Journal of the Operations Research
Society of Japan
Vol. 37, No. 1, March 1994

A RECURSIVE ALGORITHM FOR FINDING THE MINIMUM NORM POINT IN AN UNBOUNDED POLYHEDRON

Wakasa Kise
Mitsubishi Electric Corporation

Kazuyuki Sekitani
Science University of Tokyo

(Received November 25, 1991; Revised August 10, 1992)

Abstract We present a recursive algorithm for finding the minimum norm point in an unbounded, convex and pointed polyhedron defined by finite sets of points and rays. The algorithm can start at an arbitrary point of the polyhedron and does not require to solve systems of linear equations.

1. Introduction

The minimum norm point problem in a polytope arises in various fields such as pattern recognition [1], the problem of mixing [5] and the portfolio selection [9]. Several algorithms have been developed for the problem and some good computational results are reported regardless of their exponential time complexity, see [3], [4] and [11]. In this paper we consider a generalization of the minimum norm point problem, i.e., the problem of finding the minimum norm point in an unbounded convex polyhedron. This problem also has applications in the field similar to those above mentioned. Fathi and Murty [6] have proposed a critical index algorithm for finding the point in a simplicial cone which is closest to the given point and Wilhelmsen [10] has developed an algorithm for a convex polyhedral cone, which is similar to Wolfe's algorithm [11] in structure. Let P and R be a given set of finitely many points of n-dimensional Euclidean space E^n and a given set of directions of finitely many rays of E^n, respectively. Without loss of generality we can assume $\| r \| = 1$ for every $r \in R$, where $\| \cdot \|$ means the Euclidean norm. Let us denote the convex hull of P and the conical hull of R by $C(P)$ and $Cone\,(R)$, respectively. We define the sum of two sets of E^n as the sum of vectors of two sets. In this paper we assume that $Cone\,(R)$ is a pointed cone, hence $C(P) + Cone(R)$ is a pointed polyhedron. We consider the following problem:

$$\min\{ \| x \| \mid x \in C(P) + Cone(R) \}, \tag{1.1}$$

Note that Problem (1.1) has a unique solution. We denote the optimum solution of Problem (1.1) by $Nr(C(P) + Cone(R))$.

Sekitani and Yamamoto [7] developed a recursive algorithm $\mathcal{N}_1$ for finding the minimum norm point in a polytope. The algorithm $\mathcal{N}_1$ does not require to solve systems of linear equations, so that it needs neither a pivoting procedure nor an inversion of a matrix. Our proposed algorithm is an extension of the algorithm $\mathcal{N}_1$ and uses it as a subprocedure. Our algorithm is also performed without a pivoting procedure or an inversion of a matrix since its main routine has the same structure as the algorithm $\mathcal{N}_1$ has. In Section 2 we review the basic theorems of the minimum norm point problem and describe the outline of the algorithm $\mathcal{N}_1$ briefly. We provide the key theorem for the proposed algorithm. In Section 3 we propose a recursive algorithm $\mathcal{N}_P$ for Problem (1.1) and show that it provides $Nr(C(P) + Cone(R))$ after a finite number of iterations. In Section 4 we give results of computational experiments to reveal the actual behavior of the proposed algorithm $\mathcal{N}_P$.

2. Preliminary

We give the optimality condition of Problem (1.1) as follows.

Theorem 2.1. *$\hat{x} \in C(P) + Cone(R)$ is $Nr(C(P) + Cone(R))$ if and only if*

$$\hat{x}^t p \geq \|\hat{x}\|^2 \quad \text{for every } p \in P \text{ and} \tag{2.1}$$

$$\hat{x}^t r \geq 0 \quad \text{for every } r \in R. \tag{2.2}$$

Theorem 2.1 furnishes the optimality condition of minimum norm point problem in a polytope in case that $R = \emptyset$.

Corollary 2.2. *$\hat{x} \in C(P)$ is $Nr(C(P))$ if and only if*

$$\hat{x}^t p \geq \|\hat{x}\|^2 \quad \text{for every } p \in P \tag{2.3}$$

We describe the outline of the algorithm $\mathcal{N}_1$ for the minimum norm point problem in a polytope, e.g., $\min\{\, \|x\|^2 \mid x \in C(P)\,\}$. First we choose arbitrarily a point $\hat{x}$ from $C(P)$ and check if the point $\hat{x}$ satisfies the optimality condition (2.3). If there exists a point $p \in P$ such that $\hat{x}^t p < \|\hat{x}\|^2$, then we put $\hat{P} = \{\, p \in P \mid \hat{x}^t p = \min\{\, \hat{x}^t q \mid q \in P\,\}\,\}$, solve the subproblem $\min\{\, \|x\|^2 \mid x \in C(\hat{P})\,\}$ by applying the procedure $\mathcal{N}_1$ with $\hat{P}$ as the input data recursively and let $\hat{y}$ be $Nr(C(\hat{P}))$ obtained as the output of this recursive call. If $\hat{y}$ is not $Nr(C(P))$, then we calculate $\hat{\lambda} = \max\{\, \lambda \mid \{(1-\lambda)\hat{x} + \lambda\hat{y}\}^t \hat{y} \leq \{(1-\lambda)\hat{x} + \lambda\hat{y}\}^t p$ for every $p \in P \setminus \hat{P}\,\}$. We replace $\hat{x}$ by $(1-\hat{\lambda})\hat{x} + \hat{\lambda}\hat{y}$ and repeat this routine.

Taking the condition (2.2) into consideration we will extend the algorithm $\mathcal{N}_1$ to the algorithm $\mathcal{N}_P$ for Problem (1.1). For an $|R|$-dimensional coefficient vector α with component α_r corresponding to $r \in R$, let us define

$$R(\alpha) = \{\alpha_r r \mid r \in R\}.$$

Since $Cone\,(R(\alpha)) = Cone\,(R)$ whenever $\alpha > 0$, we see the following theorem.

