# Chapter 2

# Projection methods

## 2.1 Introduction

The purpose of this chapter is to provide a background about the projection methods for solving certain linear and least distance convex programming problems in which the feasible region is the intersection of a finite number of convex sets. The following least distance convex programming problem which arises in Chapters 3 and 5 is studied.

For a given point $\mathbf{f}$ find the point $\mathbf{x}^*$ which is the unique solution to the least distance problem

$$\begin{aligned} &minimize \quad \| \mathbf{f} - \mathbf{x} \|_2 \\ &subject \ \ to \ \ \mathbf{x} \in \bigcap_{i=1}^{m} K_i \end{aligned} \tag{2.1.1}$$

where $\bigcap_{i=1}^{m} K_i$ is the intersection of a finite number of convex sets $K_1$ , $K_2$ ,..., $K_m$. This minimization problem is one of a wide class of problems which arises in many applications.

Projection methods for solving this kind of problem were first given by von Neumann [1950], later improved by Dykstra [1983] and independently by Han [1988].

First some definitions relating to projections are introduced. If $K$ is a subspace in Hilbert space $H$ then define

$$K^{\perp} = \{\mathbf{x} \in H : \langle \mathbf{x}, \mathbf{y} \rangle = 0 \quad \forall \, \mathbf{y} \in K\}.$$

$K^{\perp}$ called the orthogonal complement of the set $K$.

**Definition 2.1.1**

If $K$ is a subspace in Hilbert space, then the projection of $\mathbf{x} \in H$ onto K is $\mathbf{x}_1$ where $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2$ and $\mathbf{x}_1 \in K$ and $\mathbf{x}_2 \in K^{\perp}$. The projection is denoted by $P_K(\mathbf{x}) = \mathbf{x}_1$ and $\mathbf{x}_1$ is unique.

To show that $\mathbf{x}_1$ is a unique point, let $\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 = \mathbf{y}_1 + \mathbf{y}_2$ with $\mathbf{y}_1 \in K$ and $\mathbf{y}_2 \in K^{\perp}$.

Now since $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle = 0$, it follows that $\langle \mathbf{x}_1, \mathbf{y}_1 + \mathbf{y}_2 - \mathbf{x}_1 \rangle = 0$ and hence

$$\langle \mathbf{x}_1, \mathbf{y}_1 - \mathbf{x}_1 \rangle = 0$$

since $\langle \mathbf{x}_1, \mathbf{y}_2 \rangle = 0$. Likewise from $\langle \mathbf{y}_1, \mathbf{y}_2 \rangle = 0$ it follow that

$$\langle \mathbf{y}_1, \mathbf{x}_1 - \mathbf{y}_1 \rangle = 0.$$

These two equations show that $\langle \mathbf{x}_1 - \mathbf{y}_1, \mathbf{x}_1 - \mathbf{y}_1 \rangle = 0$ and hence

$$\mathbf{x}_1 = \mathbf{y}_1.$$

**Theorem 2.1.2**

A necessary and sufficient condition that $P$ be a projection onto $K$ is that

i. $\langle P_K(\mathbf{x}), \mathbf{y} \rangle = \langle \mathbf{x}, P_K(\mathbf{y}) \rangle \quad \forall\, \mathbf{x}, \mathbf{y} \in H$

ii. $P_K^2(\mathbf{x}) = P_K(\mathbf{x}) \quad \forall\, \mathbf{x} \in H$

**Proof** (see von Neumann [1950]).

If $\mathbf{y} \in H$ then the projection map is completely characterized by the condition

$$\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{z} \rangle \leq 0 \quad \forall\, \mathbf{z} \in K. \tag{2.1.2}$$

where $\mathbf{x} = P_K(\mathbf{y})$. Clearly we can observe from the normal cone definition (1.3.7) that

$$\mathbf{y} - \mathbf{x} \in \partial K(\mathbf{x}). \tag{2.1.3}$$

Also in this chapter the following linear convex programming problem is considered.

$$\begin{aligned} &\textit{minimize} \quad \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \quad \mathbf{x} \in \bigcap_{i=1}^{m} K_i \end{aligned} \tag{2.1.4}$$

where $\mathbf{e} = [1, 1, \ldots, 1]^T \in \Re^n$.

