New Development of the Inclusive-cone-based Method for Linear Optimization

by

Ding, Ming-Fang

B.Sc.(Hons) (Shanxi U.), M.Sc. (Southwest Jiaotong U.)

Dissertation submitted in fulfilment of the requirements for the Doctor of Philosophy

School of Mathematical and Geospatial Sciences

RMIT University

Melbourne

Australia

February 7, 2014

DECLARATION

The candidate hereby declares that the work in this thesis, presented for the award of the Doctor of Philosophy and submitted in the School of Mathematical and Geospatial Sciences, RMIT University:

- **has been done by the candidate alone and has not been submitted previously, in whole or in part, in respect of any other academic award and has not been published in any form by any other person except where due reference is given, and**
- **has been carried out under the supervision of Dr Yanqun Liu**

....

Ding, Ming-Fang

Certification

This is to certify that the above statements made by the candidate are correct to the best of our knowledge.

....

Dr Yanqun Liu

Supervisor

Acknowledgements

I welcome this opportunity to thank those people who provided me with help and support throughout my candidature for PhD degree. Without them, this dissertation would not have been completed.

First and foremost, I would like to express my heartfelt gratitude to my supervisors, Dr Yanqun Liu and Dr John Gear, for their helpful guidance, constructive suggestions and encouragement.

I would also like to extend my sincere appreciation to my friend Miss Linda Nguyen for her friendship and helping install necessary study software on my laptop, which greatly facilitated my research.

Last but not least, I would like to express my thanks and love to my whole family. I am truly fortunate to have them as my families. Thank my parents and my sister for their unconditional love, support and constant encouragement, which helped me to overcome the difficulties and to get through this lengthy challenging journey. I would also say thanks to my auntie for her helpful advice whenever I needed it.

Contents

List of Figures

List of Tables

List of Publications

Publications relative to the research of this thesis:

M.-F. Ding, Y. Liu and J. A. Gear. An improved targeted climbing algorithm for linear programs. *Numerical Algebra, Control and Optimization*, 1(3): 399-405, 2011.

M.-F. Ding, Y. Liu and J. A. Gear. A modified centered climbing algorithm for linear programming. *Applied Mathematics*, 3: 1423-1429, 2012.

Y. Liu and M.-F. Ding. A ladder method for linear semi-infinite programming. *Journal of Industrial and Management Optimization.* Accepted for publication, to appear in Volume 10, No. 2, April 2014.

M.-F. Ding, Y. Liu and J. A. Gear. An inclusive-cone-based solvability criterion for linear programming. In preparation for submission.

Other publications:

M.-F. Ding, J. A. Gear and Y. Ding. Thermal storage and insulation properties of a diver dry suit. *ANZIAM Journal*, 51 (EMAC2009): C625-C639, 2010.

Summary

The purpose of this dissertation is to present a simple method for linear optimization including linear programming (LP) and linear semi-infinite programming (LSIP), which is termed “the inclusive-cone-based method”.

Using the inclusive cone as an analytic tool, theoretical aspects of linear programming are investigated. Sensitivity analysis in linear programming is examined from the perspective of an inclusive cone. The relationship of inclusiveness between correlated linear programming problems is also studied.

New inclusive-cone-based ladder algorithms are proposed to solve linear programming problems in inequality form. Numerical experiments are implemented to show effectiveness and efficiency of the new linear programming ladder algorithms.

To start the ladder method for LP, a single artificial constraint technique is introduced to find an initial ladder. Further, in the context of a new category of linear programming problems, an inclusive-cone-based solvability criterion is established to distinguish that a linear programming problem is inclusive-feasible (i.e., optimal), noninclusive-feasible (i.e., unbounded), inclusive-infeasible or noninclusive-infeasible.

The inclusive-cone-based method for linear programming is also generalized to linear semi-infinite programming. An optimality result, based upon the concept of the generalized base point, is established. With this optimality result as a theoretical foundation, a ladder algorithm for solving linear semi-infinite programming problems is developed. The new algorithm has several features: at each iteration it only deals with a small fraction of constraints; at each iteration it selects a constraint most violated along a “parameterized centerline”, by solving a one-dimensional global optimization problem using the efficient bridging algorithm; at each iteration the selection of the incoming constraint has a great degree of freedom, which is controlled by a parameter arising in the global optimization problem; it can detect infeasibility and unboundedness after a finite number of iterations; it obviates extra work for feasibility verification as it handles feasibility and optimality simultaneously. A simple convergent result is presented. Numerical behavior of the algorithm is examined on several test problems.

Chapter 1

Introduction

In this dissertation, we deal with linear optimization problems in the following form

$$(\mathrm{P_{LO}}) \qquad \begin{array}{ll} \min & c^T x \\ \text{s.t.} & a(t)x \le b(t),\ t \in T \end{array}$$

where $x \in R^n$ is the vector of decision variables, $c \in R^n$ ($c \neq 0$) is a given vector of objective function coefficients, T is a given index set, $a : T \to R^n$ and $b : T \to R$ are given functions. The above formulation includes both linear programming (LP) and linear semi-infinite programming (LSIP) problems. When T is a finite set, say $T = \{t_1, t_2, \cdots, t_m\}$, problem ($\mathrm{P_{LO}}$) represents a linear programming problem, which is conventionally expressed as

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \le b, \end{array}$$

where

$$A := \begin{bmatrix} a(t_1) \\ a(t_2) \\ \vdots \\ a(t_m) \end{bmatrix},\ b := \begin{bmatrix} b(t_1) \\ b(t_2) \\ \vdots \\ b(t_m) \end{bmatrix}.$$

If T is an infinite set, problem ($\mathrm{P_{LO}}$) is a linear semi-infinite programming problem.

Traditionally, linear programming and linear semi-infinite programming are dealt with differently and separately, both in theory and in computation. In this dissertation we present a recently developed method, namely, the inclusive-cone-based method, for both LP and LSIP. Our purpose is twofold. Firstly, we give a systematic summary of published work relevant to the inclusive-cone-based method, including both theoretical and algorithmic aspects, for both LP and LSIP. Secondly, we present new theoretical and computational results for LP and LSIP that have been developed during my PhD studies.

LP is the most fundamental class of optimization problems. It is also one of the most useful modeling paradigms. LP has been extensively applied to modeling real-world problems arising in economics, engineering, science, military and many other fields (see Gaitsgory and Quincampoix [25], Finlay et al. [22], Moghaddam and Michelot [72] and Luenberger and Ye [69]). Furthermore, there are more challenging applications that remain undiscovered. The prevalence of LP, to a great extent, is ascribable to the important role it serves as an aiding tool in constructing algorithms for more complex classes of optimization problems like integer programming and combinatorial optimization problems.

LP has been studied for several decades. Early in 1939, a special LP problem dealing with the organization and planning of production was formulated and solved by the Soviet mathematician and economist L. V. Kantorovich. However, it was not until 1947 that the general LP problems were formulated, and the first computationally efficient LP algorithm known as the primal simplex algorithm was developed, due to Dantzig's contribution [14]. Ever since, many a simplex variant has been proposed to advance Dantzig's idea. For a long time since the birth of Dantzig's algorithm, it was believed that the worst case complexity of the simplex method was polynomial. Eventually this surmise was overturned by Klee and Minty, who constructed the famous Klee-Minty cube to demonstrate that in the worst case the simplex method suffered from an exponential time complexity (see Klee and Minty [52]). The year 1979 witnessed the first LP algorithm with polynomial time complexity, that is, Khachiyan's ellipsoid algorithm

[51]. Despite of its high theoretical value, the ellipsoid algorithm performs poorly in practice. In 1984, Karmarkar [50] proposed an interior point algorithm with polynomial time complexity using a sequence of projective transformations of a polytope, which is regarded as the first practically efficient LP algorithm solving LP problems in polynomial time. Since then, interior point methods (IPMs) have been prospering in the LP community. To date, while a myriad of LP algorithms have been proposed, the simplex method and the interior point methods are still the most favorable methods for solving LP problems.

Unfortunately, any currently-existing solution method, even the most successful simplex method and the interior point methods, is only suitable for solving a certain type of LP problems. For example, the simplex method has superiority for solving small- and medium-sized LP problems, whereas the interior point methods win out for large-scale and sparse LP problems. Furthermore, it remains unknown whether there exists a strongly polynomial time LP algorithm. Due to these reasons, extensive research is still in progress to pursue LP algorithms which have better numerical performance.

On the other hand, linear semi-infinite programming, as the extension of LP, is the other case we would like to give special attention in this dissertation. In contrast to LP dealing with a finite number of variables and constraints, the LSIP we consider in this dissertation involves infinitely many constraints and finitely many variables. Since the first papers on LSIP were published in the early 1960's (see Charnes et al. [10, 11]), LSIP has been receiving researchers' attention for at least three reasons as stated by Goberna [28]: *First, for its real life and modeling applications. Second, for providing nontrivial but tractable optimization problems on which it is possible to check more general theories and methods. Finally, LSIP can be seen as a theoretical model for large scale LP problems.*

In this dissertation, we present a simple method for linear optimization (including linear programming and linear semi-infinite programming), termed "the inclusive-cone-based method". As the term "inclusive-cone-based" suggests, the method presented in this dissertation for both LP and LSIP is based on the concept of the inclusive cone. The concept of the inclusive cone was first introduced by Liu [60] for LP. Introduction of this concept resulted in the generalization of boundedness concept, and further led to a "symmetric" inclusive-cone-based strong duality theorem in the context of a new category of LP problems (see Liu [61]), which not only completes the classical strong duality theorem but also establishes the equivalent relation between primal and dual problems. The ladder method for solving LP problems (see Liu [60]) also builds upon this concept. Motivated by the aforementioned work, in this dissertation, we make further and more in-depth investigation into the inclusive-cone-based method for LP. More importantly, we generalize this approach to the linear semi-infinite programming case. Specifically speaking, what is done in this dissertation are the following:

- Examination of sensitivity analysis in LP from the perspective of an inclusive cone.
- Investigation of the relationship of inclusiveness between correlated LP problems.
- Development of new inclusive-cone-based ladder algorithms for solving LP problems with inequality constraints.
- Introduction of an initialization technique for starting the ladder method.
- Development of an inclusive-cone-based solvability criterion for LP in the context of a new category of LP problems.
- Establishment of an optimality result for LSIP, which is based upon the concept of the generalized base point.
- Development of an inclusive-cone-based ladder algorithm for solving linear semi-infinite programming problems.

All the work done in this dissertation centers around the core concept of an inclusive cone. Our goal is to provide a graphically-intuitive approach with a broad simplification and easy understanding for both LP and LSIP, which is especially suitable for classroom teaching.

This dissertation consists of two major parts: linear programming and linear semi-infinite programming. In the initial part of this dissertation, which encompasses Chapter 2–Chapter 5, we focus our attention on LP. In Chapter 2, we introduce the concept of an inclusive cone and the resulting fundamental LP theory including duality results. We then investigate the relationship of inclusiveness between correlated LP problems. We also examine sensitivity analysis in LP using an inclusive-cone-based approach. Chapter 3 gives an overview of algorithms for solving LP problems. This chapter ends with the detailed description of the ladder method, which is a basis of the subsequent algorithms to be proposed. In Chapter 4, we develop several new inclusive-cone-based ladder algorithms for solving LP problems with inequality constraints. In Chapter 5, after giving an overview of some initialization methods for starting the ladder method, we introduce a new initialization technique. Based on this technique, we further develop an inclusive-cone-based solvability criterion for LP. In Chapter 6, we generalize the inclusive-cone-based method for LP to the LSIP case. We first develop a fundamental theorem with regard to optimality for LSIP, which is an LSIP version of inclusive-cone-based optimality condition for LP. With this optimality result as a theoretical foundation, we propose a linear semi-infinite programming algorithm with convergent properties. Finally, in Chapter 7, we summarize the content and the contributions of this dissertation, concluding with a brief discussion on possible future lines of research.

Chapter 2

Fundamentals of LP Based on Inclusive Cone

2.1 The Linear Programming Problem and Conversion between its Different Forms

A linear programming (LP) problem is an optimization problem dealing with minimizing (or maximizing) a linear objective function which is subject to a set of linear equality and/or inequality constraints or restrictions. It can be formulated as any kind of the following different forms.

2.1.1 Forms of LP Problems

General form

(P$_{\text{G}}$):

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \le b \\ & Dx = g, \end{aligned}$$

where $x = [x_1; x_2; \cdots ; x_n] \in R^n$ is the vector of decision variables, $A \in R^{m\times n}$ and $D \in R^{r\times n}$ are constraint matrices, $c = [c_1; c_2; \cdots ; c_n] \in R^n$ $(c \ne 0)$ is the vector of objective function coefficients, $b \in R^m$ and $g \in R^r$ are right-hand-side (RHS) vectors. Here and the remaining of this dissertation, a vector with its elements separated by semicolons represents a column vector.

Inequality form

(P):
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \le b. \end{aligned}$$

Standard form

($\mathrm{P_S}$):
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Dx = g \\ & x \ge 0. \end{aligned}$$

Canonical form

($\mathrm{P_C}$):
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \le b \\ & x \ge 0. \end{aligned}$$

2.1.2 Conversion between Different Forms

An LP problem can be transformed from one form to another equivalent form by simple manipulations. Here we take the typical reduction of general form to inequality/standard form for examples to show how to convert one form to another.

Reduction of general form to inequality form

Consider problem ($\mathrm{P_G}$). By writing equations of the form $Dx = g$ as the equivalent inequalities $Dx \le g$ and $-Dx \le -g$, the above general form is reduced to the equivalent inequality form

($\mathrm{P'_G}$):
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & \begin{bmatrix} A \\ D \\ -D \end{bmatrix} x \le \begin{bmatrix} b \\ g \\ -g \end{bmatrix}. \end{aligned}$$

Reduction from general form to standard form

Consider problem ($\mathrm{P_G}$). Let $x = x^+ - x^-$, where $x^+, x^- \ge 0$. By introducing a vector of slack variables $s = [s_1; s_2; \cdots; s_m]$, the general LP form ($\mathrm{P_G}$) can be reduced to the standard form

$$(\mathrm{P}''_{\mathrm{G}}): \quad \min \begin{bmatrix} c^T & -c^T & 0 \end{bmatrix} \begin{bmatrix} x^+ \\ x^- \\ s \end{bmatrix}$$

$$\text{s.t.} \begin{bmatrix} A & -A & I_m \\ D & -D & 0 \end{bmatrix} \begin{bmatrix} x^+ \\ x^- \\ s \end{bmatrix} = \begin{bmatrix} b \\ g \end{bmatrix}$$

$$x^+, x^-, s \geq 0.$$

In the following and subsequent chapters for LP part of this dissertation, we focus our attention on LPs in inequality form (see problem (P) as above). We are concerned with problems in which the number of constraints is greater than or equal to that of variables and constraint matrices are of full column rank.

2.2 Inclusiveness, Boundedness, Feasibility, Optimality and Degeneracy

In terms of solution state, LP problems are usually divided into three classes:

- Problems with optimal solutions. If a problem has feasible solutions and the objective function values are bounded, then it has optimal solutions.
- Unbounded problems. A problem is called unbounded if it has feasible solutions with unbounded objective function values.
- Infeasible problems. We call an LP problem infeasible if it has no feasible solution.

We display Figure 2.1 for intuitive demonstration and subsequent comparison. As seen from the above as well as Figure 2.1, the concept of boundedness is only applicable to feasible case. Liu [61] introduced the concept of inclusiveness, consequently extending the concept of boundedness to infeasible problems. This concept is based on the concept of an inclusive cone, which we present as follows.

Definition 2.1. [60] Consider problem (P). Denote by $\mathcal{J} = \{1, 2, \cdots, m\}$ the constraint index set of problem (P). Let $J = \{j_1, j_2, \cdots, j_k\} \subset \mathcal{J}$ be an ordered

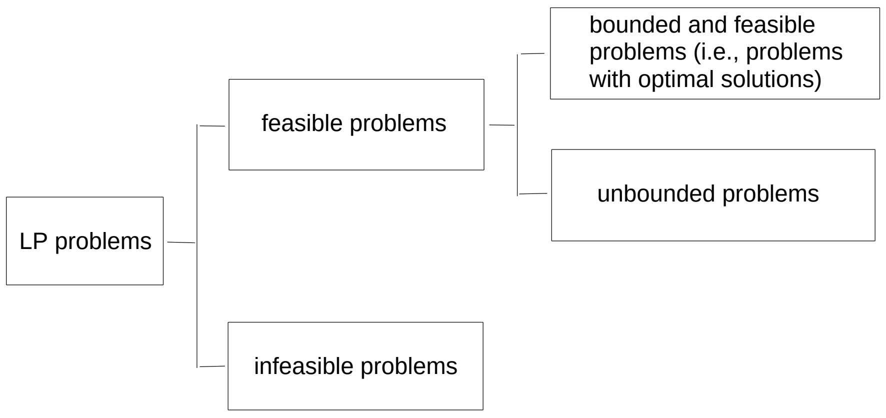
LP problems
feasible problems
infeasible problems
bounded and feasible problems (i.e., problems with optimal solutions)
unbounded problems

Figure 2.1: Classification of LP problems

subset. If the outward normal vectors of constraints corresponding to J are independent, then J is called independent. For an ordered independent subset J with $k = n$, if the convex cone spanned by its corresponding n linearly independent outward normal vectors

$$a_{j_1}^T, a_{j_2}^T, \cdots, a_{j_n}^T,$$

where $a_{j_i}^T$ $(1 \leq i \leq n)$ is the outward normal vector of the constraint $a_{j_i}x \leq b_{j_i}$ pointing away from its feasible side, contains the vector $-c$, it is said to be an inclusive cone generated by J. The generated cone is denoted by $N(J)$. If J generates an inclusive cone, the set defined by

$$L(J) = \{x \in R^n : a_j x \leq b_j, \text{for } j \in J\}$$

is called the inclusive region or the ladder associated with J. The corresponding ordered index set J is called the generator of the ladder $L(J)$. The unique solution of the linear system $a_{j_i}x = b_{j_i},\ i = 1, 2, \cdots, n$, denoted by x_J, is called the base point of the ladder $L(J)$.

We take the following two-dimensional problem for example to illustrate aforementioned concepts.

Example 2.1. Consider the problem

$$\text{(C1):} \quad \begin{aligned} \min \quad & x_2 \\ \text{s.t.} \quad & -x_1 - x_2 \le 0 \\ & x_1 - x_2 \le 0 \\ & x_1 - x_2 \le -2. \end{aligned}$$

The graph of this problem is displayed in Figure 2.2, where we put arrows at both ends of each line to indicate the feasible side of the corresponding constraint and use the red arrow to represent the outward normal vector of the corresponding constraint. (For the sake of elegance of the graphic display, we chose this special example. Note that the particularity of this example causes the outward normal vector of each constraint to be coincidentally parallel to some hyperplane.)

As shown in Figure 2.2, the problem has two inclusive cones: one is spanned by outward normal vectors of constraints $x_1 - x_2 \le 0$ and $-x_1 - x_2 \le 0$, the other by outward normal vectors of constraints $x_1 - x_2 \le -2$ and $-x_1 - x_2 \le 0$. The above problem has two ladders (inclusive regions), which are those shaded areas in Figure 2.2. x^0 and x^1 are base points associated with the ladders (inclusive regions).

We see from Figure 2.2 that the base point x^1 is as well the optimal solution of problem (C1). The relation between base points and optimal solutions was dealt with in [60]. We state the main result as follows.

Theorem 2.1. [60] (Optimality Condition) Consider problem (P). A point $x^* \in R^n$ is an optimal vertex solution of problem (P) if and only if x^* is the feasible base point of some ladder of problem (P).

Careful readers may have noticed that the concept of the inclusive cone given in the foregoing was developed for problems involving "$\le$"-type constraints. Liu [61] further extended this concept to the problems containing both inequality and equality constraints (see problem ($\mathrm{P_G}$)).

Definition 2.2. ([61]) Consider problem ($\mathrm{P_G}$) and its equivalent problem ($\mathrm{P'_G}$). An inclusive cone of problem ($\mathrm{P'_G}$) is called an inclusive cone of problem ($\mathrm{P_G}$).

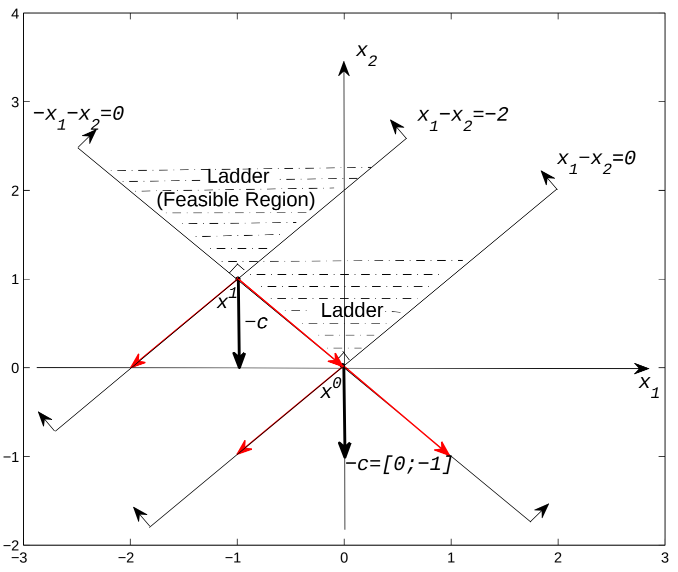

Figure 2.2: An LP example

Accordingly, we can also define the inclusive cone for problem ($\mathrm{P_S}$) in standard form (resp., problem ($\mathrm{P_C}$) in canonical form). The only work we need to do is to convert problem ($\mathrm{P_S}$) (resp., problem ($\mathrm{P_C}$)) into its equivalent problem with only "$\leq$"-type constraints, denoted by ($\mathrm{P_S'}$) (resp., ($\mathrm{P_C'}$)):

$$(\mathrm{P_S}): \quad \begin{array}{rl} \min & c^T x \\ \text{s.t.} & Dx = g \\ & x \geq 0 \end{array} \quad \Leftrightarrow \quad (\mathrm{P_S'}): \quad \begin{array}{rl} \min & c^T x \\ \text{s.t.} & Dx \leq g \\ & -Dx \leq -g \\ & -x \leq 0 \end{array}$$

$$(\mathrm{P_C}): \quad \begin{array}{rl} \min & c^T x \\ \text{s.t.} & Ax \leq b \\ & x \geq 0 \end{array} \quad \Leftrightarrow \quad (\mathrm{P_C'}): \quad \begin{array}{rl} \min & c^T x \\ \text{s.t.} & Ax \leq b \\ & -x \leq 0 \end{array}$$

Then the inclusive cone of problem ($\mathrm{P_S'}$) (resp., problem ($\mathrm{P_C'}$)) is called the inclusive cone of problem ($\mathrm{P_S}$)) (resp., problem ($\mathrm{P_C}$)).

On the basis of the concept of the inclusive cone, we now present the concept of inclusiveness.

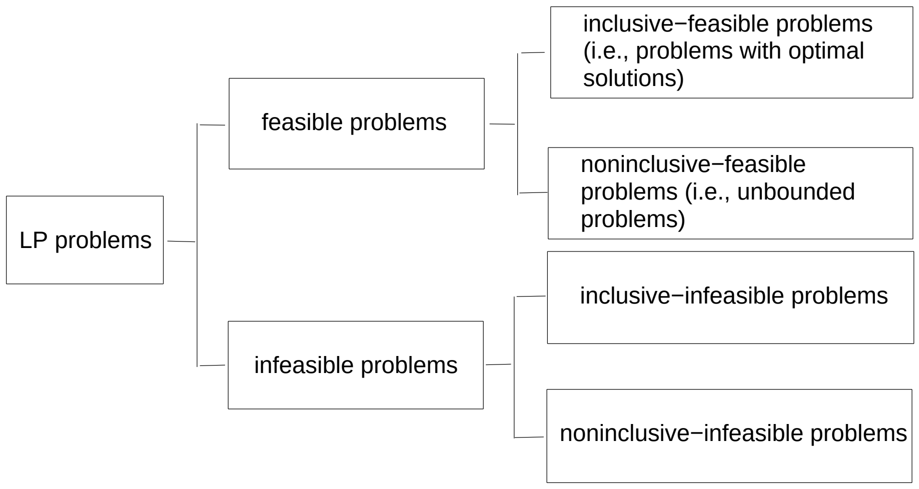
LP problems
feasible problems
infeasible problems
inclusive−feasible problems (i.e., problems with optimal solutions)
noninclusive−feasible problems (i.e., unbounded problems)
inclusive−infeasible problems
noninclusive−infeasible problems

Figure 2.3: New classification of LP problems involving inclusiveness

Definition 2.3. ([61]) A linear programming problem with at least one inclusive cone is called inclusive. We call a linear programming problem noninclusive if it is not inclusive.

By virtue of inclusiveness concept, Liu [61] extended the boundedness concept to infeasible problems, refining LP problems into four classes (see Figure 2.3):

- Inclusive-feasible problems (equivalently speaking, problems with optimal solutions). If an LP problem has feasible solutions and has at least one inclusive cone, then it is called inclusive-feasible (i.e., optimal).
- Noninclusive-feasible problems (equivalently speaking, unbounded problems). An LP problem is called noninclusive-feasible if the problem has feasible solutions and no inclusive cone.
- Inclusive-infeasible problems. We call an LP problem inclusive-infeasible if the problem has no feasible solution and has at least one inclusive cone.
- Noninclusive-infeasible problems. An LP problem is called noninclusive-infeasible if the problem has neither a feasible solution nor an inclusive cone.

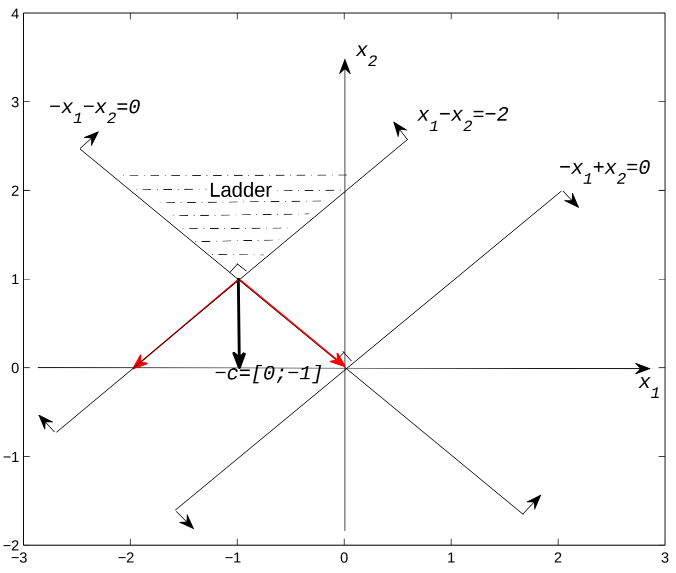

Figure 2.4: An inclusive-infeasible LP example

Note that infeasible problems are further divided into inclusive-infeasible and noninclusive-infeasible problems. We look at four examples for illustration.

First we revisit Example 2.1. This problem is inclusive-feasible, i.e., this problem has an optimal solution. Indeed, from Figure 2.2, feasibility is obvious. In addition, since the cone generated by outward normal vectors of constraints $x_1 - x_2 \leq 0$ and $-x_1 - x_2 \leq 0$ contain $-c = [0; -1]$, this problem is inclusive.

Example 2.2. This problem is inclusive-infeasible.

(C2):
$$\begin{aligned} \min \quad & x_2 \\ \text{s.t.} \quad & -x_1 - x_2 \leq 0 \\ & -x_1 + x_2 \leq 0 \\ & x_1 - x_2 \leq -2. \end{aligned}$$

The above problem is shown geometrically in Figure 2.4. Inconsistency of the second and third constraints implies the problem is infeasible. On the other hand, from the fact that the cone generated by outward normal vectors of constraints $-x_1 - x_2 \leq 0$ and $x_1 - x_2 \leq -2$ contains $-c = [0; -1]$, we know that this problem is inclusive.

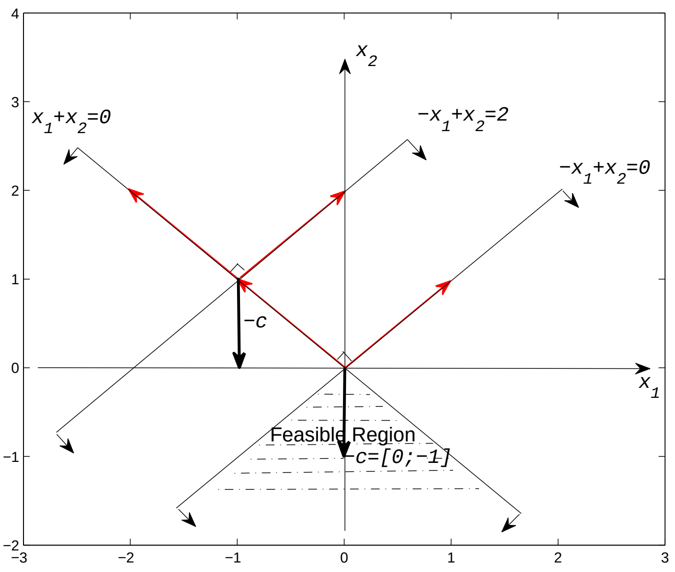

Figure 2.5: A noninclusive-feasible (unbounded) LP example

Example 2.3. This problem is noninclusive-feasible (i.e., unbounded).

(C3):
$$\begin{aligned} \min \quad & x_2 \\ \text{s.t.} \quad & x_1 + x_2 \le 0 \\ & -x_1 + x_2 \le 0 \\ & -x_1 + x_2 \le 2. \end{aligned}$$

The graphical representation of this problem is displayed in Figure 2.5. As seen in the figure, no cone generated by outward normal vectors of any two constraints contains $-c = [0; -1]$, which implies this problem is noninclusive. On the other hand, feasibility of this problem is clear. Hence the problem is noninclusive-feasible (i.e., unbounded).

Example 2.4. This problem is noninclusive-infeasible.