Theorem 2.3. *If $Cone\,(R)$ is pointed, then*

$$\{\, Nr\,(C\,(R\,(\alpha))) \mid \alpha > 0\,\} = \{\, z \mid z^t r > 0 \text{ for every } r \in R\,\} \cap Cone\,(R).$$

Proof: Let $\alpha > 0$ and $b(\alpha) := Nr\,(C\,(R\,(\alpha)))$. Since $Cone\,(R(\alpha))$ is also pointed, it follows that $0 \notin C\,(R\,(\alpha))$. Then we see that for every $r \in R$ $(\alpha_r r)^t b(\alpha) \geq \|b(\alpha)\|^2 > 0$ and that $b(\alpha) \in C\,(R\,(\alpha))$. Since $\alpha > 0$ and $C\,(R\,(\alpha)) \subseteq Cone\,(R\,(\alpha)) = Cone\,(R)$, it follows that for every $r \in R$ $r^t b(\alpha) > 0$ and $b(\alpha) \in Cone\,(R)$. Let $z \in \{\, z \mid z^t r > 0$ for every $r \in R\,\} \cap Cone\,(R)$ and $\alpha_r = \|z\|^2 / z^t r$ for every $r \in R$. Then we see that for every $r \in R$ $\alpha_r > 0$ and $\alpha_r r^t z = \|z\|^2$. Furthermore there exists an $|R|$-dimensional vector β such that $\beta \geq 0$ and $z = \sum_{r \in R} \beta_r (\alpha_r r)$. This means that $1 = \sum_{r \in R} \beta_r$. Hence it follows that $z = Nr\,(C\,(R\,(\alpha)))$ □

3. Algorithm $\mathcal{N}_P$ for Finding $Nr(C(P) + Cone(R))$

In this section we consider Problem (1.1) of finding $Nr(C(P) + Cone(R))$ for a given finite point set P and a given finite set R of directions of rays. The algorithm $\mathcal{N}_P$ first chooses arbitrarily an initial point w_0 from $C(P) + Cone(R)$. Unless the point w_0 satisfies the condition (2.2) of Theorem 2.1 with $\hat{x}$ replaced by w_0, it applies the procedure $\mathcal{N}_1$ to the convex hull $C(R)$ of the given set R of directions of rays to obtain the direction $z_0 = Nr(C(R))$. It moves a point x_0 from w_0 in the direction z_0 until the point x_0 satisfies the condition (2.2) with $\hat{x}$ replaced by x_0. In the k^{th} iteration with x_{k-1} as the current point, it keeps the condition (2.2) with $\hat{x}$ replaced by x_{k-1} and generates a subset P_k of P and a subset R_k of R. It calls itself with P_k and R_k as the input data and moves the point x_{k-1} to the point y_k obtained as the output of this recursive call. Now, we give the algorithm for finding the minimum norm point of the unbounded polyhedron $C(P) + Cone(R)$.

Algorithm $\mathcal{N}_P(P, R)$

Step 0 : Choose a point w_0 from $C(P) + Cone(R)$.
If $\min\{ w_0^t r \mid r \in R \} \geq 0$, then $x_0 := w_0$ and go to Step 1.
Call $\mathcal{N}_1(C(R))$ and $z_0 := Nr(C(R))$.
$\lambda_0 := \max\{ \lambda \mid \{(1-\lambda)z_0 + \lambda w_0\}^t r \geq 0$ for every $r \in R \}$.
$x_0 := ((1-\lambda_0)/\lambda_0) z_0 + w_0$ and $k := 1$.

Step 1 : $\alpha_k := \min\{ x_{k-1}^t p \mid p \in P \}$.
If $\|x_{k-1}\|^2 \leq \alpha_k$, then $\hat{x} := x_{k-1}$ and stop.
$P_k := \{ p \mid p \in P$ and $x_{k-1}^t p = \alpha_k \}$.
$R_k := \{ r \mid r \in R$ and $x_{k-1}^t r = 0 \}$.

Step 2 : Call $\mathcal{N}_P(P_k, R_k)$ and $y_k := Nr(C(P_k) + Cone(R_k))$.

Step 3 : $\beta_k := \min\{ y_k^t p \mid p \in P \setminus P_k \}$.
$\gamma_k := \min\{ y_k^t r \mid r \in R \setminus R_k \}$.
If $\|y_k\|^2 \leq \beta_k$ and $0 \leq \gamma_k$ then $\hat{x} := y_k$ and stop.

Step 4 : $\lambda_k :=$

$$\max\left\{ \lambda \,\middle|\, \begin{array}{lll} \{(1-\lambda)x_{k-1} + \lambda y_k\}^t y_k & \leq & \{(1-\lambda)x_{k-1} + \lambda y_k\}^t p \quad \text{for every } p \in P \setminus P_k \\ \{(1-\lambda)x_{k-1} + \lambda y_k\}^t r & \geq & 0 \quad \text{for every } r \in R \setminus R_k \end{array} \right\}.$$

$x_k := (1-\lambda_k)x_{k-1} + \lambda_k y_k$, $k := k+1$ and go to Step 1.

If $R = \emptyset$, in the algorithm $\mathcal{N}_p(P, R)$ we define $\min\{ w_0^t r \mid r \in R\} = \min\{ y_k^t r \mid r \in R \setminus R_k \} = \infty$.

Lemma 3.1. $0 < \lambda_0 < 1$.

