Department of Industrial Engineering and Management

Technical Report

No. 2013-1

The LP-Newton method for standard form linear programming problems

Tomonari Kitahara, Shinji Mizuno and Jianming Shi

March, 2013

Tokyo Institute of Technology

2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, JAPAN
http://www.me.titech.ac.jp/index-e.html

The LP-Newton method for standard form linear programming problems

Tomonari Kitahara[*], Shinji Mizuno[†], and Jianming Shi [‡].

March 2013

Abstract

Fujishige et al. propose the LP-Newton method, a new algorithm for solving linear programming problems (LPs). They address LPs which have a lower and an upper bound for each variable. They reformulate the problem by introducing a related zonotope. Their algorithm solves the problem by repeating projections to the zonotope.

In this paper, we develop the LP-Newton method for solving standard form LPs. We recast the LP by introducing a related convex cone. Our algorithm solves the problem by iterating projections to the convex cone.

Keywords: Linear programming, LP-Newton method, Wolfe's algorithm

1 Introduction

Linear programming problem (LP) is a basis of all mathematical programming problems. The simplex method and the interior point method solve LPs efficiently in practice, and the latter is a polynomial-time algorithm. However, whether there is a strongly polynomial-time algorithm for LP or

[*]Graduate School of Decision Science and Technology, Tokyo Institute of Technology, 2-12-1-W9-62, Oo-Okayama, Meguro-ku, Tokyo, 152-8552, Japan. Tel.: +81-3-5734-2896, Fax: +81-3-5734-2947, E-mail: kitahara.t.ab@m.titech.ac.jp.

[†]Graduate School of Decision Science and Technology, Tokyo Institute of Technology, 2-12-1-W9-58, Oo-Okayama, Meguro-ku, Tokyo, 152-8552, Japan. Tel.: +81-3-5734-2816, Fax: +81-3-5734-2947, E-mail: mizuno.s.ab@m.titech.ac.jp.

[‡]College of Information and Electronic Engineering, Muroran Institute of Technology (MuIT) 27-1, Mizumoto-chou, Muroran Hokkaido, 050-0071, Japan. Tel.: 0143-46-5423, Fax: 0143-46-5423, E-mail: shi@mmm.muroran-it.ac.jp.

not has been a long standing open problem. As an attempt to solve this problem, Fujishige et al. [1] propose a new algorithm for solving LPs. They address LPs which have a lower and an upper bound for each variable. In this paper, we call their algorithm LPB-Newton method. They reformulate the problem by introducing a related zonotope. LPB-Newton method solves the problem by repeating projections to the zonotope. For the projection, they use an algorithm proposed by Wolfe [3].

It is desirable to develop an algorithm which can handle more general LPs. In this paper, we develop a similar algorithm for standard form LPs. We call our algorithm LPS-Newton method. We recast the problem by introducing a related convex cone. LPS-Newton method solves the problem by applying projections to the convex cone. For the projection, we can use, for example, an algorithm developed by Wilhelmsen [2].

2 LP, zonotope, and convex cone

2.1 Zonotope and LPB-Newton method

Fujishige et al.[1] consider the following LP

$$\begin{array}{ll} \max & \boldsymbol{c}^T\boldsymbol{x}, \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}, \end{array} \tag{1}$$

where $\boldsymbol{A} \in \Re^{m\times n}$, $\boldsymbol{b} \in \Re^m$, $\boldsymbol{c} \in \Re^n$, $\boldsymbol{l} \in \Re^n$ and $\boldsymbol{u} \in \Re^n$ are given data and $\boldsymbol{x} \in \Re^n$ is a variable vector.