Such optimization problems come up in many practical situations, for example in linear programming problem where $K$ is the set of linear constraints, although projection methods are not the best for solving such problems. Here we are interested in the case where one of the $K_i$ is a positive semi–definite matrix cone. An example of this is the educational testing problem in statistics.

In Section 2.2 the algorithms of von Neumann [1950], Dykstra [1983] and Han [1988] are described. The von Neumann algorithm simply iterates using succesive projections on to each $K_i$, while in the Dykstra and Han algorithms a more complex calculation is made. This section also includes some other important results. In Section 2.3 Glunt [1991] describes a projection method for solving the linear convex programming problem (2.1.4). His idea is to construct a hyperplane in $\Re^n$ and then carry out the method of alternating projections (von Neumann's method) between the convex set $K$ and the hyperplane. His method converges globally. The general theory of Glunt's method for minimizing the linear function subject to a convex set in Hilbert space is given in that section.

## 2.2 The Dykstra algorithm

In this section the least distance convex programming problem given by (2.1.1) is considered.

The basic idea of the iterated projections was first discussed by von Neumann [1950]. He showed that if $m = 2$, $K_1$ and $K_2$ are subspaces of Hilbert space $H$ and $P_1$ and $P_2$ are respectively the orthogonal projections onto $K_1$ and $K_2$, then the sequence of alternating projections is generated by the following algorithm:

**Algorithm 2.2.1** (*von Neumann algorithm*)

Given a point $\mathbf{f}$, in each subsequent iteration 2 vectors are computed as follows :

$$\begin{aligned}
&Set \quad && \mathbf{x}_2^{(0)} \;=\; \mathbf{f} \\
&For \quad && k = 1, 2, ... \\
& && Set \quad \mathbf{x}_0^{(k)} \;=\; \mathbf{x}_2^{(k-1)} \\
& && For \quad i = 1, 2 \\
& && \qquad \mathbf{x}_i^{(k)} \;=\; P_i(\mathbf{x}_{i-1}^{(k)}) \\
& && End \\
&End. && && (2.2.1)
\end{aligned}$$

The sequence in Algorithm 2.2.1 converges to $P_{K_1 \cap K_2}(\mathbf{f})$, which is the orthogonal projection onto the intersection of $K_1$ and $K_2$.

The von Neumann algorithm can be generalised for $m$ subspaces in the following form

**Algorithm 2.2.2**

Given a point $\mathbf{f}$, subspaces $K_1,\ K_2,\ \ldots,\ K_m$ and the corresponding projections $P_1,\ P_2,\ \ldots,\ P_m$ . In each subsequent iteration $m$ vectors are computed as follows :

$$\begin{aligned}
&Set \quad && \mathbf{x}_m^{(0)} \;=\; \mathbf{f} \\
&For \quad && k = 1, 2, ... \\
& && Set \quad \mathbf{x}_0^{(k)} \;=\; \mathbf{x}_m^{(k-1)} \\
& && For \quad i = 1, 2, \ldots, m \\
& && \qquad \mathbf{x}_i^{(k)} \;=\; P_i(\mathbf{x}_{i-1}^{(k)}) \\
& && End \\
&End. && && (2.2.2)
\end{aligned}$$

Deutsch [1983] showed that the rate of convergence in Algorithm 2.2.1 decreases with the angle $\theta$ between the two subspaces, where $\theta \in [0,\ \frac{\pi}{2}]$ and is defined by

$$\theta \;=\; \cos^{-1}\{ \sup_{\mathbf{a}\in K_1,\mathbf{b}\in K_2} \frac{|\,\langle \mathbf{a}-P_{K_1\cap K_2}\mathbf{a},\; \mathbf{b}-P_{K_1\cap K_2}\mathbf{b}\rangle\,|}{\|\mathbf{a}\|\;\|\mathbf{b}\|}\}$$

where $P_{K_1\cap K_2}$ is the orthogonal projection on to $K_1 \cap K_2$. This result is derived for the case $m = 2$ and nothing is said about the rate of convergence in the general case. Also it is not easy to find the angle between the two subspaces in order to get the rate of convergence.