Proof: Clearly λ_0 is determined by

$$\lambda_0 = \max\{ \lambda \mid z_0^t r \geq \lambda(z_0^t r - w_0^t r) \text{ for every } r \in R \}. \tag{3.1}$$

It follows from Theorem 2.3 that for every $r \in R$ $z_0^t r > 0$. This means that $\lambda_0 > 0$. Furthermore since there exists a direction $\bar{r}$ of R such that $w_0^t \bar{r} < 0$, we see that $\lambda_0 \leq z_0^t \bar{r}/(z_0^t \bar{r} - w_0^t \bar{r}) < 1$. □

The following lemma states that the point x_0 satisfies the condition (2.2) with $\hat{x}$ replaced by x_0 and stays in $C(P) + Cone(R)$.

Lemma 3.2. *$x_0 \in C(P) + Cone(R)$ and $x_0^t r \geq 0$ for every $r \in R$.*

Proof: If $\min\{ w_0^t r \mid r \in R \} \geq 0$, then by the choice of the initial point w_0 and $w_0 = x_0$ we see that $x_0 \in C(P) + Cone(R)$ and $x_0^t r \geq 0$ for every $r \in R$. Otherwise the point x_0

is determined by $x_0 = w_0 + ((1-\lambda_0)/\lambda_0) z_0$. From Theorem 2.3 and Lemma 3.1 it follows that $x_0 \in C(P) + Cone(R)$. From (3.1) we see that $\lambda_0 x_0^t r = \lambda_0 w_0^t r + (1-\lambda_0) z_0^t r \geq 0$ for every $r \in R$. This means by Lemma 3.1 that $x_0^t r \geq 0$ for every $r \in R$. □

Lemma 3.3. *$0 < \lambda_k < 1$ and $x_k^t r \geq 0$ for every $r \in R$ and for $k = 1, 2, \ldots$.*

Proof: We prove the lemma by induction on the iteration. In the first iteration the lemma follows from Lemma 3.1and 3.2. Suppose that $0 < \lambda_{k-1} < 1$ and $x_{k-1}^t r \geq 0$ for every $r \in R$ as the hypothesis of induction. In the k^{th} iteration we see that

(1) for every $p \in P \setminus P_k$ $\quad x_{k-1}^t p - x_{k-1}^t y_k > 0$,

(2) for every $r \in R \setminus R_k$ $\quad x_{k-1}^t r > 0$,

(3) there exists either $\bar{p} \in P \setminus P_k$ such that $y_k^t \bar{p} < \|y_k\|^2$ or $\bar{r} \in R \setminus R_k$ such that $y_k^t \bar{r} < 0$ and

(4) $\lambda_k =$

$$\min \left\{ \begin{array}{ll} \min\{ \frac{x_{k-1}^t (p-y_k)}{(x_{k-1}-y_k)^t (p-y_k)} & \mid p \in P \setminus P_k \text{ and } (x_{k-1}-y_k)^t (p-y_k) > 0 \}, \\ \min\{ \frac{x_{k-1}^t r}{(x_{k-1}-y_k)^t r} & \mid r \in R \setminus R_k \text{ and } (x_{k-1}-y_k)^t r > 0 \} \end{array} \right\}.$$

From (1),(2) and (4) we see $\lambda_k > 0$. Furthermore it follows from (3) and (4) either $\lambda_k \leq \frac{x_{k-1}^t (\bar{p}-y_k)}{(x_{k-1}-y_k)^t (\bar{p}-y_k)} < 1$ or $\lambda_k \leq \frac{x_{k-1}^t \bar{r}}{(x_{k-1}-y_k)^t \bar{r}} < 1$. By the choice of λ_k we see that for every $r \in R \setminus R_k$ $\{(1-\lambda_k) x_{k-1} + \lambda_k y_k\}^t r \geq 0$. Since $y_k = Nr\,(C(P_k) + Cone\,(R_k))$, it follows from the hypothesis of induction that for every $r \in R_k$ $\{(1-\lambda_k) x_{k-1} + \lambda_k y_k\}^t r = (1-\lambda_k) x_{k-1}^t r + \lambda_k y_k^t r \geq 0$. □

By Lemma 3.2 and 3.3 we see by induction that each point x_{k-1} as well as y_k is contained in $C(P) + Cone(R)$.

Lemma 3.4. *$x_k \in C(P) + Cone(R)$ for $k = 1, 2, \ldots$.*

By applying Theorem 2.1 we will see that a point $\hat{x}$ obtained either in Step 1 or in Step 3 is $Nr\,(C(P) + Cone(R))$.

Lemma 3.5. *$\hat{x} = Nr\,(C(P) + Cone(R))$.*

Proof: When the algorithm terminates in Step 1, we see that $\alpha_k \geq \|x_{k-1}\|^2$, which is equivalent to ${x_{k-1}}^t p \geq \|x_{k-1}\|^2$ for every $p \in P$. This means by Lemma 3.3 and 3.4 that $x_{k-1} = Nr\,(C(P) + Cone(R))$. When it terminates in Step 3, it follows that $\beta_k \geq \|y_k\|^2$ and $\gamma_k \geq 0$. Then we see that for every $p \in P \setminus P_k$ $y_k^t p \geq \|y_k\|^2$ and for every $r \in R \setminus R_k$ $y_k^t r \geq 0$. Since $y_k = Nr\,(C(P_k) + Cone(R_k))$, it holds that for every $p \in P_k$ $\; y_k^t p \geq \|y_k\|^2$ and for every $r \in R_k$ $\; y_k^t r \geq 0$. Hence $y_k = Nr\,(C(P) + Cone(R))$. □

We will see that the sets P_k and R_k generated in Step 2 form a proper face $C(P_k) + Cone(R_k)$ of the given polyhedron $C(P) + Cone(R)$.