(C4):
$$\begin{aligned} \min \quad & x_2 \\ \text{s.t.} \quad & x_1 + x_2 \le 0 \\ & -x_1 + x_2 \le 0 \\ & x_1 - x_2 \le -2. \end{aligned}$$

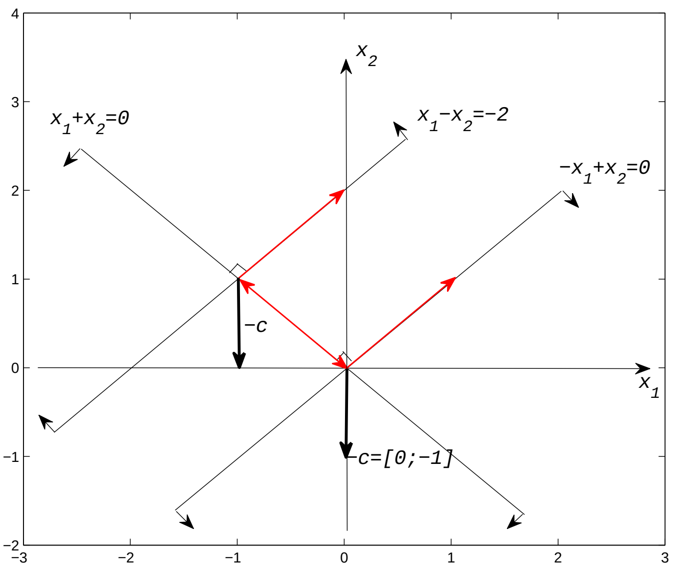

Figure 2.6: A noninclusive-infeasible LP example

This problem is shown geometrically in Figure 2.6, where the red arrow with double arrowheads indicates two overlapped outward normal vectors with opposite direction: one for constraint $-x_1 + x_2 \leq 0$ and the other for constraint $x_1 - x_2 \leq -2$. As seen from this figure, the problem is noninclusive-infeasible.

We conclude this section with the following definitions.

Definition 2.4. [60] The set defined by

$$E_k(J) = \{x : x \in L(J), a_{j_i}x \leq b_{j_i}, 1 \leq i \leq n, i \neq k\}$$

is called the k-th edge of the ladder $L(J)$, where $J = \{j_1, j_2, \cdots, j_n\}$.

From the above definition, it is clear that a ladder has n edges.

Definition 2.5. [60] A ladder $L(J)$ of problem (P) is said to be degenerate if at least one of its n edges is normal to the vector c. Problem (P) is said to be non-degenerate if it does not have a degenerate ladder.

As shown in Figure 2.2, Example 2.1 is non-degenerate since it has no degenerate ladder. If we replace the last constraint $x_1 - x_2 \leq -2$ by $-x_2 \leq -2$, the resulting problem has a degenerate ladder constructed from constraints $-x_1 - x_2 \leq 0$

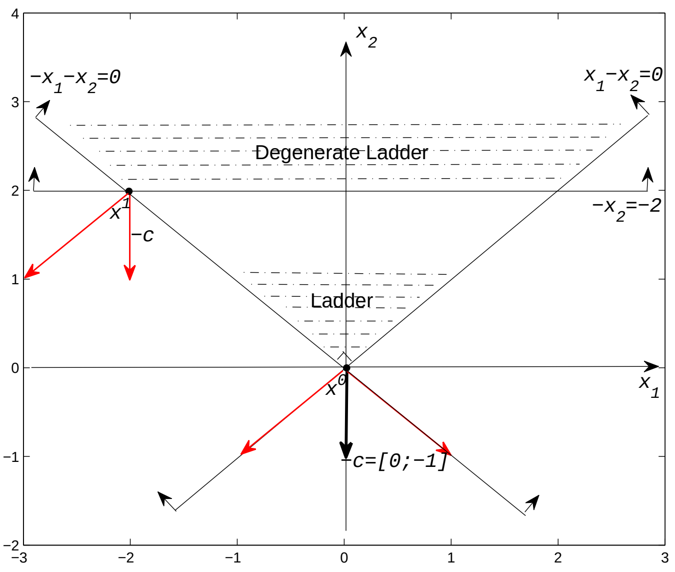

Figure 2.7: A degenerate LP example

and $-x_2 \leq -2$. Thus, the resulting problem is degenerate. See Figure 2.7 for illustration.

In the presence of degeneracy, Liu [60] illustrated that cycling may occur. However, this case can be readily treated by imposing a random perturbation on the objective function of the original problem, and that the perturbed problem has the same optimal solution as the original problem provided that the perturbation is small enough. For this reason, we only consider the case in which problem (P) is non-degenerate throughout LP part of this dissertation.

2.3 Duality Theory

2.3.1 Classical Results

Consider the following primal problem:

(P):
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b. \end{aligned}$$

The dual problem associated with (P) can be written as:

(D):
$$\begin{aligned} \min\ & b^T y \\ \text{s.t. } & A^T y = -c \\ & y \geq 0. \end{aligned}$$

With regard to the relations of (P) and (D), there are the following classical results (see Vanderbei [89]) .

Theorem 2.2. (Weak Duality Theorem) If x_0 and y_0 are feasible solutions to the primal problem (P) and its dual problem (D), then $c^T x_0 \geq -b^T y_0$.

Theorem 2.3. (Strong Duality Theorem) Given a pair of primal and dual problems (P) and (D), exactly one of the following statements holds true:

(1) If one of the primal or dual problems possesses a finite optimal solution, then both possess optimal solutions and the optimal objective function values satisfy $c^T x^* = -b^T y^*$, where x^* and y^* are the optimal solutions of the primal and dual problems (P) and (D) respectively.

(2) Both problems are infeasible.

(3) If one of the primal or dual problems possesses an unbounded optimal objective value, then the other must not have feasible solutions.

The classical strong duality theorem (or trichotomy theorem) tells us, if one of a pair of primal and dual problems is unbounded, then its counterpart is infeasible, but the reverse need not be true, which implies the strong duality theorem is not completely symmetric.

Theorem 2.4. (Complementary Slackness Theorem) Consider a pair of primal and dual problems (P) and (D). Let $x^* = [x_1^*; x_2^*; \cdots; x_n^*]$ and $y^* = [y_1^*; y_2^*; \cdots; y_m^*]$ be feasible solutions for the primal and dual programs, respectively. Then x^* and y^* are optimal solutions for their respective problems if and only if for all i $(i = 1, 2, \cdots, m)$

$$a_i x^* < b_i \Rightarrow y_i^* = 0,$$

$$y_i^* > 0 \Rightarrow a_i x^* = b_i.$$

The complementary slackness theorem tells us that if a constraint in the primal problem (P) is not binding at optimality, then the corresponding variable in the dual problem (D) must be zero. If a variable in the dual problem (D) is positive, then the corresponding constraint in the primal problem must be binding at optimality. This theorem reveals the relation of the optimal solutions of the primal and dual problems under the premise that a feasible solution pair of the primal problem and dual problem is given.

2.3.2 Inclusive-cone-based Duality Theory

Based on inclusiveness concept, Liu [61] reinvestigated the relations between the primal problem and dual problem, developing two main results which we present here as Theorem 2.5 and Theorem 2.6.

Theorem 2.5. [61] **(A duality principle)** Consider a pair of primal and dual problems (P) and (D). The following hold true:

a) Problem (P) is inclusive if and only if the corresponding dual problem (D) is feasible.

b) Problem (P) is feasible if and only if the corresponding dual problem (D) is inclusive.

Theorem 2.5 shows that inclusiveness and feasibility are two parallel, coexisting and mutually dual concepts, one of which is possessed by the primal problem if and only if the counterpart is possessed by its dual problem.

As mentioned above, the classical strong duality theorem is asymmetric, which may be one of the reasons why some current LP solvers fail to provide correct solvability information, in particular in the case that the primal problem is infeasible or unbounded. By means of the inclusiveness concept, a "symmetric" strong duality theorem termed "quadrachotomy theorem" was proposed in the framework of a new category of LP problems in [61], which we state as below:

Theorem 2.6. [61] **(A Quadrichotomy Theorem)** Given a pair of primal and dual problems (P) and (D), one of the following statements holds true:

(a) Both problem (P) and problem (D) possess finite optimal solutions

$\Leftrightarrow$ Problem (P) possesses a finite optimal solution

$\Leftrightarrow$ Problem (D) possesses a finite optimal solution

$\Leftrightarrow$ Problem (P) is inclusive-feasible

$\Leftrightarrow$ Problem (D) is inclusive-feasible

$\Leftrightarrow$ Both problem (P) and problem (D) are inclusive-feasible

$\Leftrightarrow$ Both problem (P) and problem (D) are inclusive

$\Leftrightarrow$ Both problem (P) and problem (D) are feasible

At optimality, the optimal objective function values of problem (P) and (D) satisfy $c^T x^* = -b^T y^*$.

(b) Problem (P) is inclusive-infeasible

$\Leftrightarrow$ Problem (D) is unbounded

$\Leftrightarrow$ Problem (D) is noninclusive-feasible

$\Leftrightarrow$ Problem (P) is infeasible and problem (D) is feasible

$\Leftrightarrow$ Problem (P) is inclusive and problem (D) noninclusive

(c) Problem (P) is noninclusive-infeasible

$\Leftrightarrow$ Problem (D) is noninclusive-infeasible

$\Leftrightarrow$ Both problem (P) and problem (D) are infeasible

$\Leftrightarrow$ Both problem (P) and problem (D) are noninclusive

$\Leftrightarrow$ Both problem (P) and problem (D) are noninclusive-infeasible

(d) Problem (P) is unbounded

$\Leftrightarrow$ Problem (P) is noninclusive-feasible

$\Leftrightarrow$ Problem (D) is inclusive-infeasible

$\Leftrightarrow$ Problem (P) is feasible and Problem (D) is infeasible

$\Leftrightarrow$ Problem (P) is noninclusive and Problem (D) is inclusive

Compared with the classical strong duality theorem (see Theorem 2.3), the quadrichotomy theorem not only completes the classical strong duality theorem, but also establishes the equivalent relationship between the primal problem and dual problems. More importantly, combination of the quadrichotomy theorem and an inclusive-cone-based solvability criterion to be proposed in Chapter 5 enables us to diagnose accurately the solvability of both the primal and dual problems, particularly in the case that an LP problem has no optimal solution. In Chapter 5, we will discuss in detail all the cases with regard to the solvability of LP problems in the context of the new category of LP problems and the quadrichotomy theorem.

In the foregoing we have mentioned that the complementary slackness theorem requires that a feasible solution pair of the primal and dual problems be known. With the powerful analytical tool an inclusive cone at hand, an inclusive-cone-based version of complementary slackness theorem is readily derivable.

In the following, we denote by $\mathcal{J} \setminus J_k$ the set difference of an index set J_k from a given index set $\mathcal{J}$. Given $J_k = \{j_1, j_2, \cdots, j_n\} \subseteq \mathcal{J}$, we use $A(J_k)$ to denote a matrix consisting of j_1-th, j_2-th, $\cdots$, j_n-th rows of the matrix A. Also we denote by $b(J_k)$ and $y(J_k)$ the column vectors consisting of j_1-th, j_2-th, $\cdots$, j_n-th components of the column vectors $b \in R^m$ and $y \in R^m$, respectively.

Theorem 2.7. (Complementary Slackness Theorem ——Inclusive-cone-based Version) Consider a pair of primal and dual problems (P) and (D). Let $J_k \subseteq \mathcal{J}$ be the ladder generator associated with the ladder $L(J_k)$, and let y be a feasible solution of the dual problem (D) which satisfies $y(J_k) > 0$ and $y(\mathcal{J} \setminus J_k) = 0$. Then x^k is the base point of the ladder $L(J_k)$ if and only if the following hold true:

$$(A(J_k)x^k - b(J_k))y(J_k) = 0$$

and

$$(A(\mathcal{J} \setminus J_k)x^k - b(\mathcal{J} \setminus J_k))y(\mathcal{J} \setminus J_k) = 0.$$

Proof. According to Definition 2.1, it is obvious. □

In contrast with the foregoing complementary slackness theorem (see Theorem 2.4), the inclusive-cone-based complementary slackness theorem reveals the relation between the base points (not necessarily feasible) of the primal problem (P) and dual feasible solutions, instead of only the relation between the optimal solutions of the primal and dual problems. It encompasses the original complementary slackness theorem as a special case.

2.4 Inclusiveness under Transformation

2.4.1 Motivation

We have seen from the foregoing that an LP problem can be put into different forms. A particular numerical method may be suitable for a specific form (for example, the simplex method was designed for LP problems in standard form). On the other hand, a method may have different versions corresponding to different forms of an LP problem. When an LP problem is transformed from one form to another form, some inherited properties (e.g., feasibility, boundedness and existence of an optimal solution etc.) remain unchanged. Since inclusiveness, as a generalization of boundedness concept, is a parallel and coexisting concept to feasibility, we naturally raise a question: does inclusiveness of an LP problem remain unchanged after an LP problem is transformed (or converted) from one form to another? Our aim in this section is to provide a positive answer to the above question.

In this section, we are concerned with the LP problems of the following two forms

(P):
$$\begin{aligned} &\min c^T x \\ &\text{s.t. } Ax \le b, \end{aligned}$$

(H):
$$\begin{aligned} &\min w^T x \\ &\text{s.t. } Dx = g \\ &\qquad Gx \le h, \end{aligned}$$

where $x = [x_1; x_2; \cdots; x_n] \in R^n$ is the vector of decision variables, $c \in R^n$ $(c \neq 0)$ and $w \in R^n$ $(w \neq 0)$ are vectors of objective function coefficients, $b \in R^m$, $g \in R^r$ and $h \in R^{m-r}$ are given RHS vectors, and A, D and G are given $m \times n$, $r \times n$ and $(m-r) \times n$ matrices respectively.

In this section, we assume that $m \geq n$ and rank(A)=rank$\left(\begin{bmatrix} D \\ G \end{bmatrix}\right)$=$n$.

Two cases will be investigated:

- Transforming problem (H) to a problem with only inequality constraints by eliminating the equalities.
- Converting problem (P) to a problem in standard form by adding slack variables and splitting the decision variables.

For each case, the relationship of inclusiveness between the original problem and the transformed problem is explored in the following subsection.

2.4.2 Inclusiveness under Transformation

In this subsection, we will investigate whether the property of inclusiveness keeps invariant after converting problem (P) or problem (H) to another form. The following two cases are considered.

Case 1: Convert problem (H) to an LP problem with only inequality constraints by eliminating equality constraints

Consider problem (H). We assume for this moment that rank(D)=r $(r < n)$. If rank$(D) < r$, then obviously part of equality constraints are either unsolvable or can be eliminated without changing the original problem. The occurrence of the former will cause infeasibility of problem (H). Under this circumstance there is no need to reformulate problem (H). Thus here we assume that redundant equality constraints have been removed and D is of full row rank. For simplicity of exposition and without loss of generality, we also assume that the first r columns of the matrix D are independent. In fact, if not, we just need to rearrange the n columns of the matrix D in such a manner that all the independent columns are listed first. Correspondingly, the rows of x need to be rearranged as well.

On the basis of the above assumptions, we can remove r variables of x by solving the linear system $Dx = g$ and convert problem (H) into a problem involving only $n - r$ variables but no equality constraints. This can be done as follows.

Partition the coefficient matrix D as

$$D = [B,\ P],$$

where B and P are $r \times r$ and $r \times (n-r)$ matrices, respectively. Vector x is partitioned conformably as

$$x = \begin{bmatrix} \hat{x} \\ \bar{x} \end{bmatrix}.$$

Therefore the linear system $Dx = g$ can be written as

$$B\hat{x} + P\bar{x} = g.$$

The fact that B is invertible gives

$$\hat{x} = -B^{-1}P\bar{x} + B^{-1}g.$$

Therefore, by the following transformation

$$x = \begin{bmatrix} \hat{x} \\ \bar{x} \end{bmatrix} = Q\bar{x} + q, \tag{2.1}$$

where

$$Q = \begin{bmatrix} -B^{-1}P \\ I_{n-r} \end{bmatrix} \text{ and } q = \begin{bmatrix} B^{-1}g \\ 0_{(n-r)\times 1} \end{bmatrix},$$

we obtain the problem

($\bar{\text{H}}$):
$$\begin{aligned} &\min\ \bar{w}^T\bar{x} \\ &\text{s.t. } \bar{G}\bar{x} \le \bar{h}, \end{aligned}$$

where $\bar{w} = Q^T w$, $\bar{G} = GQ$ and $\bar{h} = h - Gq$.

We want to know how the inclusiveness of problem (H) relates to the inclusiveness of ($\bar{\text{H}}$). The following theorem gives the answer.

Theorem 2.8. Problem (H) is inclusive if and only if problem ($\bar{\text{H}}$) is inclusive.

Proof. (Necessity) Suppose that problem (H) is inclusive. That means its equivalent problem (H$'$)

$$(\mathrm{H}'):\qquad \begin{aligned} &\min\ w^T x \\ &\text{s.t. } Dx \le g \\ &\quad\ -Dx \le -g \\ &\quad\ Gx \le h \end{aligned}$$

is inclusive. It follows from Theorem 2.5 that the dual problem of problem (H$'$)

$$(\mathrm{DH}):\qquad \begin{aligned} &\min \begin{bmatrix} g^T & -g^T & h^T \end{bmatrix} \begin{bmatrix} \lambda^{(1)} \\ \lambda^{(2)} \\ \lambda^{(3)} \end{bmatrix} \\ &\text{s.t. } \begin{bmatrix} D^T & -D^T & G^T \end{bmatrix} \begin{bmatrix} \lambda^{(1)} \\ \lambda^{(2)} \\ \lambda^{(3)} \end{bmatrix} = -w \\ &\quad\ \lambda^{(i)} \ge 0\ (i = 1,2,3) \end{aligned}$$

is feasible. Let $\lambda_0 = [\lambda_0^{(1)}; \lambda_0^{(2)}; \lambda_0^{(3)}]$ $(\lambda_0 \ge 0)$ be an arbitrary feasible solution of problem (DH). Then $D^T\lambda_0^{(1)} - D^T\lambda_0^{(2)} + G^T\lambda_0^{(3)} = -w$. Pre-multiplying this equality by Q^T gives $Q^T D^T(\lambda_0^{(1)} - \lambda_0^{(2)}) + Q^T G^T \lambda_0^{(3)} = -Q^T w$. Note that

$$DQ = \begin{bmatrix} B & P \end{bmatrix} \begin{bmatrix} -B^{-1}P \\ I_{n-r} \end{bmatrix} = 0_{r\times(n-r)}.$$

Therefore we have $Q^T G^T \lambda_0^{(3)} = -Q^T w$, i.e., $\bar{G}^T \lambda_0^{(3)} = -\bar{w}$, which implies that the dual problem of problem ($\bar{\mathrm{H}}$)

$$(\mathrm{D\bar{H}}):\qquad \begin{aligned} &\min\ \bar{h}^T \bar{y} \\ &\text{s.t. } \bar{G}^T \bar{y} = -\bar{w} \\ &\quad\ \bar{y} \ge 0 \end{aligned}$$

is feasible. Thus by Theorem 2.5, problem ($\bar{\mathrm{H}}$) is inclusive.

(Sufficiency) Suppose that problem ($\bar{\mathrm{H}}$) is inclusive. Then from Theorem 2.5, problem (D$\bar{\mathrm{H}}$) is feasible. Let $\bar{y}_0 \ge 0$ be an arbitrary feasible solution of problem (D$\bar{\mathrm{H}}$). Then $\bar{G}^T \bar{y}_0 = -\bar{w}$, i.e., $Q^T G^T \bar{y}_0 = -Q^T w$. That is $Q^T(G^T\bar{y}_0 + w) = 0$. Partitioning $G^T\bar{y}_0 + w$ as

$$G^T\bar{y}_0 + w = \begin{bmatrix} \hat{y}_0 \\ \tilde{y}_0 \end{bmatrix},$$

we have

$$\begin{bmatrix} -P^T[B^{-1}]^T & I_{n-r} \end{bmatrix} \begin{bmatrix} \hat{y}_0 \\ \tilde{y}_0 \end{bmatrix} = 0,$$

which yields $\tilde{y}_0 = P^T[B^{-1}]^T\hat{y}_0$. Let $\bar{\lambda}_0 = -[B^{-1}]^T\hat{y}_0$. Then

$$G^T\bar{y}_0 + w = \begin{bmatrix} \hat{y}_0 \\ \tilde{y}_0 \end{bmatrix} = \begin{bmatrix} -B^T\bar{\lambda}_0 \\ -P^T\bar{\lambda}_0 \end{bmatrix} = -\begin{bmatrix} B^T \\ P^T \end{bmatrix}\bar{\lambda}_0 = -D^T\bar{\lambda}_0.$$

Therefore $D^T\bar{\lambda}_0 + G^T\bar{y}_0 = -w$, which implies that problem (DH) is feasible. Thus it follows from Theorem 2.5 that problem (H$'$) is inclusive. Hence problem (H) is inclusive. This completes the proof. □

Case 2: Convert problem (P) to an LP problem in standard form

Consider problem (P). Let $x = x^+ - x^-$, where $x^+, x^- \geq 0$. Introducing a vector of slack variables $s = [s_1; s_2; \cdots; s_m] \geq 0$, the original problem (P) can be transformed to the following standard form

($\tilde{\text{P}}$):
$$\begin{aligned} &\min\ c^Tx^+ - c^Tx^- \\ &\text{s.t. } Ax^+ - Ax^- + s = b \\ &\qquad x^+, x^-, s \geq 0. \end{aligned}$$

For convenience of analyzing the relationship of inclusiveness between problem (P) and problem ($\tilde{\text{P}}$), we rewrite problem ($\tilde{\text{P}}$) as the equivalent inequality form

($\tilde{\text{P}}'$):
$$\min \begin{bmatrix} c^T & -c^T & 0 \end{bmatrix} \begin{bmatrix} x^+ \\ x^- \\ s \end{bmatrix}$$

$$\text{s.t.} \begin{bmatrix} A & -A & I_m \\ -A & A & -I_m \\ -I_n & 0 & 0 \\ 0 & -I_n & 0 \\ 0 & 0 & -I_m \end{bmatrix} \begin{bmatrix} x^+ \\ x^- \\ s \end{bmatrix} \leq \begin{bmatrix} b \\ -b \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

With regard to the relation of inclusiveness between problem (P) and problem ($\tilde{\text{P}}$), we have the following main result similar to Theorem 2.8.

Theorem 2.9. Problem (P) is inclusive if and only if problem ($\tilde{\mathrm{P}}$) is inclusive.

Proof. Suppose that problem (P) is inclusive. Without loss of generality, we assume that problem (P) has an inclusive cone, which is generated by outward normal vectors $a_1^T, a_2^T, \cdots, a_n^T$ of its first n constraints. (If not, we just need to rearrange constraints of problem (P) in such a manner that some n constraints associated with one of inclusive cones are listed ahead of other constraints.) Then, there exist n non-negative constants $\delta_1, \delta_2, \cdots, \delta_n$, such that

$$-c = \delta_1 a_1^T + \delta_2 a_2^T + \cdots + \delta_n a_n^T, \tag{2.2}$$

which can be equivalently written as

$$\begin{aligned}
\begin{bmatrix} -c \\ c \\ 0_{m\times 1} \end{bmatrix} &= \delta_1 \begin{bmatrix} a_1^T \\ -a_1^T \\ e_m^{(1)} \end{bmatrix} + \delta_2 \begin{bmatrix} a_2^T \\ -a_2^T \\ e_m^{(2)} \end{bmatrix} + \cdots + \delta_n \begin{bmatrix} a_n^T \\ -a_n^T \\ e_m^{(n)} \end{bmatrix} \\
&+ 0 \begin{bmatrix} -e_n^{(1)} \\ 0_{n\times 1} \\ 0_{m\times 1} \end{bmatrix} + 0 \begin{bmatrix} -e_n^{(2)} \\ 0_{n\times 1} \\ 0_{m\times 1} \end{bmatrix} + \cdots + 0 \begin{bmatrix} -e_n^{(n)} \\ 0_{n\times 1} \\ 0_{m\times 1} \end{bmatrix} \\
&+ \delta_1 \begin{bmatrix} 0_{n\times 1} \\ 0_{n\times 1} \\ -e_m^{(1)} \end{bmatrix} + \delta_2 \begin{bmatrix} 0_{n\times 1} \\ 0_{n\times 1} \\ -e_m^{(2)} \end{bmatrix} + \cdots + \delta_n \begin{bmatrix} 0_{n\times 1} \\ 0_{n\times 1} \\ -e_m^{(n)} \end{bmatrix} \\
&+ 0 \begin{bmatrix} 0_{n\times 1} \\ 0_{n\times 1} \\ -e_m^{(n+1)} \end{bmatrix} + 0 \begin{bmatrix} 0_{n\times 1} \\ 0_{n\times 1} \\ -e_m^{(n+2)} \end{bmatrix} + \cdots + 0 \begin{bmatrix} 0_{n\times 1} \\ 0_{n\times 1} \\ -e_m^{(m)} \end{bmatrix},
\end{aligned} \tag{2.3}$$

where $e_m^{(i)}$ $(1 \le i \le m)$ denotes the unit column vector with m components and its i-th element 1, and $e_n^{(j)}$ $(1 \le j \le n)$ the unit column vector with n components and its j-th element 1. It is noticeable that the $2n + m$ vectors on the right-hand-side of Equation (2.3) are outward normals of those constraints in problem ($\tilde{\mathrm{P}}'$) associated with the constraint index set $\{1, 2, \cdots, n\} \bigcup \{2m + 1, 2m + 2, \cdots, 2m + n\} \bigcup \{2m + 2n + 1, 2m + 2n + 2, \cdots, 2m + 2n + m\}$. Also note that these $2n+m$ vectors are independent, which is easily verified by means of the independence of a_i^T $(1 \le i \le n)$. Therefore, it follows from Equation (2.3) that problem $(\tilde{\mathrm{P}}^{'})$ has an inclusive cone, which is generated by aforementioned $2n+m$ outward normal vectors. Thus, problem $(\tilde{\mathrm{P}}^{'})$ is inclusive, which means that problem $(\tilde{\mathrm{P}})$ is inclusive.

Conversely, suppose that problem $(\tilde{\mathrm{P}})$ is inclusive. This means that problem $(\tilde{\mathrm{P}}^{'})$ is inclusive. It follows from Theorem 2.5 that the dual problem of problem $(\tilde{\mathrm{P}}^{'})$

$$(\tilde{\mathrm{DP}}^{'}): \qquad \min \begin{bmatrix} b^T & -b^T & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u^{(1)} \\ u^{(2)} \\ u^{(3)} \\ u^{(4)} \\ u^{(5)} \end{bmatrix}$$

$$\text{s.t.} \begin{bmatrix} A^T & -A^T & -I_n & 0 & 0 \\ -A^T & A^T & 0 & -I_n & 0 \\ I_m & -I_m & 0 & 0 & -I_m \end{bmatrix} \begin{bmatrix} u^{(1)} \\ u^{(2)} \\ u^{(3)} \\ u^{(4)} \\ u^{(5)} \end{bmatrix} = - \begin{bmatrix} c \\ -c \\ 0 \end{bmatrix}$$

$$u^{(i)} \geq 0 \quad (i = 1, 2, 3, 4, 5)$$

is feasible. Let $u_0 = \begin{bmatrix} u_0^{(1)}; & u_0^{(2)}; & u_0^{(3)}; & u_0^{(4)}; & u_0^{(5)} \end{bmatrix} \geq 0$ be an arbitrary feasible solution to problem $(\tilde{\mathrm{DP}}^{'})$. Then we have

$$\begin{bmatrix} A^T & -A^T & -I_n & 0 & 0 \\ -A^T & A^T & 0 & -I_n & 0 \\ I_m & -I_m & 0 & 0 & -I_m \end{bmatrix} \begin{bmatrix} u_0^{(1)} \\ u_0^{(2)} \\ u_0^{(3)} \\ u_0^{(4)} \\ u_0^{(5)} \end{bmatrix} = - \begin{bmatrix} c \\ -c \\ 0 \end{bmatrix},$$

i.e., the following hold:

$$A^T(u_0^{(1)} - u_0^{(2)}) - u_0^{(3)} = -c, \tag{2.4}$$

$$-A^T(u_0^{(1)} - u_0^{(2)}) - u_0^{(4)} = c, \tag{2.5}$$

$$u_0^{(1)} - u_0^{(2)} = u_0^{(5)}, \tag{2.6}$$

$$(u_0^{(i)} \geq 0 \quad i = 1, 2, 3, 4, 5).$$

From (2.4)-(2.6) we can easily deduce that $A^T u_0^{(5)} = -c$. Note that $u_0^{(5)} \geq 0$. Therefore we conclude that the dual problem of problem (P)

(DP):
$$\begin{aligned} &\min\ b^T y \\ &\text{s.t. } A^T y = -c \\ &\qquad y \geq 0 \end{aligned}$$

is feasible. Hence from Theorem 2.5, problem (P) is inclusive. This completes the proof. □

2.5 Sensitivity Analysis

2.5.1 Introduction

Sensitivity analysis is an important part of linear programming. It investigates how the optimal solution of an optimization problem changes with respect to changes in the problem data. Theoretically, sensitivity analysis of LP problems provides useful information for the decision makers.

The study of sensitivity analysis in linear programming can be traced back to the early 1950's. For pioneering works, we refer the reader to [70] by Manne and [84] by Saaty and Gass. Traditionally, the study of sensitivity analysis is based on the simplex method and therefore associated with an optimal basis (see Gal [26], Vanderbei [89] and Wendell [96]). Later, the prevalence of the interior point methods arouses the researchers' reconsideration of the sensitivity analysis in linear programming from a different point of view (see Adler and Monteiro [1], Jansen et al. [46], Greenberg [37, 38], Greenberg et al. [39] and Yildirim and Todd [100]).

In this section, we attempt to examine sensitivity analysis in linear programming from the perspective of the inclusive cone in order to explore the potentials of the new approach in sensitivity analysis.

We focus on the analysis with respect to perturbation of the objective function coefficients (OFC) or the right-hand-side (RHS) elements of the constraints. Sufficient and necessary conditions for the optimal solution invariancy with respect to perturbations of OFC as well as the optimal ladder generator invariancy with respect to perturbation of RHS elements are presented. In addition, we identify the ranges of parameter variation in which the optimal vertex solution or the optimal ladder generator remains unchanged. These results obtained are similar to the classical results. However, the approach we present here is simpler and more geometrically intuitive.

The remaining of this section is organized as follows. In Subsection 2.5.2, we present the main results with respect to the perturbation of OFC. The main results with respect to the perturbation of RHS are given in Subsection 2.5.3.

Throughout this section, we assume that problem (P) has been solved to optimality and x^* is an optimal vertex solution. Given a ladder generator J, we denote by $A(J)$ the matrix formed from rows of A with row indices in J. Also we denote by $b(J)$ the vector formed from rows of b with row indices in J.

2.5.2 Perturbation in the Objective Function Data

Let us consider the perturbed primal LP problems as follows:

P(Δc, ϵ):
$$\begin{aligned} &\min\ (c+\epsilon\Delta c)^T x \\ &\text{s.t. } Ax \le b, \end{aligned}$$

where ϵ is a real parameter, and Δc is a perturbation vector.