Let

$$\bar{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{c}^T \end{bmatrix} \tag{2}$$

and define the zonotope $\bar{Z}$ as follows.

$$\bar{Z} = \{\bar{\boldsymbol{z}} \in \Re^{m+1} |\ \bar{\boldsymbol{z}} = \bar{\boldsymbol{A}}\boldsymbol{x},\ \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}\}. \tag{3}$$

Note that the number of vertices of $\bar{Z}$ is at most 2^n. Using $\bar{Z}$, (1) can be expressed as

$$\begin{array}{ll} \max & \gamma, \\ \text{subject to} & \begin{bmatrix} \boldsymbol{b} \\ \gamma \end{bmatrix} \in \bar{Z}, \end{array} \tag{4}$$

where γ is a variable in $\Re$. Note that the set

$$L = \{\begin{bmatrix} \boldsymbol{b} \\ \gamma \end{bmatrix} |\ \gamma \in \Re\} \tag{5}$$

is a line which is parallel to the last axis of $\Re^{m+1}$. Under the formulation (4), we find the intersection point of $\bar{Z}$ and L whose $(m+1)$-st element is maximal.

LPB-Newton method generates a sequence of points $\{\bar{\boldsymbol{b}}_k\}$ on L. Compute the vector $\hat{\boldsymbol{x}} = (\hat{x}_1, \ldots, \hat{x}_n)$ by

$$\hat{x}_j = \begin{cases} u_j & \text{if } c_j > 0 \\ l_j & \text{otherwise} \end{cases} \tag{6}$$

and set $\gamma_0 = \boldsymbol{c}^T \hat{\boldsymbol{x}}$, which is the maximal value of $\boldsymbol{c}^T \boldsymbol{x}$ on $\bar{Z}$. The algorithm starts from $\bar{\boldsymbol{b}}_0 = (\boldsymbol{b}^T, \gamma_0)^T$. At the k-th iteration, given $\bar{\boldsymbol{b}}_k$, we compute $\bar{\boldsymbol{z}}_{k+1}$, the nearest point of $\bar{\boldsymbol{b}}_k$ to the zenotope $\bar{Z}$. Then $\bar{\boldsymbol{b}}_{k+1}$ is obtained as the intersection point of L and H_k, the separating hyperplane of $\bar{Z}$ at $\bar{\boldsymbol{z}}_{k+1}$.

For the projection, Fujishige et al. [1] adopt Wolfe's algorithm [3]. A formal description of LPB-Newton method is as follows.

LPB-Newton method

Input: $\boldsymbol{A} \in \Re^{m \times n}$, $\boldsymbol{b} \in \Re^m$, $\boldsymbol{c} \in \Re^n$, $\boldsymbol{l} \in \Re^n$ and $\boldsymbol{u} \in \Re^n$ of the problem (1).
Output: an optimal solution $\boldsymbol{x}^*$ of the problem (1) or the decision that the problem (1) is infeasible.

Step 1: Compute $\hat{\boldsymbol{x}}$ according to (6), set $\gamma_0 = \boldsymbol{c}^T \hat{\boldsymbol{x}}$, and $k = 0$.

Step 2: Set $\bar{\boldsymbol{b}}_k = (\boldsymbol{b}^T, \gamma_k)^T$. Compute $\bar{\boldsymbol{z}}_{k+1}$, the nearest point of $\bar{\boldsymbol{b}}_k$ to $\bar{Z}$. We express $\bar{\boldsymbol{z}}_{k+1}$ like

$$\bar{\boldsymbol{z}}_{k+1} = \begin{bmatrix} \tilde{\boldsymbol{z}}_{k+1} \\ \zeta_{k+1} \end{bmatrix}.$$

$\bar{\boldsymbol{z}}_{k+1}$ is expressed as a convex combination of affinely independent extreme points $\boldsymbol{y}_s$ $(s \in J_k)$ of $\bar{Z}$, i.e. $\bar{\boldsymbol{z}}_{k+1} = \sum_{s \in J_k} \lambda_{s,k} \boldsymbol{y}_s$ and each $\boldsymbol{y}_s$ is given by $\boldsymbol{y}_s = \bar{\boldsymbol{A}} \boldsymbol{x}_s$ with $(\boldsymbol{x}_s)_j = l_j$ or $(\boldsymbol{x}_s)_j = u_j$ for each $j = 1, \ldots, n$.
(1) If $\bar{\boldsymbol{z}}_{k+1} = \bar{\boldsymbol{b}}_k$, then put $\boldsymbol{x}^* = \sum_{s \in J_k} \lambda_{s,k} \boldsymbol{x}_s$ and return $\boldsymbol{x}^*$.
(2) If $\bar{\boldsymbol{z}}^{k+1} \neq \bar{\boldsymbol{b}}^k$ and $\zeta_{k+1} \geq \gamma_k$, declare that the problem (7) is infeasible and terminate the algorithm.

