Also, if it is initialized at any nonzero interior point, it moves toward the boundary.

which is called the *projection algorithm*. Here, G is a symmetric positive definite matrix and γ is a positive scalar. Notice that in the special case where f is the gradient of a scalar function F, the projection algorithm of Eq. (5.5) is identical with the scaled gradient projection algorithm of Section 3.3. Unlike the constrained optimization case, the projection algorithm is not guaranteed to converge, as illustrated in Fig. 3.5.3. On the other hand it is guaranteed to converge if the mapping T_G is a contraction. This is always the case if the mapping R_G is a contraction with respect to the norm $\|\cdot\|_G$, because of the nonexpansive property of the projection [Prop. 3.7(d)]. Sufficient conditions for R_G to be such a contraction are provided by Props. 1.12–1.13, specialized to the case of a single block–component ($m = 1$). In particular, we obtain the following result.

Proposition 5.4. (*Convergence of the Projection Algorithm*) Suppose that:

(a) (*Lipschitz Continuity*) There exists some constant K such that

$$\|f(x) - f(y)\|_2 \leq K\|x - y\|_2, \qquad \forall x, y \in X.$$

(b) (*Strong Monotonicity*) There exists some $\alpha > 0$ such that

$$(x - y)'\big(f(x) - f(y)\big) \geq \alpha \|x - y\|_2^2, \qquad \forall x, y \in X. \tag{5.6}$$

Let G be a symmetric positive definite matrix. Then, there exists some $\gamma_0 > 0$ such that for any $\gamma \in (0, \gamma_0]$, T_G is a contraction mapping with respect to the norm $\|\cdot\|_G$. In particular, the problem VI(X, f) has a unique solution and for $\gamma \in (0, \gamma_0]$, the sequence $\{x(t)\}$ generated by the projection algorithm (5.5) converges to it geometrically.

Proof. We use Prop. 1.12 of Subsection 3.13, for the case of a single block–component, to see that R_G is a contraction mapping with respect to the norm $\|\cdot\|_G$, when $\gamma > 0$ is sufficiently small. We then use the nonexpansive property of the projection [Prop. 3.7(d)] to conclude that T_G is also a contraction with respect to the same norm. The result follows from the convergence theorem for contracting iterations. **Q.E.D.**

If f is a function of the form $f(x) = Ax + b$, then the strong monotonicity condition (5.6) is equivalent to the nonnegative definiteness of $A - \alpha I$. In particular, the matrix A must be positive definite.

For another special case, suppose that f is the gradient of a cost function $F : \Re^n \mapsto \Re^n$. Then, the strong monotonicity assumption is equivalent to the requirement that F is strongly convex on the set X. Furthermore, for this particular case, the projection algorithm (with G being the identity matrix) is identical to the gradient algorithm of Section 3.2 (if $X = \Re^n$) or the gradient projection algorithm of Section 3.3 (if X is a convex subset of $\Re^n$). Proposition 5.4 therefore establishes the geometric convergence of the gradient and the gradient projection algorithms in the strongly convex case (Props. 2.4 and 3.5 in Sections 3.2 and 3.3, respectively).

Strong monotonicity is essential for the result of Prop. 5.4 and its geometric significance is illustrated in Fig. 3.5.4. In case f satisfies only the *monotonicity* condition

$$(x - y)'\big(f(x) - f(y)\big) \geq 0, \qquad \forall x, y \in X,$$

then convergence is not guaranteed. For instance, in the example of Fig. 3.5.3, we have $(x - y)'\big(f(x) - f(y)\big) = 0$ for all x, y, the monotonicity condition is satisfied, but strong monotonicity fails to hold and the projection algorithm does not converge. Exercise 5.1 provides a modification of the projection algorithm that converges appropriately under the monotonicity assumption.

The next result is a restatement of Prop. 5.4 for the case where f is affine.

Proposition 5.5. (*Convergence of the Projection Algorithm for Linear Problems*) Suppose that $f(x) = Ax + b$, where b is a vector in $\Re^n$ and A is a positive definite (not necessarily symmetric) $n \times n$ matrix. Then, the variational inequality VI(X, f) has a unique solution x^* and for any positive definite symmetric matrix G, the projection algorithm $x(t+1) = T_G\big(x(t)\big)$ converges to x^* geometrically, provided that γ is small enough.

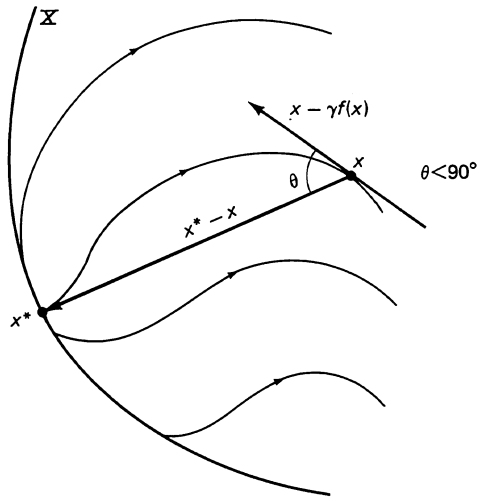

Figure 3.5.4 Interpretation of the strong monotonicity condition (5.6). Let x^* satisfy $(x - x^*)'f(x^*) \geq 0$ for all $x \in X$. Suppose that Eq. (5.6) is true and consider the trajectories determined by the vector field f. Using Eq. (5.6), with $y = x^*$, we obtain

$$\begin{aligned}(x - x^*)'f(x) &\geq (x - x^*)'f(x^*) \\ &\quad + \alpha\|x - x^*\|_2^2 \\ &\geq \alpha\|x - x^*\|_2^2.\end{aligned}$$

In particular, if $x \neq x^*$, then the angle between $-f(x)$ and $x^* - x$ is smaller than 90 degrees. This means that if the trajectories of the vector field are followed in the reverse direction, the distance from x^* decreases. The projection method for the case where G is the identity matrix can be visualized as an attempt to follow these trajectories. This is done most accurately when γ is very small. Of course, a very small value of γ is undesirable because it slows convergence. In any case, with γ sufficiently small, the projection method inherits the properties of the vector field, that is, the distance from x^* is decreased at each step.

Proof. We have $(x-y)'\big(f(x)-f(y)\big) = (x-y)'A(x-y) = \frac{1}{2}(x-y)'(A+A')(x-y) \geq \alpha\|x-y\|_2^2$ for some $\alpha > 0$, because $A + A'$ is symmetric and positive definite. The result follows from Proposition 5.4. **Q.E.D.**

We now discuss an application of the projection algorithm to constrained optimization problems with equality constraints. For simplicity, we only consider the case of a quadratic cost function and linear equality constraints, although the following discussion generalizes to broader classes of problems. Let B and C be matrices of dimensions $n \times n$ and $m \times n$, respectively. Consider the problem of minimizing the cost function $\frac{1}{2}x'Bx$ over all $x \in \Re^n$ satisfying the equality constraint $Cx = b$, where b is a vector in $\Re^m$. We assume that B is symmetric positive definite, which implies that the cost function under consideration is strictly convex. We form the Lagrangian

$$L(x,p) = \tfrac{1}{2}x'Bx + p'Cx - p'b,$$

where p is an m–dimensional vector. Let $\nabla_x L$ and $\nabla_p L$ be the vectors of partial derivatives of the Lagrangian with respect to the components of the vectors x and p, respectively. In the present context, $\nabla_x L(x,p) = Bx + C'p$ and $\nabla_p L(x,p) = Cx - b$. As shown in Appendix C and as discussed in the saddle point example in the beginning of this section, a possible approach for solving the constrained optimization problem is to look for a saddle point of the function L. This is equivalent to solving the variational inequality $\mathrm{VI}(X,f)$, where $X = \Re^{n+m}$ and the function $f : \Re^{n+m} \mapsto \Re^{n+m}$ is given by

$$f(x,p) = \begin{bmatrix} \nabla_x L(x,p) \\ -\nabla_p L(x,p) \end{bmatrix} = \begin{bmatrix} Bx + C'p \\ -Cx + b \end{bmatrix} = \begin{bmatrix} B & C' \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ p \end{bmatrix} + \begin{bmatrix} 0 \\ b \end{bmatrix}.$$

