An iterative method for solving KKT system of the semi-infinite programming

SOON-YI WU*†, DONG-HUI LI‡, LIQUN QI§ and GUANGLU ZHOU¶
†Institute of Applied Mathematics, National Cheng-Kung University, Taiwan
‡Institute of Applied Mathematics, Hunan University, China
§Department of Applied Mathematics, Hong Kong Polytechnic University, Hong Kong
¶Department of Mathematics and Statistics, Curtin University of Technology, Bentley, WA 6102, Australia

(Received October 2003; in final form December 2003)

We develop an iterative method for solving the KKT system of the semi-infinite programming (SIP) problem. At each iteration, we solve the KKT system of a nonlinear programming problem with finite constraints by a semismooth Newton method. The algorithm either terminates at a KKT point of the SIP problem in finitely many iterations or generates an infinite sequence of iterates whose any accumulation point is a KKT point of the problem. We also analyse the convergence rate of the method. Preliminary numerical results are reported.

Keywords: Semi-infinite programming; KKT system; Semismooth equation

1. Introduction

We consider the following semi-infinite programming (SIP) problem:

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g(x, t) \leq 0, \ t \in T,
\end{align*}
\]

where \( f: \mathbb{R}^n \rightarrow \mathbb{R} \) and \( g: \mathbb{R}^{n+m} \rightarrow \mathbb{R} \) are continuously differentiable and \( T \subset \mathbb{R}^m \) is a compact set. Denote by \( \mathcal{F} \) the feasible set of problem (1), i.e.,

\[
\mathcal{F} = \{ x \in \mathbb{R}^n \mid g(x, t) \leq 0, \ t \in T \}.
\]

Let \( T(x) \) denote the active set of problem (1) at \( x \), i.e.,

\[
T(x) = \{ t \in T : g(x, t) = 0 \}.
\]
The SIP problem has a strong practical background. It arises in various fields of engineering such as approximation theory, optimal control, resource allocation in decentralized systems, decision making under competition and optimum filter design in signal processing [see, e.g., [9,12–14] and references therein]. During the last decade, many numerical methods have been developed for solving the SIP problem. We refer to refs. [2,3,6–10,15,20–23] for early methods. A typical algorithm for solving problem (1) generally generates a sequence of finitely constrained auxiliary optimization problems that can be solved by standard algorithms for the NLP problem. Existing methods for equation (1) can be roughly divided into three classes according to the ways the auxiliary problems are generated: exchange methods, discretization methods and reduction-based methods. Exchange and discretization methods are numerically very expensive. The cost at each iteration even increases dramatically as the cardinality of the auxiliary problem grows. In contrast, the drawbacks of reduction-based methods are also obvious, they require strong assumptions and are often conceptual methods which can be implemented only in a rather simple form. Exchange and discretization methods, therefore, are often used only for the first stage of the solution process to generate an approximate solution of equation (1), whereas reduction-based methods are typically employed only in the final stage of the solution process in order to provide a higher accuracy of the solution and a better rate of convergence.

Let $x^*$ be a solution of equation (1). It is well known that if some constraint qualification holds at $x^*$, then there exist a nonnegative integer $p \leq n$ and multipliers $u^*_i$, where $i = 1, \ldots, p$ such that vectors $\nabla_x g(x^*, t^*_i)$ are linearly independent and that

$$\begin{align*}
\nabla f(x^*) + \sum_{i=1}^{p} u^*_i \nabla_x g(x^*, t^*_i) &= 0, \\
u^*_i &> 0, \quad g(x^*, t^*_i) = 0, \quad i = 1, \ldots, p, \\
g(x^*, t) &\leq 0, \quad \forall t \in T.
\end{align*}$$

If some sufficient conditions hold at $x^*$, then $x^*$ is a solution of equation (1).

The KKT system (2) can be reformulated as a semismooth equation in some sense. Quite recently, On the basis of the semismooth equation reformulation of equation (2), a smoothing Newton method [11] and a semismooth Newton method [19] were proposed. The global and superlinear convergence of these methods were proved. Compared with previous methods, the advantage of these methods is that only a system of linear equations needs to be solved at each iteration. However, the conditions that guarantee the accumulation point of the generated iterative sequence to be a feasible point of the problem are restricted.

In this paper, we present an iterative method for finding a KKT point of the SIP problem (1). At each iteration, we solve the KKT system of an ordinary nonlinear programming problem by a semismooth Newton method. Under mild conditions, we prove that any accumulation point of the generated iterative sequence is a KKT point of equation (1). We also analyse the convergence rate of the basic method. Preliminary numerical results are reported to illustrate the effectiveness of the algorithm.

In section 2, we propose a basic method and show its convergence. Section 3 gives a semismooth Newton method for solving the KKT system of a nonlinear programming problem with finite constraints. In section 4, we discuss the convergence rate of the basic method. In section 5, we modify the basic method to improve its efficiency. The modified methods are computationally cheaper than the basic method. We also prove the convergence of the modified method. In section 6, we present some preliminary numerical results, which show that the modified methods are promising. Some conclusions are given in section 7.
2. The basic algorithm and its convergence

In this section, we describe the algorithm and prove its convergence. Let $T_k$ be a finite subset of $T$. Consider the nonlinear programming problem:

$$
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g(x, t) \leq 0, \quad t \in T_k.
\end{align*}
$$

(3)

Suppose that some constraint qualification holds at a solution $x^k$ of equation (3). Then, there exist multipliers $u^k_i$, where $i \in T_k$ such that $(x^k, u^k)$ is a solution of the following KKT system:

$$
\begin{align*}
\nabla f(x) + \sum_{t_i \in T_k} u_i \nabla_x g(x, t_i) &= 0, \\
\int g(x, t_i) d\mu &= 0, \\
g(x, t_i) &\leq 0,
\end{align*}
$$

(4)

It is easy to see by comparing system (4) with system (2) that if $x^k$ is feasible, then, $(x^k, u^k)$ is a KKT point of equation (1). In other words, if we can find a finite subset $T_k$ of $T$ such that the solution of system (4) is feasible, then the KKT system of equation (1) reduces to the KKT system of an ordinary nonlinear programming, which can be easily solved by existing methods. Unfortunately, it seems very difficult to find such a finite subset $T_k$. In this paper, we are going to find a sequence of finite subset $T_k$ of $T$ such that any limit point of $\{x^k\}$ is a solution of the KKT system of equation (1).