Step 3: Update γ_k by

$$\gamma_{k+1} = \zeta_{k+1} - \|\boldsymbol{b} - \tilde{\boldsymbol{z}}_{k+1}\|^2 / (\gamma_k - \zeta_{k+1}).$$

Increase k by one and go to Step 2.

Fujishige et al. [1] prove that LPB-Newton method terminates in a finite number of iterations.

2.2 LP and convex cone

In this paper, we consider the standard form linear programming problem

$$\begin{array}{ll} \max & \boldsymbol{c}^T\boldsymbol{x}, \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{x} \geq \boldsymbol{0}, \end{array} \tag{7}$$

where $\boldsymbol{A} \in \Re^{m\times n}$, $\boldsymbol{b} \in \Re^m$ and $\boldsymbol{c} \in \Re^n$ are given data and $\boldsymbol{x} \in \Re^n$ is a variable vector.

Define the convex cone $\bar{K}$ as follows.

$$\bar{K} = \{\bar{\boldsymbol{z}} \in \Re^{m+1} |\ \bar{\boldsymbol{z}} = \bar{\boldsymbol{A}}\boldsymbol{x},\ \boldsymbol{x} \geq \boldsymbol{0}\}, \tag{8}$$

where $\bar{\boldsymbol{A}}$ is defined in (2). Then the problem (7) can be recast as

$$\begin{array}{ll} \max & \gamma, \\ \text{subject to} & \left[\begin{array}{c} \boldsymbol{b} \\ \gamma \end{array} \right] \in \bar{K} \end{array} \tag{9}$$

where $\gamma \in \Re$ is a variable.

In our algorithm, we need a procedure to find the nearest point in a convex cone from a given point. More precisely, let $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_N \in \Re^n$ be given vectors and consider the convex cone K

$$K = \{\sum_{i=1}^{N} \lambda_i \boldsymbol{e}_i |\ \lambda_i \geq 0,\ i = 1, 2, \ldots, N\}. \tag{10}$$

For a given point $\boldsymbol{q} \in \Re^n$, the problem is formulated as follows.

$$\begin{array}{ll} \min & \|\boldsymbol{q} - \boldsymbol{p}\|, \\ \text{subject to} & \boldsymbol{p} \in K \end{array} \tag{11}$$

There are many ways to solve the problem. Wilhelmsen [2] develops a combinatorial algorithm for the problem. It has a strong connection to Wolfe's algorithm. Wilhelmsen's algorithm is summarized as follows.

Wilhelmsen's algorithm

Input: Vectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_N \in \Re^n$ and a point $\boldsymbol{q} \in \Re^n$.
Output: The nearest point $\boldsymbol{p}^* \in K$ and coefficients $\lambda_1^*, \lambda_2^*, \ldots, \lambda_N^*$ which satisfy $\boldsymbol{p}^* = \sum_{i=1}^{N} \lambda_i^* \boldsymbol{e}_i$.