Cheney and Goldstein [1959] prove some important results. In one of these they showed that if in Algorithm 2.2.1 the subspaces are replaced by convex sets $K_1$ and $K_2$, and $P_1$ and $P_2$ are respectively the orthogonal projections onto $K_1$ and $K_2$, then they gave the following theorem

**Theorem 2.2.3**

Let $K_1$ and $K_2$ be two convex sets in Hilbert space $H$. Let $P_1$ and $P_2$ represent, respectively, the projections onto $K_1$ and $K_2$. Given any point $\mathbf{f} \in H$, then algorithm (2.2.1) generate sequences $\{\mathbf{x}_1^{(k)}\}$, $\{\mathbf{x}_2^{(k)}\}$. If one of the sets is compact or finite dimensional and if the distance between them is attained, then the sequences $\{\mathbf{x}_1^{(k)}\}$ and $\{\mathbf{x}_2^{(k}\}$ converge to points $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively such that

$$\|\mathbf{x}_1 \;-\; \mathbf{x}_2\|_2 \;=\; \inf_{\mathbf{y}_1\in K_1,\; \mathbf{y}_2\in K_2} \|\mathbf{y}_1 \;-\; \mathbf{y}_2\|_2 \qquad (2.2.3)$$

**Proof** (See Cheney and Goldstein [1959])

This result is useful in the next section.

Dykstra [1983] pointed out that if $K_1$ and $K_2$ are not subspaces then the von Neumann algorithm does not necessarily converge. Likewise, Han [1988] stated that the von Neumann algorithm cannot be applied successfully to problem (2.1.1) for general $n$. This can be seen from the following simple example in $\Re^2$.

**Example 2.2.4**

Figure 2.2.1: This example illustrates the failure of von Neumann algorithm to solve problem (2.1.1) for general $n$.

Let

$$K_1 \;=\; \{(x,y) \;:\; y \;\leq\; 0)\}$$

and

$$K_2 \;=\; \{(x,y) \;:\; x+y \;\leq\; 0)\},$$

then the straightforward projection method does not work for any point $\mathbf{f}$ outside $K_1$ and $K_2$ with $x \neq 0$ and $x \neq y$. For example if $\mathbf{x}_2^{(0)} = \mathbf{f} = (1, 1.5)$ then $\mathbf{x}_1^{(1)} = P_1((1,1.5)) = (1,0)$ and $\mathbf{x}_2^{(1)} = P_2P_1((1,1.5)) = P_2((1,0)) = (0.5,-0.5)$ hence $P_2P_1((0.5,-0.5)) = (0.5,-0.5)$ and Algorithm 2.2.1 stops at $P_2P_1(\mathbf{f}) = (0.5,-0.5)$ while $\mathbf{x}^* = (0,0)$ (see Figure 2.2.1).

Dykstra's algorithm is based on an ingeniously simple modification of Algorithm 2.2.2. Han

[1988] independently discovered the same algorithm. In both algorithms the outer normal vector $\mathbf{y}_i^{(k)}$ of the set $K_i$ at $\mathbf{x}_i^{(k)}$ is calculated and the previous outer normal $\mathbf{y}_i^{(k-1)}$ is added to $\mathbf{x}_{i-1}^{(k)}$ before projecting it to the set $K_i$. Therefore, by each projection an old outer normal vector is replaced by a new one and the sequence of normal vectors are intended to converge to a solution of a dual problem of (2.1.1). In the case when all $K_i$ are subspaces then the addition of the normal is unneccessary for the corresponding projection and Algorithm 2.2.2 is recovered (Boyle et. al. [1986]). Dykstra's and Han's algorithm can be described as follows:

**Algorithm 2.2.5** (*Dykstra–Han algorithm*)

Given a point $\mathbf{f}$, convex sets $K_1,\ K_2, ...,\ K_m$ and the corresponding projections $P_1,\ P_2, ...,\ P_m$. Set

$$\mathbf{y}_1^{(0)} \ = \ \mathbf{y}_2^{(0)} \ = \ ... \ = \ \mathbf{y}_m^{(0)} \ = \ 0$$

and

$$\mathbf{x}_m^{(0)} \ = \ \mathbf{f}$$

Each subsequent iteration will compute $2m$ vectors

$$\mathbf{x}_1^{(k)},\ \mathbf{x}_2^{(k)},\ \ldots,\ \mathbf{x}_m^{(k)}$$
$$\mathbf{y}_1^{(k)},\ \mathbf{y}_2^{(k)},\ \ldots,\ \mathbf{y}_m^{(k)}$$