The projection algorithm for this problem is given by

$$x(t+1) = x(t) - \gamma Bx(t) - \gamma C'p(t),$$
$$p(t+1) = p(t) + \gamma Cx(t) - \gamma b.$$

Even though the matrix B is assumed positive definite, the matrix

$$A = \begin{bmatrix} B & C' \\ -C & 0 \end{bmatrix}$$

is not positive definite because of the zero block in the lower right–hand corner. For this reason, the strong monotonicity condition (5.6) fails to hold. A direct calculation yields

$$\begin{bmatrix} x' & p' \end{bmatrix} \begin{bmatrix} B & C' \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ p \end{bmatrix} = x'Bx \geq 0, \qquad \forall x, p,$$

and this shows that inequality (5.6) is satisfied with $\alpha = 0$. Thus, the monotonicity (as opposed to strong monotonicity) condition holds. In particular, the extragradient method of Exercise 5.1 is applicable and is guaranteed to converge. It turns out that the projection method is also guaranteed to converge for this example, provided that the matrix C has full rank; a proof can be found in [Ber82a, p. 232].

3.5.4 Linearized Algorithms

Let x^0 be an element of X and let G be symmetric positive definite. Using the Scaled Projection Theorem [Prop. 3.7(b)], we see that $T_G(x^0)$ could be defined as the unique vector $x^1 \in X$ satisfying

$$(x - x^1)'G\Big(x^1 - x^0 + \gamma G^{-1} f(x^0)\Big) \geq 0, \qquad \forall x \in X.$$

Equivalently, since $\gamma > 0$, we see that x^1 satisfies

$$(x - x^1)'\Big(f(x^0) + \mu G(x^1 - x^0)\Big) \geq 0, \qquad \forall x \in X, \tag{5.7}$$

where $\mu = 1/\gamma$, which is again a variational inequality. In particular, it is the variational inequality problem $\mathrm{VI}(X, g)$, where $g(x) = f(x^0) + \mu G(x - x^0)$. However, it is in general easier to solve than the original variational inequality $\mathrm{VI}(X, f)$ because the function g is linear in the variable x. We can thus think of the projection algorithm as a method that solves a variational inequality by successively solving a sequence of simpler variational inequalities. Based on this observation, a variety of different algorithms are obtained

by choosing differently the variational inequality to be solved at each stage. These algorithms are classified as *linearized* or *nonlinear*, depending on whether the variational inequality solved at each stage involves a linear or a nonlinear function, respectively.

In a general linearized algorithm [Daf83], having computed $x(t)$, we compute $x(t+1)$ by solving the variational inequality $\mathrm{VI}(X, g_t)$, where the function g_t has the form

$$g_t(x) = f\big(x(t)\big) + A\big(x(t)\big)\big(x - x(t)\big).$$

Equivalently, we are looking for a vector $x(t+1)$ satisfying

$$\big(x - x(t+1)\big)'\Big(f\big(x(t)\big) + A\big(x(t)\big)\big(x(t+1) - x(t)\big)\Big) \geq 0, \qquad \forall x \in X.$$

Here, $A\big(x(t)\big)$ is a positive definite (not necessarily symmetric) scaling matrix, depending on $x(t)$. It follows (Prop. 5.5) that the previous variational inequality has a unique solution and the linearized algorithm is well–defined.

Different linearized algorithms correspond to different choices of the scaling matrices $A(x)$. [Accordingly, we will be referring to "the linearized algorithm determined by $\{A(x) \mid x \in X\}$".] Once these scaling matrices have been fixed, a linearized algorithm can be cast into the standard form

$$x(t+1) = T\big(x(t)\big),$$

where $T(x)$ is defined as the unique element of X satisfying

$$\big(y - T(x)\big)'\Big(f(x) + A(x)\big(T(x) - x\big)\Big) \geq 0, \qquad \forall y \in X. \tag{5.8}$$

As a concrete example, if $A(x) = \mu G$, for all x, where G is symmetric positive definite and $\mu > 0$, we recover the projection algorithm [see the variational inequality (5.7)]. To motivate some reasonable choices of $A(x)$, we consider the unconstrained optimization context, where $f(x) = \nabla F(x)$ for some cost function F and $X = \Re^n$. In this case, the solution of the variational inequality (5.8) is

$$T(x) = x - \big(A(x)\big)^{-1}\nabla F(x). \tag{5.9}$$

In this context, it is desirable to let $A(x)$ be an approximation of $\nabla^2 F(x) = \nabla f(x)$. Generalizing this prescription, a common choice is to let $A(x)$ be a diagonal matrix whose diagonal entries are equal to the diagonal entries of the matrix $\nabla f(x)$.

The following is a general result on the convergence of linearized algorithms, although its conditions are not always easy to verify. The proof is omitted because it is a special case of Prop. 5.8, which is proved later.

Proposition 5.6. (*Convergence of Linearized Algorithms*) Suppose that the variational inequality $\mathrm{VI}(X, f)$ has a solution x^*. Consider the linearized algorithm determined

by $\{A(x) \mid x \in X\}$. Suppose that there exists a symmetric positive definite matrix G and some $\delta > 0$ such that the matrix $A(x) - \delta G$ is nonnegative definite for every $x \in X$. Furthermore, suppose that for some $\alpha \in [0, 1)$,

$$\left\| G^{-1}\Big(f(x) - f(y) - A(y)(x - y)\Big) \right\|_G \leq \delta\alpha \|x - y\|_G, \qquad \forall x, y \in X, \tag{5.10}$$

where $\|z\|_G = (z'Gz)^{1/2}$. Then the sequence $\{x(t)\}$ generated by the linearized algorithm converges to x^* geometrically, and x^* is the unique solution of VI(X, f).

As discussed earlier, in the projection method, we have $A(x) = G/\gamma$. Thus, if we let $\delta = 1/\gamma$, then $A(x) - \delta G = 0$, which is nonnegative definite, and the hypothesis (5.10) in this proposition is equivalent to the statement that the mapping R_G, given by $R_G(x) = x - \gamma G^{-1} f(x)$, is a contraction. Thus, Prop. 5.6 generalizes our earlier results on the convergence of the projection method (Prop. 5.4).

3.5.5 The Cartesian Product Case: Parallel Implementations

From now on we assume that X is a Cartesian product $X = \prod_{i=1}^m X_i$, where each set X_i is of dimension n_i and $\sum_{i=1}^m n_i = n$. Any vector $x \in X$ is accordingly decomposed as $x = (x_1, \ldots, x_m)$, with $x_i \in X_i$. We still assume that X is nonempty, closed, and convex, and these properties are then implied for each one of the sets X_i. The Cartesian product assumption holds for several important problems such as the solution of systems of nonlinear equations in n variables, the traffic assignment problem, Nash games (see Subsection 3.5.1), as well for many important economic equilibrium problems ([Pan85], [Ahn79], [Nag87]). As in constrained optimization (Subsection 3.3.4), this assumption provides the possibility for parallel algorithms, as will be shown next.

