A New Exchange Method for Convex Semi-Infinite Programming — Source link

Liping Zhang, Soon-Yi Wu, Marco A. López

Institutions: Tsinghua University, National Cheng Kung University

Published on: 01 Aug 2010 - Siam Journal on Optimization (Society for Industrial and Applied Mathematics)

Topics: Algorithmics, Semi-infinite programming, Solver, Filter design and Lagrange multiplier

SIAM J. OPTIM. © 2010 Society for Industrial and Applied Mathematics
Vol. 20, No. 6, pp. 2959–2977

A NEW EXCHANGE METHOD FOR CONVEX SEMI-INFINITE PROGRAMMING[*]

LIPING ZHANG[†], SOON-YI WU[‡], AND MARCO A. LÓPEZ[§]

Abstract. In this paper we propose a new exchange method for solving convex semi-infinite programming (CSIP) problems. We introduce a new dropping-rule in the proposed exchange algorithm, which only keeps those active constraints with positive Lagrange multipliers. Moreover, we exploit the idea of looking for η-infeasible indices of the lower level problem as the adding-rule in our algorithm. Hence the algorithm does not require to solve a maximization problem over the index set at each iteration; it only needs to find some points such that a certain computationally-easy criterion is satisfied. Under some reasonable conditions, the new adding-dropping rule guarantees that our algorithm provides an approximate optimal solution for the CSIP problem in a finite number of iterations. In the numerical experiments, we apply the proposed algorithm to solve some test problems from the literature, including some medium-sized problems from complex approximation theory and FIR filter design. We compare our algorithm with an existing central cutting plane algorithm and with the semi-infinite solver *fseminf* in MATLAB toolbox, and we find that our algorithm solves the CSIP problem much faster. For the FIR filter design problem, we show that our algorithm solves the problem better than some algorithms that were technically established for the problem.

Key words. convex semi-infinite programming, exchange methods, adding-dropping rules, complex approximation theory, filter design

AMS subject classifications. 90C30, 49D39, 65K05, 65D15

DOI. 10.1137/090767133

1. Introduction.

A semi-infinite programming (SIP) problem is an optimization problem in finitely many variables on a feasible set described by infinitely many constraints, and it may be written as follows:

$$(\mathrm{P}) \qquad \begin{array}{ll} \min & f(x) \\ \text{s.t.} & g(x,s) \le 0 \ \ \forall s \in \Omega, \end{array}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \times \Omega \to \mathbb{R}$ are continuous functions, and Ω is a given nonempty compact set in $\mathbb{R}^p$ (or in $\mathbb{C}^p$). For further reading see, e.g., [8, 10, 11, 13, 16, 21, 27]. We shall assume that problem (P) has a nonempty feasible set and a finite optimal value. It should be pointed out that while problem (P) defined here has only one (infinite) constraint g, the developed theory in this paper can be extended to problems with several constraints (see problem (4.2) in section 4).

Problem (P) has important applications in approximation theory, optimal control, and in numerous engineering problems such as optimum filter design in signal processing, resource allocation in decentralized systems, and decision making under competition; see, e.g., [7, 10, 12, 14, 23, 25, 26, 32].

In this paper, we focus on the convex SIP problem (P) satisfying the following set of conditions.

[*]Received by the editors August 4, 2009; accepted for publication (in revised form) August 5, 2010; published electronically October 12, 2010.
http://www.siam.org/journals/siopt/20-6/76713.html

[†]Department of Mathematical Sciences, Tsinghua University, Beijing 100084, China (lzhang@math.tsinghua.edu.cn). This author's work was supported by NSFC grant 10871113.

[‡]Corresponding author. Department of Mathematics, National Cheng Kung University, Tainan, Taiwan, and National Center for Theoretical Sciences, Taiwan (soonyi@mail.ncku.edu.tw).

[§]Department of Statistics and Operations Research, Alicante University, 03080 Alicante, Spain (marco.antonio@ua.es).

Assumption A.

(i) f is convex and continuously differentiable on $\mathbb{R}^n$;

(ii) $g(\cdot, s)$ is convex for all $s \in \Omega$, and $\nabla_x g(x, s)$ exists and is continuous on $\mathbb{R}^n \times \Omega$;

(iii) Slater constraint qualification (SCQ) holds; i.e., there exists $\hat{x} \in \mathbb{R}^n$ such that $g(\hat{x}, s) < 0$ for all $s \in \Omega$.

(iv) f is level bounded on the feasible set of (P); i.e., for every scalar λ, the set $\{x \in \mathbb{R}^n | g(x, s) \leq 0, \forall s \in \Omega; \;\; f(x) \leq \lambda\}$ is bounded when nonempty.