Let $t_k$ be a global solution of the nonlinear programming:

$$
\max_{t \in T} g(x^k, t),
$$

(5)

It is obvious that $x^k$ is feasible if and only if $g(x^k, t_k) \leq 0$. Therefore, if $g(x^k, t_k) > 0$, we must have $t_k \notin T_k$. In this case, it would be reasonable to add $t_k$ to $T_k$ to get $T_{k+1}$. This motivates our algorithm, which is stated as follows:

ALGORITHM 2.1

Initial. Given a finite subset $T_0 \subset T$ such that $\{x \in \mathbb{R}^n : g(x, t) \leq 0, \forall t \in T_0\} \neq \emptyset$. Let $k = 0$.  
Step 1. Solve the KKT system (4) to get $(x^k, u^k)$. 
Step 2. Solve the global optimization problem (5) to get $t_k$. 
Step 3. Stop if $g(x^k, t_k) \leq 0$. Otherwise, let $T_{k+1} := T_k \cup \{t_k\}$. Set $k := k + 1$ and go to Step 1.

Let $T_k = \{t_0, t_1, \ldots, t_p\}$ and $T^* = \{t^*_1, \ldots, t^*_p\}$. Here $T^*$ is the index set satisfying equation (2). Define discrete measures $\mu^*$ and $\mu_k$ with finite support as follows:

$$
\mu^*(t) = \begin{cases} 
 u^*_i, & \text{if } t = t^*_i, \quad i = 1, \ldots, p \\
 0, & \text{if } t \notin T^*. 
\end{cases}
$$

(6)

and

$$
\mu_k(t) = \begin{cases} 
 u^k_i, & \text{if } t = t_i, \quad i = 0, \ldots, p_k \\
 0, & \text{if } t \notin T_k.
\end{cases}
$$

(7)

Then the KKT systems (2) and (4) can be written as

$$
\begin{align*}
\int g(x^*, t) d\mu^* &= 0, \\
\int g(x^*, t) d\mu &= 0, \\
g(x^*, t) &\leq 0, \quad \forall t \in T
\end{align*}
$$

(8)
and

\[
\begin{align*}
\nabla f(x^k) + \int \nabla_x g(x^k, t) d\mu_k &= 0, \\
\int g(x^k, t) d\mu_k &= 0, \\
g(x^k, t) &\leq 0, \quad t \in T_k
\end{align*}
\]

(9)

respectively.

The following theorem establishes the convergence of Algorithm 2.1.

**Theorem 2.1**  Let \( \{x^k\} \) be an infinite sequence generated by Algorithm 2.1. Then, any limit point of the sequence \( \{x^k\} \) is a solution of equation (2).

**Proof**  Define discrete measure \( \mu^k \) with respect to \( u^k \) and \( T_k \) as in equation (7). Let \( \{x_{nk}\} \) be a subsequence of \( \{x^k\} \) such that \( x_{nk} \to x^* \). Without loss of generality, we assume that \( \mu_{nk} \rightharpoonup \mu^* \). We first prove that

\[
\nabla f(x^*) + \int \nabla_x g(x^*, t) d\mu^* = 0.
\]

(10)

It follows from the first equality of (9) that

\[
\left\| \nabla f(x^*) + \int \nabla_x g(x^*, t) d\mu^* \right\| = \left\| (\nabla f(x_{nk})) + \int \nabla_x g(x_{nk}, t) d\mu_{nk} - (\nabla f(x^*)) + \int \nabla_x g(x^*, t) d\mu^* \right\|
\]

\[
\leq \left\| \nabla f(x_{nk}) - \nabla f(x^*) \right\| + \left\| \int \nabla_x g(x_{nk}, t) d\mu_{nk} - \int \nabla_x g(x^*, t) d\mu^* \right\|
\]

\[
+ \left\| \int \nabla_x g(x^*, t) d\mu_{nk} - \int \nabla_x g(x^*, t) d\mu^* \right\|.
\]

(11)

As \( x_{nk} \to x^* \) and \( \mu_{nk} \rightharpoonup \mu^* \), equality (10) follows from inequality (11). Similarly, we can deduce

\[
\int g(x^*, t) d\mu^* = 0
\]

by using the second equality of (9).

Next, we prove that \( x^* \) is feasible. In other words, we need to verify that \( \max_{t \in T} g(x^*, t) \leq 0 \). Let \( \bar{t} \in \arg \max_{t \in T} g(x^*, t) \). Then, it holds that

\[
g(x_{nk}, \bar{t}) \leq g(x_{nk}, t_{nk}),
\]

where \( t_{nk} \in \arg \max_{t \in T} g(x_{nk}, t) \). As \( T \) is a compact set, without loss of generality, we may assume that \( t_{nk} \to t^* \). Taking limits, we get

\[
g(x^*, \bar{t}) \leq g(x^*, t^*).
\]

(12)

We claim that for \( k = 1, 2, \ldots \), \( g(x^*, t_{nk}) \leq 0 \). Otherwise, there exists a positive integer \( N \) such that \( g(x^*, t_{nN}) > 0 \). Thus, there exists a positive integer \( \bar{N} > N \) large enough such that \( g(x_{n\bar{N}}, t_{n\bar{N}}) > 0 \). As \( \bar{N} > N \) and \( t_{n\bar{N}} \in T_{n\bar{N}} \), by the third inequality of (9), we
3. The subproblem