A key observation for the product set case is that a variational inequality decomposes into m coupled variational inequalities of smaller dimensions.

Proposition 5.7. (*Decomposition Lemma*) A vector $x^* \in X$ solves the variational inequality VI(X, f) if and only if

$$(x_i - x_i^*)' f_i(x^*) \geq 0, \qquad \forall x_i \in X_i, \ \forall i. \tag{5.11}$$

Proof. If Eq. (5.11) is satisfied for each i, we add these inequalities to conclude that $(x - x^*)' f(x^*) \geq 0$. Conversely, suppose that $x^* \in X$ solves the problem VI(X, f). Choose some vector x such that $x_j = x_j^*$ for all $j \neq i$ and $x_i \in X_i$. Because of the Cartesian product assumption, we have $x \in X$ and using the inequality $(x - x^*)' f(x^*) \geq 0$, we see that Eq. (5.11) holds. **Q.E.D.**

Consider now a linearized algorithm determined by a collection $\{A(x) \mid x \in X\}$ of scaling matrices. Such an algorithm is not easily parallelizable in general. For example, in the case of unconstrained optimization [see Eq. (5.9)], a system of linear equations has to be solved at each step and this task cannot be easily decomposed into a set

of independent subtasks. Rather, a parallel algorithm for systems of linear equations is needed at each step and this requires a greater degree of coordination between processors. On the other hand, if the matrix $A(x)$ is block–diagonal for each x, the ith block $A_i(x)$ being of dimension $n_i \times n_i$, the problem decouples naturally, as we proceed to show. Let $T : X \mapsto X$ be the mapping describing one iteration of the linearized algorithm, that is, $T(x)$ satisfies Eq. (5.8). Under the assumption that $A(x)$ is block–diagonal, the ith block–component of the function $f(x) + A(x)\big(T(x) - x\big)$ is equal to $f_i(x) + A_i(x)\big(T_i(x) - x_i\big)$, and using the Decomposition Lemma, we conclude that $T_i(x)$ satisfies

$$\big(y_i - T_i(x)\big)'\Big(f_i(x) + A_i(x)\big(T_i(x) - x_i\big)\Big) \geq 0, \qquad \forall y_i \in X_i. \tag{5.12}$$

This shows that each component of $T_i(x)$ can be found by solving a variational inequality of smaller dimension, and this can be done independently for each i. In particular, each one of these smaller variational inequalities can be solved by a different processor.

We return to the convergence analysis of linearized algorithms. Let us fix a choice of the scaling matrices $A(x)$, assumed to be block–diagonal, as discussed earlier. We assume that each diagonal block $A_i(x)$ of $A(x)$ is positive definite for each x, which guarantees that each subproblem (5.12) has a unique solution (Prop. 5.5) and the linearized algorithm is well–defined. The following result establishes the convergence of the iteration $x := T(x)$ as well as of the associated Gauss–Seidel algorithm, under the standing assumption that X is a Cartesian product and the scaling matrices $A(x)$ are block–diagonal.

Proposition 5.8. (*Linearized Algorithm Convergence in the Product Case*) Suppose that the problem VI(X, f) has a solution x^*. Suppose that there exist symmetric positive definite matrices G_i and some $\delta > 0$ such that $A_i(x) - \delta G_i$ is nonnegative definite for every i and $x \in X$, and that there exists some $\alpha \in [0, 1)$ such that

$$\Big\|G_i^{-1}\Big(f_i(x) - f_i(y) - A_i(y)(x_i - y_i)\Big)\Big\|_i \leq \delta\alpha \max_j \|x_j - y_j\|_j, \qquad \forall x, y \in X, \tag{5.13}$$

where $\|x_i\|_i = (x_i' G_i x_i)^{1/2}$. Then, the iteration mapping T of the linearized algorithm determined by $\{A(x) \mid x \in X\}$ has the property

$$\|T_i(x) - x_i^*\|_i \leq \alpha \max_j \|x_j - x_j^*\|_j, \qquad \forall x \in X,\ i = 1, \ldots, m. \tag{5.14}$$

In particular, x^* is the unique solution of VI(X, f) and the linearized algorithm $x(t+1) = T\big(x(t)\big)$, as well as the Gauss–Seidel algorithm based on T, converge to x^* geometrically.

Proof. Fix some $x \in X$. Since $x_i^* \in X_i$, inequality (5.12) yields

$$\big(x_i^* - T_i(x)\big)'\Big(f_i(x) + A_i(x)\big(T_i(x) - x_i\big)\Big) \geq 0. \tag{5.15}$$

We also have

$$\big(T_i(x) - x_i^*\big)' f_i(x^*) \geq 0, \tag{5.16}$$

because x^* solves VI(X, f) and because of the Decomposition Lemma. We add inequalities (5.15) and (5.16) and rearrange terms to obtain

$$\big(T_i(x)-x_i^*\big)' A_i(x)\big(T_i(x)-x_i^*\big) \leq \big(T_i(x)-x_i^*\big)'\Big(f_i(x^*)-f_i(x)-A_i(x)(x_i^*-x_i)\Big). \tag{5.17}$$

The left–hand side of inequality (5.17) is bounded below by $\delta\|T_i(x) - x_i^*\|_i^2$ because of the nonnegative definiteness of $A_i(x) - \delta G_i$. Also, the right–hand side of inequality (5.17) is equal to

$$\big(T_i(x) - x_i^*\big)' G_i\Big[G_i^{-1}\Big(f_i(x^*) - f_i(x) - A_i(x)(x_i^* - x_i)\Big)\Big]. \tag{5.18}$$

From the Schwartz inequality [Prop. A.28(e) in Appendix A] and inequality (5.13), the expression (5.18) is bounded above by $\|T_i(x)-x_i^*\|_i \cdot \delta\alpha \max_j \|x_j - x_j^*\|_j$. We have thus shown that

$$\delta\|T_i(x) - x_i^*\|_i^2 \leq \|T_i(x) - x_i^*\|_i \cdot \delta\alpha \max_j \|x_j - x_j^*\|_j,$$

from which inequality (5.14) follows. In particular, T is a pseudocontraction and x^* is the unique fixed point of T. The rest of the result follows from the convergence theorem for pseudocontracting iterations and their Gauss–Seidel versions (Props. 1.2 and 1.5 of Section 3.1). **Q.E.D.**

Notice that this proof remains valid under the assumption that VI(X, f) has a solution x^* and that inequality (5.13) holds when $y = x^*$. On the other hand, given that x^* is unknown, this weaker version is usually not any easier to verify.

In the special case where $A(x)$ is symmetric positive definite and independent of x we obtain the projection algorithm and the previous proposition can be strengthened a little. In particular, we do not need to assume the existence of a solution x^*.