as follows:

$$\begin{aligned}
&set \quad \mathbf{x}_0^{(k)} \ = \ \mathbf{x}_m^{(k-1)} \\
&For \quad k \ = \ 1,\ 2,\ ... \\
&\quad For \quad i \ = \ 1,\ 2,\ ...,\ m \\
&\qquad \mathbf{z}_i^{(k)} \ = \ \mathbf{x}_{i-1}^{(k)} \ + \ \mathbf{y}_i^{(k-1)} \\
&\qquad \mathbf{x}_i^{(k)} \ = \ P_i(\mathbf{z}_i^{(k)}) \\
&\qquad \mathbf{y}_i^{(k)} \ = \ \mathbf{z}_i^{(k)} \ - \ \mathbf{x}_i^{(k)} \\
&\quad End \\
&End.
\end{aligned} \tag{2.2.4}$$

The following theorem by Dykstra [1983] gives the convergence result.

**Theorem 2.2.6**

The vectors $\mathbf{x}_i^{(k)}$ converge to the solution $\mathbf{x}^*$ of (2.1.1) as $k \rightarrow \infty$ for $i = 1, 2, \ldots, m$.

**Proof** (See Dykstra [1983])

The above algorithm is not easy to deal with and an algorithm which is easier to program and cheaper to run is the following

**Algorithm 2.2.7**

Given a point $\mathbf{f}$, convex sets $K_1, K_2, ..., K_m$ and the corresponding projections $P_1, P_2, ..., P_m$.

$$
\begin{aligned}
&\textit{Let} \quad \mathbf{f}^{(0)} = \mathbf{f} \\
&\textit{For} \quad k = 1, 2, \ldots \\
&\qquad \mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + P_m \ldots P_1(\mathbf{f}^{(k)}) - P_1(\mathbf{f}^{(k)}) \\
&\textit{End}
\end{aligned}
$$

A proof of how Algorithm 2.2.7 derived from Algorithm 2.2.5 using mathematical induction is now given.

First, we going to denote for $\mathbf{f}^{(0)} = \mathbf{f}$ and $\mathbf{f}^{(k-1)} = \mathbf{z}_1^{(k)}$. Then for $k = 1$

$$\mathbf{z}_1^{(1)} = \mathbf{f}^{(0)}, \qquad \mathbf{x}_1^{(1)} = P_1(\mathbf{f}^{(0)}), \qquad \mathbf{y}_1^{(1)} = \mathbf{f}^{(0)} - P_1(\mathbf{f}^{(0)}),$$

for $i > 1$

$$
\begin{aligned}
\mathbf{z}_i^{(1)} &= \mathbf{x}_{i-1}^{(1)} + \mathbf{y}_i^{(0)} = P_{i-1} \ldots P_1(\mathbf{f}^{(0)}) \\
\mathbf{x}_i^{(1)} &= P_i(\mathbf{z}_i^{(1)}) = P_i \ldots P_1(\mathbf{f}^{(0)}) \\
\mathbf{y}_i^{(1)} &= \mathbf{z}_i^{(1)} - \mathbf{x}_i^{(1)} = P_{i-1} \ldots P_1(\mathbf{f}^{(0)}) - P_i \ldots P_1(\mathbf{f}^{(0)}).
\end{aligned}
$$

Then for $k = 2$

$$\mathbf{f}^{(1)} = \mathbf{z}_1^{(2)} = \mathbf{x}_m^{(1)} + \mathbf{y}_1^{(1)} = P_m \ldots P_1(\mathbf{f}^{(0)}) + \mathbf{f}^{(0)} - P_1(\mathbf{f}^{(0)}).$$

Assume it is true for some $k > 2$, where

$$\mathbf{f}^{(k-1)} := \mathbf{z}_1^{(k)} = P_m \ldots P_1(\mathbf{f}^{(k-2)}) + \mathbf{f}^{(k-2)} - P_1(\mathbf{f}^{(k-2)})$$

and

$$\mathbf{z}_i^{(k)} = \mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)}$$

where

$$\mathbf{y}_i^{(k-1)} := \sum_{l=0}^{k-2} \{P_{i-1} \ldots P_1(\mathbf{f}^{(l)}) - P_i \ldots P_1(\mathbf{f}^{(l)})\}.$$