In this section, we provide a semismooth Newton method for solving the semismooth equation reformulation of the KKT system (4). Let $\phi_{FB}: R^2 \to R$ be the Fischer–Burmeister function \cite{4} defined by

$$\phi_{FB}(a, b) = (a + b) - \sqrt{a^2 + b^2}.$$

It is easy to verify that $\phi_{FB}$ has the following property:

$$\phi_{FB}(a, b) = 0 \iff a \geq 0, \quad b \geq 0 \text{ and } ab = 0.$$

By using $\phi_{FB}$, the KKT system (4) can be reformulated as a system of nonlinear equations. For each $t_i \in T_k$, let $\phi_i(x, u) = \phi_{FB}(-g(x, t_i), u_i)$. Define $\Phi_1$ by

$$\Phi_1(x, u) = (\phi_i(x, u))^T_{i \in T_k}.$$

It is obvious that the KKT system (4) can be reformulated as the following system of nonlinear equations:

$$H_k(z) = \left( \begin{array}{c} L_k(x, u) \\ \Phi_1(x, u) \end{array} \right) = 0,$$

where $z = (x, u)$ and

$$L_k(z) = \nabla f(x) + \sum_{i \in I_k} u_i \nabla x g(x, t_i).$$

Function $\phi_{FB}$ is continuously differentiable everywhere except at the origin but it is semismooth at the origin. Consequently, system (13) is semismooth. For the concept of semismooth and the semismooth Newton method, we refer to refs. \cite{16,18}. Denote

$$I_0(z) = \{i: g(x, t_i) = 0 \text{ and } u_i = 0\}, \quad I_1(z) = \{i: g(x, t_i)^2 + u_i^2 \neq 0\}.$$

By direct computation, it is easy to see that for each $i \in I_1(z)$, $\phi_i$ is differentiable with derivative

$$\nabla x \phi_i(x, u) = -\left(1 + \frac{g(x, t_i)}{\sqrt{(g(x, t_i)^2 + u_i^2)}}\right) \nabla_x g(x, t_i), \quad \nabla u \phi_i(x, u) = \left(1 - \frac{u_i}{\sqrt{(g(x, t_i)^2 + u_i^2)}}\right) e_i,$$

where $e_i$ denotes the $i$th coordinate. For $i \in I_0(z)$, $\phi_i$ is semismooth with generalized derivative \cite{1}

$$\partial \phi_i(x, u) = \left\{ \begin{array}{c} \alpha \nabla x g(x, t_i) \\ \gamma e_i \end{array} : (\alpha - 1)^2 + (\gamma - 1)^2 \leq 1 \right\}.$$

Therefore, every element in $\partial H_k(z)$, where $\partial H_k$ is the generalized Jacobian of $H_k$ in the sense of Clarke \cite{1}, can be expressed as

$$\begin{pmatrix} \nabla x L_k(z) & \nabla u L_k(z) \\ \Lambda_k(z) G_k(x) & \Gamma_k(z) \end{pmatrix}.$$
where
\[ G_k(x) = (\nabla_x g(x, t_i))_{t_i \in T_i}, \quad \Lambda_k(z) = \text{diag} (\omega_i(z))_{t_i \in T_i}, \quad \Gamma_k(z) = \text{diag} (\gamma_i(z))_{t_i \in T_i}. \]

Here,
\[ \alpha_i(z) = -1 - \frac{g(x, t_i)}{\sqrt{(g(x, t_i)^2 + u_i^2)}} \quad \text{and} \quad \gamma_i(z) = 1 - \frac{u_i}{\sqrt{(g(x, t_i)^2 + u_i^2)}} \quad \text{if} \quad i \in I_1(z), \]
\[ \alpha_i(z) \quad \text{and} \quad \gamma_i(z) \quad \text{satisfy} \quad (\alpha_i(z) - 1)^2 + (\gamma_i(z) - 1)^2 \leq 1 \quad \text{if} \quad i \in I_0(z). \]

Semismooth Newton methods have been extensively studied in the recent decade. We refer to refs. [5,16–18] for this method. In what follows, we give a semismooth Newton method for solving equation (13), which is the same as our previous work [19]. Let
\[ \theta_k(z) = \frac{1}{2} H_k(z)^T H_k(z). \]
\( \theta \) is continuously differentiable with the gradient given by
\[ \nabla \theta_k(z) = W^T H_k(z), \]
where \( W \in \partial H_k(z). \)

**Algorithm 3.1**

**Initial.** Let \( z^0 \in R^{n+(m+q+1)p} \), \( \sigma, \rho \in (0, 1) \), \( \eta > 0 \), \( a > 2 \) and set \( i := 0 \).

**Step 1.** If \( H_k(z^i) = 0 \), stop. Otherwise, let \( d^i \) be a solution of
\[ H_k(z^i) + W^i d = 0, \quad W^i \in \partial H_k(z^i). \]
If problem (14) is not solvable, or if \( \nabla \theta_k(z^i)^T d^i > -\eta \|d^i\|^a \), set \( d^i = -\nabla \theta_k(z^i). \)

**Step 2.** Let \( \alpha_i = (\rho)^j \), where \( j_i \) is the smallest nonnegative integer \( j \) such that
\[ \theta_k(z^i + (\rho)^j d^i) - \theta_k(z^i) \leq \sigma (\rho)^j \nabla \theta_k(z^i)^T d^i, \]
where \( (\rho)^j \) means the \( j \)th power of \( \rho \).

**Step 3.** Let \( z^{i+1} := z^i + \alpha_i d^i \) and \( i := i + 1 \). Go to Step 1.

**4. Convergence rate**

In this section, we investigate the convergence rate of Algorithm 2.1. Throughout this section, we suppose that \( x^k \) is a local optimal solution of the nonlinear programming problem (3). This assumption holds if the second order sufficient condition holds at \( x^k \).