Proposition 5.9. Let $\gamma > 0$, let G_i, $i = 1, \ldots, m$, be symmetric positive definite matrices, and let $\|\cdot\|_i$ be the norm $\|x\|_i = (x'G_i x)^{1/2}$. Suppose that

$$\Big\|\gamma G_i^{-1}\big(f_i(x) - f_i(y)\big) - (x_i - y_i)\Big\|_i \leq \alpha \max_j \|x_j - y_j\|_j, \qquad \forall x, y \in X, \tag{5.19}$$

where $\alpha \in [0, 1)$. Then, the problem VI(X, f) has a unique solution x^*. Let G be a block–diagonal matrix whose ith diagonal block is equal to G_i. Then, the sequence $\{x(t)\}$ generated by the projection algorithm $x(t+1) = \big[x(t)-\gamma G^{-1} f\big(x(t)\big)\big]^+$ converges to x^* geometrically. The same is true concerning the corresponding Gauss–Seidel algorithm.

Proof. The condition (5.19) states that the mapping $R_G(x) = x - \gamma G^{-1} f(x)$ is a contraction with respect to the block–maximum norm $\|x\| = \max_i \|x\|_i$. It follows that the mapping $T_G(x) = [x - \gamma G^{-1} f(x)]_G^+$ is also a contraction and has a unique fixed point. Existence and uniqueness of a solution follow from our general existence and uniqueness results (Prop. 5.3). Convergence of the projection algorithm follows from the convergence theorem for contracting iterations (Prop. 1.1) and their Gauss–Seidel variants (Prop. 1.4). **Q.E.D.**

Sufficient conditions for condition (5.19) to hold are provided by Props. 1.10 and 1.11 of Subsection 3.1.3.

3.5.6 Nonlinear Algorithms

We still assume that X is a Cartesian product $X = \prod_{i=1}^m X_i$. According to the Decomposition Lemma, the variational inequality VI(X, f) is equivalent to a system of m variational inequalities that must be solved simultaneously. A nonlinear algorithm proceeds by solving for each i the ith variational inequality

$$(x_i - x_i^*)' f_i(x^*) \geq 0, \qquad \forall x_i \in X_i,$$

with respect to the ith block–component of x^*, while keeping the other block–components fixed. To be more precise, fix some $x^0 \in X$. Starting from x^0, the new value of the ith block–component, produced by an iteration of a nonlinear algorithm, is equal to some $Q_i(x^0)$ satisfying

$$\big(x_i - Q_i(x^0)\big)' f_i\Big(x_1^0, \ldots, x_{i-1}^0, Q_i(x^0), x_{i+1}^0, \ldots, x_m^0\Big) \geq 0, \qquad \forall x_i \in X_i. \tag{5.20}$$

Notice that (5.20) is itself a variational inequality. It is the problem VI(X_i, g), where $g(x_i) = f\big(x_1^0, \ldots, x_{i-1}^0, x_i, x_{i+1}^0, \ldots, x_m^0\big)$. Therefore, $Q_i(x^0)$ can be found using any one of the algorithms presented earlier. Furthermore, (5.20) should be easier to solve than the original problem (5.1) because the sets X_i are of smaller dimension.

It is assumed in the sequel that for every $x^0 \in X$, there exists some $Q_i(x^0)$ satisfying (5.20). Since (5.20) is itself a variational inequality in the unknown $Q_i(x^0)$, sufficient conditions for the existence of a solution are provided by our general existence results (Props. 5.2–5.3). In case that there are several solutions, we assume that $Q_i(x^0)$ has been arbitrarily defined to be equal to one of them. Then, the nonlinear algorithm is well defined.

We let $Q(x^0) = \big(Q_1(x^0), \ldots, Q_m(x^0)\big)$ and this determines a mapping $Q : X \mapsto X$. Accordingly, the nonlinear Jacobi algorithm is defined by the iteration

$$x(t+1) = Q\big(x(t)\big).$$

With the nonlinear Jacobi algorithm, all block–components $x_i(t+1)$ are simultaneously computed on the basis of $x(t)$. Alternatively, with a nonlinear Gauss–Seidel algorithm, the block–components of $x(t+1)$ are computed in succession. In the terminology of Section 3.1, the latter is the Gauss–Seidel algorithm based on the mapping Q.

We briefly interpret the above defined nonlinear algorithm in the context of the specific examples discussed in Subsection 3.5.1.

(a) (*Systems of equations*) Here $X = \Re^n$ and the objective is to find a solution of $f(x) = 0$. For this example, $Q_i(x)$ is the value of x_i obtained by solving the ith equation $f_i(x) = 0$, while keeping the other components of x constant.

(b) (*Optimization*) Here X is a closed convex set and $f(x) = \nabla F(x)$, where F is a continuously differentiable convex cost function. Given a current vector x^0, the new value $\tilde{x}_i = Q_i(x^0)$ of the ith block–component is obtained by solving the variational inequality

$$(y_i - \tilde{x}_i)'\nabla_i F\left(x_1^0, \ldots, x_{i-1}^0, \tilde{x}_i, x_{i+1}^0, \ldots, x_m^0\right) \geq 0, \qquad \forall y_i \in X_i.$$

This is equivalent to minimizing F with respect to the ith block–component. In particular, the nonlinear methods of this section contain as special cases the nonlinear methods of Sections 3.2 and 3.3.

(c) (*Game theory*) Consider a Nash game and let F_i be the cost function of the ith player. Assuming that each F_i is convex in the variable x_i (the strategy of the ith player), it is easily seen that at each iteration of the nonlinear algorithm, the strategy of each player is optimized while holding the strategies of the other players constant.

In Sections 3.2 and 3.3, we presented certain results on the convergence of the nonlinear Gauss–Seidel algorithm for optimization problems (Props. 2.5 and 3.9). The proof of these results was based on the descent property: the value of the cost function was nonincreasing in the course of the algorithm. For more general variational inequalities, the descent approach is inapplicable. We therefore rely instead on the theory of contraction mappings of Section 3.1 together with the observation that the nonlinear Jacobi algorithm is identical to the component solution method of Section 3.1.2, which is proved next.

Proposition 5.10. (*Nonlinear Algorithms are Component Solution Methods*) Let $T : X \mapsto X$ be the mapping corresponding to one iteration of a linearized algorithm determined by a collection of scaling matrices $\{A(x) \mid x \in X\}$ assumed to be block–diagonal and positive definite. Then, an iteration $x := Q(x)$ of the nonlinear Jacobi algorithm coincides with an iteration of the component solution method for solving the fixed point problem $x = T(x)$.

Proof. We only need to show that $Q_i(x)$ solves the ith equation of the system $x = T(x)$, that is, we need to show that the equality

$$Q_i(x) = T_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big) \tag{5.21}$$

holds for every i and every $x \in X$. Let us fix some $x \in X$. From Eq. (5.20) we have

$$\big(z_i - Q_i(x)\big)' f_i\Big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\Big) \geq 0, \qquad \forall z_i \in X_i.$$

This can be rewritten as

$$\big(z_i - Q_i(x)\big)' \Big(f_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big) + A_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big)\big(Q_i(x) - Q_i(x)\big)\Big) \geq 0, \qquad \forall z_i \in X_i,$$

which shows that $Q_i(x)$ solves (for the unknown y_i) the variational inequality

$$(z_i - y_i)' \Big(f_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big) + A_i\big(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m\big)\big(y_i - Q_i(x)\big)\Big) \geq 0, \qquad \forall z \in X_i. \tag{5.22}$$

On the other hand this is exactly the variational inequality solved by the linearized algorithm [see Eq. (5.12)], if the current vector is $(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m)$. Furthermore, a solution is unique, because $A(x)$ has been assumed positive definite (Prop. 5.5). Therefore, by the definition of T, we obtain $Q_i(x) = T_i(x_1, \ldots, x_{i-1}, Q_i(x), x_{i+1}, \ldots, x_m)$, as desired. **Q.E.D.**

Convergence of nonlinear algorithms can be now demonstrated using our general results on the convergence of component solution methods (Subsection 3.1.2). In particular, if the mapping T corresponding to a linearized algorithm is a contraction or a pseudocontraction, with respect to a block–maximum norm, the same property holds for the mapping Q describing the nonlinear Jacobi algorithm (Props. 1.7 and 1.9 in Subsection 3.1.2). Furthermore, in the case where T is a contraction, it has been shown in Section 3.1 (Prop. 1.6) that the component solutions $Q_i(x)$ are uniquely defined for every $x \in X$. We now state two results that follow directly from Prop. 5.10 and the preceding discussion.