Lemma 3.6. *$|P_k| + |R_k| < |P| + |R|$ and $C(P_k) + Cone\,(R_k)$ is a proper face of $C(P) + Cone(R)$ for $k = 1, 2, \ldots$.*

Proof: Since $\|x_{k-1}\|^2 > \alpha_k = \min\{x_{k-1}^t p \mid p \in P\}$ and $x_{k-1} \in C(P) + Cone(R)$, there exists either $\bar{p} \in P$ which does not lie on the affine hull of P_k or $\bar{r} \in R$ which is linearly independent of R_k. This means that either $|P_k| < |P|$ or $|R_k| < |R|$ and that $\dim\{C(P_k) + Cone(R_k)\} < \dim\{C(P) + Cone(R)\}$. Since $P_k = \{p \in P \mid x_{k-1}^t p = \min\{ x_{k-1}^t q \mid q \in P \}\}$, $R_k = \{r \in R \mid x_{k-1}^t r = 0\}$ and $0 < \min\{x_{k-1}^t r \mid r \in R \setminus R_k\}$, it follows from Theorem 2.4.12 in [8] that $C(P_k) + Cone(R_k)$ is a face of $C(P) + Cone(R)$. □

Lemma 3.7. *There exists either a point $\bar{p} \in P_{k+1} \setminus P_k$ such that $y_k^t \bar{p} < \|y_k\|^2$ or a direction $\bar{r} \in R_{k+1} \setminus R_k$ such that $y_k^t \bar{r} < 0$.*

Proof: We have either $\bar{p} \in P \setminus P_k$ or $\bar{r} \in R \setminus R_k$ which attains (4) of Lemma 3.3. If there exists $\bar{p} \in P \setminus P_k$ attaining (4) of Lemma 3.3, we can show $\bar{p} \in P_{k+1} \setminus P_k$ and $y_k^t \bar{p} < \|y_k\|^2$ in the same way as the proof of Lemma 2.4 in [7]. Otherwise for $\bar{r} \in R \setminus R_k$ we see from Lemma 3.3 that $x_{k-1}^t \bar{r} < (x_{k-1} - y_k)^t \bar{r}$, which implies $y_k^t \bar{r} < 0$. Furthermore by the determination of λ_k it follows that $\{(1-\lambda_k)x_k + \lambda_k y_k\}^t \bar{r} = 0$, i.e., $x_k^t \bar{r} = 0$. This means $\bar{r} \in R_{k+1}$. □

Lemma 3.8. $y_k \in C(P_{k+1}) + Cone(R_{k+1})$.

Proof: Since $y_k = Nr\,(C(P_k) + Cone(R_k))$, by the Karush-Kuhn-Tucker condition there exist a $|P_k|$-dimensional vector μ and an $|R_k|$-dimensional vector ν such that $\mu \geq 0$, $\nu \geq 0$, $\sum_{p \in P_k} \mu_p = 1$, $y_k = \sum_{p \in P_k} \mu_p p + \sum_{r \in R_k} \nu_r r$, $\mu_p\{y_k^t p - \|y_k\|^2\} = 0$ for every $p \in P_k$ and $\nu_r y_k^t r = 0$ for every $r \in R_k$. Let $z_k = \sum_{p \in P_k} \mu_p p$. Then we see from the definition of R_k that

$$\begin{aligned}
\{(1-\lambda_k)x_{k-1} + \lambda_k y_k\}^t (y_k - z_k) &= \{(1-\lambda_k)x_{k-1} + \lambda_k y_k\}^t \sum_{r \in R_k} \nu_r r \\
&= (1-\lambda_k) \sum_{r \in R_k} \nu_r x_{k-1}^t r + \lambda_k \sum_{r \in R_k} \nu_r y_k^t r \\
&= 0.
\end{aligned}$$

This means $y_k - z_k \in Cone(R_{k+1})$. We can show $z_k \in C(P_{k+1})$ in the same way as the proof of Lemma 2.5 in [7]. Hence it follows that $y_k \in C(P_{k+1}) + Cone(R_{k+1})$. □

The following lemma is the key to finite convergence of the algorithm.

Lemma 3.9. $\| y_{k+1} \| < \| y_k \|$ *for* $k = 1, 2, \ldots$.

Proof: First suppose that there exists the direction $\bar{r}$ of Lemma 3.7. Choose a λ such that $0 < \lambda < -2y_k^t \bar{r}$ and let $z = y_k + \lambda \bar{r}$. Then we see from Lemma 3.7 and 3.8 that $z \in C(P_{k+1}) + Cone(R_{k+1})$ and that

$$\begin{aligned}
\|z\|^2 &= \|y_k\|^2 + 2\lambda y_k^t \bar{r} + \lambda^2 \|\bar{r}\|^2 \\
&= \|y_k\|^2 + \lambda(2y_k^t \bar{r} + \lambda) \\
&< \|y_k\|^2.
\end{aligned}$$

This means that $\| Nr\,(C(P_{k+1}) + Cone(R_{k+1})) \| \leq \| z \| < \| y_k \|$, which implies $\| y_{k+1} \| < \| y_k \|$. Next we consider that there exists a point $\bar{p}$ of Lemma 3.7. Then we can show $\| y_{k+1} \| < \| y_k \|$ in the same way as the proof of Lemma 2.6 in [7]. □

Lemma 3.10. *When P and R consist of a single point and a single direction of ray, respectively, the algorithm $\mathcal{N}_P$ terminates within a finite number of iterations.*

Lemma 3.11. *When $R = \emptyset$, $\mathcal{N}_P$ provides the minimum norm point $Nr(C(P))$ within a finite number iterations.*

Proof: When $R = \emptyset$, $\mathcal{N}_P(P, R)$ is identical to $\mathcal{N}_1(P)$. □

Theorem 3.12. *When P and R consist of finitely many points and finitely many directions of rays, respectively, the algorithm $\mathcal{N}_P$ provides the minimum norm point $Nr(C(P) + Cone(R))$ within a finite number of iterations.*

Proof: We can assume by Lemma 3.10 that $\mathcal{N}_P(P', R')$ is finite for subsets $P' \subseteq P$ and $R' \subseteq R$, whenever either P' or R' has fewer elements than P or R respectively. Then since $|P_k| + |R_k| < |P| + |R|$ when the algorithm calls itself in Step 2, each step of the algorithm $\mathcal{N}_P(P, R)$ is finite. Since $\min\{ \| x \| \mid x \in C(P_{k+1}) + Cone(R_{k+1}) \} = \| y_{k+1} \| < \| y_k \| = \min\{ \| x \| \mid x \in C(P_k) + Cone(R_k)\}$, no $\{P_k, R_k\}$ is generated more than once. Thus $\mathcal{N}_P(P, R)$ terminates within a finite number of iterations. □

When we are given sets P and R of m_p points and m_r directions, respectively, on a plane, the unbounded polyhedron $C(P) + Cone(R)$ has at most m_p vertices and at most $m_p + 2$ facets. This observation yields the following theorem.