We hope to find the range of variation of parameter ϵ, which is called the optimal solution invariant range, such that x^* is still an optimal vertex solution for the perturbed problem P(Δc, ϵ) .

Different from ordinary sensitivity analysis in LP, which is traditionally associated with an optimal basis (as in the simplex method), we adopt the inclusive-cone-based approach to present a sufficient and necessary condition for the perturbation of c and identify the range of variation of the parameter ϵ within which the vertex solution x^* remains optimal.

To begin, we develop the following theorem.

Theorem 2.10. Let x^* be an optimal vertex solution to problem (P). Suppose that there are exactly n active constraints at the optimal vertex solution x^* to problem (P). Let J^* be a unique ladder generator associated with x^* and $N(J^*)$ be the inclusive cone associated with J^*. Then x^* is still an optimal vertex solution to the perturbed problem P(Δc, ϵ) if and only if $-(c+\epsilon\Delta c) \in N(J^*)$.

Proof. It is obvious according to Theorem 2.1. $\square$

According to Theorem 2.10, ϵ is in the optimal solution invariant range if and only if $-(c+\epsilon\Delta c) \in N(J^*)$, or equivalently there exists $\delta(\epsilon) = [\delta_1(\epsilon); \delta_2(\epsilon); \cdots ; \delta_n(\epsilon)] \geq 0$ such that

$$-(c+\epsilon\Delta c) = A^T(J^*)\delta(\epsilon). \tag{2.7}$$

Since $A(J^*)$ is invertible, we have

$$-[A^T(J^*)]^{-1}(c+\epsilon\Delta c) = \delta(\epsilon). \tag{2.8}$$

Therefore, ϵ is in the optimal solution invariant range if and only if

$$-[A^T(J^*)]^{-1}(c+\epsilon\Delta c) \geq 0,$$

equivalently,

$$\epsilon[A^T(J^*)]^{-1}\Delta c \leq -[A^T(J^*)]^{-1}c.$$

Let

$$\Delta z = [A^T(J^*)]^{-1}\Delta c,\ \ z = [A^T(J^*)]^{-1}c, \tag{2.9}$$

we have

$$\epsilon\Delta z \leq -z. \tag{2.10}$$

Since $-c \in N(J^*)$, it follows that there exists $\delta = [\delta_1; \delta_2; \cdots ; \delta_n] \geq 0$ such that

$$-c = A^T(J^*)\delta.$$

Therefore $-z = -[A^T(J^*)]^{-1}c \geq 0$.

By solving the above (2.10), we can get the range of ϵ. The solution of (2.10) is generalized into the following cases.

(1) If all the components of Δz are positive, i.e., $\Delta z > 0$, then Equation (2.10) is valid as long as

$$\epsilon \leq \min\left(-\frac{z_i}{\Delta z_i}\right), \quad (1 \leq i \leq n). \tag{2.11}$$

(2) If all the components of Δz are negative, i.e., $\Delta z < 0$, then by solving Equation (2.10) we get

$$\epsilon \geq \max\left(-\frac{z_i}{\Delta z_i}\right), \quad (1 \leq i \leq n). \tag{2.12}$$

(3) If Δz consists of both positive and negative components, then Equation (2.10) holds as long as

$$\max\left\{-\frac{z_i}{\Delta z_i}\middle|\Delta z_i < 0, 1 \leq i \leq n\right\} \leq \epsilon \leq \min\left\{-\frac{z_i}{\Delta z_i}\middle|\Delta z_i > 0, 1 \leq i \leq n\right\}. \tag{2.13}$$

Note that, in the process of discussing the solution of (2.10) we omit the case that some components of Δz are zeroes. When some components of Δz are zeroes, say $\Delta z_1 = 0$, the inequality $\epsilon\Delta z_1 \leq -z_1$ inevitably holds for arbitrary ϵ. Obviously it is trivial to consider this case since our aim is to find a bound on the parameter ϵ.

To illustrate the results developed here, consider the following example taken from [60].

Example 2.5. Consider the following LP problem

$$\begin{aligned} \min \quad & x_1 + x_2 \\ \text{s.t.} \quad & \begin{bmatrix} -2 & -1 \\ -2 & 4 \\ -1 & 3 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 4 \\ -8 \\ -7 \\ 0 \\ 0 \end{bmatrix}. \end{aligned}$$

Assume that we have got the optimal solution $x = [7; 0]$ by the targeted climbing algorithm (see Liu [60]). The optimal ladder generator is $\{3, 5\}$. Here, we see that

$$c = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad A(J^*) = \begin{bmatrix} -1 & 3 \\ 0 & -1 \end{bmatrix}.$$

Given

$$\Delta c = \begin{bmatrix} -1 \\ 4 \end{bmatrix},$$

we want to find the optimal solution invariant range of ϵ. From (2.9) we have

$$z = \begin{bmatrix} -1 \\ -4 \end{bmatrix}, \quad \Delta z = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Therefore, according to (2.13) we can get the optimal solution invariant range of ϵ

$$-4 \leq \epsilon \leq 1.$$

In the case that there are l ($l > n$) active constraints at the optimal vertex solution x^* to problem (P), we can use the following theorem to judge whether x^* is still the optimal vertex solution to the perturbed problem P(Δc, ϵ).

Theorem 2.11. Let x^* be an optimal vertex solution to problem (P). Suppose that, at the optimal solution x^*, there are l ($l > n$) active constraints and there are k inclusive cones $N(J_i^*)$ $(1 \leq i \leq k)$ with the associated ladder generators J_i^* $(1 \leq i \leq k)$. Then x^* is still the optimal vertex solution to the perturbed problem P(Δc, ϵ) if and only if $-(c + \epsilon \Delta c) \in \bigcup N(J_i^*)$, $(1 \leq i \leq k)$.

Proof. It is obvious according to Theorem 2.1. □

We can adopt the following procedure (similar to that in [27]) to find the optimal solution invariant range of ϵ when there are more than n active constraints at the optimal vertex solution x^* to problem (P).

Firstly, for every inclusive cone $N(J_i^*)$ $(1 \leq i \leq k)$, identify the range of parameter variation. The range, denoted by $\Lambda_i, (1 \leq i \leq k)$, can be derived from one of (2.11)-(2.13).

Secondly, obtain the overall range of ϵ within which x^* is the optimal vertex solution to the perturbed problem P(Δc, ϵ). The overall range is $\bigcup \Lambda_i, (1 \leq i \leq k)$.

2.5.3 Perturbation in the Right-Hand-Side Data

In this subsection, the following perturbed LP problem is considered.

$$\text{P}(\Delta b,\, \epsilon): \qquad \begin{aligned} &\min c^T x \\ &\text{s.t. } Ax \le b + \epsilon \Delta b. \end{aligned}$$

We first give a definition which will be used throughout this subsection.

Definition 2.6. Consider the linear programming problem (P). Suppose problem (P) has been solved to obtain the optimal solution x^*. The ladder generator J^* associated with the optimal solution x^* is called the optimal ladder generator of problem (P). The ladder $L(J^*)$ associated with the optimal ladder generator J^* is called the optimal ladder of problem (P).

We hope to find the range of variation of parameter ϵ, which is called the optimal ladder generator invariant range, such that J^* is still an optimal ladder generator for the perturbed problem P(Δb, ϵ).

Following a similar line to the above subsection, we first perform the optimal ladder generator invariancy sensitivity analysis for the case in which there are exactly n active constraints at the optimal vertex solution x^* to problem (P). We hope to find the range of variation of ϵ to guarantee that the perturbed problem P(Δb, ϵ) has the same optimal ladder generator as the original problem (P).

The main results in this subsection are presented as follows.

Theorem 2.12. Let x^* be an optimal vertex solution to problem (P). Suppose that there are exactly n active constraints at the optimal vertex solution x^* to problem (P). Let J^* be a unique ladder generator associated with x^* and $N(J^*)$ be the inclusive cone associated with J^*. Then J^* is still an optimal ladder generator to the perturbed problem P(Δb, ϵ) if and only if there exists a vector $\bar{x}^* \in R^n$ such that the following hold:

$$(1)\ A(J^*)\bar{x}^* = b(J^*) + \epsilon \Delta b(J^*) \tag{2.14}$$

$$(2)\ A(K^*)\bar{x}^* \le b(K^*) + \epsilon \Delta b(K^*), \tag{2.15}$$

where $K^* = \{1, 2, \cdots, m\} \setminus J^*$.

Proof. It is obvious according to Theorem 2.1. □

By virtue of Formulae (2.14) and (2.15) we can obtain the range of variation of ϵ within which the ladder generator J^* is still optimal to the perturbed problem $\mathrm{P}(\Delta b,\,\epsilon)$. In fact, from (2.14) we have

$$\bar{x}^* = [A(J^*)]^{-1}(b(J^*) + \epsilon\Delta b(J^*)).$$

Substituting the above expression in (2.15) and doing simple manipulation gives

$$\epsilon(\Delta b(K^*) - A(K^*)[A(J^*)]^{-1}\Delta b(J^*)) \geq A(K^*)[A(J^*)]^{-1}b(J^*) - b(K^*).$$

Take

$$\Delta u = \Delta b(K^*) - A(K^*)[A(J^*)]^{-1}\Delta b(J^*) \tag{2.16}$$

and

$$u = b(K^*) - A(K^*)[A(J^*)]^{-1}b(J^*), \tag{2.17}$$

we have

$$\epsilon\Delta u \geq -u. \tag{2.18}$$

Note that $u \geq 0$ in (2.18). In fact, since x^* is the optimal vertex solution to problem (P), it follows that

$$A(J^*)x^* = b(J^*), \tag{2.19}$$

and

$$A(K^*)x^* \leq b(K^*) \tag{2.20}$$

hold. From Equation (2.19), we have

$$x^* = [A(J^*)]^{-1}b(J^*).$$

Plugging this into (2.20), we can see that $u \geq 0$.

In order to obtain the range of ϵ we need only solve (2.18). We have manipulated this type of inequality in Subsection 2.5.2. Similarly, the solution of (2.18) can be divided into the following cases.

(1) If all the components of Δu are positive, i.e., $\Delta u > 0$, then Equation (2.18) is valid as long as

$$\epsilon \geq \max\left(-\frac{u_j}{\Delta u_j}\right), \quad (1 \leq j \leq m-n). \tag{2.21}$$

(2) If all the components of Δu are negative, i.e., $\Delta u < 0$, then by solving Equation (2.18), we get

$$\epsilon \leq \min\left(-\frac{u_j}{\Delta u_j}\right), \quad (1 \leq j \leq m-n). \tag{2.22}$$

(3) If components of Δu have different signs, then Equation (2.18) holds as long as

$$\max\left\{-\frac{u_j}{\Delta u_j}\middle|\Delta u_j > 0, 1 \leq j \leq m-n\right\} \leq \epsilon \leq \min\left\{-\frac{u_j}{\Delta u_j}\middle|\Delta u_j < 0, 1 \leq j \leq m-n\right\}. \tag{2.23}$$

Note that in the process of discussing the solution of (2.18) we omit the case in which some components of Δu are zeroes. The reason is same as that presented in Subsection 2.5.2. We do not restate here for conciseness.

When there are l $(l > n)$ active constraints at the optimal vertex solution x^* to problem (P), we have the following result.

Theorem 2.13. Assume that problem (P) has been solved to optimality and x^* is an optimal vertex solution to problem (P). Suppose also that, at the optimal vertex solution x^*, there are l $(l > n)$ active constraints and there are k inclusive cones $N(J_i^*)$ $(1 \leq i \leq k)$ with the associated ladder generators J_i^* $(1 \leq i \leq k)$. Then at least one J_i^* $(1 \leq i \leq k)$ is the optimal ladder generator to the perturbed problem P(Δc, ϵ) if and only if there exists a vector $\bar{x}^* \in R^n$ such that the following hold:

$$(1)\ A(J_i^*)\bar{x}^* = b(J_i^*) + \epsilon\Delta b(J_i^*), \quad (1 \leq i \leq k) \tag{2.24}$$

$$(2)\ A(K_i^*)\bar{x}^* \leq b(K_i^*) + \epsilon\Delta b(K_i^*), \quad (1 \leq i \leq k), \tag{2.25}$$

where $K_i^* = \{1, 2, \cdots, m\} \setminus J_i^*$ $(1 \leq i \leq k)$.

Proof. It is obvious according to Theorem 2.1. □

We can apply a two-step procedure similar to that presented at the end of Subsection 2.5.2 to find the optimal ladder generator invariant range of ϵ in the case that there are more than n active constraints at the optimal vertex solution x^* to problem (P). For avoidance of repetition, we omit the presentation of the similar procedure.

Chapter 3

Inclusive-cone-based Ladder Algorithms for Linear Programming

3.1 Introduction

The (primal) simplex algorithm conceived by Dantzig [13, 14] in the late 1940's was, by common consent, the first highly viable solution procedure for solving LP problems. Since its inception, dozens of simplex variants have been proposed to advance this algorithm (see Goldfarb and Reid [34], Harris [41], Forrest and Goldfarb [23], Lemke [56] and Arsham et al. [5]). Among them, the dual simplex algorithm is one of the most successful variants. However, when this algorithm was first introduced by Lemke [56] in 1954, it was not viewed as an alternative of the primal simplex method. It was not until nearly forty years later that this situation changed due to the contribution of Goldfarb [33], Fourer [24] and Forrest and Goldfarb [23]. In point of the overall performance, the dual simplex method may win out over the primal simplex method. In particular, it is effectual for solving relaxations of discrete optimization problems and has an indubitable superiority over the primal simplex method in a branch-and-bound framework.

It has turned out in practice that the simplex method is efficient and reliable

LP solution methods, particularly suitable for solving small- and medium-sized problems. Borgwardt [8, 9] showed the average number of pivot steps required by the simplex method is polynomial. However, its computational complexity is not polynomial for the worst case (see Goldfarb and Sit [35], Klee and Minty [52] and Roos [83]). Khachiyan's ellipsoid algorithm [51] developed in 1979 solves LP problems in polynomial time. Unfortunately, it performs poorly in practice. On the basis of Dikin's affine-scaling method (see Dikin [16, 17]), Karmarkar [50] proposed an interior point algorithm with polynomial time complexity in 1984 using a sequence of projective transformations of a polytope. Emergence of this algorithm is a significant breakthrough over the Khachiyan's ellipsoid method. Ever since, the interior point methods (IPMs) have been flourishing in the LP community (see Xu and Ye [99], Kojima et al. [53] and Anstreicher [3]).

To date, the simplex method and the interior point methods are still the most favored approaches, competitive to each other, for solving LP problems. As stated by Illés and Terlaky [44]: "*There is no clear champion in the race to solve LP problems in practice. Theoretical criteria, such as worst case complexity, the ability to generate strictly complementary solutions, clearly favor IPMs, but in practice pivot methods keep competing. The IPMs win for very large, sparse LP problems, while pivot algorithms are favorable for integer linear problems*".

Unfortunately, so far there has not existed a single LP algorithm which works well for solving all types of LP problems. Any currently-existing approach is only suitable for solving a certain type of LP problems. Furthermore, it remains unknown whether there exists a strongly polynomial time LP algorithm. For these reasons, a considerable amount of research is under way to pursue LP algorithms with better numerical performance (see Arsham [4], Barnes et al. [6], Liu [60, 61], Ding et al. [18, 19] and the references therein).

Recently, an inclusive-cone-based ladder method for solving LP problems was developed in [60]. The ladder method iteratively updates the inclusive cone and ladder to obtain a sequence of base points which converges to an optimal solution (if an LP problem has an optimal solution), or it concludes the problem is infeasible or unbounded. A ladder updating scheme, which involves picking up a violated constraint and dropping a constraint from the current ladder, is of crucial importance in determining the performance of a ladder algorithm. Two ladder algorithms, employing different ladder updating strategies, were proposed in the aforementioned paper. Our new algorithms to be developed in the subsequent chapter as well as Chapter 6 (for LP and LSIP) build upon the ladder method. Therefore, for sake of readability, it is necessary to give a overall review of the ladder method in the remainder of this chapter.

3.2 The Ladder Method

The ladder method (see Liu [60]) works on LP problems with inequality constraints (see problem (P) in Chapter 2) with the assumption that the problem is of full column rank. For clarification, we rewrite the problem as follows

(P):
$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \le b, \end{aligned}$$

where $x \in R^n$ is the vector of decision variables, $A \in R^{m\times n}$ $(m > n)$, $c = [c_1; c_2; \cdots ; c_n] \in R^n$ $(c \neq 0)$, and $b = [b_1; b_2; \cdots ; b_m] \in R^m$.

Throughout this section, we use the following notation:

$\mathcal{F}$: the feasible region of problem (P).

x^*: an optimal solution of problem (P).

$\mathcal{J}$: the constraint index set, $\mathcal{J} = \{1, 2, \cdots , m\}$.

J: an ordered subset of $\mathcal{J}$.

$\mathcal{J} \setminus J$: the set difference of the index set J from the index set $\mathcal{J}$.

$J(i \leftrightarrow j)$: the ordered subset with i-th entry of J replaced by $j \in \mathcal{J} \setminus J$.

a_j: the j-th row of A.

$A(J)$: the submatrix of A containing the rows a_j such that $j \in J$.

$b(J)$: the subvector of b containing the elements b_j such that $j \in J$.

P(J): the subproblem of (P) consisting of constraints associated with the index set J.

3.2.1 Preliminaries

The ladder method developed by Liu [60] is based on Definition 2.1 and Theorem 2.1 presented in the previous chapter. In addition, the following theorems were used.

Theorem 3.1. [60] For problem (P), the following statements hold true:

(1) If the problem has no optimal solution, then it is either infeasible or unbounded.

(2) If the problem has an optimal solution, it must have an optimal solution at the base point of some ladder.

Theorem 3.2. [60] For problem (P), the following properties hold true:

a) Let x_J be the base point of a ladder $L(J)$. The objective value at x_J serves as a lower bound for the optimal value if the feasible region $\mathcal{F}$ is non-empty. If x_J violates no constraint, then x_J is optimal.

b) If the feasible region $\mathcal{F}$ is non-empty and there is no bounded optimal solution, then there exists no inclusive cone and hence there exists no ladder.

c) If the problem does not have an inclusive cone, then the problem either is infeasible or has no bounded optimal solution.

d) If there exists an inclusive cone, then the following are true.

(1) Problem (P) either is infeasible or has an optimal solution at a base point.

(2) Problem (P) is infeasible if and only if all the base points are infeasible.

e) If at the base point x_J of a ladder $L(J)$ a constraint $a_s x \leq b_s$ is violated, that is, $a_s x_J > b_s$, then:

(1) Either there exists an index $j \in J$ such that $J^{'} = J(j \leftrightarrow s)$ is a ladder generator. In this case, among all such generators, there must be one $J^{'}$ such that its associated base point $x_{J^{'}}$ is in $L(J)$.

(2) Or problem $\mathrm{P}(J \bigcup \{s\})$ is infeasible (hence problem (P) is infeasible).

f) Let $J \subset \mathcal{J}$ be a ladder generator. If for some $j \in \mathcal{J} \setminus J$, $a_j^T \in N(J)$ and $a_j x_J < b_j$, then the j-th constraint of problem (P) is redundant (not binding at the optimal solution). On the other hand, if $-a_j^T \in N(J)$ and $a_j x_J > b_j$, then problem (P) is infeasible.

3.2.2 The Ladder Algorithms

In this subsection, we discuss in detail two ladder algorithms proposed in [60] (termed "the targeted climbing algorithm" and "the centered climbing algorithm" respectively). To start, we give basic steps of these two ladder algorithms.

The Targeted Climbing Algorithm (TCA):

Step 0 Initialization.

Start with a feasible reference point and a known ladder generator, which are denoted by x_r and $J_0 = \{j_1^0, j_2^0, \cdots, j_n^0\} \subset \mathcal{J}$, respectively. (Refer to Chapter 5 for how to find such a reference point and a generator if they are not immediately at hand.) Denote by $x^0 = x_{J_0}$ the base point associated with J_0. Compute the initial base point $x^0 = [A(J_0)]^{-1} b(J_0)$. Set $k = 0$ and $D_{k-1} = \emptyset$.

Step 1 Optimality criterion.

Let $V^k = \{j \in \mathcal{J} \backslash (J_k \bigcup D_{k-1}) : a_j x^k > b_j\}$.

- If $V^k = \emptyset$, exit with " Problem (P) has an optimal solution $x^* = x^k$".
- Otherwise, go to next step.

Step 2 Updating the ladder.

2.1 Selecting an index from V^k .

Set

$$t_j = \frac{a_j x^k - b_j}{a_j (x^k - x_r)}, \; j \in V^k. \tag{3.1}$$

Determine an index $p^k \in V^k$ as a pick such that $t_{p^k} = \max_{j \in V^k} \{t_j\}$.

2.2 Dropping an index from J_k .

Find an index $j_d^k \in J_k$ as a drop such that $J_{k+1} = J_k(j_d^k \leftrightarrow p^k)$ is a ladder generator and the associated base point $x^{k+1} \in L(J_k)$.

2.3 Update.

Denote the elements of J_{k+1} by $J_{k+1} = \{j_1^{k+1}, j_2^{k+1}, \cdots, j_n^{k+1}\}$. Let $D_k = \{j_d^k\}$. Compute the updated base point $x^{k+1} = [A(J_{k+1})]^{-1} b(J_{k+1})$. Set $k := k+1$. Return to Step 1.

The Centered Climbing Algorithm (CCA):

Step 0 Initialization.

Start with a known ladder generator, which is denoted by $J_0 = \{j_1^0, j_2^0, \cdots, j_n^0\} \subset \mathcal{J}$. (Refer to Chapter 5 for how to find such a generator if it is not immediately available.) Denote by $x^0 = x_{J_0}$ the base point associated with J_0. Compute the initial base point $x^0 = [A(J_0)]^{-1} b(J_0)$. Set $k = 0$ and $D_{k-1} = \emptyset$.

Step 1 Optimality criterion.

Let $V^k = \{j \in \mathcal{J} \backslash (J_k \bigcup D_{k-1}) : a_j x^k > b_j\}$.

- If $V^k = \emptyset$, exit with"Problem (P) has an optimal solution $x^* = x^k$".
- Otherwise, go to next step.

Step 2 Updating the ladder.

2.1 Selecting an index from V^k.

Let $v^k = -[A(J_k)]^{-1} \mathbf{1}_{n\times 1}$, where $\mathbf{1}_{n\times 1} = [1; 1; \cdots; 1] \in R^n$, and v^k is the center vector of the current ladder $L(J_k)$.

- If v^k is perpendicular to at least one of the a_j^T's $(j \in V^k)$, arbitrarily select $p^k \in V^k$ as a pick such that $a_{p^k} v^k = 0$.
- Otherwise, set

$$t_j = \frac{b_j - a_j x^k}{a_j v^k}. \tag{3.2}$$

  Determine $p^k \in V^k$ as a pick such that $t_{p^k} = \max_{j \in V^k} \{t_j\}$.

2.2 Dropping an index from J_k.

Try to identify an index $j_d^k \in J_k$ as a drop such that $J_{k+1} = J_k(j_d^k \leftrightarrow p^k)$ is a ladder generator and the associated base point $x^{k+1} \in L(J_k)$.

- If such an index does not exist, exit with "Problem (P) has no feasible solution."
- Otherwise, go to next step.

2.3 Update.

Denote the elements of J_{k+1} by $J_{k+1} = \{j_1^{k+1}, j_2^{k+1}, \cdots, j_n^{k+1}\}$. Let $D_k = \{j_d^k\}$. Compute the updated base point $x^{k+1} = [A(J_{k+1})]^{-1} b(J_{k+1})$. Set $k := k + 1$. Return to Step 1.

With respect to the TCA and CCA, we would like to point out:

Remark 3.1. Both the TCA and CCA require an initial ladder to start. To find a ladder $L(J)$ is to find the associated generator $J = \{j_1, j_2, \cdots, j_n\} \subset \mathcal{J}$, equivalently, to find n linearly independent outward normal vectors $a_{j_k}^T$ $(k = 1, 2, \cdots, n)$ such that there exist n constants $\lambda_{j_k} \geq 0$ $(k = 1, 2, \cdots, n)$ satisfying

$$-c = \sum_{k=1}^{n} \lambda_{j_k} a_{j_k}^T.$$

There are different techniques to obtain an initial ladder. We will discuss in detail how to find such a ladder in Chapter 5.

Remark 3.2. For the TCA, it is noticeable that existence of both an initial ladder and a feasible reference point implies that problem to be solved is inclusive-feasible, that is, the problem to be solved must have an optimal solution.

Remark 3.3. For the CCA, since this algorithm starts with a known ladder, which implies that problem (P) is inclusive, then the output information in Step 2.2 means "Problem (P) is inclusive-infeasible".

Remark 3.4. Step 2.1 of the above CCA involves the selection of a violated constraint, which is probed along the centerline of the current ladder $L(J_k)$. The centerline of the ladder $L(J_k)$ is a line emanating from the base point x^k of $L(J_k)$ and parallel to the center vector v^k of $L(J_k)$ [60]:

$$l(x^k, v^k): \qquad x = x^k + \mu v^k, \quad -\infty < \mu < \infty.$$

Remark 3.5. Different ladder updating criteria are used to choose a violated constraint in the TCA and CCA, which constitutes the main difference between these two algorithms (see Step 2.1). In the TCA, at each iteration a violated constraint is probed by means of the line joining the feasible reference point x_r and the current base point, whereas the CCA selects a violated constraint along the centerline of the current ladder. The ladder updating scheme plays a crucial role in determining the performance of a ladder algorithm. In Chapter 4, we will propose several new ladder algorithms employing various ladder updating strategies, in an effort to find a ladder algorithm with as little computational effort as possible.

Step 2.2 of the above two algorithms deals with determining a drop $j_d^k \in J_k$ such that $J_{k+1} = J_k(j_d^k \leftrightarrow p^k)$ is a ladder generator and the associated base point $x^{k+1} \in L(J_k)$. It was proven in [60] that the requirement $x^{k+1} \in L(J_k)$ is satisfied in the absence of degeneracy. As for how to identify a drop, an efficient procedure was introduced in the above-mentioned paper, which we present as below:

Procedure for identifying a drop in the absence of degeneracy:

Given the current ladder $L(J_k)$ with $J_k = \{j_1^k, j_2^k, \cdots, j_n^k\}$,

1. Calculate
$$\delta^k = [\delta_1^k; \delta_2^k \cdots; \delta_n^k] = -[A^T(J_k)]^{-1}c,$$

and

$$\gamma^k = [\gamma_1^k; \gamma_2^k \cdots ; \gamma_n^k] = [A^T(J_k)]^{-1} a_{p^k}^T.$$

2. - If at least one $\gamma_j^k > 0$ $(1 \leq j \leq n)$, determine an index d $(1 \leq d \leq n)$ such that

   $$\frac{\delta_d^k}{\gamma_d^k} = \min\left\{ \frac{\delta_j^k}{\gamma_j^k} \middle| \gamma_j^k > 0, 1 \leq j \leq n \right\}. \tag{3.3}$$

   The associated index $j_d^k \in J_k$ is the drop.

   - Otherwise, the problem is infeasible.

As seen from the TCA and CCA, at each iteration the inverse of the matrix is involved. It is not necessary to recalculate the inverse of the matrix afresh at each iteration due to the change in a row or column. In [60], the following procedure was adopted to update the inverse of the matrix. Suppose that at the k-th iteration $[A(J_k)]^{-1}$ is known, $[A(J_{k+1})]^{-1}$ is computed according to

$$[A(J_{k+1})]^{-1} = [A(J_k)]^{-1} E_k^T, \tag{3.4}$$

where

$$E_k = [e_1, e_2, \cdots , e_{d-1}, \bar{\gamma}^k, e_{d+1}, \cdots , e_n],$$

and

$$\bar{\gamma}^k = \frac{1}{\gamma_d^k}[-\gamma_1^k; -\gamma_2^k; \cdots ; -\gamma_{d-1}^k; 1; -\gamma_{d+1}^k; \cdots ; -\gamma_n^k],$$

and e_i is the unit column vector with the i-th component 1.

Chapter 4

Improved Ladder Algorithms for Linear Programming

4.1 Introduction

As previously mentioned in Chapter 3, a ladder updating scheme plays an important role in determining the performance of a ladder algorithm. In particular, the selection of a violated constraint is crucial, directly influencing the efficiency of a ladder algorithm. In this chapter, we propose new ladder algorithms employing various ladder updating criteria for selecting a violated constraint, in an effort to improve the ladder method. Still we are concerned with problems with inequality constraints, which are restated here

$$
\text{(P)}: \qquad \begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \leq b, \end{array}
$$

where $A \in R^{m\times n}$ with $m \geq n$. Throughout this chapter, we assume the problem is non-degenerate and use the same notation as introduced in Section 3.2 in Chapter 3. Still, we assume that rank$(A) = n$. We denote by $\|\cdot\|$ the Euclidean norm in R^n.

This chapter is organized as follows. We develop a ladder algorithm termed “an improved targeted climbing algorithm ”(ITCA) in Section 4.2. As the term suggests, the ITCA is a variant of the original targeted climbing algorithm. To show the efficiency of this new algorithm, we also provide an illustration, as well as computational implementation. In Section 4.3, we introduce several ladder updating rules for picking up a violated constraint in the framework of CCA, and examine these rules on the famous Klee-Minty problem and randomly generated test problems.

4.2 An Improved Targeted Climbing Algorithm (ITCA)

4.2.1 Motivation

In the TCA, at each iteration, a fixed reference point is used in the ladder updating scheme to determine a violated constraint. We present in the following a variant of the targeted climbing algorithm. Instead of using a fixed reference point in the ladder updating scheme, as in the original targeted climbing algorithm, we use a different reference point (at each iteration). The reference points used in the new algorithm are feasible and have improved objective values. We can see geometrically (see Figure 4.1) that a reference point with a smaller objective value provides a better chance of choosing a more efficient constraint to form the next ladder and hence enhance the rate of convergence. In particular, the proposed algorithm is efficient for problems with a number of constraints near an optimal solution.

4.2.2 Description of Algorithm

An Improved Targeted Climbing Algorithm (ITCA):

Step 0 Initialization.

Start with a feasible reference point and a known ladder generator, which are denoted by x_r^0 and $J_0 = \{j_1^0, j_2^0, \cdots, j_n^0\} \subset \mathcal{J}$, respectively. (Refer to

Chapter 5 for how to find such a reference point and generator if they are not immediately available.) Denote by $x^0 = x_{J_0}$ the base point associated with J_0. Compute the initial base point $x^0 = [A(J_0)]^{-1}b(J_0)$. Set $k = 0$ and $D_{k-1} = \emptyset$.

Step 1 Optimality criterion.