Step 0: If $\boldsymbol{q}^T\boldsymbol{e}_i \leq 0$ for all $i = 1, 2, \ldots, N$ then set $p^* = \boldsymbol{0}$, $\lambda_i^* = 0$ $(i = 1, 2, \ldots, N)$ and stop the algorithm. Otherwise, take an index s such that $\boldsymbol{q}^T\boldsymbol{e}_s > 0$. Set $I_1 = \{s\}$, $k = 1$ and compute $\boldsymbol{p}_k = \frac{\boldsymbol{q}^T\boldsymbol{e}_s}{\|\boldsymbol{e}_s\|^2}\boldsymbol{e}_s$. Also set $\lambda_{s,1} = \frac{\boldsymbol{q}^T\boldsymbol{e}_s}{\|\boldsymbol{e}_s\|^2}$, $\lambda_{i,1} = 0$ $(i \neq s)$.
Step 1: Set $\boldsymbol{h}_k = \boldsymbol{q} - \boldsymbol{p}_k$. If $\boldsymbol{h}_k = \boldsymbol{0}$ or $(\boldsymbol{h}_k)^T\boldsymbol{e}_i \leq 0$ for all $i = 1, 2, \ldots, N$, then $\boldsymbol{p}^* = \boldsymbol{p}_k$, $\lambda_i^* = \lambda_{i,k}$ $(i = 1, 2, \ldots, N)$ and stop the algorithm. Otherwise, find $\boldsymbol{e}_l$ such that $(\boldsymbol{h}_k)^T\boldsymbol{e}_l > 0$, set $J = I_k \cup \{l\}$.
Step 2: Find the nearest point $\boldsymbol{r}$ from $\boldsymbol{q}$ to the linear subspace spanned by vectors $\boldsymbol{e}_i$ $(i \in J)$. Let $\boldsymbol{r} = \sum_{i \in J} \beta_i \boldsymbol{e}_i$. If $\beta_i \geq 0$, $i \in J$, then set $I_{k+1} = \{i \in J : \beta_i > 0\}$, $\boldsymbol{p}_{k+1} = \boldsymbol{r}$, $\lambda_{i,k+1} = \beta_i$ $(i \in I_{k+1})$, $\lambda_{i,k+1} = 0$ $(i \notin I_{k+1})$. Increase k by one and go to Step 1. Otherwise, compute

$$\begin{aligned} \rho_i &= \begin{cases} 1 & \beta_i \geq 0 \\ \lambda_{i,k}/(\lambda_{i,k} - \beta_i) & \beta_i < 0, \end{cases} \\ \rho &= \min_{i \in J} \rho_i \end{aligned}$$

and

$$\gamma_i = (1 - \rho)\lambda_{i,k} + \rho\beta_i, \ i \in J.$$

Step 3: Update J by the set $\{i \in J : \gamma_i > 0\}$. Also set $\lambda_{i,k} = \gamma_i$ $(i \in J)$, $\lambda_{i,k} = 0$ $(i \notin J)$ and go to Step 2.

3 LPS-Newton method

3.1 A description of the algorithm

In this subsection, we give a formal description of our algorithm, LPS-Newton method for the problem (7). The algorithm starts from $(\boldsymbol{b}^T, \gamma_0)^T \in L$ where γ_0 is chosen so that it is larger than the optimal value of (7). We note that our algorithm even works if this assumption is not satisfied or the problem (7) is infeasible. In Figure 1 we show a geometrical example of LPS-Newton method. The current point $\bar{\boldsymbol{b}}_k$ is on the line L. Then $\bar{\boldsymbol{z}}_{k+1}$, the nearest point of $\bar{\boldsymbol{b}}_k$ to $\bar{K}$ is computed. The next point $\bar{\boldsymbol{b}}_{k+1}$ is obtained as the intersection of the separating hyperplane at $\bar{\boldsymbol{z}}_{k+1}$ and L, and so on. In this example, $\bar{\boldsymbol{b}}_{k+2}$ is the optimal solution of (9).

A formal description of LPS-Newton method is summarized as follows.