Theorem 3.13. *The algorithm $\mathcal{N}_P$ finds $Nr(C(P) + Cone(R))$ with $O((m_p+m_r)^2)$ time complexity when $P \subseteq E^2$ and $R \subseteq E^2$.*

Proof: Each step excluding Step 0 and 2 of $\mathcal{N}_P$ is of $O(m_p + m_r)$ time complexity. It follows from Lemma 3.6 that the polyhedron defined by (P_k, R_k) of Step 2 is of either zero or one dimension. Clearly the problem on a zero dimensional polyhedron, that is a single point, is solved with a constant time complexity. Suppose that $C(P_k) + Cone(R_k)$ is of 1-dimension. Then we see that $\mathcal{N}_P$ does not repeat Step 2 more than once. Since $\|r\|^2 = 1$ for all $r \in R_k \subseteq R$ we also see $|R_k| \leq 1$. Then Step 0 of $\mathcal{N}_P(P_k, R_k)$ is done with a constant time. It follows from Lemma 3.6 that Step 2 of $\mathcal{N}_P(P_k, R_k)$ is done within a constant time. Hence the algorithm $\mathcal{N}_P(P_k, R_k)$ provides $Nr(C(P_k) + Cone(R_k))$ within a linear time of $|P_k| + |R_k|$.

Each step excluding Step 0 of $\mathcal{N}_P(P, R)$ is of at most $O(m_p + m_r)$ time complexity. Theorem 2.9 in [7] shows that Step 0 of $\mathcal{N}_P(P, R)$ is executed within $O(m_r^2)$. Since the algorithm $\mathcal{N}_P(P, R)$ does not generate the same facet or same vertex of $C(P) + Cone(R)$, it provides the solution $Nr(C(P) + Cone(R))$ with $O((m_p + m_r)^2)$ time complexity. □

4. Computational Experiment

In this section we demonstrate the efficiency of the algorithm and reveal its behavior by some computational experiments. We first randomly choose a point z from the n-cube $\{ x \in E^n \mid -ne \leq x \leq ne \}$ and make an n-cube $\{ x \in E^n \mid -\sqrt{n}e + z \leq x \leq \sqrt{n}e + z \}$ translated by z. The set P consists of m_p points which are randomly chosen from this n-cube. We generate a set R of m_r points in the following manner. The first $n-1$ components $r_1, \ldots, r_{n-1}$ of each point are randomly chosen from the interval $[-n, n]$ and the last component r_n is set $3n - \sum_{i=1}^{n-1} r_i$. We then normalize the m_r points thus generated on the hyperplane $\{ x \in E^n \mid e^t x = 3n \}$ to the unit length. Note that the cone generated by R is pointed.

Type 1 : We consider $n = 10$ and the ratio $m_p/m_r = 3/7$. We have varied the total number $m_p + m_r$ as 100, 200, 400, 600, 800, 1000, 1500 and 2000 and have solved 10 problems for each.

Type 2 : We consider the 2-dimensional problem and the ratio $m_p/m_r = 3/7$. We have changed the total number $m_p + m_r$ as 1000, 5000, 10000, 15000, 20000, 25000, 30000 and have carried out 10 trials for each.

Type 3 : We have dealt with the minimum norm point problem of the cone for $n = 10$. We have varied the number of directions, m_r, 100, 200, 400, 600, 800, 1000, 1500 and 2000 and have examined 10 problems for each.

Since Kise [4] reports that for the minimum norm point problem in a polytope the several devices of the algorithm $\mathcal{N}_1$ is useful to reduce a computational time, for the implementation of $\mathcal{N}_P$ we have made the following similar devices.

(1) Let us consider the case where $R_k = \emptyset$. When $|P_k| \geq 3$, by Lemma 3.11 we call the subprocedure $\mathcal{N}_1(P_k)$ instead of $\mathcal{N}_P(P_k, R_k)$. When $|P_k| \leq 2$, we did not call $\mathcal{N}_1$ recursively to suppress the over-head time of recursive calls. In fact when $|P_k| = 1$, i.e., $P_k = \{p\}$, then $p = Nr(C(P_k) + Cone(R_k))$. For the case when $|P_k| = 2$, e.g., P_k consists of p^1 and p^2, let us define

$$\lambda^* = \frac{(p^1)^t(p^1 - p^2)}{\|p^1 - p^2\|^2}.$$

Then $Nr(C(P_k) + Cone(R_k))$ is determined in the following way:

(1.1.a) $Nr(C(P_k) + Cone(R_k)) = p^2$ if $\lambda^* \geq 1$,

(1.1.b) $Nr(C(P_k) + Cone(R_k)) = p^1$ if $\lambda^* \leq 0$ and

(1.1.c) $Nr(C(P_k) + Cone(R_k)) = (1 - \lambda^*)p^1 + \lambda^* p^2$ if $0 < \lambda^* < 1$.