Let \( z^{n_k} = (x^{n_k}, u^{n_k}) \) converge to \( z^* = (x^*, u^*) \). As \( T_{n_k} \subset T \), the feasible set \( \mathcal{F} \) of problem (1) is contained in the feasible set of problem (3). Without loss of generality, we assume that \( f(x^*) \geq f(x^{n_k}) \) throughout this section.

**Definition 4.1** For the SIP problem (1), a feasible point \( \bar{x} \in \mathcal{F} \) is said to be a strict local minimizer of order \( q > 0 \) if there is a neighbourhood \( U \) of \( \bar{x} \) and a constant \( m > 0 \) such that
\[ f(x) - f(\bar{x}) \geq m \|x - \bar{x}\|^q, \quad \forall x \in \mathcal{F} \cap U. \]
A local minimizer of order \( q = 1 \) is also called strongly unique local minimizer.
We say that the strong Mangasarian–Fromovitz constraint qualification (sMFCQ) is valid at \( \bar{x} \in F \) if there exists a vector \( \xi \in \mathbb{R}^n \) such that
\[
\xi^T \nabla_x g(\bar{x}, t) \leq -1, \quad \forall t \in T(\bar{x}).
\] (16)

**CONDITION A1** The gradient \( \nabla_x g(x, t) \) is continuous on \( U^* \times T \), where \( U^* \) is a neighbourhood of \( x^* \), and sMFCQ is valid at the local minimizer \( x^* \) of equation (1).

Let
\[
\delta_i = \max_{t \in T} g(x^i, t).
\]
It follows from \( x^{n_k} \to x^* \), \( \delta_{n_k} > 0 \) and \( x^* \in F \) that \( \delta_{n_k} \to 0 \).

**LEMMA 4.1** Let \( x^{n_k} \to x^* \). Suppose Condition A1 holds. Then for any \( \rho > 2 \), when \( n_k \) is sufficiently large, \( \hat{x}^{n_k} = x^{n_k} + \rho \delta_{n_k} \xi \) is feasible for equation (1), i.e., \( g(\hat{x}^{n_k}, t) \leq 0 \) for all \( t \in T \), where \( \xi \) is a vector defined as in equation (16) with \( \bar{x} \) replaced by \( x^* \).

**Proof** For \( \epsilon > 0 \), we consider the relative open set
\[
T^\epsilon(x^*) = \{ t \in T : \| t - \bar{t} \| < \epsilon, \ \text{for some} \ \bar{t} \in T(x^*) \}.
\]
By the sMFCQ and the continuity of \( \nabla_x g \), there is some \( \epsilon > 0 \) such that
\[
\xi^T \nabla_x g(x, t) \leq -\frac{1}{2},
\]
for any \( t \in T^\epsilon(x^*) \) and any \( x \) satisfying \( \| x - x^* \| < \epsilon \). For given \( \rho > 0 \), when \( n_k \) is sufficiently large, it holds that \( \| \rho \delta_{n_k} \xi \| < \epsilon \). Therefore, for any \( t \in T^\epsilon(x^*) \),
\[
g(\hat{x}^{n_k}, t) = g(x^{n_k}, t) + \rho \delta_{n_k} \xi^T \nabla_x g(x^{n_k}, t) + o(\rho \delta_{n_k})
\]
\[
\leq \delta_{n_k} - \frac{1}{2} \rho \delta_{n_k} + o(\rho \delta_{n_k})
\]
\[
= \left( 1 - \frac{1}{2} \rho \right) \delta_{n_k} + o(\rho \delta_{n_k})
\]
\[
\leq 0,
\] (17)
where the last inequality follows from \( \rho > 2 \).

Now we consider the compact set \( T \setminus T^\epsilon(x^*) \). By the continuity of \( g \), there exists \( \epsilon_1 > 0 \) such that
\[
g(x, t) < 0, \quad \forall t \in T \setminus T^\epsilon(x^*) \quad \text{and} \quad x \text{ satisfying} \ \| x - x^* \| < \epsilon_1.
\]
As \( x^{n_k} \) converges to \( x^* \), for \( n_k \) sufficiently large, we have \( g(\hat{x}^{n_k}, t) < 0, \forall t \in T \setminus T^\epsilon(x^*) \).
Together with equation (17), \( g(\hat{x}^{n_k}, t) \leq 0, \forall t \in T \). The proof is completed.

**THEOREM 4.1** Let \( x^* \in F \) be a strict local minimizer of order \( q \geq 1 \). Suppose that Condition A1 holds. Then there exists a \( \sigma > 0 \) such that for all \( \delta_{n_k} > 0 \) small enough,
\[
\| x^{n_k} - x^* \| \leq \sigma \delta_{n_k}^{1/q}.
\]
Proof It follows from Lemma 4.1 that there exists a $\rho > 0$ such that for small enough $\delta_{n_k} > 0$, $x^{n_k} + \rho \delta_{n_k} \xi$ is feasible, i.e.,

$$g(x^{n_k} + \rho \delta_{n_k} \xi, t) \leq 0, \quad \forall t \in T,$$

where $\xi$ is a vector defined as in equation (16) with $\bar{x}$ replaced by $x^*$. For a given sufficiently small $\epsilon > 0$, we can take $\delta_{n_k}$ small enough such that $\|x^{n_k} + \rho \delta_{n_k} \xi - x^*\| < \epsilon$. Then we have

$$f(x^{n_k} + \rho \delta_{n_k} \xi) \geq f(x^*).$$

By the Lipschitz continuity of $f$, it follows that

$$0 \leq f(x^*) - f(x^{n_k}) \leq f(x^{n_k} + \rho \delta_{n_k} \xi) - f(x^{n_k}) = O(\delta_{n_k}).$$