LPS-Newton method

Input: $\boldsymbol{A} \in \Re^{m \times n}$, $\boldsymbol{b} \in \Re^m$, $\boldsymbol{c} \in \Re^n$ and $\bar{\gamma}_0$.
Output: an optimal solution $\boldsymbol{x}^*$ of the problem (7) or the decision that

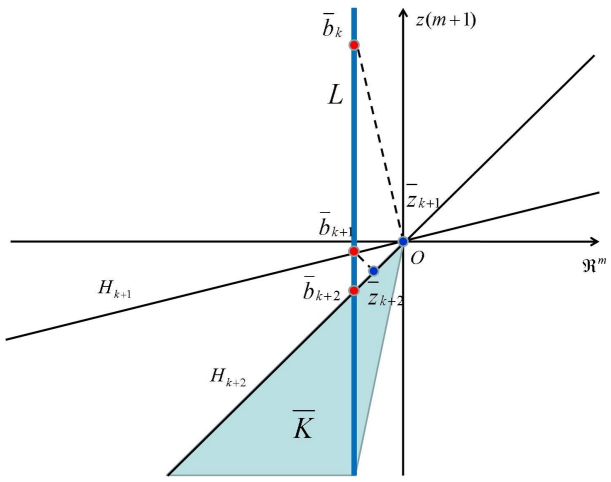

Figure 1: A geometrical example of LPS-Newton method

either the problem (7) is infeasible or the optimal value v^* is larger than γ_0.

Step 1: Set $\gamma_0 = \bar{\gamma}$ and $k = 0$.

Step 2: Set $\bar{\boldsymbol{b}}_k = (\boldsymbol{b}^T, \gamma_k)^T$. Compute the optimal solution $\tilde{\boldsymbol{z}}_{k+1}$ for the problem (11) with vectors $\bar{\boldsymbol{a}}_1, \bar{\boldsymbol{a}}_2, \ldots, \bar{\boldsymbol{a}}_n$ ($\bar{\boldsymbol{a}}_i$ is the i-th column vector of $\bar{\boldsymbol{A}}$) and the point $\bar{\boldsymbol{b}}_k$. Let $\bar{\boldsymbol{z}}_{k+1}$ can be written as $\bar{\boldsymbol{z}}_{k+1} = \bar{\boldsymbol{A}}\tilde{\boldsymbol{x}}_{k+1}$. In the vector form, we express $\bar{\boldsymbol{z}}_{k+1}$ as $(\tilde{\boldsymbol{z}}_{k+1}^T, \zeta_{k+1})^T \in \Re^m \times \Re$.
(C1) If $\tilde{\boldsymbol{z}}_{k+1} = \boldsymbol{b}$ and $\gamma_k > \zeta_{k+1}$, output $\tilde{\boldsymbol{x}}_{k+1}$ as an optimal solution and terminate the algorithm. When $k = 0$, $\tilde{\boldsymbol{x}}_1$ is an optimal solution or $v^* > \gamma_0$.
(C2) If either ($\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\zeta_{k+1} = \gamma_k$) or $\zeta_{k+1} > \gamma_k$ holds, declare that the problem (7) is infeasible and terminate the algorithm. When $k = 0$, (7) is infeasible or $v^* > \gamma_0$.

Step 3: Update γ_k by

$$\gamma_{k+1} = \zeta_{k+1} - \|\boldsymbol{b} - \tilde{\boldsymbol{z}}_{k+1}\|^2/(\gamma^k - \zeta_{k+1}). \tag{12}$$

Increase k by one and go to Step 2.

We call $\bar{\boldsymbol{b}}_k$ and $\bar{\boldsymbol{z}}_k$ the target point and the projection point, respectively.

3.2 Proof of the finite termination

Let $\bar{\boldsymbol{b}}_k = (\boldsymbol{b}^T, \gamma_k)^T$ $(k = 0, 1, \dots)$ be the sequence of target points and $\bar{\boldsymbol{z}}^k = (\tilde{\boldsymbol{z}}_k^T, \zeta_k)$ $(k = 1, 2, \dots,)$ be the sequence of projection points generated by LPS-Newton method. The next lemma is a technical result. Figure 1 might be helpful to understand the lemma.