Then for $k+1$, it is clear that

$$P_i(\mathbf{y}_i^{(k-1)}) = 0$$

then

$$\begin{aligned} \mathbf{x}_1^{(k)} &= P_1(\mathbf{z}_1^{(k)}) = P_1(\mathbf{f}^{(k-1)}) \\ \mathbf{x}_i^{(k)} &= P_i(\mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)}) = P_i(\mathbf{x}_{i-1}^{(k)}) \\ &= P_i \ldots P_1(\mathbf{f}^{(k-1)}). \quad for\ i \geq 2 \end{aligned}$$

Also

$$\begin{aligned} \mathbf{y}_i^{(k)} &= \mathbf{z}_i^{(k)} - \mathbf{x}_i^{(k)} \\ &= \mathbf{x}_{i-1}^{(k)} + \mathbf{y}_i^{(k-1)} - \mathbf{x}_i^{(k)} \\ &= P_{i-1} \ldots P_1(\mathbf{f}^{(k-1)}) + \\ &\qquad \sum_{l=0}^{k-2} \{P_{i-1} \ldots P_1(\mathbf{f}^{(l)}) - P_i \ldots P_1(\mathbf{f}^{(l)})\} - \\ &\qquad\qquad P_i \ldots P_1(\mathbf{f}^{(k-1)}) \\ &= \sum_{l=0}^{k-1} \{P_{i-1} \ldots P_1(\mathbf{f}^{(l)}) - P_i \ldots P_1(\mathbf{f}^{(l)}). \end{aligned}$$

Therefore

$$\begin{aligned} \mathbf{f}^{(k)} \;=\; \mathbf{z}_1^{(k+1)} \;&=\; \mathbf{x}_m^{(k)} \;+\; \mathbf{y}_1^{(k)} \\ &=\; P_m \;\ldots\; P_1(\mathbf{f}^{(k-1)}) \;+\; \mathbf{z}_1^{(k)} \;-\; \mathbf{x}_1^{(k)} \\ &=\; P_m \;\ldots\; P_1(\mathbf{f}^{(k-1)}) \;+\; \mathbf{f}^{(k-1)} \;-\; P_1(\mathbf{f}^{(k-1)}). \end{aligned}$$

Which is Algorithm 2.2.7. □

In Algorithm 2.2.7 the vectors $\mathbf{y}_i$ for $i > 1$ are not used and saved from calculation which makes it cheaper. Using Boyle and Dykstra [1986] convergence result, we have the following theorem.

**Theorem 2.2.8**

Given $\mathbf{f}$ and the sequence $\{\mathbf{f}^{(k)}\}$ generated by Algorithm 2.2.7 then $P_i \;\ldots\; P_1(\mathbf{f}^{(k)}) \;\rightarrow\; \mathbf{d}^*$ the optimal solution of (2.1.1), for any $i \geq 1$.

**Proof** (See Boyle and Dykstra [1986])

In Example 2.2.4 Figure 2.2.2 shows how Algorithm 2.2.7 works successfully. In Figure 2.2.2 $\mathbf{f}^{(k)}$ is projected onto $K_1$ then onto $K_2$ and then subtracting $P_1(\mathbf{f}^{(k)})$ from $\mathbf{f}^{(k)} \;+\; P_1P_2(\mathbf{f}^{(k)})$ produces the new $\mathbf{f}^{(k+1)}$. It is clear from Figure 2.2.2 that $P_1(\mathbf{f}^{(k)})$ converges to the optimal solution $\mathbf{0}$ on the $x$–axis. Also $P_1P_2(\mathbf{f}^{(k)})$ converges to $\mathbf{0}$ on the $x \;=\; -\,y$ axis. Clearly $\mathbf{f}^{(k)}$ converges to $\mathbf{f}^* \;\neq\; \mathbf{0}$.