Proposition 5.11. Let $T : X \mapsto X$ be the mapping corresponding to one iteration of a linearized algorithm determined by a collection of scaling matrices $\{A(x) \mid x \in X\}$ assumed to be block–diagonal and positive definite. Suppose also that T is a pseudocontraction, with respect to a block–maximum norm. Let x^* be the fixed point of T. If the nonlinear Jacobi algorithm $x(t+1) = Q\big(x(t)\big)$ is well defined [meaning that the variational inequality (5.20) always has a solution] then the sequence $\{x(t)\}$ converges to x^* geometrically, and the same is true for the nonlinear Gauss–Seidel algorithm.

It should be recalled here that Prop. 5.8 provides sufficient conditions for a linearized algorithm to be a pseudocontraction with respect to a block–maximum norm.

Proposition 5.12. Let $T : X \mapsto X$ be the mapping corresponding to one iteration of a linearized algorithm determined by a collection of scaling matrices $\{A(x) \mid x \in X\}$ assumed to be block–diagonal and positive definite. Suppose also that T is a contraction with respect to a block–maximum norm. Then the problem VI(X, f) has a unique solution x^*, the nonlinear Jacobi and Gauss–Seidel algorithms are well defined, and they converge to x^* geometrically.

An example of a linearized algorithm that is a block–maximum norm contraction is the projection algorithm under the assumption (5.19) of Prop. 5.9. Conditions for the latter assumption to hold have been presented in Subsection 3.1.3.

3.5.7 Decomposition Methods for Variational Inequalities

Some of the decomposition techniques developed in Section 3.4 for convex constrained optimization problems can be extended to more general variational inequality problems. As an example, consider the following natural extension of the separable problem of Subsection 3.4.4 (Example 3.4), where we want to find a vector $x^* = (x_1^*, \ldots, x_m^*)$ in a product set $P_1 \times \cdots \times P_m$ which satisfies the linear coupling constraints

$$e_j' x^* = s_j, \qquad j = 1, \ldots, r,$$

and solves the separable variational inequality

$$\sum_{i=1}^{m} f_i(x_i^*)'(x_i - x_i^*) \geq 0, \qquad \forall\, x \in P_1 \times \cdots \times P_m, \text{ with } e_j' x = s_j,\ j = 1, \ldots, r. \quad (5.23)$$

Here P_i is a polyhedral subset of $\Re^{n_i}$, $e_j \in \Re^n$ are given vectors, and $s_j \in \Re$ are given scalars, where $n = n_1 + \cdots + n_m$. If $f_i(x_i) = \nabla F_i(x_i)$, where $F_i : \Re^{n_i} \mapsto \Re$ is a convex function for each i, we obtain a separable optimization problem.

Let e_{ji} be the subvector of e_j that corresponds to x_i, let

$$I(j) = \{i \mid e_{ji} \neq 0\}, \qquad j = 1, \ldots, r,$$

and let m_j be the number of elements of $I(j)$. In the typical iteration of the natural alternating direction method of multipliers (cf. Subsection 3.4.4), we obtain for $i = 1, \ldots, m$, a solution $x_i(t+1) \in P_i$ of the variational inequality

$$\Big[f_i\big(x_i(t+1)\big) + \sum_{\{j \mid i \in I(j)\}} e_{ji}\big[\lambda_j(t) + c\big(e_{ji}'\big(x_i(t+1) - x_i(t)\big) + w_j(t)\big)\big]\Big]'\big(x_i - x_i(t+1)\big) \geq 0,$$

$$\forall\, x_i \in P_i, \quad (5.24)$$

where

$$w_j(t) = \frac{1}{m_j}\big(e_j' x(t) - s_j\big), \qquad j = 1, \ldots, r, \quad (5.25)$$

and we update $\lambda_j(t)$ according to

$$\lambda_j(t+1) = \lambda_j(t) + cw_j(t+1), \qquad j = 1, \ldots, r. \tag{5.26}$$

Here c is a positive scalar and the initial vectors $x_i(0)$ and $\lambda_j(0)$ are arbitrary. Note that this is a highly parallelizable method.

To establish the validity of the preceding method we consider (cf. Subsection 3.4.4) the variational inequality problem of finding $x^* \in \Re^n$ such that $x^* \in C_1$, $Ax^* \in C_2$ and

$$f(x^*)'(x - x^*) \geq 0, \qquad \forall\, x \in C_1 \cap \{\xi \mid A\xi \in C_2\}. \tag{5.27}$$

Here A is an $m \times n$ matrix, and $C_1 \subset \Re^n$ and $C_2 \subset \Re^m$ are nonempty polyhedral sets.

There is a natural extension of the alternating direction method of multipliers of Subsection 3.4.4 for the above problem. In the typical iteration of this method, we obtain $x(t+1)$ that solves the variational inequality

$$\Big[f\big(x(t+1)\big) + A'\big[p(t) + c\big(Ax(t+1) - z(t)\big)\big]\Big]'\big(x - x(t+1)\big) \geq 0, \qquad \forall\, x \in C_1, \tag{5.28}$$

and we update $z(t)$ and $p(t)$ by

$$z(t+1) = \arg\min_{z \in C_2} \left\{-p(t)'z + \frac{c}{2}\|Ax(t+1) - z\|_2^2\right\}, \tag{5.29}$$

$$p(t+1) = p(t) + c\big(Ax(t+1) - z(t+1)\big). \tag{5.30}$$

The parameter c is assumed positive, and the initial vectors $p(0)$ and $z(0)$ are arbitrary.

As in Subsection 3.4.4, the method (5.24)–(5.26) is obtained as a special case of the method (5.28)–(5.30) with the identifications

$$f(x) = \sum_{i=1}^{m} f_i(x_i), \qquad C_1 = P_1 \times P_2 \times \cdots \times P_m,$$

$$C_2 = \left\{ z \,\middle|\, \sum_{i \in I(j)} z_{ji} = s_j,\ j = 1, \ldots, r \right\},$$

and with A being the matrix that maps x into the vector having coordinates $e'_{ji}x_i$, $j = 1, \ldots, r$, $i \in I(j)$.

We will make the following assumption (cf. Assumption 4.1 in Subsection 3.4.4):

Assumption 5.1. The optimal solution set X^* of problem (5.27) is nonempty, and the function f is Lipschitz continuous and monotone, that is, for some K

$$\|f(x) - f(y)\|_2 \leq K\|x - y\|_2, \qquad (x - y)'\big(f(x) - f(y)\big) \geq 0, \qquad \forall\, x, y \in C_1. \tag{5.31}$$

Furthermore, either C_1 is bounded, or else the matrix $A'A$ is invertible.