Lemma 1 *Let k be a nonnegative integer. If LPS-Newton method does not terminate at the k-th iteration, then we have*

$$v^* \leq \gamma_{k+1} < \zeta_{k+1} < \gamma_k, \tag{13}$$

where v^* is the optimal value of the problem.

Proof: Note that $\bar{\boldsymbol{z}}_{k+1} = (\tilde{\boldsymbol{z}}_{k+1}^T, \zeta_{k+1}^T)$ is the projection point from $\bar{\boldsymbol{b}}_k = (\boldsymbol{b}^T, \gamma_k)^T$ to $\bar{K}$ and that $\bar{\boldsymbol{b}}_{k+1} = (\boldsymbol{b}^T, \gamma_{k+1})^T$ is the intersection point of L and the hyperplane

$$\{\bar{\boldsymbol{z}} \in \Re^{m+1} |\ (\bar{\boldsymbol{b}}_k - \bar{\boldsymbol{z}}_{k+1})^T(\bar{\boldsymbol{z}} - \bar{\boldsymbol{z}}_{k+1}) = 0\},$$

which separates $\bar{\boldsymbol{b}}_k$ and $\bar{K}$. There are four possibilities, namely

(i) $\tilde{\boldsymbol{z}}_{k+1} = \boldsymbol{b}$ and $\gamma_k < \zeta_{k+1}$,

(ii) $\tilde{\boldsymbol{z}}_{k+1} = \boldsymbol{b}$ and $\gamma_k \geq \zeta_{k+1}$,

(iii) $\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\gamma_k < \zeta_{k+1}$,

(iv) $\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\gamma_k \geq \zeta_{k+1}$.

Since the algorithm does not terminate at the k-th iteration, the conditions **(C1)** and **(C2)** do not hold. Thus we see that (iv) holds and **(C2)** excludes the case $\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\gamma_k = \zeta_{k+1}$. In summary, we have $\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\gamma_k > \zeta_{k+1}$. For any $\bar{\boldsymbol{z}} = (\tilde{\boldsymbol{z}}^T, \delta)^T \in \bar{K}$, we have

$$(\bar{\boldsymbol{b}}_k - \bar{\boldsymbol{z}}_{k+1})^T(\bar{\boldsymbol{z}} - \bar{\boldsymbol{z}}_{k+1}) \leq 0. \tag{14}$$

If $\bar{\boldsymbol{z}} = (\boldsymbol{b}^T, \delta)^T \in \bar{K} \cap L$, since $\gamma_k > \zeta_{k+1}$, (14) is equivalent to

$$\delta \leq \zeta_{k+1} - \|\boldsymbol{b} - \tilde{\boldsymbol{z}}_{k+1}\|^2/(\gamma_k - \zeta_{k+1}) = \gamma_{k+1}. \tag{15}$$

The inequality (15) holds for $(\boldsymbol{b}^T, v^*)^T \in \bar{K} \cap L$. From this observation and $\gamma_k > \zeta_{k+1}$, we have

$$v^* \leq \gamma_{k+1} < \zeta_{k+1} < \gamma_k.$$

■

Corollary 1 *If the problem (7) has an optimal solution and the optimal value v^* is larger than γ_0, LPS-Newton method terminates at the 0-th iteration.*

Proof: Assume that the problem (7) has an optimal solution with the optimal value $v^* > \gamma_0$ and LPS-Newton method does not terminate at the 0-th iteration. Then from (13) we have

$$\gamma_0 < v^* \leq \gamma_1 < \zeta_1 < \gamma_0,$$

which is a contradiction. ■

Next we address the case where the condition **(C1)** is true.

Lemma 2 *Let k be a positive integer. If* **(C1)** *holds at the k-th iteration of LPS-Newton method, then $\tilde{\boldsymbol{x}}_{k+1}$ is an optimal solution of the problem (7). In addition, if* **(C1)** *holds at the 0-th iteration, then either $\tilde{\boldsymbol{x}}_1$ is an optimal solution or $v^* > \gamma_0$.*

Proof: Since the algorithm does not terminate at the $(k-1)$-th iteration, we have $v^* \leq \gamma_k$ from Lemma 1. Thus the statement readily holds if $\gamma_k = \zeta_{k+1}$. When $k = 0$, we see that (7) has the feasible solution $\tilde{\boldsymbol{x}}_1$ with the objective value γ_0. So we conclude that $\tilde{\boldsymbol{x}}_1$ is an optimal solution or $v^* > \gamma_0$.