Let $V^k = \{j \in \mathcal{J} \backslash (J_k \bigcup D_{k-1}) : a_j x^k > b_j\}$.

- If $V^k = \emptyset$, exit with "Problem (P) has an optimal solution $x^* = x^k$".
- Otherwise, go to Step 2.

Step 2 Updating the ladder.

2.1 Selecting an index from V^k.

Identify an index $p^k \in V^k$ as a pick such that $t_{p^k} = \max_{j \in V^k} \{t_j\}$ where

$$t_j = \frac{a_j x^k - b_j}{a_j(x^k - x_r^k)}, \; j \in V^k. \tag{4.1}$$

2.2 Updating the reference point.

Set

$$x_r^{k+1} = x^k + t_{p^k}(x_r^k - x^k). \tag{4.2}$$

2.3 Dropping an index from J_k.

Find an index $j_d^k \in J_k$ as a drop such that $J_{k+1} = J_k(j_d^k \leftrightarrow p^k)$ is a ladder generator and the associated base point $x^{k+1} \in L(J_k)$.

2.4 Updating.

Denote the elements of J_{k+1} by $J_{k+1} = \{j_1^{k+1}, j_2^{k+1}, \cdots, j_n^{k+1}\}$. Let $D_k = \{j_d^k\}$. Compute the updated base point $x^{k+1} = [A(J_{k+1})]^{-1}b(J_{k+1})$. Set $k := k + 1$. Return to Step 1.

Note that, existence of both an initial ladder and a feasible reference point implies that problem (P) is inclusive-feasible, that is, problem (P) can achieve optimality.

Remark 4.1. The proposed algorithm is different from the TCA (see Subsection 3.2.2) in the following aspect. In the original targeted climbing algorithm a stationary reference point is adopted during the process of iterations, whereas in the proposed algorithm the reference point is dynamic and is updated at each iteration (see Step 2.2). Due to the fact that the sequence of reference points moves in the descending direction, changing the reference point at each iteration increases the chance that the active constraints at the optimal solution are picked up, thus speeding up the solution process and reducing the number of iterations.

Remark 4.2. In order to implement the improved targeted climbing algorithm, there must be an initial ladder and a feasible reference point. We can adopt the method introduced in Chapter 5 to obtain a known ladder and a feasible point. For details, see Chapter 5.

Remark 4.3. The iterative process possesses the following properties:

i) The sequence of reference points $\{x_r^k\}_{k=0}^N$ converges to the optimal solution x^*, where N denotes the number of iterations.

ii) The sequence $\{c^T x^k - c^T x^*\}_{k=0}^N$ is non-positive and increasing.

iii) The sequence $\{c^T x_r^k - c^T x^*\}_{k=0}^N$ is non-negative and decreasing.

Remark 4.4. In Step 2.3 a drop needs to be identified. We can follow the procedure presented in Subsection 3.2.2 to determine a drop.

Remark 4.5. In light of the above properties ii) and iii), it is clear that the improved targeted climbing algorithm is finite on non-degenerate LP problems.

4.2.3 An Illustration

For illustration, we present the following simple example to show how the improved targeted climbing algorithm works.

Example 4.1. Consider the following linear programming problem:

$$\begin{aligned} \min \quad & x_1 + 8x_2 \\ \text{s.t.} \quad & x_1 - x_2 \leq 0 \\ & -x_1 - x_2 \leq 0 \\ & -4x_1 - 3x_2 \leq -12 \\ & 5x_1 - 4x_2 \leq -20 \\ & -x_1 \leq 1. \end{aligned}$$

Take $x_r^0 = [0; 10]$ as the initial reference point. An initial ladder generator $J_0 = \{1, 2\}$ with base point $x^0 = [0; 0]$ is easily identified. Following the improved targeted climbing algorithm, we need to update the ladder generator twice (the subsequent ladder generators are $J_1 = \{2, 4\}$ and $J_2 = \{3, 4\}$) and change the reference point once (the updated reference point is $x_r^1 = [0; 5]$) for obtaining the optimal solution $x^* = [-\frac{12}{31}; \frac{140}{31}]$. See Figure 4.1 for a geometric illustration.

For comparison, the targeted climbing algorithm (for details, see [60]) is also implemented to solve the same problem. Still we take $[0; 10]$ as the initial reference point. Starting from the initial ladder generator $J_0 = \{1, 2\}$ with base point $x^0 = [0; 0]$, the optimal solution is reached by three iterations. The subsequent ladder generators are $J_1 = \{2, 4\}$, $J_2 = \{4, 5\}$ and $J_3 = \{3, 4\}$.

4.2.4 Computational Experiments

In this section, we present results of numerical tests (without dealing with large-scale and sparse problems) to show the efficiency of the improved targeted climbing algorithm. We implemented the proposed algorithm in the MATLAB environment (MATLAB 7.7.0 (R2008b)) and ran the tests on a desk-top computer (HP Compaq, Intel Core 2 Duo, 3.16GHz, 3.48GB RAM) under the Microsoft Windows XP operating system. The machine precision used is 16 decimal places. For sake of comparison, the original targeted climbing algorithm, the linprog solver (the medium-scale simplex algorithm was implemented) in MATLAB optimization toolbox (Version 4.1 (R2008b)) and lpSimplex solver in TOMLAB (Version 7.6) were used for solving the same test problems. Our objective is to

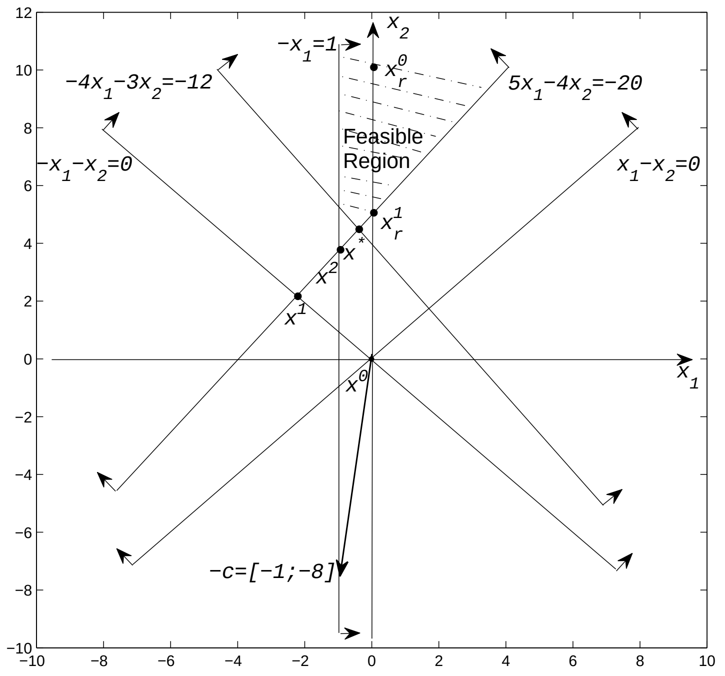

Figure 4.1: Geometric illustration of Example 4.1

compare the CPU time. Since our algorithms and simplex algorithms actually work on problems of different forms and dimensions, the number of iterations does not provide us much helpful information. Therefore, here we do not take a comparison of iteration numbers into consideration.

Example 4.2. [60] (Randomly generated feasible problems) Generate a linear programming problem by specifying $A \in R^{m\times n}$, $c = [c_1; c_2; \cdots ; c_n] \in R^n$, and $b = [b_1; b_2; \cdots ; b_m] \in R^m$ in the following method.

1. Randomly generate $c \in R^n$ and a vector $\bar{x} \in R^n$ such that components of c take values between -25 and 25, and components of $\bar{x}$ between 0 and 20.

2. Generate A and b by two steps.

   (a) For $1 \le j \le n$, the j-th row a_j of A is $a_j = -c^T + 2\text{sign}(c_j)e_j$, where e_j is the j-th row of the $n \times n$ identity matrix. Then, b_j is defined by $b_j = a_j\bar{x} + \kappa_j$, where κ_j is a random number in (0, 1).

   (b) For $n + 1 \le j \le m$, randomly generate a row vector $\alpha_j \in R^n$ and a number $\beta_j \in R$ such that β_j and all the components of α_j are between

-25 and 25. If $\alpha_j \bar{x} \leq \beta_j$, then the j-th row a_j of A and the j-th element of b are defined by $a_j = \alpha_j,\ b_j = \beta_j$. Otherwise, they are defined by $a_j = -\alpha_j,\ b_j = -\beta_j$.

Computational results for running 20 problems with less than 1000 variables and constraints are reported in Table 4.1. We had to give up test problems with greater dimensions due to the limitation of computer memory.

As seen from Table 4.1, numerical test results are encouraging. Our tests show that for problems whose number of variables (n) is between 100 and 200, the current algorithm is faster than linprog solver and lpSimplex solver when the number of constraints (m) is approximately within the range of $1.4n < m \leq 4n$ (for $100 \leq n \leq 200$). For problems with at least 200 variables, our algorithm performs better when the number of constraints is at least 50 more than the number of variables. It seems when n increases, the range for m also increases.

It is worth mentioning that the proposed ladder algorithm has significant advantages over the simplex method for solving LP problems with overly stringent constraints. This class of problems can be solved by the proposed ladder algorithm and ladder algorithms in general, while the linprog solver which implements the the medium-scale simplex algorithm from MATLAB optimization toolbox fails to solve the same problems.

We would also like to point out that in the present code we adopt the traditional technique of the inverse of matrix to calculate base points. If advanced numerical techniques were adopted, the algorithmic performance could be improved.

4.3 Ladder Updating Criteria in the Framework of CCA

In the framework of CCA (see Subsection 3.2.2 in Chapter 3), we restrict attention to the k-th iteration of this ladder algorithm and examine several ladder updating criteria for picking up a violated constraint in Step 2.1.

4.3.1 Description of Criteria

- **Criterion 1**

We can select a violated constraint along c direction. The detailed rule is introduced as follows.

If c is orthogonal to at least one of the a_j^T's ($j \in V^k$), arbitrarily choose $p^k \in V^k$ such that $a_{p^k} c = 0$.

Otherwise, select $p^k \in V^k$ such that $t_{p^k} = \max\limits_{j \in V^k} \{t_j\}$, where

$$t_j = \frac{b_j - a_j x^k}{a_j c}. \tag{4.3}$$

A distinct feature of the above rule is its simplicity and ease of implementation. ■

- **Criterion 2**

Select $p^k \in V^k$ such that $t_{p^k} = \max\limits_{j \in V^k} \{t_j\}$, where

$$t_j = \frac{a_j v^k}{\|a_j\|}, \tag{4.4}$$

and v^k is the center vector of the current ladder $L(J_k)$.

At the k-th iteration, a violated constraint is selected whose associated outer normal vector forms the minimum angle with the center vector.■

4.3.2 A Specific Example

We take the famous Klee-Minty problem with $n = 3$ (see, e.g., Vanderbei [89], Klee and Minty [52]) for illustration to examine the efficiency of the above criteria. For comparison, we also use the simplex method to solve the same problem.

Example 4.3. Consider the following Klee-Minty problem with $n = 3$

$$\begin{aligned} \min \quad & -100x_1 - 10x_2 - x_3 \\ \text{s.t.} \quad & x_1 \le 1 \\ & 20x_1 + x_2 \le 100 \end{aligned}$$

$$
\begin{aligned}
&200x_1 + 20x_2 + x_3 \leq 10000\\
&x_1,\ x_2,\ x_3 \geq 0.
\end{aligned}
$$

On the one hand, we use the simplex method to solve the above problem. Introducing the slack variables $s_1,\ s_2,\ s_3 \geq 0$, write the above problem as the standard form

$$
\begin{aligned}
\min\quad & -100x_1 - 10x_2 - x_3\\
\text{s.t.}\quad & x_1 \qquad\qquad\quad + s_1 = 1\\
& 20x_1 + x_2 \qquad\quad + s_2 = 100\\
& 200x_1 + 20x_2 + x_3 \quad + s_3 = 10000\\
& x_1,\ x_2,\ x_3,\ s_1,\ s_2,\ s_3 \geq 0.
\end{aligned}
$$

Tables 4.2–4.3 show that the simplex method with the most negative rule requires $2^n - 1 = 2^3 - 1 = 7$ iterations to attain optimality.

On the other hand, we solve the same problem using the ladder method. Firstly we rewrite all constraints as $\leq$-type:

$$
\begin{aligned}
\min\quad & -100x_1 - 10x_2 - x_3\\
\text{s.t.}\quad & x_1 \leq 1\\
& 20x_1 + x_2 \leq 100\\
& 200x_1 + 20x_2 + x_3 \leq 10000\\
& -x_1 \leq 0\\
& -x_2 \leq 0\\
& -x_3 \leq 0.
\end{aligned}
$$

Here an initial ladder is not immediately available for this problem. To find an initial ladder, we add an artificial constraint $x_1 + x_2 + x_3 \leq M$. (See Section 5.4, Chapter 5 for construction detail of the initial ladder.) This results in the problem with the additional constraint as below:

$$
\begin{aligned}
\min\quad & -100x_1 - 10x_2 - x_3\\
\text{s.t.}\quad & x_1 \leq 1\\
& 20x_1 + x_2 \leq 100\\
& 200x_1 + 20x_2 + x_3 \leq 10000\\
& -x_1 \leq 0\\
& -x_2 \leq 0
\end{aligned}
$$

$$-x_3 \leq 0$$

$$x_1 + x_2 + x_3 \leq M,$$

where M is a sufficiently large real number.

In the next chapter we will discuss in detail how to construct an initial ladder. The interested reader can temporarily skip to Section 5.4 in Chapter 5 to acquire the construction detail of an initial ladder.

It is easy to verify that the index set $\{7,\ 5,\ 6\}$ is an initial ladder generator. With the known ladder generator at hand, it takes four iterations to reach an optimal solution using the ladder algorithm with Criterion 1. And it takes only two iterations to reach an optimal solution using the ladder algorithm with Criterion 2. For solution details, see Table 4.4 and Table 4.5.

Clearly, the ladder algorithms using the above-proposed criteria are much more efficient for solving the Klee-Minty problem. Firstly, our algorithms require no additional variables (slack, surplus and artificial variables). Secondly, the number of iterations is reduced greatly. In addition, we would like to address that although here we use an example with non-negative variables to illustrate the efficiency of our algorithms, there is no non-negativity requirement for variables in our problem form. Thus, our algorithms are suitable for a wide range of LP problems.

4.3.3 Computational Experiments

In this subsection, we carry out computational experiments on randomly generated problems from Example 4.2 to examine numerical behavior of the new ladder algorithms (with the foregoing two ladder updating criteria implemented). The following codes were tested and compared against one another:

- LCR1: Using the ladder algorithm with Criterion 1.
- LCR2: Using the ladder algorithm with Criterion 2.
- CCA: Using the centered climbing algorithm.
- linprog: Using the medium-scale simplex algorithm.

Except for the linprog solver which is from the MATLAB optimization toolbox (Version 5.1 (R2010b)), the others were coded in the MATLAB environment (MATLAB 7.11.0 (R2010b)). Tests were run on a desk-top computer (HP Intel(R) Core(TM), i7-2600 CPU@3.40GHz, 3.39GHz, 3.24GB of RAM) under the Microsoft Windows XP operating system. The machine precision used is 16 decimal places. Tables 4.6–4.7 present computational results for 20 test problems with various dimensions. The average CPU time is reported in seconds. Since our algorithm and the simplex method actually work on problems with different forms and dimensions, the number of iterations does not provide much helpful information. Therefore, here we do not take a comparison of iteration numbers into consideration.

Tables 4.6–4.7 reveal that, the ladder algorithm using Criterion 2 has surprisingly excellent performance, particularly for the case where $m - n = 100$. We would like to point out that in the present code we adopt the traditional technique of the inverse of matrix to calculate base points. If advanced numerical techniques were incorporated into the current code, then computational performance may be improved.

Table 4.1: Average CPU time (in seconds) for Example 4.2

Size	Algorithms			
$(m,\ n)$	ITCA	TCA	linprog	lpSimplex
(130, 100)	1.2559	1.2725	1.1221	1.1397
(150, 100)	1.5773	1.6648	1.8008	2.1820
(280, 100)	5.2721	7.2318	8.2370	20.4518
(470, 100)	19.1250	26.9861	18.9201	74.6580
(165, 125)	1.8799	1.9597	1.8380	1.7936
(175, 125)	2.0703	2.1893	2.6809	2.5811
(245, 125)	3.8125	5.2272	7.4808	9.0012
(525, 125)	25.6830	40.8002	32.9319	128.9565
(180, 150)	2.2432	2.3193	1.8086	2.3193
(370, 150)	11.6641	20.3105	24.2402	67.1992
(750, 150)	70.3969	122.8031	84.9000	395.3281
(220, 200)	3.4229	3.4323	2.2198	1.1927
(280, 200)	5.6547	7.6266	11.3234	18.8266
(400, 200)	13.9340	29.7750	42.2090	135.2770
(600, 200)	44.3320	99.0390	93.8906	521.7660
(335, 300)	8.5156	9.0495	7.3919	7.4297
(350, 300)	9.6853	10.9476	10.4174	13.6071
(550, 300)	34.2148	100.2161	154.4714	471.2057
(900, 350)	177.0365	588.9167	580.5000	1138.6563
(550, 500)	29.3465	32.7013	37.1572	38.5689

Table 4.2: The tableau obtained from simplex for Example 4.3

Iteration		x_1	x_2	x_3	s_1	s_2	s_3	rhs
0	z	-100	-10	-1	0	0	0	0
	s_1	1	0	0	1	0	0	1
	s_2	20	1	0	0	1	0	100
	s_3	200	20	1	0	0	1	10000
1	z	0	-10	-1	100	0	0	100
	x_1	1	0	0	1	0	0	1
	s_2	0	1	0	-20	1	0	80
	s_3	0	20	1	-200	0	1	9800
2	z	0	0	-1	-100	10	0	900
	x_1	1	0	0	1	0	0	1
	x_2	0	1	0	-20	1	0	80
	s_3	0	0	1	200	-20	1	8200
3	z	100	0	-1	0	10	0	1000
	s_1	1	0	0	1	0	0	1
	x_2	20	1	0	0	1	0	100
	s_3	-200	0	1	0	-20	1	8000
4	z	-100	0	0	0	-10	1	9000
	s_1	1	0	0	1	0	0	1
	x_2	20	1	0	0	1	0	100
	x_3	-200	0	1	0	-20	1	8000
5	z	0	0	0	100	-10	1	9100
	x_1	1	0	0	1	0	0	1
	x_2	0	1	0	-20	1	0	80
	x_3	0	0	1	200	-20	1	8200

Table 4.3: The tableau obtained from simplex for Example 4.3 (to continue Table 4.2)

Iteration		x_1	x_2	x_3	s_1	s_2	s_3	rhs
6	z	0	10	0	-100	0	1	9900
	x_1	1	0	0	1	0	0	1
	s_2	0	1	0	-20	1	0	80
	x_3	0	20	1	-200	0	1	9800
7	z	100	10	0	0	0	1	10000
	s_1	1	0	0	1	0	0	1
	s_2	20	1	0	0	1	0	100
	x_3	200	20	1	0	0	1	10000

Table 4.4: The table obtained from the ladder algorithm with Criterion 1 for Example 4.3

Iteration	Ladder generator	Base point	Optimal value
0	{7, 5, 6}	$[M; 0; 0]$	
1	{7, 1, 6}	$[1; M-1; 0]$	
2	{7, 1, 3}	$[1; \frac{9801-M}{19}; \frac{-9820+20M}{19}]$	
3	{7, 5, 3}	$[\frac{10000-M}{199}; 0; \frac{-10000+200M}{199}]$	
4	{4, 5, 3}	$[0; 0; 10000]$	-10000

Table 4.5: The table obtained from the ladder algorithm with Criterion 2 for Example 4.3

Iteration	Ladder generator	Base point	Optimal value
0	{7, 5, 6}	$[M; 0; 0]$	
1	{7, 5, 3}	$[\frac{10000-M}{199}; 0; \frac{-10000+200M}{199}]$	
2	{4, 5, 3}	$[0; 0; 10000]$	-10000

Table 4.6: Average CPU time (in seconds) for test problems in Example 4.2 ($m = 2n$)

Size	Algorithms			
$(m,\ n)$	LCR1	LCR2	CCA	linprog
$(40,\ 20)$	0.0906	0.0843	0.0953	0.6406
$(80,\ 40)$	0.3875	0.3937	0.4375	1.2609
$(120,\ 60)$	0.6281	0.4921	0.5093	2.1656
$(160,\ 80)$	0.9062	0.8484	1.2281	3.9640
$(200, 100)$	1.3875	1.4312	1.775	7.0625
$(240, 120)$	1.8906	2.4078	2.9609	10.3703
$(280, 140)$	4.9375	4.4765	4.7421	18.5488
$(320,\ 160)$	5.8437	6.6347	7.6992	28.5644
$(360, 180)$	8.75	10.9609	12.3066	42.7851
$(400,\ 200)$	18.3812	16.1437	16.9937	61.7343

Table 4.7: Average CPU time (in seconds) for test problems in Example 4.2 ($m - n = 100$)

Size	Algorithms			
$(m,\ n)$	LCR1	LCR2	CCA	linprog
$(600, 500)$	101.1484	35.8554	40.8424	126.6692
$(650, 550)$	113.3549	44.75	57.9263	148.9732
$(700, 600)$	212.5703	52.25	72.5273	185.5312
$(750, 650)$	198.9479	62.2291	73.6718	225.5625
$(800, 700)$	190.0729	75.7187	118.7760	239.6666
$(850, 750)$	799.7265	97.3125	99.4531	240.1953
$(900, 800)$	69.4531	94.9375	122.8281	252.8281
$(950, 850)$	987.1563	112.5312	155.4531	280.9843
$(1000, 900)$	95.625	117.5937	147.9531	345.1093

Chapter 5

Initialization Techniques

The ladder algorithms presented in the previous chapters require an initial ladder at hand, which is not readily available in many cases. Some work may need to be done to find an initial ladder. In this chapter, we introduce several techniques on how to obtain an initial ladder for starting the ladder method.

Throughout this chapter, the reader needs to keep in mind that, to find a ladder $L(J)$ is to find the associated generator $J = \{j_1, j_2, \cdots, j_n\} \subset \mathcal{J}$, equivalently, to find n independent outward normal vectors a_{j_k} $(k = 1, 2, \cdots, n)$ such that there exist n constants $\lambda_{j_k} \geq 0$ $(k = 1, 2, \cdots, n)$ satisfying

$$-c = \sum_{k=1}^{n} \lambda_{j_k} a_{j_k}^T.$$

5.1 Finding an Initial Ladder in Special Cases

For the case that problem (P) includes constraints taking the form of $-\text{sign}(c_i)x_i \leq d_i$ for all $c_i \neq 0$, $i = 1, 2, \cdots, n$, where d_i are constants, any independent set of n indices including those corresponding to the non-zero rows of diag$[-\text{sign}(c)]$ is a ladder generator, where $\text{sign}(c) = [\text{sign}(c_1); \text{sign}(c_2); \cdots ;$ $\text{sign}(c_n)]$.

For another case that the feasible region $\mathcal{F}$ of problem (P) satisfies

$$\mathcal{F} \subset \{x \in R^n : (-\text{sign}(c_i) - |\text{sign}(c_i)| + 1)x_i \leq d_i,\ i = 1, 2, \cdots, n\},$$

adding n constraints with the form of $(-\text{sign}(c_i) - |\text{sign}(c_i)| + 1)x_i \leq d_i,\ i = 1, 2, \cdots, n$, obtains a new problem with $m + n$ constraints (which has the same solutions as the original problem). The corresponding index set associated with n added constraints generates a ladder. For details, see [60].

5.2 Finding an Initial Ladder by Means of the Dual System

Consider the dual system of problem (P)

(D):
$$\begin{aligned} &\min \quad b^T y \\ &\text{s.t. } A^T y = -c \\ &\qquad y = [y_1; y_2; \ldots; y_m] \geq 0. \end{aligned}$$

Observing the dual system (D), we know that $J = \{j_1, j_2, \cdots, j_n\} \subset \mathcal{J}$ is a ladder generator of problem (P) if and only if the dual problem (D) has a solution y^* which satisfies

$$y_j^* \geq 0 \quad \text{for} \quad j \in J \quad \text{and} \quad y_j^* = 0 \quad \text{for} \quad j \in \mathcal{J} \setminus J.$$

In the following, let us see how such a y^* can be found if it does exist, which will lead to a ladder of problem (P). To achieve this goal, Liu [60] constructed a problem as below:

(P1):
$$\begin{aligned} &\min \quad \tilde{b}^T y \\ &\text{s.t.} \begin{bmatrix} A^T \\ -I_m \end{bmatrix} y \leq \begin{bmatrix} -c \\ 0_{m\times 1} \end{bmatrix}, \end{aligned}$$

where $\tilde{b} = -[\sum_{i=1}^n a_{1i}; \sum_{i=1}^n a_{2i}; \cdots; \sum_{i=1}^n a_{mi}]$. Noting that $-A^T y \geq c$ implies $\tilde{b}^T y \geq \sum_{i=1}^n c_i$, it is easy to see that the dual system (D) allows an optimal solution y^* if and only if y^* is an optimal solution of problem (P1) with the optimal value $\tilde{b}^T y^* = \sum_{i=1}^n c_i$.

Also note that, any independent index set of the form $J = \{1, 2, \cdots, n, n + i_1, n + i_2, \cdots, n + i_{m-n}\} \subset \{1, 2, \cdots, m + n\}$ is a ladder generator of the above problem (P1), where $\{i_1, i_2, \cdots, i_{m-n}\} \subset \{1, 2, \cdots, m\}$. Therefore, problem (P1) can be solved using CCA or any ladder algorithm in the frame of CCA (see Subsection 4.3.1). In the progress of solving problem (P1), it can be detected whether problem (P) has a ladder and an initial ladder of (P) can be found by means of (P1) if it has. The following three cases can occur.

- *Case A*: Problem (P1) is infeasible. (Note that the case of unboundedness cannot occur since problem (P1) has an ladder.) In this case problem (D) is infeasible. Hence problem (P) has no inclusive cone, which implies problem (P) is infeasible or unbounded.

- *Case B*: Problem (P1) has an optimal solution y^* with the optimal value $\tilde{b}^T y^* > \sum_{i=1}^n c_i$. In this case problem (D) has no solution. Hence problem (P) has no inclusive cone, which implies problem (P) is infeasible or unbounded.

- *Case C*: Problem (P1) has an optimal solution y^* satisfying the optimal value $\tilde{b}^T y^* = \sum_{i=1}^n c_i$. Let $K = \{j : y_j^* > 0, 1 \leq j \leq m\}$. In view of the special form of constraints of problem (P1), clearly at least $m - n$ of the active constraints are from $-I_m y \leq 0$. Therefore, y^* has at least $m - n$ zero components and hence at most n positive components. This means the number of elements in K, denoted by k_1, satisfies $k_1 \leq n$. Noting that y^* is also an optimal solution of the dual system (D), we have $-c = \Sigma_{k=1}^{k_1} y_{j_k}^* a_{j_k}^T$. Therefore, a ladder generator J can be easily obtained by expanding K to an independent set of n indices.

5.3 Finding an Initial Ladder by Means of Transformation

We can use the following technique to find an initial ladder and a feasible point for starting ITCA. This method was first proposed by Liu [60] to start TCA, which we restate as follows.

Combining primal problem (P) and its dual problem (D), the following problem is constructed:

$$
\text{(P2):}\qquad
\begin{aligned}
\min\quad & \xi \\
\text{s.t.}\quad & -(c^T x_0 + b^T y_0)\xi + c^T x + b^T y = 0 \\
& -(Ax_0 + z_0 - b)\xi + Ax + z = b \\
& -(c + A^T y_0)\xi + A^T y = -c \\
& -\xi \le 0 \\
& -y \le 0 \\
& -z \le 0,
\end{aligned}
$$

where $x_0 \in R^n, y_0 \in R^m_+$ and $z_0 \in R^m_+$ are chosen in such a way that y_0 and z_0 have strictly positive components and $c^T x_0 + b^T y_0 \neq 0$.

Observing the first constraint of problem (P2), we know that $(c^T x_0 + b^T y_0)\xi$ represents duality gap between primal problem (P) and its dual problem (D). It is clear that problem (P2) has a feasible solution $[\xi; x; y; z] = [1; x_0; y_0; z_0]$. In addition, noting that problem (P2) contains constraint $-\xi \le 0$, it is easy to know that problem (P2) is inclusive. Hence, problem (P2) can achieve optimality. Denote by $[\xi^*; x^*; y^*; z^*]$ an optimal solution of problem (P2). If $\xi^* = 0$, then we have

$$c^T x^* + b^T y^* = 0,$$

$$Ax^* \le b,$$

$$A^T y^* = -c,$$

which implies that x^* and y^* are the optimal solutions of problem (P) and its dual problem (D), respectively. Otherwise, if $\xi^* > 0$, then problem (P) is infeasible or unbounded since the duality gap $(c^T x_0 + b^T y_0)\xi^* \neq 0$.

In the remaining of this section, we focus on how to reduce problem (P2) to a problem with less constraints and variables, which contains a feasible reference point and an initial ladder, and thus can be solved using TCA or ITCA.