From equation (15),

$$\|x^{n_k} + \rho \delta_{n_k} \xi - x^*\|^q \leq \frac{f(x^{n_k} + \rho \delta_{n_k} \xi) - f(x^*)}{m} \leq \frac{f(x^{n_k}) - f(x^{n_k})}{m} = O(\delta_{n_k}).$$

Thus,

$$\|x^{n_k} - x^*\| \leq \|x^{n_k} + \rho \delta_{n_k} \xi - x^*\| + \|\rho \delta_{n_k} \xi\| = O(\delta_{n_k}^{1/q}) + O(\delta_{n_k}) = O(\delta_{n_k}^{1/q}).$$

The proof is completed. ■

**CONDITION A2** For any $t \in T$, function $g(\cdot, t)$ is convex.

**LEMMA 4.2** Suppose that Condition A2 holds and there exist $\bar{x} \in \mathbb{R}^n$ and $\bar{\epsilon} > 0$ such that $g(\bar{x}, t) \leq -\bar{\epsilon}$ for all $t \in T$. Then, when $n_k$ is sufficiently large,

$$\frac{\bar{\epsilon} x^{n_k}}{\bar{\epsilon} + \delta_{n_k}} + \frac{\delta_{n_k}}{\bar{\epsilon} + \delta_{n_k}} \bar{x}$$

is a feasible point for problem (2).

**Proof** For sufficiently large $n_k$, as $\delta_{n_k} g(\bar{x}, t) \leq -\bar{\epsilon} \delta_{n_k}$ and $g(\cdot, t)$ is convex for all $t \in T$,

$$g\left(\frac{\bar{\epsilon} x^{n_k}}{\bar{\epsilon} + \delta_{n_k}} + \frac{\delta_{n_k}}{\bar{\epsilon} + \delta_{n_k}} \bar{x}, t\right) \leq \frac{\bar{\epsilon}}{\bar{\epsilon} + \delta_{n_k}} g(x^{n_k}, t) + \frac{\delta_{n_k}}{\bar{\epsilon} + \delta_{n_k}} g(\bar{x}, t) \leq \frac{\bar{\epsilon} \delta_{n_k}}{\bar{\epsilon} + \delta_{n_k}} - \frac{\bar{\epsilon} \delta_{n_k}}{\bar{\epsilon} + \delta_{n_k}} = 0.$$

This completes the proof. ■

**THEOREM 4.2** Suppose that the conditions of Lemma 4.4 hold. If $x^* \in \mathcal{F}$ is a strict local minimizer of order $q \geq 1$, then there exists a constant $\tilde{\sigma} > 0$ such that for all $\delta_{n_k}$ small enough,

$$\|x^{n_k} - x^*\| \leq \bar{\sigma} \delta_{n_k}^{1/q}.$$
Method for solving KKT system of the SIP

Proof It follows from Lemma 4.4 that \( \left( \frac{1}{\bar{\epsilon}} + \delta_{nk} \right) \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) \) is feasible for equation (1) when \( n_k \) is sufficiently large. For a given sufficiently small \( \epsilon > 0 \), we can take \( \delta_{nk} \) small enough such that

\[
\left\| \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) - x^* \right\| < \epsilon.
\]

Then

\[
f \left( \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) \right) \geq f(x^*).
\]

By the Lipschitz continuity of \( f \), it follows that

\[
0 \leq f(x^*) - f(x^{nk}) \leq f \left( \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) \right) - f(x^{nk}) = O(\delta_{nk}).
\]

From equation (15),

\[
\left\| \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) - x^* \right\|^q \leq \frac{1}{m} \left( f \left( \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) \right) - f(x^*) \right) = \frac{1}{m} \left( f \left( \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) \right) - f(x^{nk}) \right) - \frac{1}{m} \left( f(x^*) - f(x^{nk}) \right) = O(\delta_{nk}).
\]

Thus,

\[
\left\| x^{nk} - x^* \right\| = \left\| \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) + \frac{\delta_{nk}}{\bar{\epsilon} + \delta_{nk}} x^{nk} - x^* - \frac{\delta_{nk}}{\bar{\epsilon} + \delta_{nk}} \bar{x} \right\| \leq \frac{1}{\bar{\epsilon} + \delta_{nk}} \left( \bar{\epsilon} x^{nk} + \delta_{nk} \bar{x} \right) - x^* \right\| + \frac{\delta_{nk}}{\bar{\epsilon} + \delta_{nk}} \left\| x^{nk} - \bar{x} \right\| = O(\delta_{nk}^{1/q}) + O(\delta_{nk}) = O(\delta_{nk}^{1/q}).
\]

The proof is completed. ■

5. Improvement to the basic algorithm

The purpose of this section is to enhance the effectiveness of Algorithm 2.1. We note that at each iteration, a global optimization problem (5) has to be solved. Generally, it is very difficult to find a global minimizer of problem (5). In addition, as \( k \) goes to infinity, the index set \( T_k \) becomes very large. As a result, the subproblem (4) becomes very large as well.

In this section, we first introduce a technique to avoid solving problem (5) exactly. Recall the KKT systems (2) and (4). If \((x^k, u^k)\) is a solution of subproblem (4) and \( x^k \) is feasible for equation (1), then \((x^k, u^k)\) is a KKT point of problem (1). In Algorithm 2.1, the role of the optimization problem (5) is to get an index \( t_k \) such that the next iterate \( x^{k+1} \) is closer to the feasible set \( \mathcal{F} \). If for some small constant \( \delta > 0 \), the solution \((x^k, u^k)\) of problem (4) satisfies \( g(x^k, t) \leq \delta, \forall t \in T \), then \( x^k \) is approximately feasible for SIP (1), and consequently, \((x^k, u^k)\) is an approximate KKT point of problem (1). This observation motivates the following algorithm.
Algorithm 5.1

Initial. Given a prescribed number \( \delta > 0 \) and a finite subset \( T_0 \subset T \) such that \( \{ x \in \mathbb{R}^n : g(x, t) \leq 0, \forall t \in T_0 \} \neq \emptyset \). Let \( k = 0 \).