Dykstra [1983], shows that if the $K_i$ are convex cones, then the $\mathbf{x}_i$ converge to the nearest point to the initial point $\mathbf{f}$ in the intersection of the $K_i$. This result has been extended by Boyle et. al. [1986] to the case where some of the $K_i$ are convex sets. Han [1988] has shown that the algorithm works for general convex sets $K_i$ given that the intersection has nonempty interior. Gaffke and Mathar [1989], show that the interior point condition may be omitted. The result of Boyle et. al. [1986] is enough for our application of projection method in Chapters 3, 5 and 6.

Figure 2.2.2: Illustrates the success of Dykstra–Han algorithm to solve problem (2.1.1) for general $n$.

## 2.3 A projection algorithm for linear convex programming problems

This section describes a projection method due to Glunt [1991] for solving the linear convex programming problem (2.1.4). He uses a particular choice of convex sets in an ingenious way. One important linear convex programming problem is the educational testing problem which will be solved by the method of this section in Chapter 6.

Glunt's idea is to take account of the function $\mathbf{e}^T\mathbf{x}$ by defining the hyperplane

$$L_\tau \;=\; \{\mathbf{y} \;\in\; \Re^n |\; f(\mathbf{y}) \;=\; \tau\} \tag{2.3.1}$$

where $f(\mathbf{y}) \;=\; \mathbf{e}^T\mathbf{y}$. If $\tau$ is chosen such that

$$\tau \;<\; \min_{\mathbf{x}\in K} f(\mathbf{x}) \tag{2.3.2}$$

then the sets $K$ and $L_\tau$ are disjoint. Given $\mathbf{f} \;\in\; \Re^n$ Glunt then applies the von Neumann Algorithm 2.2.1 to the problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \|\mathbf{f} \;-\; \mathbf{x}\|_2 \\ &\textit{subject to} \quad \mathbf{x} \;\in\; K \;\cap\; L_\tau \end{aligned} \tag{2.3.3}$$

which has no feasible solution. It follows from the Theorem 2.2.3 of Cheney and Goldstein [1959] that the iterates $\mathbf{x}_1^{(k)}$ and $\mathbf{x}_2^{(k)}$ will converge to points $\mathbf{x}_1^* \;\in\; L_\tau$ and $\mathbf{x}_2^* \;\in\; K$ such that $\|\mathbf{x}_1 \;-\; \mathbf{x}_2\|_2$ attains the minimum distance between $K$ and $L_\tau$. It can then be deduced from the relationship of $L_\tau$ and $\mathbf{e}^T\mathbf{x}$ (2.3.1), that $\mathbf{x}_2^*$ solves problem (2.1.4).

The von Neumann algorithm involves computing alternately the projections onto $L_\tau$ and $K$. That onto $L_\tau$ is straightforward. Glunt suggests that the projection on the $K \;=\; \bigcap_{i=1}^m K_i$ is computed by using an inner iteration based on the Dykstra algorithm. It follows from Theorem 2.2.6 that the resulting method is globally convergent.

The following is a statement of the outer (von Neumann) algorithm.

**Algorithm 2.3.1**

Given an arbitrary $\mathbf{g} \;\in\; \Re^n$, convex sets $K_1,\; K_2, ...,\; K_m$ and the corresponding projections $P_1,\; P_2, ...,\; P_m$ .

$$\begin{aligned} &\textit{Set} \qquad \mathbf{x}_2^{(0)} \;=\; \mathbf{g} \\ &\textit{For} \qquad k \;=\; 1,\; 2,\; ... \\ &\qquad\quad \textit{Set} \quad \mathbf{x}_0^{(k)} \;=\; \mathbf{x}_2^{(k-1)} \\ &\qquad\quad \mathbf{x}_1^{(k)} \;=\; P_{L_\tau}(\mathbf{x}_0^{(k)}) \\ &\qquad\quad \mathbf{x}_2^{(k)} \;=\; P_K(\mathbf{x}_1^{(k)}) \\ &\textit{End} \end{aligned} \tag{2.3.4}$$

where $K = \bigcap_{i=1}^{m} K_i$

In this algorithm in every outer iteration $P_K(\mathbf{x}_1^{(k)})$ is calculated by solving the following problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \|P_{L_\tau}(\mathbf{x}_0^{(k)}) - \mathbf{x}\|_2 \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \quad \mathbf{x} \in K = \bigcap_{i=1}^{m} K_i. \end{aligned} \tag{2.3.5}$$

where $\mathbf{f}$ (in problem (2.1.1)) $= P_{L_\tau}(\mathbf{x}_0^{(k)}) = \mathbf{x}_1^{(k)}$ which is an initial point in every outer iteration. Problem (2.3.5) is solved using Algorithm 2.2.7.