Rewrite problem (P2) as the equivalent matrix form as below:

$$\begin{aligned}
&\min \quad \xi \\
&\text{s.t.} \begin{bmatrix} -(c^T x_0 + b^T y_0) & c^T & b^T & 0_{1\times m} \\ -(Ax_0 + z_0 - b) & A & 0_{m\times m} & I_m \\ -(c + A^T y_0) & 0_{n\times n} & A^T & 0_{n\times m} \end{bmatrix} \begin{bmatrix} \xi \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ b \\ -c \end{bmatrix} \\
&\qquad \begin{bmatrix} -1 & 0_{1\times n} & 0_{1\times m} & 0_{1\times m} \\ 0_{m\times 1} & 0_{m\times n} & -I_m & 0_{m\times m} \\ 0_{m\times 1} & 0_{m\times n} & 0_{m\times m} & -I_m \end{bmatrix} \begin{bmatrix} \xi \\ x \\ y \\ z \end{bmatrix} \le 0_{(2m+1)\times 1}.
\end{aligned}$$

With the assumption that $c^T x_0 + b^T y_0 = 1$ and ξ and z eliminated, the above problem is reduced to

$$\begin{aligned}
&\min \begin{bmatrix} c^T & b^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \\
&\text{s.t.} \begin{bmatrix} (c + A^T y_0)c^T & (c + A^T y_0)b^T - A^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = c \qquad (5.1) \\
&M_1 \left(M_2 \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0_{(m+n+1)\times 1} \\ b \end{bmatrix} \right) \le 0_{(2m+1)\times 1},
\end{aligned}$$

where

$$M_1 = -\begin{bmatrix} 1 & 0_{1\times n} & 0_{1\times 2m} \\ 0_{2m\times 1} & 0_{2m\times n} & I_{2m} \end{bmatrix}$$

$$M_2 = \begin{bmatrix} c^T & b^T \\ I_n & 0_{n\times m} \\ 0_{m\times n} & I_m \\ (Ax_0 + z_0 - b)c^T - A & (Ax_0 + z_0 - b)b^T \end{bmatrix}.$$

Further we solve the system of linear equations (5.1) to remove n of y's components. That is, we can express n of y's components in terms of x and $\hat{y}$, where $\hat{y}$ consists of y's remaining $m - n$ components (see Appendix A for elimination procedure). Suppose this gives

$$\begin{bmatrix} x \\ y \end{bmatrix} = M_3 \begin{bmatrix} x \\ \hat{y} \end{bmatrix} + c_0^{'},$$

where M_3 is an $(m+n) \times m$ matrix and $c_0^{'}$ is an $(m+n)$-vector. This will lead to the following problem

(P3): $\min \quad \hat{c}^T \hat{x}$

$\text{s.t.} \quad \hat{A}\hat{x} \leq \hat{b},$

where $\hat{x} = [x;\ \hat{y}]$, $\hat{c} = M_3^T[c;\ b]$, $\hat{b} = -M_1(M_2 c_0^{'} + [0_{(m+n+1)\times 1};\ b])$, and $\hat{A} = M_1 M_2 M_3$. Noting that the normal vector of the first constraint in problem (P3) is $-\hat{c}^T$, it is obvious that any m independent indices including index 1 is a ladder generator of (P3). Meanwhile, a feasible interior point of (P3) is given by $\hat{x}_r = [x_0;\ \hat{y}_0]$, where x_0 satisfies $c^T x_0 + b^T y_0 = 1$ and $\hat{y}_0$ is formed from y_0 in the same way as $\hat{y}$ is formed from y.

With an initial ladder and a feasible point at hand, problem (P3) can be solved using ITCA or TCA. Denote by $\hat{x}^* = [x^*;\ \hat{y}^*]$ an optimal solution to problem (P3). Let

$$\xi^* = [c^T\ b^T]M_3[x^*;\ \hat{y}^*] + [c^T\ b^T]c_0^{'}.$$

If $\xi^* = 0$, then problem (P) has an optimal solution x^*. Otherwise, problem (P) is infeasible or unbounded.

5.4 Constructing an Artificial Ladder

We can also adopt the single artificial constraint technique to construct an initial ladder for starting the ladder method. This technique is typically used in the dual simplex method for finding a dual feasible basis. While applying this technique to the dual simplex method, an extra constraint taking the form of $\sum_{i=1}^{n} x_i + x_{n+1} = M$ needs to be appended to the original problem in the standard form (see problem ($\mathrm{P_S}$) in Subsection 2.1.1), where $x_{n+1} \geq 0$ is an artificial variable and M is a sufficiently large number. Undoubtedly, introduction of the artificial variable results in extra computational load. In the following, we will adopt a similar technique to construct an artificial ladder for starting the ladder method. Different from the aforementioned technique adopted in the dual simplex method, the ladder construction technique does not involve any additional variable. More importantly, as we will see in Subsection 5.4.2, this technique will lead to an inclusive-cone-based solvability criterion for LP in the context of the new category of LP problems, which enables us to identify accurately that an LP problem is inclusive-feasible (i.e., optimal), noninclusive-feasible (unbounded), inclusive-infeasible or noninclusive-infeasible.

5.4.1 Approach to Construction

In most practical settings variables are usually bounded. Therefore we first deal with the cases in which variables are bounded, then we consider the general case in which variables are unrestricted in sign.

Case 1: Variables with upper bounds

Here we consider the case where variables of problem (P) have upper bounds. Temporarily, we assume that at least one component of c is positive. For convenience of discussion, write the problem as below:

(P4):
$$\begin{aligned} \min\ & c_1x_1 + c_2x_2 + \cdots + c_dx_d + \cdots + c_nx_n \\ \text{s.t.}\ & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\ & \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ & a_{(m-n)\times 1}x_1 + a_{(m-n)\times 2}x_2 + \cdots + a_{(m-n)\times n}x_n \le b_{m-n} \\ & x_i \le b_{m-n+i},\ i = 1, 2, \cdots, d, \cdots, n. \end{aligned}$$

With this assumption that c contains at least one positive component, it is easily seen that the index set $\{m-n+1, m-n+2, \cdots, m-n+d, \cdots, m\}$ (corresponding to the bound constraints $x_i \le b_{m-n+i},\ i = 1, 2, \cdots, n$) is not a ladder generator. In order to obtain a ladder for the above problem, we add an artificial constraint $-\sum_{i=1}^{n} x_i \le M$, where M is a sufficiently large number. For clarity, display the problem with the additional constraint as follows:

$$
\begin{aligned}
\text{(PM4):} \qquad & \min \ c_1x_1 + c_2x_2 + \cdots + c_dx_d + \cdots + c_nx_n \\
& \text{s.t.} \ \ a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\
& \qquad a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\
& \qquad \cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
& \qquad a_{(m-n)\times 1}x_1 + a_{(m-n)\times 2}x_2 + \cdots + a_{(m-n)\times n}x_n \le b_{m-n} \\
& \qquad x_i \le b_{m-n+i}, \ i = 1, 2, \cdots, d, \cdots, n \\
& \qquad -x_1 - x_2 - \cdots - x_n \le M.
\end{aligned}
$$

Executing the following simple procedure, we can readily find an initial ladder for problem (PM4). To begin, take $J = \{m-n+1, m-n+2, \cdots, m-n+d, \cdots, m\}$. Let $c_d = \max\limits_{1\le i\le n} \{c_i\} > 0$ $(1 \le d \le n)$. Select $j = m - n + d$ as a drop (the associated constraint is $x_d \le b_{m-n+d}$) and $p = m + 1$ as a pick (the associated constraint is $-\sum_{i=1}^{n} x_i \le M$). It is easy to verify that $J(j \leftrightarrow p)$ is a ladder generator of problem (PM4). Indeed, from

$$
-\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{d-1} \\ c_d \\ c_{d+1} \\ \vdots \\ c_n \end{bmatrix}
=
\begin{bmatrix}
1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\
0 & 1 & \cdots & 0 & -1 & 0 & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & \cdots & 1 & -1 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & -1 & 1 & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & \cdots & 0 & -1 & 0 & \cdots & 1
\end{bmatrix}
\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{d-1} \\ \lambda_d \\ \lambda_{d+1} \\ \vdots \\ \lambda_n \end{bmatrix},
$$

we have

$$
\lambda_i = c_d - c_i \ge 0 \ (i \ne d), \quad \lambda_d = c_d > 0,
$$

which implies $J(j \leftrightarrow p)$ is a ladder generator of the above problem.

Case 2: Variables with lower bounds

In this part, we consider the case in which variables of problem (P) have lower bounds. For convenience of discussion, we rewrite the problem in the following form:

$$
\begin{aligned}
\text{(P5):} \qquad & \min \ c_1x_1 + c_2x_2 + \cdots + c_dx_d + \cdots + c_nx_n \\
& \text{s.t.} \ a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\
& \quad a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\
& \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
& \quad a_{(m-n)\times 1}x_1 + a_{(m-n)\times 2}x_2 + \cdots + a_{(m-n)\times n}x_n \le b_{m-n} \\
& \quad -x_i \le b_{m-n+i}, \ i = 1, 2, \cdots, d, \cdots, n.
\end{aligned}
$$

Note that here we use the same notations in problem (P5) as in (P4) for convenience. In this case, we temporarily assume that c contains at least one negative component. With this assumption, it is easy to be seen that the index set $\{m-n+1, m-n+2, \cdots, m-n+d, \cdots, m\}$ (corresponding to the bound constraints $-x_i \le b_{m-n+i}, \ i = 1, 2, \cdots, n$) is not a ladder generator. Adding an artificial constraint $\sum_{i=1}^{n} x_i \le M$, where M is a sufficiently large number, we get the following system:

$$
\begin{aligned}
\text{(PM5):} \qquad & \min \ c_1x_1 + c_2x_2 + \cdots + c_dx_d + \cdots + c_nx_n \\
& \text{s.t.} \ a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\
& \quad a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\
& \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
& \quad a_{(m-n)\times 1}x_1 + a_{(m-n)\times 2}x_2 + \cdots + a_{(m-n)\times n}x_n \le b_{m-n} \\
& \quad -x_i \le b_{m-n+i}, \ i = 1, 2, \cdots, d, \cdots, n \\
& \quad x_1 + x_2 + \cdots + x_n \le M.
\end{aligned}
$$

Performing a similar procedure as in the above Case 1, we can easily obtain an initial ladder for this problem with additional constraint. Initially, take $J = \{m-n+1, m-n+2, \cdots, m-n+d, \cdots, m\}$. Let $c_d = \min_{1 \le i \le n} \{c_i\} < 0 \ (1 \le d \le n)$. Choose $j = m-n+d$ as a drop (the associated constraint is $-x_d \le b_{m-n+d}$) and $p = m+1$ as a pick (the associated constraint is $\sum_{i=1}^{n} x_i \le M$). It is easy to verify that $J(j \leftrightarrow p)$ is a ladder generator of this problem. In fact, from

$$-\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{d-1} \\ c_d \\ c_{d+1} \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} -1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & -1 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & -1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & -1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{d-1} \\ \lambda_d \\ \lambda_{d+1} \\ \vdots \\ \lambda_n \end{bmatrix},$$

we have

$$\lambda_i = -c_d + c_i \geq 0 \ (i \neq d), \quad \lambda_d = -c_d > 0,$$

which implies $J(j \leftrightarrow p)$ is a ladder generator of the above problem.

For illustration, we consider the famous Klee-Minty problem with $n = 3$:

$$\begin{aligned} \min \quad & -100x_1 - 10x_2 - x_3 \\ \text{s.t.} \quad & x_1 \leq 1 \\ & 20x_1 + x_2 \leq 100 \\ & 200x_1 + 20x_2 + x_3 \leq 10000 \\ & x_1,\ x_2,\ x_3 \geq 0. \end{aligned}$$

Rewriting all constraints as $\leq$-type, yields

$$\begin{aligned} \min \quad & -100x_1 - 10x_2 - x_3 \\ \text{s.t.} \quad & x_1 \leq 1 \\ & 20x_1 + x_2 \leq 100 \\ & 200x_1 + 20x_2 + x_3 \leq 10000 \\ & -x_1 \leq 0 \\ & -x_2 \leq 0 \\ & -x_3 \leq 0. \end{aligned}$$

An initial ladder is not immediately at hand for this problem. To find an initial ladder, we add an artificial constraint $x_1+x_2+x_3 \leq M$. For clarity, write the problem with the additional constraint as below.

$$\begin{aligned}
\min\quad & -100x_1 - 10x_2 - x_3 \\
\text{s.t.}\quad & x_1 \leq 1 \\
& 20x_1 + x_2 \leq 100 \\
& 200x_1 + 20x_2 + x_3 \leq 10000 \\
& -x_1 \leq 0 \\
& -x_2 \leq 0 \\
& -x_3 \leq 0 \\
& x_1 + x_2 + x_3 \leq M.
\end{aligned}$$

Following the procedure in Case 2, it is easy to verify that the index set {7, 5, 6} is an initial ladder generator.

If the variables in problem (P) are bounded from both below and above, that is, problem (P) contains bound constraints taking the form of $L_i \leq x_i \leq U_i$ $(1 \leq i \leq n)$, where L_i and U_i denote the lower and upper bounds of x_i and $L_i < U_i$, then after rewriting the above constraints as two constraints $-x_i \leq -L_i$ and $x_i \leq U_i$ we can follow the procedure in either Case 1 or Case 2 to obtain an initial ladder in order to start the ladder method.

Case 3: Variables unrestricted in sign

Now we deal with problem (P) in which the variables are unrestricted in sign. In this case, we can use the conventional variable transformation techniques to convert this problem to an equivalent problem in which the variables have lower bounds. For the equivalent problem, we can take the procedure introduced in Case 2 to construct an initial ladder in order to start the ladder method. We discuss in detail in the following.

Suppose that there are k variables $x_1, x_2, \cdots, x_k$ that are unrestricted in sign in problem (P). We can let $x_i = x_i^+ - x_i^-$, $(x_i^+ \geq 0,\ x_i^- \geq 0,\ i = 1, 2, \cdots, k)$. Replacing each x_i by $x_i^+ - x_i^-$ wherever it appears in this problem, we obtain an equivalent problem which can be formulated into the same form as problem (P5). Then, we can follow the procedure introduced in Case 2 to construct an initial ladder for the converted problem.

Alternatively, we can let $x_i = x_i^{'} - x^{''}$ $(x_i^{'} \geq 0,\ x^{''} \geq 0,\ i = 1, 2, \cdots, k)$. Replacing x_i by $x_i^{'} - x^{''}$ wherever they appear in the problem, we get an equivalent problem which can also be formulated into the same form as problem (P5). Again, using the procedure in Case 2, we can readily construct an initial ladder for the converted problem.

For illustration, we give the following two-dimensional example.

$$\begin{aligned} \min \quad & x_2 \\ \text{s.t.} \quad & x_1 - x_2 \le 1 \\ & -x_1 - x_2 \le 0. \end{aligned}$$

Note that both x_1 and x_2 are unrestricted in sign in the above problem. Since this problem has only two variables, we may readily solve it graphically. However, here we would rather apply our current procedure to solve it, in order to make the reader understand better the aforementioned ladder construction technique.

Applying the first method of variable transformation, we get the following equivalent problem

$$\begin{aligned} \min \quad & x_2^+ - x_2^- \\ \text{s.t.} \quad & x_1^+ - x_1^- - x_2^+ + x_2^- \le 1 \\ & -x_1^+ + x_1^- - x_2^+ + x_2^- \le 0 \\ & -x_1^+ \le 0 \\ & -x_1^- \le 0 \\ & -x_2^+ \le 0 \\ & -x_2^- \le 0. \end{aligned}$$

An initial ladder generator is not immediately at hand. Thus, we add the constraint $x_1^+ + x_1^- + x_2^+ + x_2^- \le M$, leading to the problem

$$\begin{aligned} \min \quad & x_2^+ - x_2^- \\ \text{s.t.} \quad & x_1^+ - x_1^- - x_2^+ + x_2^- \le 1 \\ & -x_1^+ + x_1^- - x_2^+ + x_2^- \le 0 \\ & -x_1^+ \le 0 \\ & -x_1^- \le 0 \\ & -x_2^+ \le 0 \\ & -x_2^- \le 0 \\ & x_1^+ + x_1^- + x_2^+ + x_2^- \le M. \end{aligned}$$

Following the procedure in Case 2, we know $J = \{3, 4, 5, 7\}$ is an initial ladder generator. Then the ladder method can be used to solve the above transformed problem, leading to an optimal solution $[x_1^+; x_1^-; x_2^+; x_2^-] = [80.1359; 79.6359; 69.4768; 69.9768]$. According to $x_i = x_i^+ - x_i^-$, we can reconstruct the solution of the original problem as $[x_1; x_2] = [0.5; -0.5]$.

As an alternative, we can also let $x_i = x_i^{'} - x^{''}, x_i^{'} \geq 0, x^{''} \geq 0$ $(i = 1, 2)$, resulting in the problem as below

$$\begin{aligned} \min \quad & x_2^{'} - x^{''} \\ \text{s.t.} \quad & x_1^{'} - x_2^{'} \leq 1 \\ & -x_1^{'} - x_2^{'} + 2x^{''} \leq 0 \\ & -x_1^{'} \leq 0 \\ & -x_2^{'} \leq 0 \\ & -x^{''} \leq 0. \end{aligned}$$

An initial ladder generator is not immediately available. Thus, introducing the constraint $x_1^{'} + x_2^{'} + x^{''} \leq M$, we obtain the problem

$$\begin{aligned} \min \quad & x_2^{'} - x^{''} \\ \text{s.t.} \quad & x_1^{'} - x_2^{'} \leq 1 \\ & -x_1^{'} - x_2^{'} + 2x^{''} \leq 0 \\ & -x_1^{'} \leq 0 \\ & -x_2^{'} \leq 0 \\ & -x^{''} \leq 0 \\ & x_1^{'} + x_2^{'} + x^{''} \leq M. \end{aligned}$$

Following the procedure in Case 2, we know $J = \{3, 4, 6\}$ is an initial ladder generator. Then the ladder method can be used to solve the above transformed problem, resulting in an optimal solution $[x_1^{'}; x_2^{'}; x^{''}] = [73.0235; 72.0235; 72.5235]$. From $x_i = x_i^{'} - x^{''}$, we know that the solution of the original problem is $[x_1; x_2] = [0.5; -0.5]$.

5.4.2 An Inclusive-cone-based Solvability Criterion for LP

The classical strong duality theorem (or trichotomy theorem) tells us, if one of a pair of primal and dual problems is unbounded, then its counterpart is infeasible, but the reverse is not necessarily true. Asymmetry of the strong duality theorem may be one of the reasons why some current LP solvers fail to provide correct solvability information, in particular in the case that an LP problem is infeasible or unbounded. By virtue of the inclusiveness concept, Liu [61] extended the boundedness concept to infeasible problems, refining an LP problem into four classes: inclusive-feasible (i.e., optimal), noninclusive-feasible (i.e., unbounded), inclusive-infeasible and noninclusive-infeasible problems (note that infeasible problems are further divided into inclusive-infeasible and noninclusive-infeasible problems). Further, the author proposed a "symmetric" strong duality theorem, termed "a quadrachotomy theorem". It follows from the quadrachotomy theorem that: (1) an LP problem is inclusive-feasible (i.e., optimal) if and only if its dual is inclusive-feasible (i.e., optimal); (2) an LP problem is noninclusive-feasible (i.e., unbounded) if and only if its dual is inclusive-infeasible; (3) an LP problem is inclusive-infeasible if and only if its dual is noninclusive-feasible (i.e., unbounded); (4) an LP problem is noninclusive-infeasible if and only if its dual is noninclusive-infeasible. In the context of the new category of LP problems and the quadrachotomy theorem, in this subsection we aim to propose an inclusive-cone-based solvability criterion to detect that an LP problem is inclusive-feasible (i.e., optimal), noninclusive-feasible (i.e., unbounded), inclusive-infeasible or noninclusive-infeasible.

At the moment, we restrict our attention to the pair of problems (P4) and (PM4) (see Subsection 5.4.1). Recall that an initial ladder is available for problem (PM4). With an initial ladder at hand, we can solve problem (PM4) using the ladder method. In the following, we shall discuss how we identify the solvability of the original problem (P4) while solving the auxiliary problem (PM4). (Note that the "solvability" includes the aforementioned four cases!) From now on until the end of this subsection, those notations introduced for solving problem (P) in Section 3.2 will carry over to problem (PM4) for use. Also, we denote by J_a the constraint index set associated with the artificial constraint in problem (PM4).

We would like to point out that, while solving problem (PM4), if at some iteration there exist multiple candidates for a drop (including the index associated with the artificial constraint), we choose the index associated with the added constraint as a priority for a drop since the artificial constraint itself is not part of problem (P4) and naturally we do not hope that the index associated with the additional constraint remains in the ladder generator. In fact, in order to reduce the amount of computation, we can drive out the index associated with the artificial constraint at the start as long as it is a candidate for a drop. We have the following main result with regard to the solvability of the original problem (P4).

Theorem 5.1. Consider the pair of problems (P4) and (PM4). The following statements hold:

(1) If problem (PM4) achieves optimality and $J^* \cap J_a = \emptyset$, where J^* is the ladder generator at an optimal solution of problem (PM4), then problem (P4) is inclusive-feasible (i.e., optimal).

(2) If problem (PM4) achieves optimality and $J^* \cap J_a \neq \emptyset$, then problem (P4) is noninclusive-feasible (i.e., unbounded).

(3) At some iteration k, if $\gamma^k = [A^T(J_k)]^{-1}a_{p^k}^T \leq 0$ and $J_k \cap J_a = \emptyset$, then problem (P4) is inclusive-infeasible.

(4) At some iteration k, if $\gamma^k = [A^T(J_k)]^{-1}a_{p^k}^T \leq 0$ and $J_k \cap J_a \neq \emptyset$, then problem (P4) is noninclusive-infeasible.

Proof. On the one hand, from the fact that problem (PM4) has a ladder, we know that the solvability of problem (PM4) includes two cases:

- Problem (PM4) is inclusive-feasible (equivalently, problem (PM4) can achieve optimality). (This tells us that problem (P4) is feasible.)
- Problem (PM4) is inclusive-infeasible, which means at some iteration k, $\gamma^k = [A^T(J_k)]^{-1}a_{p^k}^T \leq 0$ (see Subsection 3.2.2). (If so, we can assert that problem (P4) is infeasible since M is sufficiently large.)

On the other hand, noting that both $J^* \cap J_a = \emptyset$ and $J_k \cap J_a = \emptyset$ imply that problem (P4) is inclusive, it is straightforward to come to the above four conclusions.

$\square$

Remark 5.1. The above theorem is proposed to identify the solvability of problem (P4) by solving the auxiliary problem (PM4). Likewise, for the pair of problems (P5) and (PM5), we can develop the same result as above which can be used to recognize the solvability of problem (P5) by solving the auxiliary problem (PM5). For avoidance of repetition, we omit the presentation of the similar result. We would like to emphasize once more that the "solvability" we refer to here includes four cases, that is, inclusiveness-feasibility (i.e., optimality), noninclusiveness-feasibility (i.e. unboundedness), inclusiveness-infeasibility and noninclusiveness-infeasibility, instead of the conventional three cases—optimality, unboundedness and infeasibility. In view of the function of the proposed theorem, which is to diagnose the solvability of an LP problem in the context of the new category of LP problems, we prefer to call it "an inclusive-cone-based solvability criterion for LP" hereafter.

Remark 5.2. Combining the proposed solvability criterion and the quadrichotomy theorem, we are capable of identifying accurately the solvability of both primal and dual problems, especially in the case that an LP problem has no optimal solution, as we will see from the subsequent illustrations.

5.4.3 Illustrations

In this subsection, we will illustrate the ability that the above new criterion distinguishes the solvability of an LP problem.

Example 5.1. Consider the following LP problem

(E1):
$$\begin{aligned} \min \quad & x_1 + \tfrac{1}{2}x_2 \\ \text{s.t.} \quad & x_1 - x_2 \leq -1 \\ & -x_1 + x_2 \leq -1 \end{aligned}$$

$$x_1 \le 1$$
$$x_2 \le 0.$$

Adding the constraint $-x_1 - x_2 \le M$, yields the following problem:

$$\begin{aligned} \min \quad & x_1 + \tfrac{1}{2}x_2 \\ \text{s.t.} \quad & x_1 - x_2 \le -1 \\ & -x_1 + x_2 \le -1 \\ & x_1 \le 1 \\ & x_2 \le 0 \\ & -x_1 - x_2 \le M. \end{aligned}$$

We solve this problem with the additional constraint to detect that problem (E1) is inclusive-feasible (i.e., optimal), noninclusive-feasible (i.e., unbounded), inclusive-infeasible, or noninclusive-infeasible. The ladder algorithm implementing the ladder updating Criterion 2 (see Section 4.3, Chapter 4) is adopted. Here, $\mathcal{J} = \{1, 2, \cdots, 5\}$, $J_a = \{5\}$.

Initialization:

Following the procedure in Subsection 5.4.1, obtains the initial ladder generator

$$J_0 = \{5, 4\},$$
$$A(J_0) = \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix},$$
$$x^0 = A^{-1}(J_0)b(J_0) = \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} M \\ 0 \end{bmatrix} = \begin{bmatrix} -M \\ 0 \end{bmatrix}.$$

Since $V^0 = \{2\} \ne \emptyset$, iteration starts.

Iteration 1

Select a pick:

With only one element in V^0, we choose

$$p^0 = 2.$$

Select a drop:

Calculate

$$\delta^0 = [A^T(J_0)]^{-1}(-c) = \begin{bmatrix} -1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{2} \end{bmatrix}$$

and

$$\gamma^0 = [A^T(J_0)]^{-1}a_{p^0}^T = \begin{bmatrix} -1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}.$$

By means of Formula (3.3), we get

$$j_d^0 = 4.$$

Updating the ladder:

$$J_1 = \{5, 2\}, \quad A(J_1) = \begin{bmatrix} -1 & -1 \\ -1 & 1 \end{bmatrix},$$

$$x^1 = A^{-1}(J_1)b(J_1) = \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} M \\ -1 \end{bmatrix} = \begin{bmatrix} \frac{-M+1}{2} \\ \frac{-M-1}{2} \end{bmatrix}.$$

Since $V^1 = \{1\} \neq \emptyset$, iteration continues.

Iteration 2

Select a pick:

Since only one element is in V^1, we see

$$p^1 = 1.$$

Select a drop:

Calculate

$$\delta^1 = [A^T(J_1)]^{-1}(-c) = \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} -1 \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{3}{4} \\ \frac{1}{4} \end{bmatrix}$$

and

$$\gamma^1 = [A^T(J_1)]^{-1}a_{p^1}^T = \begin{bmatrix} -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}.$$

Note that $\gamma^1 \leq 0$ and $J_1 \bigcap J_a = \{5, 2\} \bigcap \{5\} \neq \emptyset$. We conclude that the original problem (E1) is noninclusive-infeasible. Further, according to Liu's quadrichotomy theorem (see Theorem 2.6), we can conclude that the dual problem of problem (E1) is noninclusive-infeasible.

We also used the linprog solver in MATLAB optimization toolbox (Version 5.1 (R2010b)) to solve the same problem. When the simplex algorithm was implemented, the output was as follows:

```
>> options=optimset('LargeScale', 'off', 'Simplex', 'on');
[x0, v0]=linprog([1; 0.5], [1 -1; -1 1; 1 0; 0 1], [-1; -1; 1; 0], [ ], [ ], [ ], [ ], [ ],
options)
Exiting: The constraints are overly stringent; no feasible starting point found.
```

Example 5.2. Consider the following LP problem

(E2):
$$\begin{aligned} \min \quad & x_1 - 4x_2 \\ \text{s.t.} \quad & -2x_1 - x_2 \le 4 \\ & -2x_1 + 4x_2 \le -8 \\ & -x_1 + 3x_2 \le -7 \\ & -x_1 \le 0 \\ & -x_2 \le 0. \end{aligned}$$

Adding the constraint $x_1 + x_2 \le M$, yields the following transformed problem:

$$\begin{aligned} \min \quad & x_1 - 4x_2 \\ \text{s.t.} \quad & -2x_1 - x_2 \le 4 \\ & -2x_1 + 4x_2 \le -8 \\ & -x_1 + 3x_2 \le -7 \\ & -x_1 \le 0 \\ & -x_2 \le 0 \\ & x_1 + x_2 \le M. \end{aligned}$$

Following the same line as the above example, we solve this problem with the additional constraint. Here, $\mathcal{J} = \{1, 2, \cdots, 6\}$, $J_a = \{6\}$.

Initialization:

By executing the procedure in Subsection 5.4.1, we find the initial ladder generator

$$J_0 = \{4, 6\},$$

$$A(J_0) = \begin{bmatrix} -1 & 0 \\ 1 & 1 \end{bmatrix},$$

$$x^0 = A^{-1}(J_0)b(J_0) = \begin{bmatrix} -1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ M \end{bmatrix} = \begin{bmatrix} 0 \\ M \end{bmatrix}.$$

Since $V^0 = \{2, 3\} \neq \emptyset$, iteration starts.

Iteration 1

Select a pick:

Using the ladder updating criterion 2, obtains

$$p^0 = 3.$$

Select a drop:

Calculate

$$\delta^0 = [A^T(J_0)]^{-1}(-c) = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 4 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$$

and

$$\gamma^0 = [A^T(J_0)]^{-1}a_{p^0}^T = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}.$$

According to Formula (3.3), we choose

$$j_d^0 = 4.$$

Updating the ladder:

$$J_1 = \{3, 6\}, \quad A(J_1) = \begin{bmatrix} -1 & 3 \\ 1 & 1 \end{bmatrix},$$

$$x^1 = A^{-1}(J_1)b(J_1) = \begin{bmatrix} -\frac{1}{4} & \frac{3}{4} \\ \frac{1}{4} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} -7 \\ M \end{bmatrix} = \begin{bmatrix} \frac{7+3M}{4} \\ \frac{-7+M}{4} \end{bmatrix}.$$

Since $V^1 = \emptyset$, the transformed problem achieves optimality. $J^* = J_1 = \{3, 6\}$. Note that $J^* \bigcap J_a = \{3, 6\} \bigcap \{6\} \neq \emptyset$. We conclude that the original problem (E2) is noninclusive-feasible (i.e., unbounded). Further, according to Liu's quadrichotomy theorem (see Theorem 2.6), we can conclude that the dual problem of problem (E2) is inclusive-infeasible.

When using CPLEX Dual Simplex LP solver in TOMLAB (Version 7.7 (R7.7.0)) to solve the same problem, there appeared the following incorrect output:

```
[x0, v0]=linprog([1; -4], [-2 -1; -2 4; -1 3; -1 0; 0 -1], [4; -8; -7; 0; 0])
linprog (CPLEX): The problem is infeasible.
```

Example 5.3. Consider the following LP problem

$$\text{(E3):} \quad \begin{aligned} \min \quad & -x_1 + x_2 \\ \text{s.t. } & x_1 - x_2 \leq 2 \\ & x_1 \leq 1 \\ & x_2 \leq 1. \end{aligned}$$

Adding the constraint $-x_1 - x_2 \leq M$, yields the following transformed problem:

$$\begin{aligned} \min \quad & -x_1 + x_2 \\ \text{s.t. } & x_1 - x_2 \leq 2 \\ & x_1 \leq 1 \\ & x_2 \leq 1 \\ & -x_1 - x_2 \leq M. \end{aligned}$$

Following the same line as previous examples, we solve this problem with the additional constraint. Here, $\mathcal{J} = \{1, 2, 3, 4\}, J_a = \{4\}$.

Initialization:

By the procedure in Subsection 5.4.1, we obtain the initial ladder generator

$$J_0 = \{2, 4\},$$

$$A(J_0) = \begin{bmatrix} 1 & 0 \\ -1 & -1 \end{bmatrix},$$

$$x^0 = A^{-1}(J_0)b(J_0) = \begin{bmatrix} 1 & 0 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ M \end{bmatrix} = \begin{bmatrix} 1 \\ -M-1 \end{bmatrix}.$$

Since $V^0 = \{1\} \neq \emptyset$, iteration starts.