Furthermore when $|P_k| = 1$ and $|R_k| = 1$, e.g., p^1 and r^1 respectively, we did not call $\mathcal{N}_P$ recursively but calculate

$$\mu^* = (p^1)^t r^1.$$

Then $Nr(C(P_k) + Cone(R_k))$ is determined in the following way:

(1.2.a) $Nr(C(P_k) + Cone(R_k)) = p^1$ if $\mu^* \geq 0$ and

(1.2.b) $Nr(C(P_k) + Cone(R_k)) = p^1 - \mu^* r^1$ if $\mu^* < 0$.

(2) Let us split P_{k+1} into two parts, $\hat{P}_{k+1} = P_{k+1} \setminus P_k$ and $\bar{P}_{k+1} = P_{k+1} \cap P_k$, and R_{k+1} into two parts, $\hat{R}_{k+1} = R_{k+1} \setminus R_k$ and $\bar{R}_{k+1} = R_{k+1} \cap R_k$. Define

$$\lambda_k^p = \min\left\{ \frac{x_{k-1}^t(p - y_k)}{(x_{k-1} - y_k)^t(p - y_k)} \,|\, p \in P \setminus P_k \text{ and } (x_{k-1} - y_k)^t(p - y_k) > 0 \right\}$$

and

$$\lambda_k^r = \min\left\{ \frac{x_{k-1}^t r}{(x_{k-1} - y_k)^t r} \,|\, r \in R \setminus R_k \text{ and } (x_{k-1} - y_k)^t r > 0 \right\},$$

then $\lambda_k = \min\{ \lambda_k^p, \lambda_k^r \}$ from (4) of Lemma 3.3. $\hat{P}_{k+1}$ and $\hat{R}_{k+1}$ are determined in the following manner:

(2.a) $$\begin{array}{l} \hat{P}_{k+1} = \left\{ p \in P \setminus P_k \,|\, \frac{(x_{k-1})^t(p-y_k)}{(x_{k-1}-y_k)^t(p-y_k)} = \lambda_k^p \right\} \\ \hat{R}_{k+1} = \emptyset \end{array} \quad \text{if } \lambda_k^p < \lambda_k^r,$$

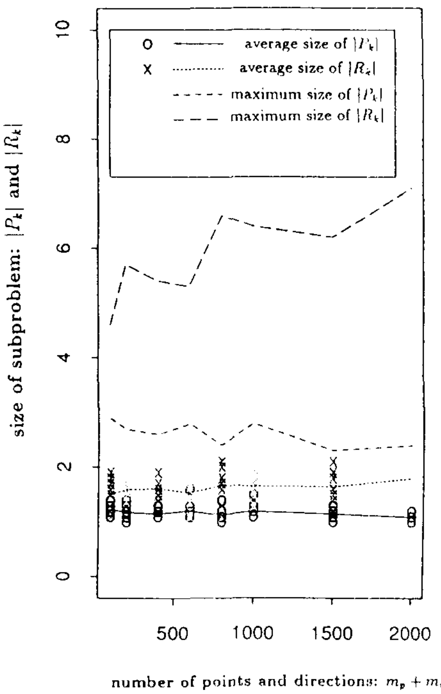

Figure 4.1 Plot of $|P_k|$ and $|R_k|$ v.s. $m_p + m_r$ for Type 1

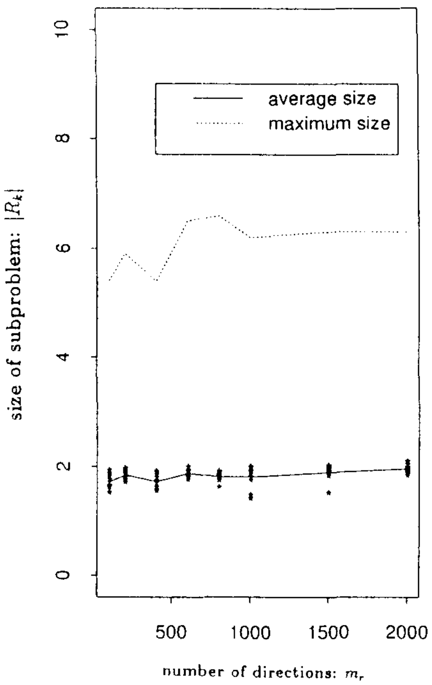

Figure 4.2 Plot of $|R_k|$ v.s. m_r for Type 3

$$
\text{(2.b)}\quad \begin{array}{l} \hat{P}_{k+1} = \emptyset \\ \hat{R}_{k+1} = \left\{ r \in R \setminus R_k \mid \frac{x_{k-1}^t r}{(x_{k-1}-y_k)^t r} = \lambda_k^r \right\} \end{array} \qquad \text{if } \lambda_k^r < \lambda_k^p \text{ and}
$$

$$
\text{(2.c)}\quad \begin{array}{l} \hat{P}_{k+1} = \left\{ p \in P \setminus P_k \mid \frac{(x_{k-1})^t (p-y_k)}{(x_{k-1}-y_k)^t (p-y_k)} = \lambda_k^p \right\} \\ \hat{R}_{k+1} = \left\{ r \in R \setminus R_k \mid \frac{x_{k-1}^t r}{(x_{k-1}-y_k)^t r} = \lambda_k^r \right\} \end{array} \qquad \text{if } \lambda_k^r = \lambda_k^p.
$$

Hence $\hat{P}_{k+1}$ or $\hat{R}_{k+1}$ is found as the set of points or the set of directions attaining λ_k in Step 4. To make $\bar{P}_{k+1}$ we have to evaluate $x_k^t p$ only for points of P_k and to collect those points satisfying $x_k^t p = x_k^t y_k$. In the similar way as $\bar{P}_{k+1}$ we collect the directions r satisfying $x_k^t r = 0$ in R_k to make $\bar{R}_{k+1}$.

Fig. 4.1 and 4.2 show the size of the subproblems solved in Step 2 for the problems of Type 1 and 3, respectively.