It is possible to show that under Assumption 5.1, a solution $x(t+1)$ of the variational inequality of Eq. (5.28) exists. Indeed if C_1 is compact, existence follows from Prop. 5.2. If $A'A$ is invertible the variational inequality of Eq. (5.28) involves a strongly monotone map because f is monotone and $A'A$ is positive definite, so existence (as well as uniqueness) follows from Prop. 5.4. The following convergence result parallels Prop. 4.2 in Subsection 3.4.4.

Proposition 5.13. Let Assumption 5.1 hold. A sequence $\{x(t), z(t), p(t)\}$ generated by the algorithm (5.28)–(5.30) is bounded, and every limit point of $\{x(t)\}$ is a solution of the original variational inequality (5.27).

Proof. The proof closely resembles the proof of Prop. 4.2 in Subsection 3.4.4. Let x^* be a solution of the variational inequality (5.27), let $z^* = Ax^*$, and let p^* be a Lagrange multiplier associated with the equality constraint $z = Ax$ in the problem

$$\begin{aligned} &\text{minimize} \quad f(x^*)'x \\ &\text{subject to } x \in C_1,\ z \in C_2,\ z = Ax. \end{aligned} \tag{5.32}$$

Then (x^*, z^*) is an optimal solution of the above optimization problem. By using the multiplier update formula (5.30) we can write the condition (5.28) as

$$f\big(x(t+1)\big)'\big(x(t+1)-x\big)+\big[p(t+1)+c\big(z(t+1)-z(t)\big)\big]'A\big(x(t+1)-x\big) \le 0, \qquad \forall\, x \in C_1, \tag{5.33}$$

while by using the necessary optimality condition of Prop. 3.1 of Section 3.3 we obtain from Eq. (5.29)

$$p(t+1)'\big(z - z(t+1)\big) \le 0, \qquad \forall\, z \in C_2. \tag{5.34}$$

By applying Eq. (5.33) with $x = x^*$ we have

$$f\big(x(t+1)\big)'\big(x(t+1) - x^*\big) + \big[p(t+1) + c\big(z(t+1) - z(t)\big)\big]'A\big(x(t+1) - x^*\big) \le 0,$$

and by applying Eq. (5.34) with $z = z^*$ we have

$$p(t+1)'\big(z^* - z(t+1)\big) \le 0.$$

By adding these two relations, and using also the fact $Ax^* = z^*$, we obtain

$$\begin{aligned} f\big(x(t+1)\big)'\big(x(t+1) - x^*\big) + p(t+1)'\big(Ax(t+1) - z(t+1)\big) \\ + c\big(z(t+1) - z(t)\big)'A\big(x(t+1) - x^*\big) \le 0. \end{aligned} \tag{5.35}$$

Since (x^*, z^*) is an optimal solution and p^* is a Lagrange multiplier for problem (5.32), we have

$$0 \leq f(x^*)'(x(t+1) - x^*) + p^{*\prime}(Ax(t+1) - z(t+1)), \qquad \forall\, t. \tag{5.36}$$

By adding Eqs. (5.35) and (5.36) we obtain

$$\big[f(x(t+1)) - f(x^*)\big]'(x(t+1) - x^*) + (p(t+1) - p^*)'(Ax(t+1) - z(t+1)) + c(z(t+1) - z(t))'A(x(t+1) - x^*) \leq 0. \tag{5.37}$$

Using the monotonicity of f, we see that the first term on the left–hand side of Eq. (5.37) is nonnegative, so we obtain

$$(p(t+1) - p^*)'(Ax(t+1) - z(t+1)) + c(z(t+1) - z(t))'A(x(t+1) - x^*) \leq 0. \tag{5.38}$$

We now denote for all t

$$\bar{x}(t) = x(t) - x^*, \qquad \bar{z}(t) = z(t) - z^*, \qquad \bar{p}(t) = p(t) - p^*,$$

and we use the calculation given in the proof of Prop. 4.2 to obtain

$$\|\bar{p}(t+1) - \bar{p}(t)\|_2^2 + c^2\|\bar{z}(t+1) - \bar{z}(t)\|_2^2 \leq \big(\|\bar{p}(t)\|_2^2 + c^2\|\bar{z}(t)\|_2^2\big) - \big(\|\bar{p}(t+1)\|_2^2 + c^2\|\bar{z}(t+1)\|_2^2\big). \tag{5.39}$$

It follows that $\{z(t)\}$ and $\{p(t)\}$ are bounded and

$$\bar{p}(t+1) - \bar{p}(t) \to 0, \qquad \bar{z}(t+1) - \bar{z}(t) \to 0. \tag{5.38}$$

Since $Ax(t) - z(t) = \big(\bar{p}(t) - \bar{p}(t-1)\big)/c \to 0$ and $\{z(t)\}$ is bounded, we obtain by using Assumption 5.1 that $\{x(t)\}$ is bounded. Furthermore, for every limit point $(\tilde{x}, \tilde{z})$ of $\big\{\big(x(t), z(t)\big)\big\}$ we have $A\tilde{x} = \tilde{z}$. Let $(\tilde{x}, \tilde{z}, \tilde{p})$ be a limit point of the sequence $\big\{\big(x(t), z(t), p(t)\big)\big\}$. Then by taking the limit in Eq. (5.33) we obtain

$$f(\tilde{x})'(\tilde{x} - x) + \tilde{p}'A(\tilde{x} - x) \leq 0, \qquad \forall\, x \in C_1, \tag{5.39}$$

while by taking the limit in Eq. (5.34) we obtain

$$\tilde{p}'(z - \tilde{z}) \leq 0, \qquad \forall\, z \in C_2. \tag{5.40}$$

By adding Eqs. (5.39) and (5.40), and by using the fact $A\tilde{x} = \tilde{z}$ we obtain

$$f(\tilde{x})'(x - \tilde{x}) + \tilde{p}'(Ax - z) \geq 0, \qquad \forall\, x \in C_1,\ z \in C_2. \tag{5.41}$$

By using the Lagrange Multiplier Theorem of Appendix C we see that Eq. (5.41) implies that $\tilde{x}$ and $\tilde{z}$ are an optimal solution of the problem

$$\begin{aligned} &\text{minimize} \quad f(\tilde{x})'x \\ &\text{subject to} \quad x \in C_1,\ z \in C_2,\ z = Ax, \end{aligned}$$

or equivalently that $\tilde{x}$ is a solution of the original variational inequality (5.27). **Q.E.D.**

We note that it is possible to show convergence of the sequences $\{z(t)\}$ and $\{p(t)\}$ as in the proof of Prop. 4.2. Exercise 5.3 provides an alternating direction method of multipliers for a generalized version of the variational inequality (5.27).

EXERCISES

5.1. (**The Extragradient Method [Kor76].**) This exercise provides a modification of the projection algorithm that converges appropriately to a solution of VI(X, f), assuming that the monotonicity condition

$$(x - y)'\big(f(x) - f(y)\big) \geq 0, \qquad \forall x, y \in X,$$

holds, and that there exists a constant A such that

$$\|f(x) - f(y)\|_2 \leq A\|x - y\|_2, \qquad \forall x, y \in X.$$

(As shown in Fig. 3.5.3, the projection algorithm need not converge, for any value of the stepsize, under the preceding conditions; a strong monotonicity condition is needed.)

Consider the modified projection method

$$x(t+1) = \Big[x(t) - \gamma f\big(\bar{x}(t)\big)\Big]^+,$$

where $\bar{x}(t)$ is given by

$$\bar{x}(t) = \Big[x(t) - \gamma f\big(x(t)\big)\Big]^+,$$

and γ is a positive scalar. [Thus, the method, at each iteration, uses the value of f at $\bar{x}(t)$ rather than the one at $x(t)$.]