Step 1. Solve the KKT system (4) to get \((x^k, u^k)\).

Step 2. Find a \( t_k \in T \) such that \( g(x^k, t_k) > \delta \) and let \( T_{k+1} := T_k \cup \{ t_k \} \). Set \( k := k + 1 \) and go to Step 1. Otherwise, if \( g(x^k, t) \leq \delta \) for all \( t \in T \), then stop.

The following theorem shows that Algorithm 5.1 terminates at a point \( x_k \) that satisfies subproblem (4) and \( g(x^k, t_k) \leq \delta \). If \( \delta \) is small, then such a point \((x^k, u^k)\) can be regarded as an approximate KKT point of problem (1). Compared with Algorithm 2.1, the advantage of Algorithm 5.1 lies in that it does not need to solve the optimization problem (5) exactly. Finding a \( t_k \in T \) such that \( g(x^k, t_k) > \delta \) can be done by an optimization method in finite iterations if such a \( t_k \in T \) exists. We only solve the optimization problem (5) in the final iteration.

Theorem 5.1 Suppose that the sequence \( \{x^k\} \) generated by Algorithm 5.1 is bounded. Then Algorithm 5.1 terminates at a finite number of iterations.

Proof Assume that Algorithm 5.1 does not terminate at a finite number of iterations. Let \( \{x^{n_k}\} \) be a subsequence of \( \{x^k\} \) such that \( x^{n_k} \to x^* \). We claim that for \( k = 1, 2, \ldots, g(x^*, t_{n_k}) \leq 0 \). Otherwise, there exists a positive integer \( N \) such that \( g(x^*, t_{n_k}) > 0 \). Therefore, there exists a positive integer \( \bar{N} \geq N \) large enough such that \( g(x^{n_{\bar{N}}}, t_{n_{\bar{N}}}) > 0 \). As \( \bar{N} > N \), we have \( g(x^{n_{\bar{N}}}, t_{n_{\bar{N}}}) \leq 0 \), which yields a contradiction. This contradiction shows that for \( k = 1, 2, \ldots, g(x^*, t_{n_k}) \leq 0 \). Taking limit in this inequality yields \( g(x^*, t^*) \leq 0 \). From the Step 2 of Algorithm 5.1, we know \( g(x^{n_k}, t_{n_k}) > \delta \) for all \( k = 1, 2, \ldots \). Taking limits in this inequality yields \( g(x^*, t^*) \geq \delta \). Thus, we have \( \delta \leq g(x^*, t^*) \leq 0 \). This is a contradiction. Therefore, Algorithm 5.1 terminates at a finite number of iterations.

Algorithm 5.1 terminates at an approximate KKT point of problem (1) in finitely many iterations if the condition of Theorem 5.1 holds. Moreover, this algorithm does not need to solve a global optimization problem exactly except at the final iteration. In this sense, it is computationally cheaper than Algorithm 2.1. We replace \( \delta \) by \( \delta_k \) in Algorithm 5.1, by this way, we assume that Algorithm 5.1 terminates at the point \((x^k, u^k)\). To get an exact KKT point of problem (1), we may adjust the parameter \( \delta_k \) such that it converges to zero and repeat Algorithm 5.1. In this way, we get an infinite sequence \( \{(x^k, u^k)\} \) whose limit points are KKT points of problem (1). However, the index set \( T_k \) may be very large when \( \delta_k \) is very small. Consequently, the corresponding subproblem (4) may be very large as well. In what follows, we will further modify Algorithm 5.1 such that the subproblem (4) has as less constraints as possible.

It is well known that the inactive constraints at a solution \( x^* \) do not affect the solution of the problem. In other words, if we remove all the inactive constraints at \( x^* \), then \( x^* \) is also a solution of the new problem. Taking this into account, we use an active set strategy to decrease the number of the constraints of the subproblem (4).

Suppose that \( x^* \) is a solution of equation (1). It is easy to see that when \( x^k \) is sufficiently close to \( x^* \), the active constraints in equation (4) obtained at the \( k \)th iteration also could be possibly active for problem (4) in the \((k + 1)\)th iteration. Moreover, for a sufficiently small constant \( \delta_k > 0 \), the constraints satisfying \( g(x^k, t) > \delta_k \) must also be active for the next iteration. On the basis of this observation, we may use the set \( E_k = \{ t \in T_k \mid g(x^k, t) = 0 \} \cup \{ t_k \} \), where \( t_k \) satisfies that \( g(x^k, t_k) > \delta_k \) as an estimate of the active set. We then use \( E_k \) instead of \( T_{k+1} \) and solve the KKT system (4). The modified algorithm is stated as follows.
ALGORITHM 5.2

Initial. Given a prescribed number $\delta_0 > 0$ and $\epsilon \in (0, 1)$. Given a finite subset $T_0 \subset T$ such that $\{x \in \mathbb{R}^n : g(x, t) \leq 0, \forall t \in T_0 \} \neq \emptyset$. Let $k := 0$.

Step 1. Solve the KKT system (4) to get $(x^k, u^k)$. Stop if $g(x^k, t) \leq 0$ for all $t \in T$.

Step 2. Find a $t_k \in T$ such that $g(x^k, t_k) > \delta_k$ and let $T_k := T_k \cup \{t_k\}$. Then go to Step 3. If there does not exist a $t_k \in T$ such that $g(x^k, t_k) > \delta_k$, then go to Step 4.

Step 3. Let $E_k = \{t \in T_k \mid g(x^k, t) = 0\}$ and $\delta_{k+1} = \epsilon \delta_k$ and go to Step 4.