Iteration 1

Select a pick:

With only one element in V^0, it is clear that

$$p^0 = 1.$$

Select a drop:

Calculate

$$\delta^0 = [A^T(J_0)]^{-1}(-c) = \begin{bmatrix} 1 & -1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

and

$$\gamma^0 = [A^T(J_0)]^{-1} a_{p^0}^T = \begin{bmatrix} 1 & -1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

According to Formula (3.3), here we have two candidates for a drop ($j_d^0 = 2$ or $j_d^0 = 4$). We prefer to choose $j_d^0 = 4$ since our ultimate task is to solve the original problem and naturally hope the index associated with the additional constraint is driven out.

Updating the ladder:

$$J_1 = \{2, 1\}, \quad A(J_1) = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix},$$

$$x^1 = A^{-1}(J_1) b(J_1) = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Since $V^1 = \emptyset$, the transformed problem achieves optimality. $J^* = J_1 = \{2, 1\}$. Note that $J^* \bigcap J_a = \{2, 1\} \bigcap \{4\} = \emptyset$. We conclude that the original problem (E3) is inclusive-feasible (i.e., optimal). Further, according to Liu's quadrichotomy theorem (see Theorem 2.6), we can conclude that the dual problem of problem (E3) is inclusive-feasible (i.e., optimal).

Example 5.4. Consider the following LP problem

$$\text{(E4):} \qquad \begin{aligned} \min \quad & x_1 - 2x_2 \\ \text{s.t.} \ & x_1 - x_2 \leq -1 \\ & -x_1 + x_2 \leq -1 \\ & x_1 \leq 1 \\ & x_2 \leq 1. \end{aligned}$$

Adding the constraint $-x_1 - x_2 \leq M$, yields the following transformed problem:

$$\begin{aligned} \min \quad & x_1 - 2x_2 \\ \text{s.t.} \quad & x_1 - x_2 \leq -1 \\ & -x_1 + x_2 \leq -1 \\ & x_1 \leq 1 \\ & x_2 \leq 1 \\ & -x_1 - x_2 \leq M. \end{aligned}$$

Following the same procedure as previous examples, we solve this problem with the additional constraint. Here, $\mathcal{J} = \{1, 2, \cdots, 5\}$, $J_a = \{5\}$.

Initialization:

Executing the procedure in Subsection 5.4.1, we obtain the initial ladder generator

$$J_0 = \{5, 4\},$$

$$A(J_0) = \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix},$$

$$x^0 = A^{-1}(J_0)b(J_0) = \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} M \\ 1 \end{bmatrix} = \begin{bmatrix} -M - 1 \\ 1 \end{bmatrix}.$$

Since $V^0 = \{2\} \neq \emptyset$, iteration starts.

Iteration 1

Select a pick:

Since only one element is in V^0, clearly we choose

$$p^0 = 2.$$

Select a drop:

Calculate

$$\delta^0 = [A^T(J_0)]^{-1}(-c) = \begin{bmatrix} -1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

and

$$\gamma^0 = [A^T(J_0)]^{-1} a_{p^0}^T = \begin{bmatrix} -1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}.$$

According to Formula (3.3), we have

$$j_d^0 = 5.$$

Updating the ladder:

$$J_1 = \{2, 4\}, \quad A(J_1) = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix},$$

$$x^1 = A^{-1}(J_1)b(J_1) = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

Since $V^1 = \{1, 3\} \neq \emptyset$, iteration continues.

Iteration 2

Select a pick:

Using the ladder updating criterion 2, obtains

$$p^1 = 1.$$

Select a drop:

Calculate

$$\delta^1 = [A^T(J_1)]^{-1}(-c) = \begin{bmatrix} -1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

and

$$\gamma^1 = [A^T(J_1)]^{-1} a_{p^1}^T = \begin{bmatrix} -1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}.$$

Note that $\gamma^1 \leq 0$ and $J_1 \bigcap J_a = \{2,4\} \bigcap \{5\} = \emptyset$. We conclude that the original problem (E4) is inclusive-infeasible. Further, according to Liu's quadrichotomy theorem (see Theorem 2.6), we can conclude that the dual problem of problem (E4) is noninclusive-feasible (i.e., unbounded).

We also used the linprog solver in MATLAB optimization toolbox (Version 5.1 (R2010b)) to solve the same problem. When the simplex algorithm was implemented, the output was as follows:

```
>> options=optimset('LargeScale','off','Simplex','on');
[x0, v0]=linprog([1; -2], [1 -1; -1 1; 1 0; 0 1], [-1; -1; 1; 1], [ ], [ ], [ ], [ ], [ ],
options)
Exiting: The constraints are overly stringent; no feasible starting point found.
```

When using CPLEX Dual Simplex LP solver in TOMLAB (Version 7.7 (R7.7.0)) to solve the same problem, there appeared the following incorrect output:

```
>> [x0, v0]=linprog([1; -2], [1 -1; -1 1; 1 0; 0 1], [-1; -1; 1; 1])
linprog (CPLEX): The problem is infeasible.
```

Chapter 6

Inclusive-cone-based Method for Linear Semi-infinite Programming

In the previous chapters, we used the inclusive-cone-based method for linear programming. In this chapter we apply this method to the linear semi-infinite programming (LSIP) case, obtaining an optimality result for LSIP and a ladder algorithm for solving LSIP problems. The main content of this chapter is to be published in [62].

This chapter is organized as follows. In Section 6.1, we give an overview of semi-infinite programming (SIP) (specifically, linear semi-infinite programming), including applications and numerical methods. We then introduce the problem formulation we are concerned with and end this section with the outline of our work. In Section 6.2, we give notation and preliminaries. We present an optimality result for LSIP in Section 6.3, which lays a theoretical foundation for our new algorithm. In the beginning of Section 6.4, we discuss a technique for finding an initial ladder, followed by the main algorithm and some remarks. We conclude this section with a simple convergence result. In Section 6.5, we test our algorithm on several numerical examples to demonstrate the efficiency and performance of the proposed algorithm. We make a brief summary for this chapter in Section 6.6.

6.1 Introduction

A semi-infinite programming (SIP) problem with finitely many variables subject to an infinitely many constraints can be formulated as

$$\text{(SIP):} \qquad \begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f(x,t) \leq 0, \text{ for all } t \in T \end{array}$$

where $x \in R^n$, $f_0 : R^n \to R$, $T \subseteq R^m$ is an infinite parameter set, and $f : R^n \times T \to R$.

SIP has been an active area of research for decades due to its wide range of real-world applications in a variety of fields. Approximation theory (specifically, Chebyshev approximation) may be the field where SIP found its inceptive application (see Polak [78], Reemtsen and R*ü*ckmann [82] and Stein [85]). Other fields in which SIP has been put into application include:

- air pollution control (see Hettich and Kortanek [43] and Vaz and Ferreira [91]);
- robotics (see Vaz et al. [90], Haaren-Retagne [40] and Marin [71]);
- optimal control (see Reemtsen and R*ü*ckmann [82] and Liu et al. [63]);
- gemstone cutting industry (see Winterfeld [97]);
- signal processing (see Moulin et al. [73], Potchinkov [79] and Vo et al. [92]);
- geometry (see Juhnke and Sarges [49]);
- mathematical physics (see Dambrine and Pierre [12]);
- game theory (see Llorca et al. [67] and Timmer et al. [88]);
- probability and statistics (see Noubiap and Seidel [74]);
- mathematical economics (see Stein and Still [86], Kortanek and Medvedev [54] and Jerez [48]).

On the numerical aspect, the most commonly known categories of solution methods for SIP problems are discretization methods (by grids and by cutting planes) and methods based on local reduction.

In a discretization method, a sequence of relaxed subproblems is solved by substituting the infinite parameter set T with $T_k \subset T$, where T_k is a finite index set such that $\mathrm{dist}(T_k, T) := \sup_{t\in T}\inf_{t_k\in T_k}\|t - t_k\|$ tends to zero as the iteration number k goes to ∞ (see Hettich [42], Teo et al. [87] and Reemtsen [81]). A general discretization method is computationally expensive, since T_k must be sufficiently dense in T when k is sufficiently large. The time, spent on checking feasibility with respect to the subproblems and solving the subproblems, increases dramatically with increasing size of the subproblems (unless an efficient constraint dropping strategy is employed).

A local reduction approach replaces a semi-infinite programming problem with a nonlinear system of equations and possibly some inequalities with finitely many unknowns, which is usually solved by means of a quasi-Newton method (see Price [80], Gramlich et al. [36] and Watson [94]). One advantage of reduction based methods is that they have good local convergence properties, hence they are typically employed in a final stage to improve the solution.

Beyond the methods discussed above, the exchange methods (including the primal exchange methods and dual exchange methods) are also a well-known family of methods. This class of methods can be viewed as a compromise between discretization methods and local reduction methods. Hence, exchange methods are usually more efficient than pure discretization methods (see Watson [93], Okuno et al. [75], Wu et al. [98] and Anderson and Lewis [2]).

Other numerical methods include (but not limited to): dual parametrization methods (see Liu and Teo [65] and Liu et al. [66]), non-differentiable optimization methods (see Polak [77, 78] and Lin et al. [57]), constraint transcription methods (see Jennings and Teo [47]), exact penalty methods (see Yu et al. [102] and Lin et al. [58]) and interior point methods (see Liu [59], Ferris and Philpott [20, 21] and Stein and Still [86]). Surveys on the development of SIP methods can be found in López and Still [55] and Hettich and Kortanek [43]. For a comprehensive knowledge of SIP, we refer the interested reader to several monographs in which SIP is discussed in an exhaustive way (see Goberna and López [30], Stein [85] and Reemtsen and Rückmann [82]).

Linear semi-infinite programming (LSIP) is an important special case of problem (SIP), in which both the objective function $f_0(x)$ and constraint function $f(x,t)$ are linear in x for any fixed $t \in T$. According to Goberna [28], *there are at least three reasons that justify researchers' interest in LSIP. First, for its real life and modeling applications. Second, for providing nontrivial but tractable optimization problems on which it is possible to check more general theories and methods. Finally, LSIP can be seen as a theoretical model for large scale LP problems.* For these reasons, considerable efforts are under way to develop it (see Betrò [7], Oskoorouchi et al. [76], Goberna et al. [32], Liu and Ding [62] and the references therein). Development of LSIP theoretical and numerical aspects can be found in recent survey articles [31, 28]. For a more in-depth treatment of this topic, the interested reader is invited to consult [29, 82].

In this chapter, we particularly focus on LSIP problems which can be expressed as below:

$$\text{(LSIP):} \qquad \begin{array}{ll} \min & c^T x \\ \text{s.t.} & a(t)x \leq b(t), \ \text{for all } t \in T = [\alpha, \beta] \subset R \end{array}$$

where $x, c \in R^n$ $(c \neq 0)$. $a : T \to R^n$ and $b : T \to R$. Clearly when T is a finite set, the above LSIP is reduced to a linear programming problem.

Our main aim is to develop a new method for LSIP, which is the generalization of the inclusive-cone-based method for LP. On the basis of the concept of the generalized base point, we first develop a fundamental theorem with regard to optimality for LSIP, which is an LSIP version of inclusive-cone-based optimality condition for LP. With this optimality result as a theoretical foundation, we propose a ladder algorithm for solving LSIP problems, termed "the centered climbing ladder algorithm for LSIP". This new algorithm approaches an optimal solution of an LSIP problem, which is essentially a feasible generalized base point, through a sequence of improved base points which is obtained by iteratively updating the inclusive cone as well as the ladder. The new algorithm has the features as follows:

1. The algorithm only deals with a fixed number of constraints at each iteration.

2. Instead of selecting the most violated constraint as required in most of solution methods for LSIP, at each iteration we pick up a violated constraint along the “parameterized centerline”, by solving a one-dimensional global optimization problem. It is worth emphasizing that the selection of the entering constraint has a great degree of freedom, which is controlled by a parameter arising in the global optimization problem.

3. It can detect infeasibility and unboundedness after a finite number of iterations.

4. It obviates extra work for feasibility check, as it handles feasibility and optimality simultaneously.

6.2 Notation and Preliminaries

In the case that elements of a matrix need to be displayed, we will often present them in row form, with entries in the same row separated by commas and different rows separated by semicolons. For example, $A = [a, b; c, d]$ represents a 2×2 matrix, in which the first row elements are a and b and the second row elements are c and d. If $a_1, a_2, \cdots, a_m$ are n-dimensional row vectors and $b_1, b_2, \cdots, b_n$ are m-dimensional column vectors, then $A = [a_1; a_2; \cdots; a_m]$ and $B = [b_1, b_2, \cdots, b_n]$ are both $m \times n$ matrices. Furthermore, with this notation, the elements of the row vector a_1 and the column vector b_1 above can be conveniently displayed as $a_1 = [a_{11}, a_{12}, \cdots, a_{1n}]$ and $b_1 = [b_{11}; b_{21}; \cdots; b_{m1}]$. We will use R^n and R_n to represent the n-column and the n-row vector spaces, respectively.

Throughout the remaining of this dissertation, we make the following assumptions:

Assumption 1: The functions $a(t)$ and $b(t)$ are continuously differentiable on $[\alpha, \beta]$ and $a(t) \neq 0$ for all $t \in [\alpha, \beta]$.

Assumption 2: There exist $t_1, t_2, \cdots, t_n \in [\alpha, \beta]$ such that $a(t_1), a(t_2), \cdots, a(t_n)$ form a basis for R_n.

Assumption 3: If the feasible region of problem (LSIP) is non-empty, it contains at least a Slater point.

The main results to be proposed in this chapter build upon the concept of the inclusive cone, which has been presented in Chapter 2 for LP case. This concept is applicable directly to LSIP case as well. For convenience of discussion, we restate this concept for problem (LSIP) as follows.

Definition 6.1. Consider problem (LSIP). Let $J = \{t_1, t_2, \cdots, t_n\} \subset T$ be a finite subset. A convex cone generated by n linearly independent vectors

$$a^T(t_1), a^T(t_2), \cdots, a^T(t_n)$$

is said to be an inclusive cone generated by J if it contains the vector $-c$. If J generates an inclusive cone, the set defined by

$$L(J) = \{x \in R^n : a(t_j)x \leq b(t_j), \text{for } t_j \in J, j = 1, 2, \cdots, n\}$$

is called the inclusive region or the ladder associated with J. The corresponding finite set J is called the generator of $L(J)$, and the unique solution of the linear system $a(t_j)x = b(t_j), j = 1, 2, \cdots, n$, denoted by x_J, is called the base point of the ladder $L(J)$.

For LP case, we know that a feasible base point is essentially an optimality solution of an LP problem (see Theorem 2.1). In view of the fact that the feasible region of an LSIP problem is in general no longer a polytope, the inclusive-cone-based optimality result for LP case (see Theorem 2.1) may not automatically apply to LSIP. For a proper generalization, it is necessary to introduce the following concept.

Definition 6.2. A point $x \in R^n$ is called a generalized base point of problem (LSIP) if there exists a sequence of base points $x^k, k = 1, 2, \cdots$, with corresponding sequence of ladders $L(J_k)$ such that $x^k \to x$ $(k \to \infty)$.

It is clear that base points are special generalized base points. The feasible region of an LSIP problem may not contain a base point. However, it must contain a generalized base point if an optimal solution exists, which we will prove in the following Section 6.3. We have seen from the previous chapters that the concept of base point plays an important role in development of the ladder method for LP. In fact, the ladder method for LP was designed to generate a sequence of base points, which eventually converges to a feasible base point (namely, an optimal solution) if the problem can achieve optimality. Likewise, we will perceive in the subsequent two sections the importance of the concept of the generalized base point in development of both the optimality result and the ladder algorithm for LSIP.

6.3 An Optimality Result

To begin, we present the following theorem with respect to optimality.

Theorem 6.1. Consider problem (LSIP). The following statements hold true:

(a) A generalized base point is an optimal solution if and only if it is feasible.

(b) Let Assumptions 1-3 be satisfied. Problem (LSIP) has an optimal solution if and only if it has one at a feasible generalized base point.

(c) Let Assumptions 1-3 be satisfied. Problem (LSIP) has no optimal solution if and only if it either has no generalized base point or all its generalized base points are infeasible.

Proof. (a) From the fact that the objective function value at a generalized base point is a lower bound of the optimal value, we know that a feasible generalized base point must be an optimal solution. Therefore, this statement is obvious.

(b) We only need to prove the necessity. Suppose that problem (LSIP) has an optimal solution x^*. Since the feasible region of problem (LSIP) contains a Slater point (see Assumption 3), it follows from the KKT conditions (see, e.g., Theorem 6 in [45]), there exist $t_1, t_2, \cdots, t_p \in [\alpha, \beta]$ for some $p \ (1 \leq p \leq n)$ such that the following hold:

$$a(t_j)x^* = b(t_j), \ j = 1, 2, \cdots, p \tag{6.1}$$

and

$$-c = \sum_{j=1}^{p} \lambda_j a^T(t_j), \ \lambda_j > 0, \ j = 1, 2, \cdots, p \tag{6.2}$$

where $a(t_j) \ \ (1 \leq j \leq p)$ are linearly independent. By Assumption 2, we can span $\{a(t_1), a(t_2), \cdots, a(t_p)\}$ into $\{a(t_1), a(t_2), \cdots, a(t_n)\}$ to form a basis for R_n. It is clear that $J = \{t_1, t_2, \cdots, t_n\}$ generates a ladder $L(J)$. We take two cases into consideration.

Case 1: Problem (LSIP) has a non-degenerate ladder $L(J)$.

In this case, none of the edges of $L(J)$ is orthogonal to $-c$. Let $B^k = J \bigcup \{\alpha + \frac{i}{2^k}(\beta - \alpha) \big| i = 0, 1, \cdots, 2^k\} \ \ (k = 1, 2, \cdots)$. Consider the following problem (P_{B^k}):

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & a(t)x \leq b(t), \ \text{for } t \in B^k \end{array}$$

For each k, problem (P_{B^k}) has an optimal solution since it is feasible and has a ladder $L(J)$. Denote by x^k the optimal solution to problem (P_{B^k}). It is clear that

$$x^k \in X \triangleq \{x | x \in L(J) \ \text{ and } \ c^T x \leq c^T x^*\}.$$

Noting that the set X in the above expression is compact, we know that the sequence $\{x^k\}$ has a convergent subsequence. Denote by $\{x^{k_i}\}_{i=1}^{\infty}$ such a subsequence. Let $x^{k_i} \to \tilde{x}$. In view of Assumption 1, we know that $\tilde{x}$ satisfies $a(t)\tilde{x} \leq b(t)$ for all $t \in \bigcup_{k=1}^{\infty} B^k$. Since $\bigcup_{k=1}^{\infty} B^k$ is dense in $[\alpha, \beta]$, we then have that $a(t)\tilde{x} \leq b(t)$ holds for all $t \in [\alpha, \beta]$, which implies $\tilde{x}$ is feasible. On the other hand, it follows from $c^T x^k \leq c^T x^*$ that $c^T \tilde{x} = c^T x^*$. Hence, $\tilde{x}$, as a generalized base point, is an optimal solution of problem (LSIP).

Case 2: All ladders of problem (LSIP) (especially $L(J)$) are degenerate.

From Formulae (6.1) and (6.2), it follows that the base point x^0 associated with the ladder $L(J)$ satisfies $c^T x^0 = c^T x^*$. Consider the following system of linear equations:

$$a(t_j)x = b(t_j), \quad j = 1, 2, \cdots, p. \tag{6.3}$$

Note that $a(t_j),\ j = 1, 2, \cdots, p$, are linearly independent. Therefore, by solving system (6.3), we can express p components of x in terms of its remaining $n-p$ components which we denote by $y = [y_1; y_2; \cdots; y_{n-p}]$. In other words, for the above system, there exist some $n \times (n-p)$ matrix H and a vector $h_0 \in R^n$ such that

$$x = Hy + h_0.$$

We consider the following problem:

($\overline{\text{LSIP}}$):
$$\begin{aligned} \min \quad & \bar{c}^T y \\ \text{s.t.} \quad & \bar{a}(t)y \le \bar{b}(t), \ \text{for all } t \in [\alpha, \beta] \end{aligned}$$

where

$$\bar{c} = -\bar{a}^T(t_{p+1}) - \bar{a}^T(t_{p+2}) - \cdots - \bar{a}^T(t_n),$$

$$\bar{a}(t) = a(t)H,$$

and

$$\bar{b}(t) = b(t) - a(t)h_0.$$

It is clear that each ladder $L(r_1, r_2, \cdots, r_{n-p})$ of the above problem with base point y^0 corresponds to a ladder $L(t_1, t_2, \cdots, t_p, r_1, r_2, \cdots, r_{n-p})$ of problem (LSIP) with base point $x^0 = Hy^0 + h_0$, and each feasible point y of the above problem corresponds to an optimal solution $x = Hy + h_0$ of problem (LSIP).

With $\bar{c}$ constructed as above, we see that $\bar{J} = \{t_{p+1}, t_{p+2}, \cdots, t_n\} \subset J$ generates a non-degenerate ladder for problem ($\overline{\text{LSIP}}$). According to Case 1, problem ($\overline{\text{LSIP}}$) has an optimal solution y^*, which is also a generalized base point of problem ($\overline{\text{LSIP}}$). Let $L(r_1^i, r_2^i, \cdots, r_{n-p}^i)\ \ (i = 1, 2, \cdots)$ denote the sequence of ladders of problem ($\overline{\text{LSIP}}$) with the corresponding base point sequence $\{y^i\}$ convergent to y^*. Then, from the above discussion it follows that $x^* = Hy^* + h_0$ is an optimal solution of problem (LSIP) and $x^i = Hy^i + h_0\ \ (i = 1, 2, \cdots)$ is a sequence of base points satisfying

$$\lim_{i\to\infty} x^i = \lim_{i\to\infty}(Hy^i + h_0) = Hy^* + h_0 = x^*$$

which implies that x^* is a generalized base point of problem (LSIP) as well. This completes the proof.

(c) This is a direct consequence of (b).

$\square$

6.4 An LSIP Ladder Algorithm

In this section, we develop a ladder algorithm for solving problem (LSIP). The new algorithm is a natural extension of CCA for LP (see Section 3.2). The main idea is to produce a sequence of ladders $\{L(J_k)\}$, $k = 0, 1, 2, \cdots$, such that the corresponding sequence of base points $\{x^k\}$ converges to a generalized base point which, according to Theorem 6.1, must be an optimal solution of problem (LSIP). At the start of solution process, an initial ladder $L(J_0)$ with the associated base point x^0 is artificially constructed. Then, this ladder is iteratively updated in such a way that $L(J_{k+1})$ is obtained from $L(J_k)$ by climbing from the current base point x^k toward the feasible region along the so-called 'centerline' of $L(J_k)$. As will be seen below, the ladder updating scheme for this new algorithm derives from CCA for LP. For this reason, we call this algorithm "the centered climbing ladder algorithm for LSIP".

Before presenting the new algorithm, we first introduce how to construct an initial ladder.

6.4.1 Initialization

To start the ladder algorithm for LSIP, we need to get a ladder at hand. It is natural to try to find such a ladder from $\big\{a(t_i)\big|t_i \in T_N\big\}$, where $T_N = \big\{\alpha + i\frac{\beta-\alpha}{N} : i = 0, 1, 2, \cdots, N\big\}$ and N is a positive integer. Unfortunately, it is not always feasible to obtain a ladder in this way. In fact, in the discretization methods proposed by Hettich [42] and Reemtsen [81], the authors assume that the discretized problem

$$(\mathrm{P}_{T_N}): \qquad \begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & a(t)x \le b(t), \ \text{for } t \in T_N \end{aligned}$$

has an optimal solution for sufficiently large N, which means problem (P_{T_N}) is inclusive-feasible, and hence the original problem has at least one ladder which can be constructed from $\big\{a(t_i)\big|t_i \in T_N\big\}$. However, the following counterexample shows this strong assumption is easily violated and an initial ladder can not always be obtained from $\big\{a(t_i)\big|t_i \in T_N\big\}$.

Example 6.1. Consider the problem given by

$$\begin{array}{ll} \min & x_2 \\ \text{s.t.} & x_1 \sin\left(t-\sqrt{\frac{\pi}{2}}\right)^2 - x_2 \cos\left(t-\sqrt{\frac{\pi}{2}}\right)^2 \leq 1, \text{ for } 0 \leq t \leq 2. \end{array}$$

Discretizing this problem by taking $t_i \in T_N = \big\{\alpha + i\frac{\beta-\alpha}{N} : i = 0, 1, 2, \cdots, N\big\}$, where $\alpha = 0$ and $\beta = 2$, we attempt to find an initial ladder for the above problem. Noting that the discretized problem

$$\begin{array}{ll} \min & x_2 \\ \text{s.t.} & x_1 \sin\left(t-\sqrt{\frac{\pi}{2}}\right)^2 - x_2 \cos\left(t-\sqrt{\frac{\pi}{2}}\right)^2 \leq 1, \text{ for } t \in T_N \end{array}$$

does not include the constraint at $t = \sqrt{\frac{\pi}{2}}$, we know that this discretized problem is unbounded for arbitrarily large N, even though the original problem has infinitely many solutions. This means that this discretized problem is noninclusive-feasible. Hence, we cannot find an initial ladder for the original problem from $\big\{a(t_i)\big|t_i \in T_N\big\}$.

As shown above, it is not always workable to obtain an initial ladder by discretization. In the following, we construct an artificial ladder which applies to general LSIP problems.

Consider problem (LSIP). We introduce an additional constraint $-c^T x \leq M$ to problem (LSIP), where M is a sufficiently large number. Clearly, for sufficiently large M, addition of this constraint does not change optimal solutions of the problem (if they exist) or solvability (if the problem is bounded or infeasible). From Assumption 2, we can span $-c$ into n linearly independent vectors, say, $-c, a^T(t_{j_1}), a^T(t_{j_2}), \cdots, a^T(t_{j_{n-1}})$. It is clear that these n vectors form an inclusive cone, and hence the associated constraints produce a ladder with the corresponding base point given by

$$x^0 = \begin{bmatrix} -c^T \\ a(t_{j_1}) \\ \vdots \\ a(t_{j_{n-1}}) \end{bmatrix}^{-1} \begin{bmatrix} M \\ b(t_{j_1}) \\ \vdots \\ b(t_{j_{n-1}}) \end{bmatrix}. \tag{6.4}$$

For notational reasons, here we define $a(\alpha - 1) = -c$, $b(\alpha - 1) = M$, and $T = \{\alpha - 1\} \bigcup [\alpha, \beta]$. Then, we can conveniently present any ladder generator, including that of the artificial ladder, in the form of $J = \{t_1, t_2, \cdots, t_n\}$, where $t_j \in T,\ j = 1, 2, \cdots, n$.

6.4.2 Description of Algorithm

Consider problem (LSIP). Let $L(J_k)$ denote the current ladder with the corresponding base point x^k, where $J_k = \{t_1^k, t_2^k, \cdots, t_n^k\}$. The center vector of $L(J_k)$ is defined as

$$v^k = -A(J_k)^{-1}\mathbf{1}_{n\times 1},$$

where $\mathbf{1}_{n\times 1} = [1; 1; \cdots; 1] \in R^n$ and $A(J_k) = [a(t_1^k); a(t_2^k); \cdots; a(t_n^k)]$. Here, in order for $A(J_k)$ to be well defined, we may consider J_k as ordered sets. The centerline of $L(J_k)$ is defined as a line emanating from x^k and parallel to v^k:

$$l(x^k, v^k): \qquad x = x^k + \mu v^k, \quad -\infty < \mu < \infty.$$

If x^k is not an optimal solution, the current ladder needs to be updated to obtain a new base point. In CCA for solving LP problems, in order to update the current ladder $L(J_k)$ to $L(J_{k+1})$ and the current base point x^k to x^{k+1}, an entering constraint index (called 'a pick' in CCA for LP) is determined along the centerline of the current ladder. The selected index corresponds to the most violated constraint along the centerline (see Liu [60]). For LSIP problems, the most violated constraint index $t^k \in [\alpha, \beta]$ along the centerline is the global minimizer of the following optimization problem:

P(J_k): $\quad \min \phi(t, J_k)$

$\quad$ s.t. $\alpha \le t \le \beta$,

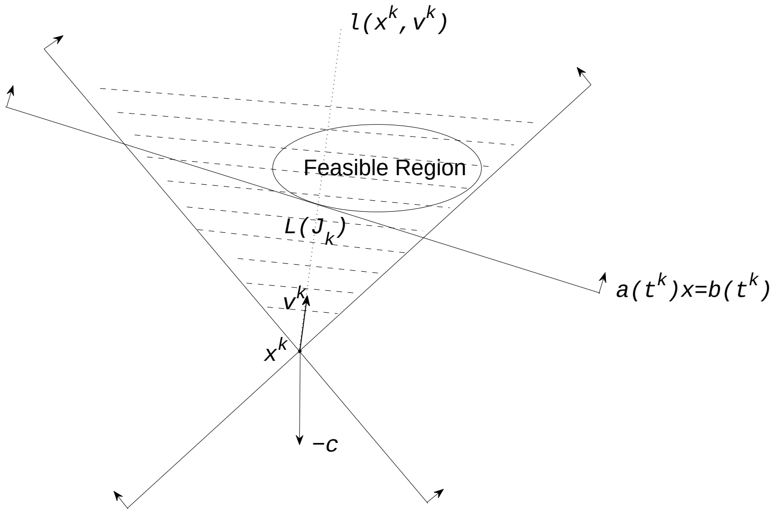

Figure 6.1: Finding a pick along the centerline

where

$$\phi(t, J_k) = \frac{b(t) - a(t)x^k}{|a(t)v^k|}.$$

Geometrically, if the global minimizer t^k of the above problem exists, then $a(t^k)x \leq b(t^k)$ is the constraint most violated at x^k along the centerline of $L(J_k)$ (see Figure 6.1). However, problem $\mathrm{P}(J_k)$ may not be well posed since the denominator $|a(t)v^k|$ of $\phi(t, J_k)$ in problem $\mathrm{P}(J_k)$ may take zero value for some values of t. For this reason, we modify $\phi(t, J_k)$ as below:

$$\phi(t, J_k, \tau) = \frac{b(t) - a(t)x^k}{\sqrt{\tau + (1-\tau)(a(t)v^k)^2}}, \quad \text{for} \ \ 0 < \tau \leq 1. \tag{6.5}$$

Note that, when the parameter $\tau = 0$ we obtain the objective function $\phi(t, J_k)$ in problem $\mathrm{P}(J_k)$. For $0 < \tau \leq 1$, the function $\phi(t, J_k, \tau)$ is well defined and continuously differentiable on $T = [\alpha, \beta]$. Therefore, the global minimizer of the following modified problem $\mathrm{P}(J_k, \tau)$ always exists.