We say that the proposed algorithm executes at level 0 when the stack of the recursive call is empty and that it does at level $l+1$ when it calls itself recursively at level l. We have measured the number of the transitions from level l to level $l+1$ of the algorithm and plotted their weighted average v.s. $m_p + m_r$ in Fig. 4.3. Fig. 4.4, 4.5 and 4.6 are the logarithmic plot of the computational time t, exclusive of input and the output, of the algorithm for the problems of Type 1, 2 and 3, respectively, as a function of $m_p + m_r$. From these figures we have the following three observations about the algorithm $\mathcal{N}_P$ with the devices of (1) and (2).

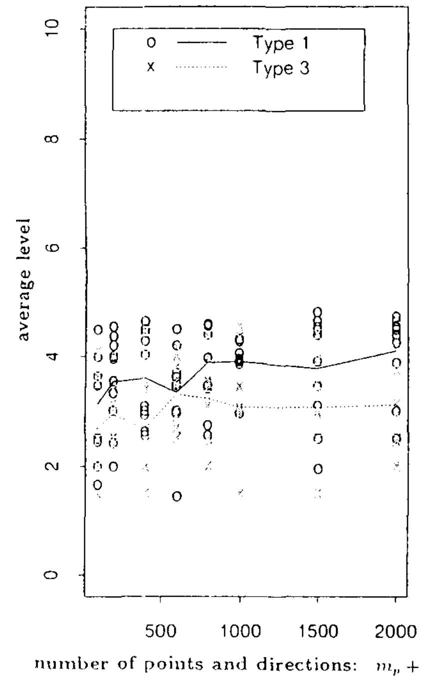

Figure 4.3 Plot of average level v.s. $m_p + m_r$ for Type 1 and 3

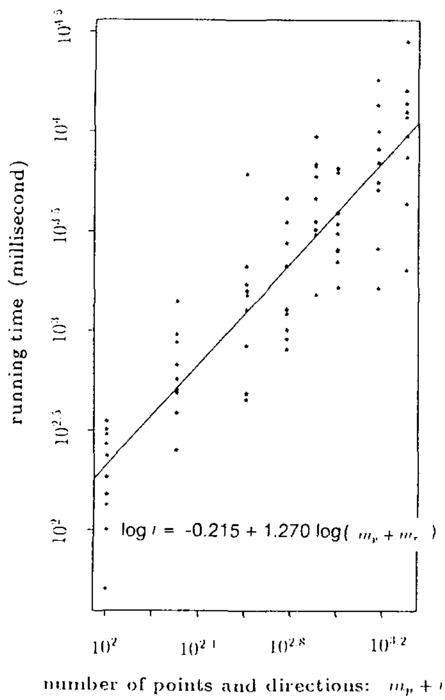

Figure 4.4 Plot of $\log t$ vs. $\log m_p + m_r$ for Type 1

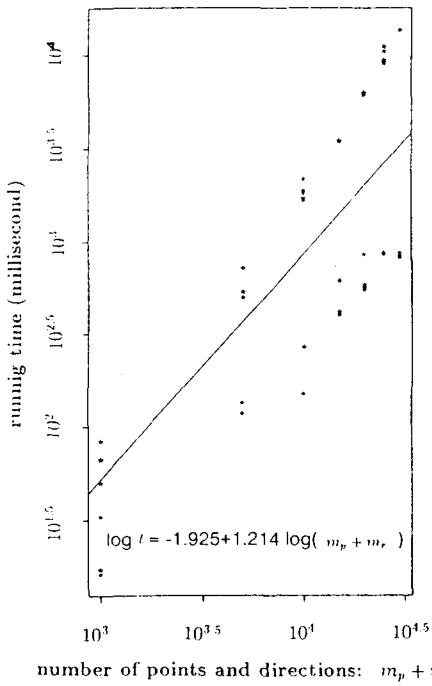

Figure 4.5 Plot of $\log t$ vs. $\log m_p + m_r$ for Type 2

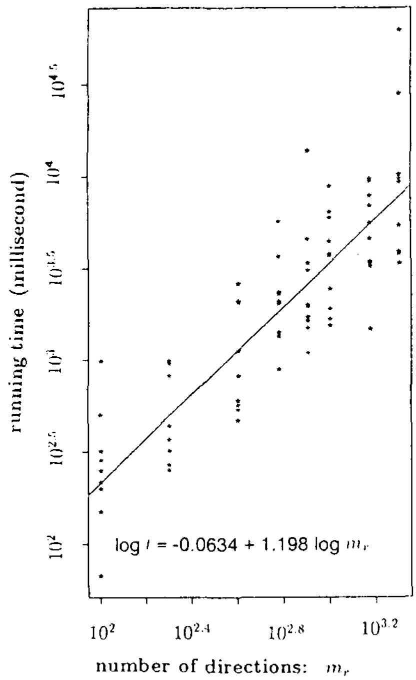

Figure 4.6 Plot of $\log t$ vs. $\log m_r$ for Type 3

(A) We observe from Fig. 4.1 and 4.2 that the maximum size of the subproblem solved in Step 2 is invariant against $m_p + m_r$ and is less than the dimension n and that the average size is much lower than the maximum size since most of the subproblem are of size 1 or 2.

(B) We see from Fig 4.3 that the average level where the proposed algorithm calls itself recursively is invariant against $m_p + m_r$ and it is almost half the dimension $n = 10$.

(C) In Fig 4.4, 4.5 and 4.6 we have three regression lines $0.609(m_p + m_r)^{1.270}$, $0.0011(m_p + m_r)^{1.274}$ and $0.864(m_p + m_r)^{1.198}$ for type 1,2 and 3, respectively. Especially $0.609(m_p + m_r)^{1.270}$ grows more slowly than the worst case analysis in Theorem 3.13.