Step 4. Find a $t_k \in T$ such that $g(x^k, t_k) > \delta_{k+1}$, and let $T_{k+1} = E_k \cup \{t_k\}$. Set $k := k + 1$, then go to Step 1.

Step 5. If there does not exist a $t_k \in T$ such that $g(x^k, t_k) > \delta_{k+1}$, then let $\delta_{k+1} := \epsilon \delta_{k+1}$, go to Step 4.

We note that, if Algorithm 5.2 does not terminate in finitely many iterations, then the sequence $\{\delta_k\}$ must go to zero. Moreover, it follows from Theorem 5.1 that the cycling between Steps 1 and 2 is finite under the conditions of Theorem 5.1. After we get $(x^k, u^k)$, we drop all unnecessary inactive constraints from $T_k$ in Step 4. The index set $T_{k+1}$ then consists of the active set of problem (5) plus one extra constraint. As such, at $(k + 1)$th iteration, the number of the constraint of the subproblem (4) is decreased. Consequently, from computational point of view, Algorithm 5.2 is much more effective than Algorithm 5.1.

Note that for each $t \in T$ and each $k$,

$$g(x^k, t) \leq \delta_k \quad \text{and} \quad \delta_k \to 0. \quad (18)$$

In the next theorem, we shall show that any limit point of the infinite sequence $\{x^k, u^k\}$ generated by Algorithm 5.2 is a KKT point of equation (1).

THEOREM 5.2 Suppose that the infinite sequence $\{x^k\}$ generated by Algorithm 5.2 is bounded. Then, there exists a subsequence of $\{x^k\}$ converging to a KKT point of problem (1).

Proof Define discrete measure $\mu^k$ with respect to $u^k$ and $T_k$ as in equation (7). Let $\{(x^{n_k}, \mu^{n_k})\}$ be a subsequence of $\{(x^k, \mu^k)\}$ such that $x^{n_k} \to x^*$ and $\mu^{n_k} \text{ weakly} \to \mu^*$. Similar to the proof of Theorem 2.1, we have

$$\nabla f(x^*) + \int \nabla_x g(x^*, t) d\mu^* = 0,$$

and

$$\int g(x^*, t) d\mu^* = 0.$$

As $\delta_k \to 0$ as $k \to \infty$, it follows from equation (18) that $g(x^*, t) \leq 0, \forall t \in T$. ■

Now we discuss some issues about implementation of Algorithms 5.1 and 5.2. At $k$th iteration of these two algorithms, the KKT system (4) can be solved by Algorithm 3.1. The other task that we need to do is finding a $t^k \in \tilde{T}$ such that $g(x^k, t^k) > \delta_k$. This task can be done by many approaches in finite iterations if such a $t_k \in T$ exists but it can not be done finitely when such a $t_k \in T$ does not exist. In what follows, we give a finite procedure to check if there is a $t^k \in T$ such that $g(x^k, t^k) > \delta_k$ approximately. Let

$$\text{dist}(\tilde{T}, T) := \max_{t \in \tilde{T}} \min_{\tilde{t} \in \tilde{T}} \|t - \tilde{t}\|,$$

where $\tilde{T}$ is a finite subset of $T$. 
Given a constant $\varepsilon > 0$ and a very small constant $0 < \varepsilon_0 < \varepsilon$, say $\varepsilon_0 = 10^{-4}$, let $\bar{T}$ satisfy $\text{dist}(\bar{T}, T) \leq \varepsilon$. We test each point in $\bar{T}$ to see whether there is a point satisfying $g(x^k, t^k) > \delta_k$. If all points fail, then we refine $\bar{T}$ such that $\bar{T}$ satisfies $\text{dist}(\bar{T}, T) \leq r\varepsilon$, where $r \in (0, 1)$ and test each point in $\bar{T}$ to find a point satisfying $g(x^k, t^k) > \delta_k$. If this fails again, then we refine $\bar{T}$ such that $\bar{T}$ satisfies $\text{dist}(\bar{T}, T) \leq r^2\varepsilon$ and so on until $\bar{T}$ satisfies $\text{dist}(\bar{T}, T) \leq \varepsilon_0$. (Only in the worst situation, $\bar{T}$ may have large number of points for finding $t_k$.) If a point $t^k$ satisfying $g(x^k, t^k) > \delta_k$ is found, then we add $t^k$ into $T^k$ and go to the next iteration.

6. Preliminary numerical experiments

In this section, we report some preliminary numerical test results. We implemented Algorithms 5.1 and 5.2 described in section 5 in MATLAB and the numerical experiments were done by using a Pentium II 450 MHz personal computer. We compared the performance of Algorithms 5.1 and 5.2 with \texttt{fseminf} that is a solver for SIP based on an implementation of the discretization SQP method in MATLAB toolbox. We test four problems.

Problem 6.1

$$\min \quad f(x) = \frac{1}{2} x^T x$$

s.t. \quad $g(x, t) = 3 + 4.5 \sin\left(\frac{4.7\pi(t - 1.23)}{8}\right) - \sum_{i=1}^{n} x_i t^{i-1} \leq 0, \quad \forall t \in [0, 200],$

where $n = 20$.

Problem 6.2

$$\min \quad f(x) = \frac{1}{2} x^T x$$

s.t. \quad $g(x, t) = 3 + 4.5 \cos\left(\frac{4.7\pi(t - 1.23)}{8}\right) - \sum_{i=1}^{n} x_i t^{i-1} \leq 0, \quad \forall t \in [0, 200],$

where $n = 20$.

Problem 6.3

$$\min \quad f(x) = \sum_{i=1}^{n} \exp(x_i)$$

s.t. \quad $g(x, t) = \frac{1}{1 + t^2} - \sum_{i=1}^{n} x_i t^{i-1} \leq 0, \quad \forall t \in [0, 200],$

where $n = 10$. 
Problem 6.4

\[
\min f(x) = \sum_{i=1}^{n} \exp(x_i)
\]

s.t. \[g(x, t) = \cos(t) - \sum_{i=1}^{n} x_i t^{i-1} \leq 0, \ \forall t \in [0, 200],\]

where \( n = 10 \).