$\mathrm{P}(J_k, \tau)$:

$$\begin{aligned} &\min \ \phi(t, J_k, \tau) \\ &\text{s.t.} \ \ \alpha \leq t \leq \beta. \end{aligned}$$

Still denote by t^k the global minimizer of problem $\mathrm{P}(J_k, \tau)$. Once t^k is found, an entering constraint $a(t^k)x \leq b(t^k)$ is determined. Consistent with [60], we call t^k a 'pick'. We would like to point out that the value of the parameter τ needs to be preset for solving problem $\mathrm{P}(J_k, \tau)$. Note that different τ values may result in different entering constraint, and hence different base point. Naturally we may expect that some τ values are better than others, thus enabling us to find a more effective entering constraint. A number of experiments show that, for the problems solved in Section 6.5, the algorithm performs much better for $0.9 \leq \tau \leq 1$ than for $0 < \tau \leq 0.2$. In fact, the best τ value seems to be problem-dependent, and can be anywhere in (0, 1].

In general, finding a global optimal solution of a non-linear optimization problem is intractable. However, for one-dimensional optimization problems, reliable and efficient numerical methods are available (see Liu and Teo [64], Locatelli and Schoen [68] and Yiu et al. [101]). In this dissertation, we use the right bridging algorithm developed by Liu and Teo (see [64]) to solve the global optimization problem $\mathrm{P}(J_k, \tau)$. To put it briefly, the right bridging algorithm proceeds as follows. Starting the bridging process from $t = \alpha$, the right bridging algorithm minimizes a bridged function which is initially constructed at the left end of the interval. By minimizing the bridged function, points that are not better than the current solution are skipped and a local minimum, better than the starting point, is found. Then, by updating the bridged function at the local minimum, better local minimum solutions are located. The entire process continues until a global optimal solution is found. For details of the bridging method, we invite the interested readers to consult [64]. After selecting a 'pick', that is, the global minimizer t^k, the procedure of ladder updating is the same as that for the LP case. In the following, we describe the centered climbing ladder algorithm for LSIP in detail.

A Centered Climbing Ladder Algorithm for LSIP:

Step 0 Initialization.

Start with an artificial ladder $L(J_0)$ with the associated generator

$J_0 = \{t_1^0, t_2^0, \cdots, t_n^0\}$ and base point x^0. (Refer to the foregoing subsection for how to find such a ladder.)

Set $k = 0$. Let η be a prescribed small number such that $0 < \eta < 1$.

Set the parameter value $\tau \in (0, 1]$.

Step 1 Checking optimality.

1.1 Find the global minimizer t^k of problem $\mathrm{P}(J_k, \tau)$ with the global optimal value V_k.

1.2 If $V_k > -\eta$, then output x^k,

- If $c^T a^T(t^k) > 0$, then exit with "the problem is infeasible."
- Else if $c^T x^k = -M$, then exit with "the problem is unbounded."
- Else exit with "the problem has an optimal solution $x^* = x^k$".

1.3 Otherwise, go to next step.

Step 2 Updating the ladder.

2.1 Identifying an index from J_k.

Try to find an index $t_{j_1}^k \in J_k$ $(1 \le j_1 \le n)$ as a drop such that $J_{k+1} = \{t_1^{k+1}, t_2^{k+1}, \cdots, t_n^{k+1}\}$ is a ladder generator and the associated base point $x^{k+1} \in L(J_k)$, where

$$t_j^{k+1} = \begin{cases} t_j^k, & j \neq j_1 \\ t^k, & j = j_1. \end{cases}$$

and

$$x^{k+1} = [a(t_1^{k+1}); a(t_2^{k+1}); \cdots ; a(t_n^{k+1})]^{-1}[b(t_1^{k+1}); b(t_2^{k+1}); \cdots ; b(t_n^{k+1})].$$

- If such an index does not exist, exit with "the problem is infeasible."
- Otherwise, go to next step.

2.2 Set $k := k + 1$. Return to Step 1.

Remark 6.1. In Step 0, by means of the artificial constraint $-c^T x \leq M$, we construct an initial ladder to start this algorithm. Specifically speaking, we take $t_1^0 = \alpha - 1$ (note that $a(\alpha - 1) = -c^T$) and select randomly $t_j^0 \in [\alpha, \beta]$ $(j = 2, 3, \cdots, n)$ to find a ladder. We would like to point out that this way of obtaining a ladder is not rigorous. However, in many cases it is a remarkably effective method. An alternative procedure for obtaining an initial ladder, which is more formal but less efficient, is introduced as follows.

- Find n linearly independent outward normal vectors $a^T(t_j^0)$ $(j = 1, 2, \cdots, n)$, where $t_j^0 \in [\alpha, \beta]$.
- Calculate $[\lambda_1; \lambda_2; \cdots; \lambda_n] = -[a^T(t_1^0), a^T(t_2^0), \cdots, a^T(t_n^0)]^{-1}c$.
  - If $\lambda_j \geq 0$ $(j = 1, 2, \cdots, n)$, then $J_0 = \{t_1^0, t_2^0, \cdots, t_n^0\}$ generates a ladder $L(J_0)$.
  - Otherwise, identify any $\lambda_{j_0} \neq 0$. Set $t_{j_0}^0 = \alpha - 1$. Then $a(t_j^0)$ $(j = 1, 2, \cdots, n)$ are linearly independent, which implies $J_0 = \{t_1^0, t_2^0, \cdots, t_n^0\}$ generates a ladder $L(J_0)$.

Remark 6.2. Step 1 requires solving a global optimization problem $\mathrm{P}(J_k, \tau)$ to find a pick t^k as well as the global minimum value V_k of the function $\phi(t, J_k, \tau)$ over the interval $[\alpha, \beta]$, which is the main computational task of this algorithm. As we have seen earlier, V_k is the negative of the 'maximum constraint violation' along the parameterized centerline emanating from the current base point x^k. Hence, if $-V_k$ is less than a prescribed accuracy η, the current base point x^k is taken as an approximate optimal solution of problem (LSIP). Otherwise, the global minimizer t^k of problem $\mathrm{P}(J_k, \tau)$ determines a constraint which is to enter the current ladder. Then, the current ladder is updated and the new base point is calculated with ease.

Remark 6.3. Step 2.1 involves identifying a drop, which can be fulfilled by the procedure introduced in Subsection 3.2.2, Chapter 3.

Remark 6.4. This algorithm has the following advantages:

1. It can detect infeasibility and unboundedness after a finite number of iterations.

2. At each iteration, the entering constraint results in, generally speaking, an effective improvement of the current base point. Although identifying the entering constraint requires solving a global optimization problem, it turns out that the bridging method is an efficient algorithm for finding a global optimum as the computational effort involved is comparable to a few linear searches.

3. This algorithm does not require extra feasibility checks as feasibility information can be drawn from Step 1.1.

We conclude this section with the following theorem with regard to the convergence of the proposed algorithm.

Theorem 6.2. Let $\{x^k\}$ be the sequence of base points generated by the above algorithm. The following statements hold true:

(a) Once the algorithm terminates, the output x^k is either an optimal solution, or the problem is infeasible or unbounded.

(b) If $\{x^k\}$ is an infinite sequence, then the following are true:

  (b.1) The objective value sequence $\{c^T x^k\}$ is non-decreasing. If it diverges, i.e. $c^T x^k \to \infty$ $(k \to \infty)$, then problem (LSIP) is infeasible.

  (b.2) Suppose, in the artificial constraint $-c^T x \leq M$, M is sufficiently large. If the output x^k satisfies that $-c^T x^k = M$, then problem (LSIP) is unbounded.

  (b.3) Assume that $\{c^T x^k\} \to \hat{V}$, and $\max_{t \in T} \{a(t)x^k - b(t)\} \to 0$ $(k \to \infty)$. Then any convergent subsequence of $\{x^k\}$ converges to an optimal solution of problem (LSIP).

Proof. Statements (a), (b.1) and (b.2) are obvious. Statement (b.3) is true because in this situation any limiting point will be a feasible generalized base point, and hence is optimal according to Theorem 6.1. □

6.5 Numerical Experiments

In this section, we solve several approximation problems to examine the computational behavior of the algorithm described in the previous section. The first example is the famous one-sided L^1 approximation of the tangent function on [0,1] by polynomials with limited order. This is a typical LSIP test problem and appears quite often in literature (see Goberna and López [29], Ito et al. [45], den Hertog et al. [15] and Betrò [7]). The main intention for presenting this example is to compare the proposed algorithm with other existing algorithms. The other two examples we give in this section are one-sided approximations of a given smooth function by sine and cosine series of given orders. The inclusion of these two examples is to demonstrate that our algorithm is capable of dealing with problems of larger size. To the best of the authors' knowledge, these two examples have not been documented in the research literature.

We implemented the proposed algorithm in the MATLAB environment (MATLAB 7.11.0 (R2010b)) and ran experiments on a desk-top computer (HP Intel(R) Core(TM), i7-2600 CPU@3.40GHz, 3.39GHz, 3.24GB of RAM) under the Microsoft Windows XP operating system. All calculations were done at 16 digit precision. For the proposed algorithm, we set η =1.0e-7. As mentioned in the foregoing, we used the bridging method [64] to solve the one-dimensional global optimization problem in Step 1 of the new algorithm, which was coded in the same MATLAB environment. A number of experimental tests showed that it is a computationally efficient and reliable global optimization algorithm, capable of finding the global optimal solution of the optimization problem in Step 1 efficiently and accurately.

We report numerical test results for this section in Table 6.1–6.4. In all these tables, we employ the following abbreviations for simplicity:

- Iters.: Number of iterations.
- CPU(s): CPU time elapsed in seconds.
- Max. Const. Vio.: The maximum constraint violation, $\max_{t\in T} \{a(t)x^*-b(t)\}$.

- Opt. Dist.: The optimal L^1 distance.

For convenience of reporting, computational results displayed were rounded to at most 8 significant figures. We would like to point out that, for all test problems, the number of iterations and CPU time depend not only on the problem dimension and the value of parameter τ, but also on the initial ladder constructed randomly in the initial step. Therefore, there may be noticeable change among different runs.

Example 6.2. Find the best L^1 approximation to the tangent function $b_+(t) = \tan(t)$ by polynomial functions $g_p(t)$ of order less than n from above over the interval [0,1]:

$$\inf_{g_p}\left\{\|g_p - b_+\|_1 \middle| g_p \text{ is a polynomial of order} < n, \text{and } g_p(t) \geq b_+(t) \text{ for } t \in [0,1]\right\}$$

Let $g_p(t) = \sum_{i=1}^{n} x_i t^{i-1}$. For $g_p(t) \geq b_+(t)$, we have

$$\begin{aligned}\|g_p - b_+\|_1 &= \int_0^1 |g_p(t) - b_+(t)|dt \\ &= \int_0^1 (\sum_{i=1}^{n} x_i t^{i-1} - \tan(t))dt \\ &= \sum_{i=1}^{n} \frac{x_i}{i} + \ln(\cos 1).\end{aligned}$$

Therefore, the above problem can be formulated as the following LSIP problem :

$$\begin{aligned}(\mathrm{E}_1): \qquad & \min \quad \sum_{i=1}^{n} \frac{x_i}{i} \\ & \text{s.t.} \quad -\sum_{i=1}^{n} t^{i-1} x_i \leq -\tan t, \quad t \in [0,1].\end{aligned}$$

For each n, problem (E_1) has a unique optimal solution $x^* = [x_1^*; x_2^*; \cdots; x_n^*]$, which determines a best L^1 approximation $g_p^*(t) = \sum_{i=1}^{n} x_i^* t^{i-1}$ to the tangent curve from above on [0, 1]. However, it is noticeable that, for $n \geq 3$ the boundary of its feasible region becomes very flat in the $-c$ direction near the optimal solution, and for $n \geq 6$ the numerical solution becomes extremely sensitive (see Watson [95]).

We tested this example for $n = 1, 2, \cdots, 18$. Here, we only report the test results for the cases $n = 6$, $n = 8$ and $10 \leq n \leq 18$ with different choice of parameter τ, which are summarized in Table 6.1 and 6.2. The last column of Table 6.1 and 6.2 reports the optimal L^1 distance, which is calculated by $\|g_p^* - b_+\|_1 = \sum_{i=1}^n \frac{1}{i} x_i^* + \ln(\cos(1))$. From the above expression, the optimal value of problem (E_1) can be readily obtained, which is, $\|g_p^* - b_+\|_1 - \ln(\cos(1))$.

For $1 \leq n \leq 16$ our computational results are accurate in terms of both feasibility and objective optimality. As mentioned in the foregoing, different choices of parameter τ may bring in different entering constraint, thus influencing the computational efficiency of our algorithm. For this problem, we tested quite a few values of τ, in the hope to find the best value of τ at which our algorithm achieves the best performance. It seems that, for this example, $\tau = 0.95$ is close to the best possible choice in terms of CPU time consumption when the same level of accuracy of solution is taken into consideration.

The same problem has been considered in much literature. Most of the previous literature only reports computational results for $n \leq 9$. Goberna and López [29] solved the test problem for $n = 8$ by the grid discretization and the cutting plane method, reporting the optimal solutions with accuracy 1.0e-3 and 1.0e-4, respectively. In comparison, our algorithm takes much less CPU time to achieve the optimal solution of accuracy 1.0e-4 with higher feasibility accuracy. Though Ito et al. [45] reported optimal solutions with better accuracy, the authors did not consider the CPU assumption. Adopting a logarithmic barrier cutting plane algorithm, den Hertog et al. [15] tested the same problem for larger n values ($n = 10,\ 20,\ 30$). Even though satisfactory duality gaps for subproblems were achieved with this algorithm, the feasibility of obtained solutions was not guaranteed. By contrast, the proposed algorithm is capable of finding optimal solutions with ignorable constraint violation for $1 \leq n \leq 16$. However, we have to admit that, for $n \geq 17$, we frequently encountered the ill-conditioned linear systems in the progress of solving the base points.

Example 6.3. Find the best L^1 approximation of a given continuously differentiable function $b_+(t)$ from above by a Fourier sine series of order $\leq n$, in the form of $g_s(t) = \sum_{i=1}^n x_i \sin(it)$, over a given interval $[\alpha, \beta]$.

The problem is to determine the coefficient vector $x = [x_1; x_2; \cdots ; x_n]$ of $g_s(t)$ such that the L^1 distance between $g_s(t)$ and $b_+(t)$

$$\|g_s - b_+\|_1 = \int_\alpha^\beta |g_s(t) - b_+(t)| dt,$$

is minimized subject to $g_s(t) \geq b_+(t)$ for all $t \in [\alpha, \beta]$.

Here, we take a special case into consideration in which $b_+(t) = -\sqrt{\pi t - t^2}$ and $[\alpha, \beta] = [0, \pi]$. For $g_s(t) \geq b_+(t)$, we have

$$\begin{aligned}\|g_s - b_+\|_1 &= \int_0^\pi |g_s(t) - b_+(t)| dt \\ &= \sum_{i=1}^n x_i \int_0^\pi \sin(it) dt + \int_0^\pi \sqrt{\pi t - t^2} dt \\ &= \sum_{i=1}^n \frac{1 - \cos(i\pi)}{i} x_i + \frac{\pi^3}{8}.\end{aligned}$$

Therefore, the problem can be formulated as an LSIP problem as below:

$$\begin{aligned}(\mathrm{E}_2): \qquad & \min \quad \sum_{i=1}^n \frac{1 - \cos(i\pi)}{i} x_i \\ & \text{s.t.} \quad -\sum_{i=1}^n \sin(it) x_i \leq \sqrt{\pi t - t^2}, \quad t \in [0, \pi].\end{aligned}$$

For each n, an optimal solution x^* of problem (E_2) determines a best L^1 approximation $g_s^*(t) = \sum_{i=1}^n x_i^* \sin(it)$. We solve problem ($\mathrm{E}_2$) for $1 \leq n \leq 200$ and report computational results for various n values in Table 6.3.

To demonstrate the effect of approximation, we also depict the graphs of the best approximation $g_s^*(t)$ against that of $b_+(t)$ with $\tau = 1$ for $n = 5, 10, 15, 25, 55,$ and 200 in Figure 6.2 and Figure 6.3. As seen in Figure 6.3, the degree of approximation by $g_s^*(t)$ with $n = 55$ is satisfactory. And the effect of approximation by $g_s^*(t)$ for $n = 200$ is ideal.

A number of tests show that the algorithm is numerically stable (at least for tested cases). However, we would like to point out that, when n approaches 200, some effort needs to be made to find an initial ladder that does not involve a singularity (that is, the associated linear system is not ill-conditioned).

Once the optimal solution x^* of problem (E$_2$) is found, the minimum L^1 distance is given by

$$\|g_s^* - b_+\|_1 = \sum_{i=1}^{n} \frac{1-\cos(i\pi)}{i} x_i^* + \frac{\pi^3}{8}.$$

Example 6.4. Find the best L^1 approximation of a given continuously differentiable function $b_+(t)$ from above by a Fourier cosine series of order $\leq n-1$, in the form of $g_c(t) = \sum_{i=1}^{n} x_i \cos((i-1)t)$, over a given interval $[\alpha, \beta]$.

As in the above example, we take $b_+(t) = -\sqrt{\pi t - t^2}$ and $[\alpha, \beta] = [0, \pi]$. By simple manipulation (similar to that in Example 6.3), we can formulate this problem as an LSIP problem as following:

$$\begin{array}{lll} (\text{E}_3): & \min & \pi x_1 \\ & \text{s.t.} & -\sum_{i=1}^{n} \cos((i-1)t)x_i \leq \sqrt{\pi t - t^2}, \quad t \in [0, \pi]. \end{array}$$

We solve this problem for $1 \leq n \leq 200$ and summarize computational results for various n values in Table 6.4. In order to demonstrate the accuracy of approximation, we also present the graphs of the best approximation $g_c^*(t) = \sum_{i=1}^{n} x_i^* \cos((i-1)t)$ against that of $b_+(t)$ with $\tau = 1$ for $n = 5, 10, 15, 25, 55$ and 200 in Figure 6.4 and Figure 6.5. As seen in Figure 6.5, the degree of approximation by $g_c^*(t)$ with $n = 200$ is satisfactory.

For tested cases, the algorithm is computationally efficient. However, we would like to say that, when n is close to 200, some effort needs to be made to find an initial ladder which does not involve a singularity.

Once the optimal solution x^* of problem (E$_3$) is found, the optimal L^1 distance is calculated by

$$\|g_c^* - b_+\|_1 = \pi x_1^* + \frac{\pi^3}{8}.$$

6.6 Summary

In this chapter, we have extended the inclusive-cone-based method for LP to the LSIP case and done the following work:

Firstly, we introduced the concept of the generalized base point for LSIP. This concept builds upon the concept of base point for LP. As we have seen from the foregoing discussion, the concept of the generalized base point plays a significant role in development of the optimality result and the ladder algorithm for solving LSIP problems.

Secondly, we established an optimality result, which tells us that an LSIP problem with continuous linear inequality constraints has an optimal solution if and only if it has one at a feasible generalized base point.

Last but not the least, we developed a centered climbing ladder algorithm for solving LSIP problems, which is an LSIP version of a center climbing algorithm for LP. This algorithm iteratively updates the inclusive cone as well as the ladder to generate a sequence of base points, which ultimately converges to a feasible generalized base point, that is, an optimal solution of the problem. This new LSIP ladder algorithm combines an LP ladder algorithm and an efficient one-dimensional global optimization method—— the bridging method. At each iteration, the bridging method is capable of finding the exact global optimal solution which is corresponding to a violated constraint. (This global optimization method works faster than the linear search strategy over a set of dense grid index point used in discretization methods.) Computational results demonstrate that our algorithm has surprisingly good performance in terms of both CPU time and the accuracy of the solution.

Table 6.1: Computational results for Example 6.2 ($M = 50$)

n	τ	Iters.	CPU(s)	Max. Const. Vio.	Opt. Dist.
6	0.01	18	0.658005	3.026419e-15	4.586812e-4
	1	15	0.536005	1.268621e-15	4.586810e-4
	0.95	16	0.682806	2.344241e-16	4.586811e-4
8	0.01	20	1.135620	2.178005e-14	2.675330e-5
	1	19	0.995219	0	2.675325e-5
	0.95	20	0.932818	1.955741e-16	2.675318e-5
10	0.01	27	1.754113	0	1.589166e-6
	1	21	1.021227	3.851927e-17	1.589145e-6
	0.95	23	1.008900	4.058102e-16	1.589148e-6
11	0.01	31	2.124825	8.627571e-14	3.787589e-7
	1	19	1.009225	5.048957e-14	3.737784e-7
	0.95	20	0.946825	9.700573e-15	3.742882e-7
12	0.01	33	2.207455	6.047939e-14	1.145817e-7
	1	20	1.753125	2.354116e-13	1.112370e-7
	0.95	23	1.807025	2.438008e-14	1.123466e-7
13	0.01	35	2.956250	2.051373e-10	4.049794e-7
	1	20	1.725000	4.913628e-10	1.590831e-7
	0.95	18	1.523750	3.237483e-10	8.011145e-8
14	0.01	39	3.268125	3.262518e-10	9.787721e-7
	1	32	2.843750	1.839627e-10	6.052919e-7
	0.95	22	1.671650	7.038231e-10	7.424762e-8

Table 6.2: Computational results for Example 6.2 (to continue Table 6.1)

n	τ	Iters.	CPU(s)	Max. Const. Vio.	Opt. Dist.
15	0.01	41	3.687751	2.893904e-10	2.745783e-7
	1	28	2.675305	8.132726e-10	4.655475e-7
	0.95	30	2.031225	6.046296e-10	2.924098e-7
16	0.01	45	4.076100	7.884831e-10	8.275940e-7
	1	36	3.425000	8.493327e-10	2.667424e-7
	0.95	38	3.121875	8.174934e-10	1.616550e-7
17	0.01	37	2.500000	7.549062e-8	4.366051e-6
	1	30	2.458435	7.482757e-8	5.687806e-6
	0.95	31	2.346875	2.127828e-8	1.464241e-6
18	0.01	45	3.062500	7.048341e-8	7.458815e-5
	1	52	3.578125	9.554832e-8	1.977110e-4
	0.95	40	3.250000	7.360533e-8	5.147018e-5

Table 6.3: Computational results for Example 6.3 ($M = 5,\ \tau = 1$)

n	Iters.	CPU(s)	Max. Const. Vio.	Opt. Dist.
5	21	1.078125	9.992007e-15	0.200884
10	57	2.703125	5.206305e-13	0.101011
15	65	2.687500	1.907363e-13	0.052233
20	81	2.984375	9.769962e-15	0.037951
25	98	5.328125	9.066787e-14	0.025969
35	147	8.140625	7.838174e-14	0.016149
55	210	11.859375	2.719638e-12	0.008424
95	360	21.593750	1.731947e-13	0.003782
200	521	37.203125	2.856625e-8	0.001253

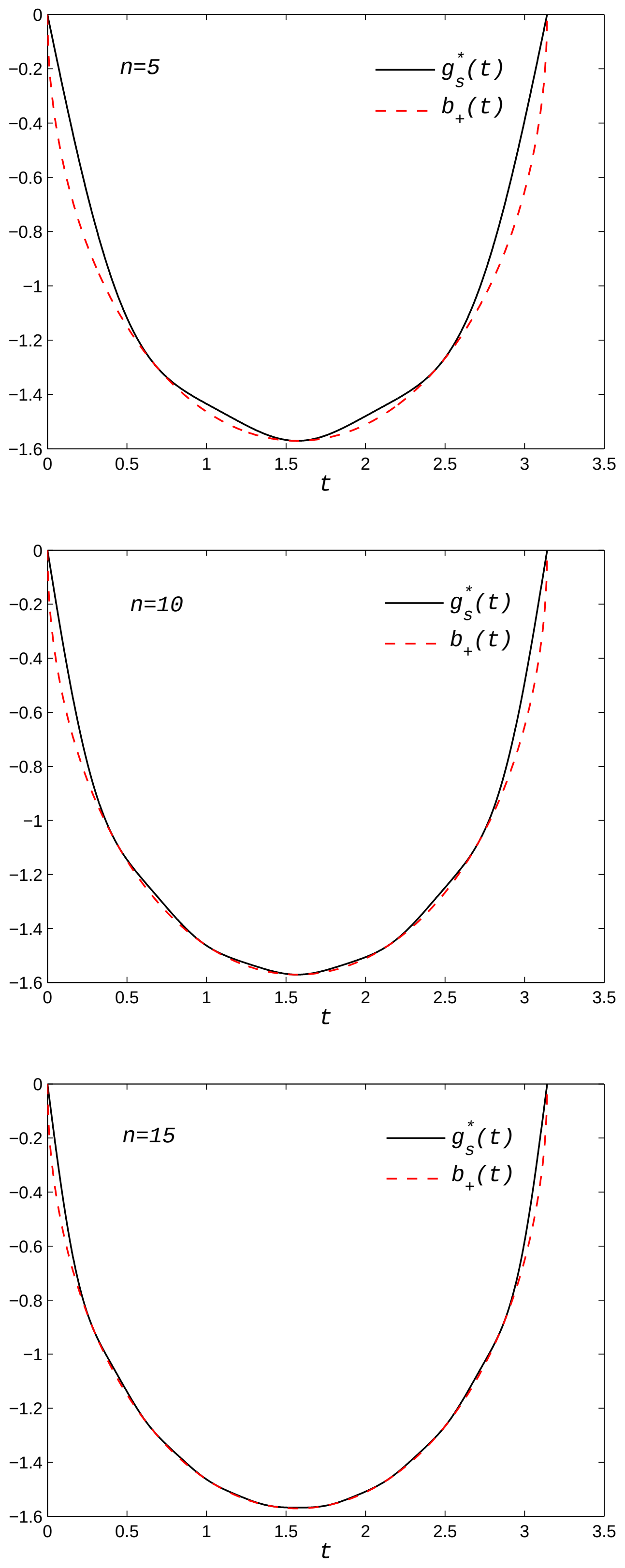

Figure 6.2: Approximating functions $g_s^*(t)$ against $b_+(t)$ in Example 6.3 (n=5, 10, 15)

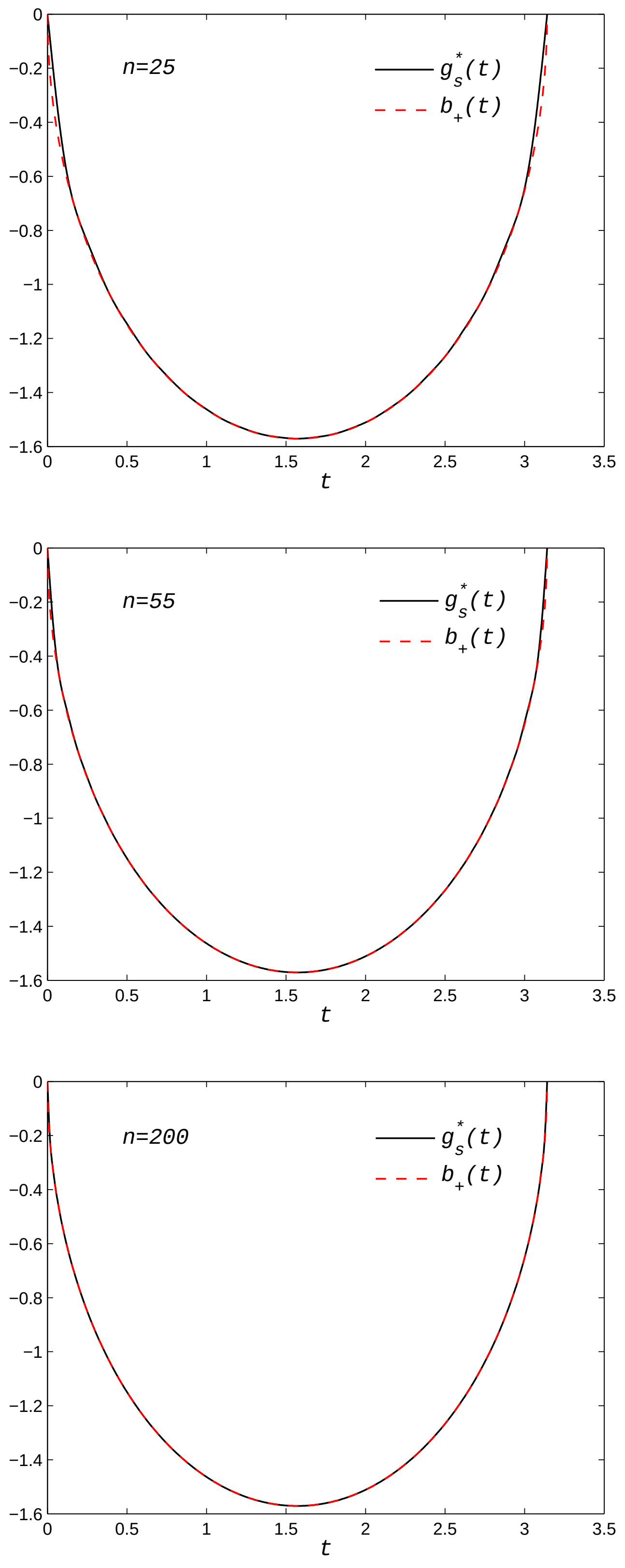

Figure 6.3: Approximating functions $g_s^*(t)$ against $b_+(t)$ in Example 6.3 (n=25, 55, 200)

Table 6.4: Computational results for Example 6.4 ($M = 5,\ \tau = 1$)

n	Iters.	CPU(s)	Max. Const. Vio.	Opt. Dist.
5	14	0.718750	6.661338e-15	0.774061
10	39	1.765625	2.220446e-16	0.362608
15	86	3.140625	1.079370e-12	0.179977
20	120	5.031250	2.238653e-12	0.128976
25	120	6.421875	2.344791e-13	0.087129
35	133	7.281750	2.176868e-10	0.053535
55	252	14.125000	6.944051e-14	0.027616
95	404	23.906250	7.016610e-14	0.012366
200	766	55.468750	9.083422e-8	0.004103