The following three figures show how Algorithm 2.3.1 works under different circumstances. In Figure 2.3.1 the convex set $K$ is nonsmooth at the solution and it is seen that Algorithm 2.3.1 terminates in 2 iterations with the point $\mathbf{x}_2^*$ as solution. In Figures 2.3.2 and 2.3.3 $K$ is smooth at the solution and it seen that the solution point $\mathbf{x}_2^*$ is the limit point of the sequence $\{\mathbf{x}_2^{(k)}\}$. It can also be observed that if we make $\tau$ smaller (as in Figure 2.3.3 $(\tau = -3)$) then a more rapid rate of convergence is obtained. However a study of the numerical results indicates that the order of convergence is linear or slower. Similar features are observed when Glunt's method is applied to the educational testing problem as described in Chapter 6.

The following theorem, due to Glunt [1991], gives the convergence result for the linear convex programming problems.

**Theorem 2.3.2**

For any $\mathbf{g} \in \Re^n$, the sequences $\{\mathbf{x}_1^{(k)}\}$ and $\{\mathbf{x}_2^{(k}\}$ generated by Algorithm 2.3.1 converge to $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively. Also the sequence $\{\mathbf{x}_2^{(k)}\}$ converges to the solution of the problem

$$\begin{aligned} &\textit{minimize} \quad f(\mathbf{x}) = \mathbf{e}^T\mathbf{x} \qquad \mathbf{x} \in \Re^n \\ &\textit{subject to} \quad \mathbf{x} \in K. \end{aligned} \tag{2.3.6}$$

The function values $f(\mathbf{x}_2^{(k)})$ decrease strictly monotonically to the minimal value.

Figure 2.3.1: Algorithm 2.3.1 terminates for a nonsmooth convex set.

Figure 2.3.2: Algorithm 2.3.1 converges for a smooth convex set.

Figure 2.3.3: Making $\tau$ smaller gives faster convergence.

**Proof** (Glunt [1991])

The convergence of the two sequences $\{\mathbf{x}_1^{(k)}\}$ and $\{\mathbf{x}_2^{(k}\}$ follows from Theorem 2.2.3. Set

$$\begin{aligned} \mathbf{x}_1^* &= \lim_{k\to\infty} \mathbf{x}_1^{(k)} \\ \mathbf{x}_2^* &= \lim_{k\to\infty} \mathbf{x}_2^{(k)}. \end{aligned}$$

Let $\mathbf{x}_2 = P_K(\mathbf{x}_1)$ then from the characterization of the projection map (2.1.2)

$$\langle \mathbf{x}_2 \ - \ \mathbf{x}_1, \ \mathbf{x}_2 \ - \ \mathbf{z} \rangle \ \leq \ 0. \qquad \forall \ \mathbf{z} \ \in \ K \tag{2.3.7}$$

Now $\mathbf{x}_1 \ = \ P_{L_\tau}(\mathbf{x}_2)$, and $L_\tau$ is a hyperplane with the unit vector $\mathbf{e}$, so $P_{L_\tau}(.)$ is easy to compute

$$P_{L_\tau}(\mathbf{x}_2) \ = \ \mathbf{x}_2 \ + \ \frac{\tau \ - \ \mathbf{e}^T\mathbf{x}_2}{\|\mathbf{e}\|^2}\mathbf{e}. \tag{2.3.8}$$