For all test problems, we chose the vector of ones as the starting point.

Throughout the computational experiments, the parameters used in Algorithm 3.1 are \( \rho = 0.5, a = 2.1, \eta = 10^{-8} \) and \( \sigma = 10^{-4} \), and we use \( \|H(z_k)\| \leq 10^{-6} \) as the stopping criterion for Algorithm 3.1. For Algorithms 5.1 and 5.2, if \( T = [a, b] \), \( T^0 \) and \( \bar{T} \) are set to be

\[
T^0 = \left\{ \frac{t_i = a + i(b - a)}{N}: i = 0, 1, 2, \ldots, N \right\},
\]

and

\[
\bar{T} = \left\{ \frac{t_i = a + i(b - a)}{\bar{N}}: i = 0, 1, 2, \ldots, \bar{N} \right\},
\]

where \( N = 2000 \) and \( \bar{N} = 10,000 \). At each iteration of Algorithms 5.1 and 5.2, we find a \( t^k \in T \) such that \( g(x^k, t^k) > \delta_k \) in the following way. We test each point in \( \bar{T} \) to see whether there is a point satisfying \( g(x^k, t^k) > \delta_k \). If all points fail, then we let \( \bar{N} = 20,000 \) and test each point in \( \bar{T} \) to find a point satisfying \( g(x^k, t^k) > \delta_k \). If this fails again, then we let \( \bar{N} = 30,000 \), etc.

In Algorithm 5.1, we set \( \delta = 10^{-6} \) and stop our iteration when \( \max_{t \in \bar{T}} g(x^k, t) \leq 10^{-6} \), where \( \bar{T} \) is defined previously with \( \bar{N} = 200,000 \). In Algorithm 5.2, we set \( \delta_0 = 0.01 \) and \( \epsilon = 0.01 \). We stop the iteration of Algorithm 5.2 when \( \delta_k \leq 10^{-6} \) and \( \max_{t \in \bar{T}} g(x^k, t) \leq 10^{-6} \), where \( \bar{T} \) is defined previously with \( \bar{N} = 200,000 \).

The test results are summarized in Table 1, where \( n_{\text{KKT}} \) denotes the number of the KKT system (4) solved, cpu the CPU time (in seconds) for solving each problem, \( f(x^k) \) and \( g(x^k, t^k) \) the values of the objective function and the function \( \max_{t \in \bar{T}} g(x^k, t) \), where \( \bar{T} \) is defined previously with \( \bar{N} = 200,000 \), at the final iteration, respectively.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>( n_{\text{KKT}} )</th>
<th>cpu (s)</th>
<th>( f(x^k) )</th>
<th>( g(x^k, t^k) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.1</td>
<td>5</td>
<td>13.47</td>
<td>3.57e−02</td>
<td>1.28e−16</td>
</tr>
<tr>
<td>1</td>
<td>5.2</td>
<td>5</td>
<td>11.28</td>
<td>3.57e−02</td>
<td>7.63e−17</td>
</tr>
<tr>
<td>1</td>
<td>fseminf</td>
<td>29.65</td>
<td>3.57e−02</td>
<td>−3.85e−06</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5.1</td>
<td>5</td>
<td>13.54</td>
<td>7.76e+00</td>
<td>5.05e−17</td>
</tr>
<tr>
<td>2</td>
<td>5.2</td>
<td>5</td>
<td>10.46</td>
<td>7.76e+00</td>
<td>1.74e−16</td>
</tr>
<tr>
<td>2</td>
<td>fseminf</td>
<td>31.48</td>
<td>7.76e+00</td>
<td>−5.89e−07</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5.1</td>
<td>7</td>
<td>12.03</td>
<td>1.13e+01</td>
<td>3.34e−13</td>
</tr>
<tr>
<td>3</td>
<td>5.2</td>
<td>8</td>
<td>8.91</td>
<td>1.13e+01</td>
<td>8.82e−14</td>
</tr>
<tr>
<td>3</td>
<td>fseminf</td>
<td>13.93</td>
<td>1.13e+01</td>
<td>3.10e−07</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5.1</td>
<td>3</td>
<td>5.01</td>
<td>1.10e+01</td>
<td>8.80e−13</td>
</tr>
<tr>
<td>4</td>
<td>5.2</td>
<td>3</td>
<td>4.48</td>
<td>1.10e+01</td>
<td>8.80e−13</td>
</tr>
<tr>
<td>4</td>
<td>fseminf</td>
<td>6.39</td>
<td>1.10e+01</td>
<td>7.74e−07</td>
<td></td>
</tr>
</tbody>
</table>
The results reported in table 1 show that Algorithms 5.1 and 5.2 perform very well. These algorithms are able to obtain KKT points of the test problems. From the cpu column of the table, we can see that Algorithm 5.2 use less CPU time than Algorithm 5.1 and fseminf for the four test problems. This shows that the active set technique used in Algorithm 5.2 is effective.

7. Conclusions

In this paper, we developed an iterative method for solving the KKT system of semi-infinite programming. At each iteration, we solved the KKT system of a nonlinear programming problem with finite constraints by a semismooth Newton method. Compared with the methods proposed in [11,19], the method presented in the paper can guarantee any limit point of the generated sequence to be a KKT point of the SIP problem. The numerical tests reported in the paper are very preliminary. Further experience with testing and with actual applications will be necessary.

Acknowledgements

The work of D.-H. Li was partially supported by the National Natural Science Foundation of China via grant 10171030. L. Qi was supported by the Research Grant Council of Hong Kong. G. Zhou was supported by Australian Research Council.

References