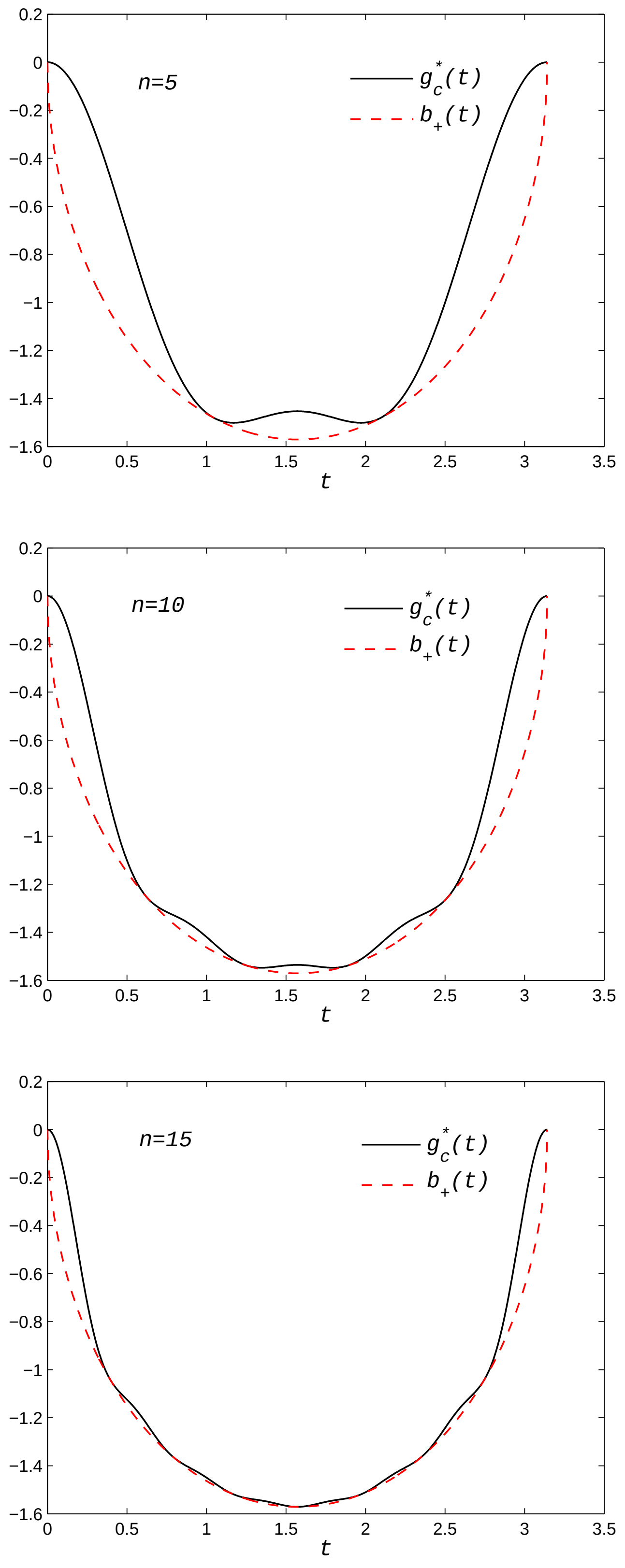

Figure 6.4: Approximating functions $g_c^*(t)$ against $b_+(t)$ in Example 6.4 (n=5, 10, 15)

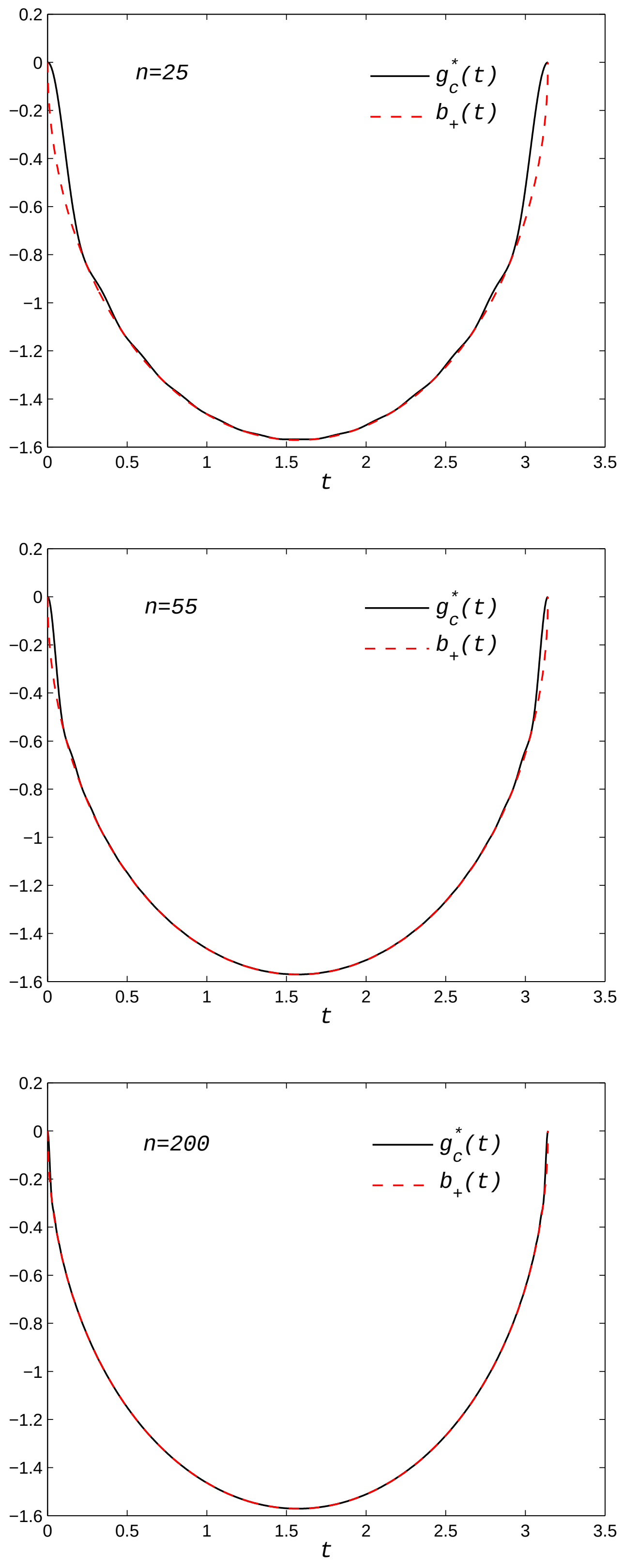

Figure 6.5: Approximating functions $g_c^*(t)$ against $b_+(t)$ in Example 6.4 (n=25, 55, 200)

Chapter 7

Conclusion and Future Works

In this dissertation, we have proposed a simple method for linear programming as well as linear semi-infinite programming, which is called the inclusive-cone-based method. Applying this method, we have done the following work.

In Chapter 2, we investigated theoretical aspects of linear programming from the perspective of the inclusive cone, demonstrating that the concept of the inclusive cone plays an indispensable role in deriving a set of inclusive-cone-based LP theory (especially the "symmetric" strong duality theory). Using the inclusive cone as an analytic tool, we examined sensitivity analysis in LP. In addition, we established the relationship of inclusiveness between correlated LP problems.

In Chapter 3, we presented an overview of LP algorithms. In Chapter 4, we developed several inclusive-cone-based ladder algorithms for solving LP problems. Numerical test results reveal the new algorithms outperform the simplex method in terms of CPU time for the tested problems.

In order to get the ladder algorithms started, we introduced some methods on how to find an initial ladder in Chapter 5. In particular, we proposed a simple and efficient initialization technique in Section 5.4. The famous Klee-Minty problem illustrated the effectiveness of this technique. Further, we developed an inclusive-cone-based solvability criterion in the context of the new category of LP problems, which enables us to diagnose accurately that an LP problem is inclusive-feasible (i.e., optimal), noninclusive-feasible (i.e., unbounded), inclusive-infeasible or noninclusive-infeasible.

We further generalized the inclusive-cone-based method for LP to the linear semi-infinite case in Chapter 6. On the basis of the concept of generalized base point, we established an optimality result for LSIP. With the optimality result as a theoretical foundation, we developed a new ladder algorithm for solving LSIP problems, termed "the centered climbing ladder algorithm for LSIP". The new LSIP ladder algorithm is a natural extension of a ladder algorithm for LP. At each iteration, the new algorithm chooses an incoming constraint most violated along a "parameterized centerline", by solving a one-dimensional global optimization problem using the efficient bridging algorithm. It is worth pointing out that the selection of the entering constraint allows a great degree of freedom, which is controlled by a parameter arising in the global optimization problem. We tested the proposed algorithm on several problems (the order of some tested problems is up to 200). Computational results show that our algorithm is efficient and has a superior numerical performance in terms of accuracy and CPU time.

Several possible directions are to be taken into consideration for future research. One direction is to adopt advanced coding techniques in our algorithms for further improvements in performance of the proposed ladder algorithms. For example, in the present codes we adopt the traditional technique of the inverse of matrix to calculate the sequence of base points x^k $(k = 0, 1, 2, \cdots,)$. If the LU decomposition technique was incorporated to calculate the sequence x^k, algorithmic performance could be improved. We would like to point out that, current numerical tests for new LP ladder algorithms are done on randomly generated problems. Once advanced coding techniques are adopted, we will test our algorithms on NETLIB problems. Another research direction is to apply the inclusive-cone-based method to parametric linear semi-infinite programming. As we have seen earlier (see Section 2.5, Chapter 2), the inclusive cone is a simple and useful tool for examining sensitivity analysis in LP. We will attempt to use this analytic tool to study parametric LSIP, in the hope to obtain some encouraging results. In addition, generalizing this inclusive-cone-based approach to general convex optimization problems is also one main direction for our future research.

Bibliography

[1] I. Adler and R. D. C. Monteiro. A geometric view of parametric linear programming. *Algorithmica*, 8:161–176, 1992.

[2] E. J. Anderson and A. S. Lewis. An extension of the simplex algorithm for semi-infinite linear programming. *Mathematical Programming*, 44(1-3):247–269, 1989.

[3] K. M. Anstreicher. Interior-point algorithms for a generalization of linear programming and weighted centring. *Optimization Methods & Software*, 27(4-5):605–612, 2012.

[4] H. Arsham. A hybrid gradient and feasible direction pivotal solution algorithm for general linear programs. *Applied Mathematics and Computation*, 188(1):596–611, 2007.

[5] H. Arsham, T. Damij, and J. Grad. An algorithm for simplex tableau reduction: the push-to-pull solution strategy. *Applied Mathematics and Computation*, 137(2-3):525–547, 2003.

[6] E. Barnes, V. Chen, B. Gopalakrishnan, and E. L. Johnson. A least-squares primal-dual algorithm for solving linear programming problems. *Operations Research Letters*, 30(5):289–294, 2002.

[7] B. Betrò. An accelerated central cutting plane algorithm for linear semi-infinite programming. *Mathematical Programming*, 101(3):479–495, 2004.

[8] K.-H. Borgwardt. The average number of pivot steps required by the simplex method is polynomial. *Zeitschrift für Operations Research*, 26:157–177, 1982.

[9] K.-H. Borgwardt. Some distribution-independent results about the asymptotic order of the average number of pivot steps of the simplex method. *Mathematics of Operations Research*, 7(3):441–462, 1982.

[10] A. Charnes, W. W. Cooper, and K. Kortanek. Duality, Haar programs and finite sequence spaces. *Proceedings of the National Academy of Sciences of the United States of America*, 48(5):783–786, 1962.

[11] A. Charnes, W. W. Cooper, and K. Kortanek. Duality in semi-infinite programs and some works of Haar and Carathéodory. *Management Science*, 9(2):209–228, 1963.

[12] M. Dambrine and M. Pierre. About stability of equilibrium shapes. *Mathematical Modelling and Numerical Analysis*, 34(4):811–834, 2000.

[13] G. B. Dantzig. Maximization of a linear function of variables subject to linear inequalities. In T. C. Koopmans, editor, *Activity Analysis of Production and Allocation*, pages 339–347, New York, 1951. John Wiley & Sons, Inc. Proceedings of a Conference.

[14] G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, Princeton, NJ, 1963.

[15] D. den Hertog, J. Kaliski, C. Roos, and T. Terlaky. A logarithmic barrier cutting plane method for convex programming. *Annals of Operations Research*, 58(2):69–98, 1995.

[16] I. I. Dikin. Iterative solution of problems of linear and quadratic programming. *Doklady Akademii Nauk SSSR*, 174:747–748, 1967. In Russian. English Translation: *Soviet Mathematics Doklady*, 8: 674-675, 1967.

[17] I. I. Dikin. On the speed of an iterative process. *Upravlaemye Sistemy*, 12:54–60, 1974. In Russian.

[18] M.-F. Ding, Y. Liu, and J. A. Gear. An improved targeted climbing algorithm for linear programs. *Numerical Algebra, Control and Optimization*, 1(3):399–405, 2011.

[19] M.-F. Ding, Y. Liu, and J. A. Gear. A modified centered climbing algorithm for linear programming. *Applied Mathematics*, 3:1423–1429, 2012.

[20] M. C. Ferris and A. B. Philpott. An interior point algorithm for semi-infinite linear programming. *Mathematical Programming*, 43:257–276, 1989.

[21] M. C. Ferris and A. B. Philpott. On affine scaling and semi-infinite programming. *Mathematical Programming*, 56(1-3):361–364, 1992.

[22] L. Finlay, V. Gaitsgory, and I. Lebedev. Duality in linear programming problems related to deterministic long run average problems of optimal control. *SIAM Journal on Control and Optimization*, 47(4):1667–1700, 2008.

[23] J. J. Forrest and D. Goldfarb. Steepest-edge simplex algorithms for linear programming. *Mathematical Programming*, 57(1-3):341–374, 1992.

[24] R. Fourer. Notes on the dual simplex method. 1994. Unpublished.

[25] V. Gaitsgory and M. Quincampoix. Linear programming approach to deterministic infinite horizon optimal control problems with discounting. *SIAM Journal on Control and Optimization*, 48(4):2480–2512, 2009.

[26] T. Gal. *Postoptimal Analysis, Parametric Programming and Related Topics.* McGraw-Hill, New York, 1979.

[27] T. Gal. Shadow prices and sensitivity analysis in linear programming under degeneracy. *OR Spectrum*, 8(2):59–71, 1986.

[28] M. A. Goberna. Linear semi-infinite optimization: recent advances. In V. Jeyakumar and A. Rubinov, editors, *Continuous Optimization: Current Trends and Modern Applications*, volume 99 of *Applied Optimization*, pages 3–22, New York, 2005. Springer.

[29] M. A. Goberna and M. A. López. *Linear Semi-Infinite Optimization.* John Wiley & Sons, Chichester, 1998.

[30] M. A. Goberna and M. A. López. *Semi-Infinite Programming: Recent Advances*, volume 57 of *Nonconvex Optimization and Its Applications.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[31] M. A. Goberna and M. A. López. Linear semi-infinite programming theory: An updated survey. *European Journal of Operational Research*, 143(2):390–405, 2002.

[32] M. A. Goberna, T. Terlaky, and M. I. Todorov. Sensitivity analysis in linear semi-infinite programming via partitions. *Mathematics of Operations Research*, 35(1):14–26, 2010.

[33] D. Goldfarb. Using the steepest-edge simplex algorithm to solve sparse linear programs. In J. R. Bunch and D. J. Rose, editors, *Sparse Matrix Computations*, pages 227–240, New York, 1976. Academic Press.

[34] D. Goldfarb and J. K. Reid. A practicable steepest-edge simplex algorithm. *Mathematical Programming*, 12(1):361–371, 1977.

[35] D. Goldfarb and W. Y. Sit. Worst case behavior of the steepest edge simplex method. *Discret Applied Mathematics*, 1(4):277–285, 1979.

[36] G. Gramlich, R. Hettich, and E. W. Sachs. Local convergence of SQP methods in semi-infinite programming. *SIAM Journal on Optimization*, 5(3):641–658, 1995.

[37] H. J. Greenberg. The use of the optimal partition in a linear programming solution for postoptimal analysis. *Operations Research Letters*, 15(4):179–185, 1994.

[38] H. J. Greenberg. Simultaneous primal-dual right-hand-side sensitivity analysis from a strictly complementary solution of a linear program. *SIAM Journal on Optimization*, 10(2):427–442, 2000.

[39] H. J. Greenberg, A. G. Holder, K. Roos, and T. Terlaky. On the dimension of the set of rim perturbations for optimal partition invariance. *SIAM Journal on Optimization*, 9(1):207–216, 1998.

[40] E. Haaren-Retagne. *A semi-infinite programming algorithm for robot trajectory planning.* Ph.D. Thesis, University of Trier, 1992.

[41] P. M. J. Harris. Pivot selection methods of the devex lp code. *Mathematical Programming*, 5(1):128, 1973.

[42] R. Hettich. An implementation of a discretization method for semi-infinite programming. *Mathematical Programming*, 34(3):354–361, 1986.

[43] R. Hettich and K. O. Kortanek. Semi-infinite programming: theory, methods and applications. *SIAM Riview*, 35(3):380–429, 1993.

[44] T. Illés and T. Terlaky. Pivot versus interior point methods: Pros and cons. *European Journal of Operational Research*, 140(2):170–190, 2002.

[45] S. Ito, Y. Liu, and K. L. Teo. A dual parametrization method for convex semi-infinite programming. *Annals of Operations Research*, 98(1-4):189–213, 2000.

[46] B. Jansen, J. J. de Jong, C. Roos, and T. Terlaky. Sensitivity analysis in linear programming: just be careful! *European Journal of Operational Research*, 101(1):15–28, 1997.

[47] L. S. Jennings and K. L. Teo. A computational algorithm for functional inequality constrained optimization problems. *Automatica*, 26(2):371–375, 1990.

[48] B. Jerez. A dual characterization of incentive efficiency. *Journal of Economic Theory*, 112(1):1–34, 2003.

[49] F. Juhnke and O. Sarges. Minimal spherical shells and linear semi-infinite optimization. *Contributions to Algebra and Geometry*, 41(1):93–105, 2000.

[50] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[51] L. G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20(1):191–194, 1979.

[52] V. Klee and G. J. Minty. How good is the simplex algorithm? In O. Shisha, editor, *Inequalities III*, pages 159–175, New York, 1972. Academic Press.

[53] M. Kojima, N. Megiddo, and S. Mizuno. A primal-dual infeasible-interior-point algorithm for linear programming. *Mathematical Programming*, 61(1-3):263–280, 1993.

[54] K. O. Kortanek and V. G. Medvedev. *Building and using dynamic interest rate models*. Wiley Finance Series. John Wiley & Sons, Ltd., Chichester, United Kingdom, 2001.

[55] M. López and G. Still. Semi-infinite programming. *European Journal of Operational Research*, 180(2):491–518, 2007.

[56] C. E Lemke. The dual method of solving the linear programming problem. *Naval Research Logistics Quarterly*, 1(1):36–47, 1954.

[57] C.-J. Lin, S.-C. Fang, and S.-Y. Wu. An unconstrained convex programming approach to linear semi-infinite programming. *SIAM Journal on Optimization*, 8(2):443–456, 1998.

[58] Q. Lin, R. Loxton, K. L. Teo, Y. H. Wu, and C. Yu. A new exact penalty method for semi-infinite programming problems. *Journal of Computational and Applied Mathematics*, 261:271–286, 2014. Accepted for publication, to appear in May 2014.

[59] G.-X. Liu. A homotopy interior point method for semi-infinite programming problems. *Journal of Global Optimization*, 37(4):631–646, 2007.

[60] Y. Liu. An exterior point linear programming method based on inclusive normal cones. *Journal of Industrial and Management Optimization*, 6(4):825–846, 2010.

[61] Y. Liu. Duality in linear programming: From trichotomy to quadrichotomy. *Journal of Industrial and Management Optimization*, 7(4):1003–1011, 2011.

[62] Y. Liu and M.-F. Ding. A ladder method for linear semi-infinite programming. *Journal of Industrial and Management Optimization*, 10(2), 2014. Accepted for publication, to appear in April 2014.

[63] Y. Liu, S. Ito, H. W. J. Lee, and K. L. Teo. Semi-infinite programming approach to continuously-constrained linear-quadratic optimal control problems. *Journal of Optimization Theory and Applications*, 108(3):617–632, 2001.

[64] Y. Liu and K. L. Teo. A bridging method for global optimization. *The Journal of the Australian Mathematical Society. Series B*, 41(1):41–57, 1999.

[65] Y. Liu and K. L. Teo. An adaptive dual parametrization algorithm for quadratic semi-infinite programming problems. *Journal of Global Optimization*, 24(2):205–217, 2002.

[66] Y. Liu, K. L. Teo, and S. Y. Wu. A new quadratic semi-infinite programming algorithm based on dual parametrization. *Journal of Global Optimization*, 29(4):401–413, 2004.

[67] N. Llorca, S. Tijs, and J. Timmer. Semi-infinite assignment problems and related games. *Mathematical Methods of Operations Research*, 57(1):67–78, 2003.

[68] M. Locatelli and F. Schoen. An adaptive stochastic global optimization algorithm for one-dimensional functions. *Annals of Operations Research*, 58(4):261–278, 1995.

[69] D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming.* International Series in Operations Research and Management Science. Springer, New York, 3rd edition, 2008.

[70] A. S. Manne. *Notes on parametric linear programming.* RAND Report P-468, The Rand Corporation, Santa Monica, CA, 1953.

[71] S. P. Marin. Optimal parametrization of curves for robot trajectory design. *IEEE Transactions on Automatic Control*, 33(2):209–214, 1988.

[72] A. T. N. Moghaddam and C. Michelot. A contribution to the linear programming approach to joint cost allocation: Methodology and application. *European Journal of Operational Research*, 197(3):999–1011, 2009.

[73] P. Moulin, M. Anitescu, K. O. Kortanek, and F. A. Potra. The role of linear semi-infinite programming in signal-adapted QMF bank design. *IEEE Transactions on Signal Processing*, 45(9):2160–2174, 1997.

[74] R. F. Noubiap and W. Seidel. An algorithm for calculating Γ-minimax decision rules under generalized moment conditions. *The Annals of Statistics*, 29(4):1094–1116, 2001.

[75] T. Okuno, S. Hayashi, and M. Fukushima. A regularized explicit exchange method for semi-infinite programs with an infinite number of conic constraints. *SIAM Journal on Optimization*, 22(3):1009–1028, 2012.

[76] M. R. Oskoorouchi, H. R. Ghaffari, T. Terlaky, and D. M. Aleman. An interior point constraint generation algorithm for semi-infinite optimization with health-care application. *Operations Research*, 59(5):1184–1197, 2011.

[77] E. Polak. On the mathematical foundations of nondifferentiable optimization in engineering design. *SIAM Review*, 29(1):21–89, 1987.

[78] E. Polak. *Optimization: Algorithms and Consistent Approximation*, volume 124 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1997.

[79] A. W. Potchinkov. Design of optimal linear phase FIR filters by a semi-infinite programming technique. *Signal Processing*, 58(2):165–180, 1997.

[80] C. J. Price. *Non-Linear Semi-Infinite Programming.* University of Canterbury, New Zealand, 1992.

[81] R. Reemtsen. Discretization methods for the solution of semi-infinite programming problems. *Journal of Optimization Theory and Applications*, 71(1):85–103, 1991.

[82] R. Reemtsen and J.-J. Rückmann. *Semi-infinite Programming.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[83] C. Roos. An exponential example for Terlaky's pivoting rule for the criss-cross simplex method. *Mathematical Programming*, 46(1-3):79–84, 1990.

[84] T. Saaty and S. Gass. Parametric objective function (part 1). *Journal of the Operations Research Society of America*, 2(3):316–319, 1954.

[85] O. Stein. *Bi-level Strategies in Semi-infinite Programming*, volume 71 of *Nonconvex Optimization and its Applications.* Kluwer Academic Publishers, Boston, 2003.

[86] O. Stein and G. Still. Solving semi-infinite optimization problems with interior point techniques. *SIAM Journal on Control and Optimization*, 42(3):769–788, 2003.

[87] K. L. Teo, X. Q. Yang, and L. S. Jennings. Computational discretization algorithms for functional inequality constrained optimization. *Annals of Operations Research*, 98(1-4):215–234, 2000.

[88] J. Timmer, S. Tijs, and N. Llorca. Games arising from infinite production situations. *International Game Theory Review*, 2(1):97–105, 2000.

[89] R. J. Vanderbei. *Linear Programming–Foundations and Extensions*. International Series in Opterations Research and Management Science. Springer, New York, 3rd edition, 2008.

[90] A. I. F. Vaz, E. M. G. P. Fernandes, and M. P. S. F. Gomes. Robot trajectory planning with semi-infinite programming. *European Journal of Operational Research*, 153(3):607–617, 2004.

[91] A. I. F. Vaz and E. C. Ferreira. Air pollution control with semi-infinite programming. *Applied Mathematical Modelling*, 33(4):1957–1969, 2009.

[92] B.-N. Vo, A. Cantoni, and K. L. Teo. Envelope constrained filter with linear interpolator. *IEEE Transactions on Signal Processing*, 45(6):1405–1414, 1997.

[93] G. A. Watson. A multiple exchange algorithm for multivariate Chebyshev approximation. *SIAM Journal on Numerical Analysis*, 12(1):46–52, 1975.

[94] G. A. Watson. Globally convergent methods for semi-infinite programming. *BIT Numerical Mathematics*, 21(3):362–373, 1981.

[95] G. A. Watson. Lagrangian methods for semi-infinite programming problems. In E. J. Anderson and A. B. Philpott, editors, *Infinite Programming*, volume 259 of *Lecture Notes in Economics and Mathematical Systems*, pages 90–107. Springer Berlin-Heidelberg, 1985.

[96] R. E. Wendell. Using bounds on the data in linear programming: The tolerance approach to sensitivity analysis. *Mathematical Programming*, 29(3):304–322, 1984.

[97] A. Winterfeld. Application of general semi-infinite programming to lapidary cutting problems. *European Journal of Operational Research*, 191(3):838–854, 2008.

[98] S. Y. Wu, S. C. Fang, and C. J. Lin. Relaxed cutting plane method for solving linear semi-infinite programming problems. *Journal of Optimization Theory and Applications*, 99(3):759–779, 1998.

[99] X. Xu and Y. Ye. A generalized homogeneous and self-dual algorithm for linear programming. *Operations Research Letters*, 17(4):181–190, 1995.

[100] E. A. Yildirim and M. J. Todd. An interior-point approach to sensitivity analysis in degenerate linear programs. *SIAM Journal on Optimization*, 12(3):692–714, 2002.

[101] K. F. C. Yiu, Y. Liu, and K. L. Teo. A hybrid descent method for global optimization. *Journal of Global Optimization*, 28(2):229–238, 2004.

[102] C. Yu, K. L. Teo, L. Zhang, and Y. Bai. A new exact penalty function method for continuous inequality constrained optimization problems. *Journal of Industrial and Management Optimization*, 6(4):895–910, 2010.

Appendix A

Procedure of Elimination for Linear System (5.1)

In this appendix we demonstrate how to eliminate n of y's components from the system of linear equations (5.1) (see Section 5.3, Chapter 5).

In the following, we use MATLAB function command $[R, jb]$=rref($\bullet$) to produce the reduced row echelon form R from the matrix $\bullet$, where jb denotes the index set of basic columns of R. Given an index set $J = \{1, 2, \cdots, m\}$, we denote by $J \setminus jb$ the set difference of index set jb from index set J. Supposing $jb = \{i_1, i_2, \cdots, i_n\} \subseteq J$, we use $I_m(:, jb)$ (alternatively, $I_m(:, i_1 : i_n)$) to denote a matrix consisting of i_1-th, i_2-th, $\cdots$, i_n-th columns of the identity matrix I_m. Also we denote by $y(jb)$ a column vector consisting of i_1-th, i_2-th, $\cdots$, i_n-th components of the column vector $y \in R^m$.

Rewrite (5.1) as

$$\begin{bmatrix} (c + A^T y_0)b^T - A^T & (c + A^T y_0)c^T \end{bmatrix} \begin{bmatrix} y \\ x \end{bmatrix} = c. \tag{A.1}$$

Let R be the reduced row echelon form of the matrix

$$\begin{bmatrix} (c + A^T y_0)b^T - A^T & (c + A^T y_0)c^T & c \end{bmatrix}$$

and let jb be the set of indices of the basic columns of R. That is

$$[R,\ jb] = \text{rref}\left(\begin{bmatrix} (c + A^T y_0)b^T - A^T & (c + A^T y_0)c^T & c \end{bmatrix} \right).$$

Let c_0 be the last column of R. $c_0 = R(:, m+n+1)$, $\hat{jb} = \{1, 2, \cdots, m\} \backslash jb$, $\bar{y} = y(jb)$, and $\hat{y} = y(\hat{jb})$. Then $y = I_m(:, jb)\bar{y} + I_m(:, \hat{jb})\hat{y}$, and

$$R(:, jb)\bar{y} + R(:, \hat{jb})\hat{y} + R(:, (m+1) : (m+n))x = c_0.$$

Since $R(:, jb) = I_n$, we have

$$\bar{y} = -R(:, \hat{jb})\hat{y} - R(:, (m+1) : (m+n))x + c_0$$

$$= -\begin{bmatrix} R(:, (m+1) : (m+n)) & R(:, \hat{jb}) \end{bmatrix} \begin{bmatrix} x \\ \hat{y} \end{bmatrix} + c_0.$$

Therefore

$$\begin{aligned} y &= I_m(:, jb)\bar{y} + I_m(:, \hat{jb})\hat{y} \\ &= I_m(:, jb)\left(-\begin{bmatrix} R(:, (m+1) : (m+n)) & R(:, \hat{jb}) \end{bmatrix} \begin{bmatrix} x \\ \hat{y} \end{bmatrix} + c_0 \right) + I_m(:, \hat{jb})\hat{y} \\ &= \begin{bmatrix} -I_m(:, jb)R(:, (m+1) : (m+n)) & -I_m(:, jb)R(:, \hat{jb}) + I_m(:, \hat{jb}) \end{bmatrix} \begin{bmatrix} x \\ \hat{y} \end{bmatrix} \\ &\quad + I_m(:, jb)c_0. \end{aligned}$$

This gives

$$\begin{aligned} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} I_n & 0_{n\times(m-n)} \\ -I_m(:, jb)R(:, (m+1) : (m+n)) & -I_m(:, jb)R(:, \hat{jb}) + I_m(:, \hat{jb}) \end{bmatrix} \\ &\quad \times \begin{bmatrix} x \\ \hat{y} \end{bmatrix} + \begin{bmatrix} 0_{n\times 1} \\ I_m(:, jb)c_0 \end{bmatrix} \\ &= M_3 \begin{bmatrix} x \\ \hat{y} \end{bmatrix} + c_0^{'}, \end{aligned}$$

where

$$M_3 = \begin{bmatrix} I_n & 0_{n\times(m-n)} \\ -I_m(:, jb)R(:, (m+1) : (m+n)) & -I_m(:, jb)R(:, \hat{jb}) + I_m(:, \hat{jb}) \end{bmatrix},$$

$$c_0^{'} = \begin{bmatrix} 0_{n\times 1} \\ I_m(:, jb)c_0 \end{bmatrix}.$$