So from (2.3.7)

$$\langle \mathbf{x}_2 \ - \ (\mathbf{x}_2 \ + \ \frac{\tau \ - \ \mathbf{e}^T\mathbf{x}_2}{\|\mathbf{e}\|^2}\mathbf{e}, \ \mathbf{x}_2 \ - \ \mathbf{z} \rangle \ \leq \ 0 \qquad \forall \ \mathbf{z} \ \in \ K$$

$$\Rightarrow \qquad \langle (\mathbf{e}^T\mathbf{x}_2 \ - \ \tau)\mathbf{e}, \ \mathbf{x}_2 \ - \ \mathbf{z} \rangle \ \leq \ 0 \qquad \forall \ \mathbf{z} \ \in \ K$$

$$\Rightarrow \qquad (\mathbf{e}^T\mathbf{x}_2 \ - \ \tau) \ \langle (\mathbf{e}, \ \mathbf{x}_2 \ - \ \mathbf{z} \rangle \ \leq \ 0 \qquad \forall \ \mathbf{z} \ \in \ K$$

But

$$\tau \ < \ \min_{\mathbf{z} \in K} \mathbf{e}^T\mathbf{z}$$

hence $\mathbf{e}^T\mathbf{x}_2 \ - \ \tau \ \geq \ 0$. Therefore

$$\langle \mathbf{e}, \ \mathbf{x}_2 \ - \ \mathbf{z} \rangle \ \leq \ 0. \qquad \forall \ \mathbf{z} \ \in \ K$$

or

$$\mathbf{e}^T\mathbf{x}_2 \ \leq \ \mathbf{e}^T\mathbf{z}. \qquad \forall \ \mathbf{z} \ \in \ K$$

Thus $\mathbf{x}_2$ solves (2.3.6).

To demonstrate that $f(\mathbf{x}_2^{(k)})$ for $k \ = \ 1, 2, \ \ldots$ is monotonically decreasing, consider $\mathbf{x}_2^{(k)}$ and write

$$\begin{aligned} \mathbf{x}_1^{(k)} \ &= \ P_{L_\tau}(\mathbf{x}_2^{(k-1)}) \\ \mathbf{x}_2^{(k)} \ &= \ P_K(\mathbf{x}_1^{(k)}). \end{aligned}$$

Thus $\mathbf{x}_2^{(k)}$ is the nearest point in $K$ to $\mathbf{x}_1^{(k)}$. Therefore unless $\mathbf{x}_2^* = \mathbf{x}_2^{(k-1)} \ = \ \mathbf{x}_2^{(k)}$,

$$\|\mathbf{x}_2^{(k)} \ - \ \mathbf{x}_1^{(k)}\| \ < \ \|\mathbf{x}_2^{(k-1)} \ - \ \mathbf{x}_1^{(k)}\|. \tag{2.3.9}$$

Similarly, $\mathbf{x}_1^{(k)}$ is the nearest point in $L_r$ to $\mathbf{x}_2^{(k-1)}$, so unless $\mathbf{x}_1^* = \mathbf{x}_1^{(k-1)} = \mathbf{x}_1^{(k)}$,

$$\|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k+1)}\| < \|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k)}\|. \tag{2.3.10}$$

Thus from (2.3.9) and (2.3.10)

$$\|\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k+1)}\| < \|\mathbf{x}_2^{(k-1)} - \mathbf{x}_1^{(k)}\|. \tag{2.3.11}$$

But

$$\begin{aligned}
\mathbf{x}_2^{(k)} - \mathbf{x}_1^{(k+1)} &= \mathbf{x}_2^{(k)} - P_{L_\tau}(\mathbf{x}_2^{(k)}) \\
&= \mathbf{x}_2^{(k)} - (\mathbf{x}_2^{(k)} + \frac{\tau - \mathbf{e}^T\mathbf{x}_2^{(k)}}{\|\mathbf{e}\|^2}\mathbf{e}) \\
&= \frac{\mathbf{e}^T\mathbf{x}_2^{(k)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e}.
\end{aligned}$$

Similarly

$$\mathbf{x}_2^{(k-1)} - \mathbf{x}_1^{(k)} = \frac{\mathbf{e}^T\mathbf{x}_2^{(k-1)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e}.$$

Hence from (2.3.11)

$$\frac{\mathbf{e}^T\mathbf{x}_2^{(k)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e} < \frac{\mathbf{e}^T\mathbf{x}_2^{(k-1)} - \tau}{\|\mathbf{e}\|^2}\mathbf{e}.$$

or

$$\mathbf{e}^T\mathbf{x}_2^{(k)} < \mathbf{e}^T\mathbf{x}_2^{(k-1)}$$

proving that $f(\mathbf{x})$ is monotonically decreasing. □