If $\gamma_k > \zeta_{k+1}$, from (15), we have $v^* \leq \zeta_{k+1}$, which proves the statement. Note the conclusion also holds when $k = 0$. ■

In the next lemma, we address the case when the condition **(C2)** is true. We give Figure 2 to understand the situation.

Lemma 3 *Let k be a positive integer. If* **(C2)** *holds at the k-th iteration of LPS-Newton method, then the problem is infeasible. In addition, if* **(C2)** *holds at the 0-th iteration of LPS-Newton method, either the problem (7) is infeasible or $v^* > \gamma_0$*

Proof: For the case of $\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\zeta_{k+1} = \gamma_k$, from (14) for $\bar{\boldsymbol{z}} = (\tilde{\boldsymbol{z}}^T, \delta)^T \in \bar{K}$ we have

$$(\boldsymbol{b} - \tilde{\boldsymbol{z}}_{k+1})^T(\tilde{\boldsymbol{z}} - \tilde{\boldsymbol{z}}_{k+1}) \leq 0,$$

which is not true for any point on L, so (7) is infeasible. Note that the conclusion also holds the case $k = 0$.

We show the second case. In this case, $\gamma_k < \zeta_{k+1}$ holds and from (14) we have

$$\delta \geq \zeta_{k+1} - \|\boldsymbol{b} - \tilde{\boldsymbol{z}}_{k+1}\|^2/(\gamma_k - \zeta_{k+1}) = \gamma_{k+1}$$

for $(\boldsymbol{z}^T, \delta)^T \in \bar{K} \cap L$. Thus we have

$$v^* \geq \gamma_{k+1} \geq \zeta_{k+1} > \gamma_k, \tag{16}$$

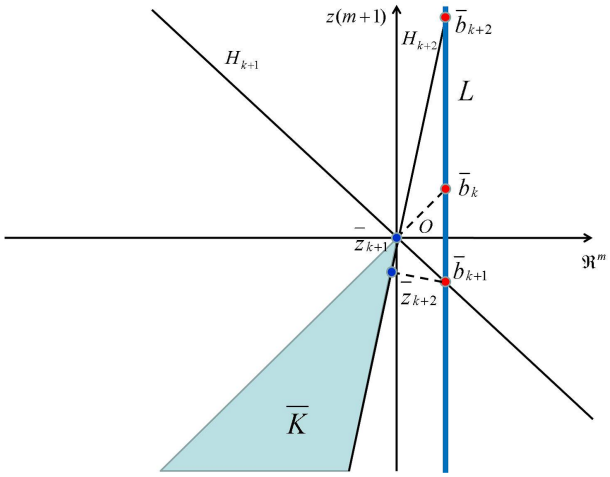

Figure 2: LPS-Newton method applied to infeasible LP

which contradicts $v^* \leq \gamma_k$. When $k = 0$, (16) implies that $v^* > \gamma_0$ if (7) is feasible. ■

Theorem 1 *The number of iterations of LPS-Newton method is at most the number of faces of the cone* $\bar{K}$.