We cannot measure the overhead time the algorithm needs to provide $Nr(C(P) + Cone(R))$. If the recursive call requires heavy overhead, we could save it by applying Wolfe's algorithm [11] instead of the subprocedure $\mathcal{N}_1$ of the implemented version $\mathcal{N}_P$ with (1.1.a), (1.1.b), (1.1.c), (2.a) and (2.b). To make the proposed algorithm more efficient various suggestions and further experiments should be needed. This task is beyond the aim of this paper, so we leave it for future research.

5. Concluding Remarks

Let us consider the minimum norm point problem in a polyhedral cone,

$$\min\{ \|x\|^2 \mid \{w_0\} + Cone(R) \}. \tag{5.1}$$

Problem (5.1) is a special case of Problem (1.1). However, it is interesting that Problem (5.1) is rewritten as $\min\{ \|x + w_0\|^2 \mid x \in Cone(R) \}$ and this problem is reduced to a linear complementarity problem [6]. We can simplify the recursive algorithm $\mathcal{N}_P$ for Problem (5.1) as follows.

Algorithm $\mathcal{N}_C(R)$

Step 0 : If $\min\{ w_0^t r \mid r \in R \} \geq 0$, then $\hat{x} := w_0$ and stop.

Step 1 : Call $\mathcal{N}_1(R)$ and $z_0 := Nr(C(R))$.
$\lambda_0 := \max\{ \lambda \mid \{(1-\lambda)z_0 + \lambda w_0\}^t r \geq 0$ for every $r \in R \}$.
$x_0 := w + ((1-\lambda_0)/\lambda_0) z_0$ and $k := 1$.

Step 2 : If $x_{k-1}^t w_0 \geq \|x_{k-1}\|^2$, then $\hat{x} := x_{k-1}$ and stop.
$R_k := \{ r \mid r \in R$ and $x_{k-1}^t r = 0 \}$.

Step 3 : Call $\mathcal{N}_C(R_k)$ and $y_k := Nr(\{w_0\} + Cone(R_k))$.

Step 4 : $\beta_k := \min\{ y_k^t r \mid r \in R \setminus R_k \}$.
If $0 \leq \beta_k$, then $\hat{x} := y_k$ and stop.

Step 5 : $\lambda_k := \max\{ \lambda \mid \{(1-\lambda)x_{k-1} + \lambda y_k\}^t r \geq 0$ for every $r \in R \setminus R_k \}$.
$x_k := (1-\lambda_k)x_{k-1} + \lambda_k y_k$, $k := k+1$ and go to Step 2.

The given point w_0 itself serves as the initial point in the algorithm $\mathcal{N}_C$. So we need not examine the optimality condition (2.1) with the set P and $\hat{x}$ replaced by $\{w_0\}$ and the

k^{th} iterate x_{k-1}, respectively. In exactly the same way as Section 3 we obtain the similar lemmas and finite convergence of $\mathcal{N}_C$. So we omit the proof.

It is well known, e.g., [2], that the problem of finding a nearest pair of points in two polytopes is reduced to the minimum norm point problem. Let Q be a set of finitely many points of E^n. It is seen that the minimum distance problem $\min\{ \| x - y \| \mid x \in C(P),\ y \in C(Q) + Cone(R) \}$ is equivalent to $\min\{ \| z \| \mid z \in C(P - Q) + Cone(-R) \}$, where $-R = \{ -r \mid r \in R \}$. Thus we can solve the minimum distance problem by the algorithm $\mathcal{N}_P$.

Acknowledgments

The authors are grateful to Professor Yoshitsugu Yamamoto who has given them this theme, encouraged them with many useful comments and helped to improve the presentation of this paper. They also wish to thank Professor Satoru Fujishige for valuable comments.

References

[1] L.C. Barbosa and E. Wong, "On a class of iterative algorithms for linear inequalities with applications to pattern classification", in: *Proceedings of the First Annual Princeton Conference of Information Sciences and Systems* (1967) pp.86-89.

[2] M.D. Canon and C.D. Cullum, "The determination of optimum separating hyperplanes I. A finite step procedure" RC 2023, IBM Watson Research Center, Yorktown Heights, NY (February 1968).

[3] S. Fujishige and P. Zhan, "A dual algorithm for finding the minimum-norm point in a polytope", *Journal of the Operations Research Society of Japan* 33 (1990) 188-195.

[4] W. Kise, "An extension and practical improvements of the recursive algorithm for the solving the minimum norm point problem", Master thesis, University of Tsukuba (Tsukuba, Ibaraki,1993).

[5] E. Klafszky, J. Mayer and T. Terlaky, "On mathematical programming model of mixing", *European Journal of Operational Research* 42 (1989) 254-267.

[6] K.G. Murty and Y. Fathi, "A critical index algorithm for nearest point problems on simplicial cones", *Mathematical Programming* 23 (1982) 206-215.

[7] K. Sekitani and Y. Yamamoto, "A recursive algorithm for finding the minimum norm point in a polytope and a pair of closest points in two polytopes", *Mathematical Programming* 61 (1993) 233-249.

[8] J. Stoer and C. Witzgall, *Convexity and Optimization in Finite Dimension I* (Springer-Verlag, Berlin, 1978).

[9] H. Takehara, "An interior point algorithm for finding the minimum norm point in a polytope", MTEC Working Paper No. T911, MTB Investment Technology Institute, Co., Ltd, Tokyo (1991).

[10] D.R. Wilhelmsen, "A nearest point algorithm for convex polyhedral cones and applications to positive linear approximations", *Mathematics of Computation* 30 (1976) 48-57.

[11] P. Wolfe, "Finding the nearest point in a polytope", *Mathematical Programming* 11 (1976) 128-149.

Kazuyuki SEKITANI:
Department of Industrial Administration
Science University of Tokyo
2641, Noda, Chiba 278, Japan