(a) Show that for any solution x^* of VI(X, f) we have

$$\|x(t+1) - x^*\|_2^2 \leq \|x(t) - x^*\|_2^2 - (1 - \gamma^2 A^2)\|x(t) - \bar{x}(t)\|_2^2,$$

and conclude that for $\gamma \in (0, 1/A)$, the method converges to some solution of VI(X, f), if at least one such solution exists. *Hint:* The monotonicity condition and the fact that x^* is a solution imply that

$$\begin{aligned} 0 \le \left(f\left(\bar{x}(t)\right) - f(x^*)\right)'\left(\bar{x}(t) - x^*\right) &= f\left(\bar{x}(t)\right)'\left(\bar{x}(t) - x^*\right) - f(x^*)'\left(\bar{x}(t) - x^*\right) \\ &\le f\left(\bar{x}(t)\right)'\left(\bar{x}(t) - x^*\right) \\ &= f\left(\bar{x}(t)\right)'\left(\bar{x}(t) - x(t+1)\right) \\ &\quad + f\left(\bar{x}(t)\right)'\left(x(t+1) - x^*\right), \end{aligned}$$

and, finally,

$$f\left(\bar{x}(t)\right)'\left(x^* - x(t+1)\right) \le f\left(\bar{x}(t)\right)'\left(\bar{x}(t) - x(t+1)\right). \tag{5.42}$$

Since $x(t+1)$ is the projection of $x(t) - \gamma f\left(\bar{x}(t)\right)$ on X and $x^* \in X$ we have, also using Eq. (5.42),

$$\begin{aligned} \|x(t+1) - x^*\|_2^2 &\le \|x(t) - \gamma f\left(\bar{x}(t)\right) - x^*\|_2^2 - \|x(t) - \gamma f\left(\bar{x}(t)\right) - x(t+1)\|_2^2 \\ &= \|x(t) - x^*\|_2^2 - \|x(t) - x(t+1)\|_2^2 + 2\gamma f\left(\bar{x}(t)\right)'\left(x^* - x(t+1)\right) \\ &\le \|x(t) - x^*\|_2^2 - \|x(t) - \bar{x}(t)\|_2^2 - \|\bar{x}(t) - x(t+1)\|_2^2 \\ &\quad - 2\left(x(t) - \bar{x}(t)\right)'\left(\bar{x}(t) - x(t+1)\right) + 2\gamma f\left(\bar{x}(t)\right)'\left(\bar{x}(t) - x(t+1)\right) \end{aligned}$$

and, finally,

$$\begin{aligned} \|x(t+1) - x^*\|_2^2 \le \|x(t) - x^*\|_2^2 - \|x(t) - \bar{x}(t)\|_2^2 - \|\bar{x}(t) - x(t+1)\|_2^2 \\ + 2\left(x(t+1) - \bar{x}(t)\right)'\left(x(t) - \gamma f\left(\bar{x}(t)\right) - \bar{x}(t)\right). \end{aligned} \tag{5.43}$$

Since $\bar{x}(t)$ is the projection of $x(t) - \gamma f\left(x(t)\right)$ on X and $x(t+1) \in X$ we have, also using the Lipschitz continuity of f,

$$\begin{aligned} &\left(x(t+1) - \bar{x}(t)\right)'\left(x(t) - \gamma f\left(\bar{x}(t)\right) - \bar{x}(t)\right) \\ &\quad = \left(x(t+1) - \bar{x}(t)\right)'\left(x(t) - \gamma f\left(x(t)\right) - \bar{x}(t)\right) \\ &\qquad + \gamma\left(x(t+1) - \bar{x}(t)\right)'\left(f\left(x(t)\right) - f\left(\bar{x}(t)\right)\right) \qquad (5.44) \\ &\quad \le \gamma\left(x(t+1) - \bar{x}(t)\right)'\left(f\left(x(t)\right) - f\left(\bar{x}(t)\right)\right) \\ &\quad \le \gamma A\|x(t+1) - \bar{x}(t)\|_2 \cdot \|x(t) - \bar{x}(t)\|_2. \end{aligned}$$

Using inequality (5.44) to strengthen inequality (5.43), we obtain

$$\begin{aligned}\|x(t+1)-x^*\|_2^2 &\le \|x(t)-x^*\|_2^2 - \|x(t)-\bar{x}(t)\|_2^2 - \|\bar{x}(t)-x(t+1)\|_2^2 \\ &\quad + 2\gamma A\|x(t)-\bar{x}(t)\|_2 \cdot \|\bar{x}(t)-x(t+1)\|_2 \\ &= \|x(t)-x^*\|_2^2 - (1-\gamma^2A^2)\|x(t)-\bar{x}(t)\|_2^2 \\ &\quad - \Big(\gamma A\|x(t)-\bar{x}(t)\|_2 - \|\bar{x}(t)-x(t+1)\|_2\Big)^2 \\ &\le \|x(t)-x^*\|^2 - (1-\gamma^2A^2)\|x(t)-\bar{x}(t)\|_2^2.\end{aligned}$$

(b) Demonstrate the convergence of the method when applied to the example of Fig. 3.5.3.

5.2. **(Separable Constrained Optimization Problems.)** Consider the separable, convex constrained optimization problem

$$\begin{aligned}&\text{minimize} && \sum_{i=1}^m F_i(x_i) \\ &\text{subject to} && a_j'x \le t_j, \quad j=1,\dots,r, \\ &&& x_i \in P_i, \quad i=1,\dots,m,\end{aligned}$$

similar to the one considered in Subsection 3.4.4, where $F_i : \Re^{n_i} \mapsto \Re$ is a convex differentiable function.

(a) Verify that this problem is equivalent to the variational inequality problem of finding $x_i^* \in P_i$ and $u_j^* \ge 0$ such that

$$\sum_{i=1}^m \Big(\nabla F_i(x_i^*) + \sum_{j=1}^r u_j^* a_{ji}\Big)'(x_i - x_i^*) + \sum_{j=1}^r (t_j - a_j'x^*)(u_j - u_j^*) \ge 0,$$

$$\forall x_i \in P_i,\ u_j \ge 0.$$

(b) Discuss the use of the extragradient method of Exercise 5.1 for solving the problem.

5.3. Consider the problem of finding $x^* \in \Re^n$ such that $x^* \in C_1$, $Ax^* \in C_2$ and

$$f(x^*)'(x-x^*) + G(Ax) - G(Ax^*) \ge 0, \qquad \forall\, x \in C_1 \cap \{\xi \mid A\xi \in C_2\}.$$

Here $G : \Re^m \mapsto \Re$ is a convex function, A is an $m \times n$ matrix, and $C_1 \subset \Re^n$, $C_2 \subset \Re^m$ are nonempty polyhedral sets. Let Assumption 5.1 hold, and consider the algorithm (5.28)–(5.30) except that Eq. (5.29) is replaced by

$$z(t+1) = \arg\min_{z \in C_2} \Big\{G(z) - p(t)'z + \frac{c}{2}\|Ax(t+1) - z\|_2^2\Big\}.$$

Show that a sequence $\{x(t), z(t), p(t)\}$ generated by the algorithm is bounded, and every limit point of $\{x(t)\}$ is a solution of the above inequality.