Proof: Assume, to the contrary, the number of iterations of LPS-Newton method exceeds the number of faces of $\bar{K}$. Then there exists a face S of $\bar{K}$ such that two projection points $\bar{\boldsymbol{z}}_{k_1}$ and $\bar{\boldsymbol{z}}_{k_2}$ $(k^1 < k^2)$ belong to the relative interior of S. In the following, for a positive integer k, we set $\boldsymbol{p}_k = \bar{\boldsymbol{b}}_{k-1} - \boldsymbol{z}_k$ and express the separating hyperplane H^k of $\bar{K}$ at $\bar{\boldsymbol{z}}^k$ as $H_k = \{\bar{\boldsymbol{z}} \in \Re^{m+1} | \ \boldsymbol{p}_k^T \bar{\boldsymbol{z}} = \alpha_k\}$. Then we have $S \subset H_{k_1} \cap H_{k_2}$. Note that since the algorithm never ends, $\tilde{\boldsymbol{z}}_{k+1} \neq \boldsymbol{b}$ and $\gamma_k > \zeta_{k+1}$ hold at each iteration. From this observation and Lemma 1, we see that γ_k is strictly decreasing. These facts lead to the following two inequalities.

$$\boldsymbol{p}_{k_1}^T \bar{\boldsymbol{b}}_{k_2-1} \leq \alpha_{k_1} \tag{17}$$

and

$$\boldsymbol{p}_{k_2}^T \bar{\boldsymbol{b}}_{k_1-1} > \alpha_{k_2}. \tag{18}$$

From the inequality (17) and $\boldsymbol{p}_{k_1}^T \bar{\boldsymbol{z}}_{k_2} = \alpha_{k_1}$, we have

$$\boldsymbol{p}_{k_1}^T \boldsymbol{p}_{k_2} = \boldsymbol{p}_{k_1}^T (\bar{\boldsymbol{b}}_{k_2-1} - \bar{\boldsymbol{z}}_{k_2}) \leq 0 \tag{19}$$

But, from (19) and $\boldsymbol{p}_{k_2}^T \bar{\boldsymbol{z}}_{k_1} = \alpha_{k_2}$, we have

$$\begin{aligned} \boldsymbol{p}_{k_2}^T \bar{\boldsymbol{b}}_{k_1-1} - \alpha_{k_2} &= \boldsymbol{p}_{k_2}^T (\bar{\boldsymbol{b}}_{k_1-1} - \bar{\boldsymbol{z}}_{k_1}) \\ &= \boldsymbol{p}_{k_1}^T \boldsymbol{p}_{k_2} \leq 0, \end{aligned}$$

which contradicts the inequality (18). Thus LPS-Newton method terminates in a finite number of iterateions. ■

3.3 A comparison of the two algorithms

Here we compare LPB-Newton method with LPS-Newton method.

- Formulation: Fujishige et al. address LP with bounded constraints (1) and they reformulate the problem by introducing a related zenotope $\bar{Z} \subset \Re^{m+1}$ (3). In contrast, we address the standard form LP (7) and recast the problem using a related convex cone $\bar{K} \subset \Re^{m+1}$ (8).

- Projection algorithm: For projections in the algorithms, LPB-Newton method adopts Wolfe's algorithm. To implement LPS-Newton method, we can use, for example Wilhelmsen's algorithm.

- Finite termination: Both algorithms terminates in a finite number of iterations.

4 Conclusion

In this paper, we propose LPS-Newton method for the standard form linear programming problem (7). The development of LPS-Newton method is motivated by LPB-Newton method by Fujishige et al. [1]. They reformulate the LP with bound constraints (1) by introducing the zonotope $\bar{Z}$ (3), which has at most 2^n vertices. On the other hand, we recast the problem (7) using the convex cone (8) which is constructed from at most n rays. Each algorithm repeats a projection to the zonotope or the convex cone and finds an optimal solution. We expect that the number of iterations of LPS-Newton method is smaller than that of LPB-Newton method, since the number of faces of the convex cone is much smaller than that of the zonotope. As a future work, we would like to implement our algorithm and compare the two algorithm.

Acknowledgment
This research is supported in part by Grant-in-Aid for Young Scientists (B) 23710164 and Grant-in-Aid for Science Research (A) 20241038 of Japan Society for the Promotion of Science.

References

[1] S. Fujishige, T. Hayashi, K. Yamashita and U. Zimmermann, Zonotopes and the LP-Newton method, *Optimization and Engineering* 10 (2009) 193–205.

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

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