5.4. Develop an alternating direction method of multipliers for the variational inequality

$$\sum_{i=1}^{m} f_i(x^*)'(x - x^*) \geq 0, \qquad \forall\, x \in \cap_{i=1}^{m} P_i,$$

based on the algorithm (5.24)–(5.26).

NOTES AND SOURCES

3.1. Most of the material in this section is classical; see, e.g., [OrR70] and [Lue69]. Some of the sufficient conditions for contraction mappings (Props. 1.10–1.13) are inspired from related results of [PaC82a].

3.2. Unconstrained optimization algorithms are discussed in many textbooks, e.g., [Avr76], [Ber82a], [DeS83], [GMW81], [KoO68], [Lue84], [OrR70], [Pol71], [Pol87], [Zan69], and [Zou76]. A special type of approximate Newton method, called the *truncated Newton method* and based on the conjugate gradient method is analyzed in [DES82]. An example showing that some form of strict convexity assumption is needed to guarantee convergence of the nonlinear Gauss–Seidel method is given in [Pow73]. Parallel conjugate direction methods for unconstrained optimization are given in [ChM70], [Sut83], [Han86], and [BSS88]. For an overview of parallel optimization methods, see [LoR88] and [MeZ88].

3.3. Many of the above mentioned books on unconstrained optimization also contain discussions of constrained optimization. The gradient projection method was proposed independently in [Gol64] and [LeP65]. For some further developments and analysis see [Ber76a] and [Dun81]. Variants of the gradient projection method that aim at acceleration of its convergence rate while maintaining its simplicity are given in [Ber82b], [BeG83], [Bon83], [GaB84], and [CaM87]. The results on the well–posedness of the nonlinear Jacobi and Gauss–Seidel methods and their global convergence to a minimizing point (Prop. 3.10) appear to be new.

We have not discussed in this section or elsewhere in this book the simplex method for linear programming because this method contains some operations that are difficult to parallelize. In particular each iteration of the simplex method consists of two steps (see e.g. [Dan63], [Chv83], [Lue84]): a) choosing a new basic variable and b) performing a pivot operation. The first of these steps lends itself to parallelization but the second generally does not. While much depends on the parallelization approach used and the structure of the problem solved, parallel versions of the simplex method have produced thus far only limited speedup in computational experiments. For a representative computational study that discusses various approaches see [Pet88].

3.4. The uses of duality in large–scale optimization are discussed in numerous sources, including the books [Las70], [MMT70], [Sin77], and [FBB80], the edited volumes [Wis71] and [HoM76], and the journal special issue [IEE78].

The dual quadratic programming algorithm of Subsection 3.4.1 is given in [Hil57]; for related work see [Lue69], [Cry71], [Man77], [MaD86], and [MaD87]. The given implementation for sparse problems has been used extensively (with some variations) in image processing applications [HLL78], [HeL78], [Cen81], and [CeH87].

Dual methods have been used extensively for the solution of various types of separable problems. An early reference is [Eve63]; see also [Las70], [Wis71], [Geo72], [Las73], [BLT76], and [Lue84]. Nondifferentiable optimization methods (see, e.g., [Pol69], [BaW75], [Sha79], and [Pol87]) have also been popular in this context.

The proximal minimization algorithm was introduced in [Mar70], and was also understood through the studies of its dual equivalent, the method of multipliers. Reference [Ber82a] provides an extensive treatment of the latter method, and contains a large number of references on the subject; see also the survey papers [Ber76b] and [Roc76c]. The finiteness of the method when applied to linear programs was shown independently in [PoT74] and [Ber75]. The method of multipliers has been advocated for linear programs with structure that is unfavorable for the use of the simplex method [Ber76c], [BLS83], [Man84]. An extension of the proximal minimization algorithm for nonconvex problems is given in [Ber79a]; see also [TaM85].

The problem of solving systems of linear inequalities or, more generally, finding a point in the intersection of several convex sets has a long history; see [Cim38], [Agm54], [MoS54], [Bre67], [Tan71], [Aus76], [Jer79], [Elf80], and [Gof80]. References [Spi85a], [Spi87], and [Han88] are similar in spirit to the material in this section.

The alternating direction method of multipliers was proposed in [GlM75] and [GaM76], and was further developed in [Gab79]. It was generalized in [LiM79], where the connection with alternating direction methods for solving differential equations was pointed out. Discussions of the method and its applications in large boundary–value problems are given in [FoG83] and [GlL87]. Related work includes [Gol85b], [Spi85b], [Gol86a], [HaL88], and [RoW87]. An extension which addresses the problem of finding a zero of the sum of monotone operators is given in [Gol87b]. The decomposition algorithms for separable and linear programs of Subsection 3.4.4 (derived with assistance from J. Eckstein) are simpler and involve updating fewer variables than other related algorithms in the literature.

A generalization of the proximal minimization algorithm, called the *proximal point algorithm* was introduced in [Mar70] and [Mar72]. This algorithm applies also to variational inequalities under monotonicity assumptions. An extensive analysis and further development of the algorithm is given in [Roc76a], [Roc76b]. A rate of convergence analysis is given in [Luq84]. Extensions are given in [Luq86a] and [Luq86b]; in the case of the proximal minimization algorithm, these extensions involve the use of a nonquadratic additive term. The proximal point algorithm solves the problem of finding a zero of a maximal monotone operator and contains as special cases all of the algorithms discussed in Subsections 3.4.3 and 3.4.4 together with several other related methods. In particular the proximal minimization algorithm and the method of multipliers are special cases as shown in [Roc76a], [Roc76b]. The method of partial inverses of [Spi85b] was also developed as a special case of the proximal point algorithm. One of the splitting algorithms of [LiM79] contains as special cases both the method of partial inverses and

the alternating direction method of Section 3.4.4. It is shown in [Eck88] that the proximal point algorithm contains as a special case the algorithm of [LiM79] and *a fortiori* the alternating direction method. Therefore a substantial amount of analysis can be shared by all of the methods mentioned above. As an example, we could use known results on the proximal point algorithm to obtain a more elegant and insightful convergence analysis for the alternating direction method of multipliers than the one we gave here. We have not pursued this analysis because of its advanced mathematical character.

As mentioned in Subsection 3.4.4, the quadratic term used in the method of multipliers tends to affect adversely the separability structure of the problem. The alternating direction method can be viewed as one way of coping with this difficulty for some separable problems. A different approach, which also lends itself well to parallelization, was introduced in [StW75], and was further developed in [Sto77], [WNM78], and [CoZ83]; see also [Coh78].

3.5. For more background on variational inequalities, see [Aus76], [KiS80], and [GLT81]. Methods for solving the traffic assignment problem are given in [AaM81], [BeG82], [BeG83], [CaG74], [Daf71], [Daf80], [FlN74], [HLN84], [HLV87], and [LaH84]. The projection method for variational inequalities satisfying the strong monotonicity assumption of Prop. 5.4 was proposed in [Sib70]. This assumption was relaxed somewhat in [BeG82]. The extragradient method of [Kor76] (see Exercise 5.1) bypasses altogether the need for strong monotonicity, and is therefore applicable to linear programs (see Exercise 5.2). A general class of algorithms, generalizing the projection method has been introduced and analyzed in [Daf83]. The results concerning the product set case and nonlinear algorithms are adapted from [PaC82a], [PaC82b], and [Pan85], although the derivations here, using the general theory of contraction mappings, are different. The alternating direction method for variational inequalities was proposed in [Gab79]; see also [Gab83].