The main difficulty for solving a SIP problem is precisely that it has infinitely many constraints. Despite this serious drawback, many algorithms have been proposed for solving problem (P); see, for example, [5, 10, 13, 14, 15, 17, 19, 24, 27, 28, 29, 30, 31, 32, 33]. Among them, discretization and reduction based methods are two common approaches. Discretization methods have the advantage to internally work only with finite subsets of Ω (see, e.g., [14, 29, 30]). However, they are computationally costly, and the cost per iteration increases dramatically as the size of these finite subsets grows. Globally convergent reduction based methods (see, e.g., [13, 27]), on the other hand, require strong assumptions and are often conceptual methods which can be implemented merely in a rather simplified form. Beyond discretization and reduction based methods, one of the most important families of methods is the so-called exchange method family. See, for instance, [10, 13, 27], for a description of these methods and their history starting from the first algorithm of Remes [4]. Laurent and Carasso [20] proposed in 1978 a general exchange algorithm for the minimization of a certain type of convex function (supremum of affine functions) with infinitely many constraints. For convex quadratic SIP with linear constraints an exchange method was proposed in [15], but this algorithm performs a finite prefixed number of iterations. In [32] a kind of cutting plane algorithm for solving a convex SIP with a strictly convex quadratic objective function is given, which can be considered an exchange method as long as nonbinding constraints are deleted. Most of the computational effort in this algorithm is devoted to approximatively solve the so-called auxiliary problem (finding the most violated constraint at each iteration). The central cutting plane (CCP) algorithm [17] is the most important exchange method for convex SIP in the literature. Betrò [3] improved it for linear SIP and obtained a faster convergence rate.

Here we propose an exchange method for the convex SIP problem (P) whose main feature is that only those active constraints with positive Lagrange multipliers are kept, and no global optimization needs to be carried out at each iteration to detect the (almost) most violated constraint. Our algorithm has to find indexes whose associated constraints are only slightly violated. This adding-dropping rule ensures that our algorithm is promising in saving computational time. Moreover, we prove that, under certain conditions, the algorithm provides an approximate optimal solution of problem (P) in a finite number of iterations. We also give some numerical results to see the performance of the algorithm. Especially, we have compared our algorithm with the CCP algorithm and with the SIP solver *fseminf*, and we have observed that our algorithm solves convex SIP much faster. For the FIR filter design problem, we find that our algorithm has better performance than the algorithm in [26] that was technically established for the FIR filter design problem. Our paper is motivated by [19], but here we provide additional results about finite termination and convergence for convex SIP problems. Moreover, our analysis technique is quite different and mainly relies on the convexity. In addition to the analysis technique, our algorithm has two advantages when compared with [19]. First, we introduce the new adding-rule in our algorithm. Second, the assumptions that are required for the finite termination analysis are weakened. We do not need here the assumptions that each subproblem has unique optimal solution and its corresponding Jacobian matrix of constraints has a nonsingular submatrix. These assumptions were used in [19].

This paper is organized as follows. In section 2, we propose an exchange method for solving problems (P). In section 3, we establish the convergence property of the proposed algorithm. In section 4, we give some numerical results to see the performance of the algorithm. We finish with some concluding remarks in section 5.

2. Algorithm description.

In this section we present our exchange method for solving the convex SIP problem (P).

The algorithm solves a finitely constrained convex programming problem at each iteration. Associated with each finite set $E = \{s_1, \ldots, s_m\} \subset \Omega$, we define the finitely constrained convex programming problem

$$P(E): \quad \begin{array}{ll} \min & f(x) \\ \text{s.t.} & g(x, s_i) \le 0, \quad i = 1, \ldots, m. \end{array}$$

It is well known that the following theorem holds (see, e.g., [22]).

THEOREM 2.1. *Let $x^* \in \mathbb{R}^n$ be a feasible solution of $P(E)$ and suppose that Assumption A holds. Then, x^* is optimal if and only if there exist multipliers $\nu^* \in \mathbb{R}^m$ such that (x^*, ν^*) satisfies the following Karush-Kuhn-Tucker (KKT) conditions:*

$$\begin{aligned} &\nabla f(x) + \sum_{i=1}^{m} \nu(s_i) \nabla_x g(x, s_i) = 0, \\ &\nu(s_i) \ge 0, \quad g(x, s_i) \le 0, \quad g(x, s_i)\nu(s_i) = 0, \quad i = 1, \ldots, m. \end{aligned} \tag{2.1}$$

According to [1, Lemma 3.1], there exists a finite subset $\Omega_0 \subset \Omega$ such that f is level bounded on the feasible set of $P(\Omega_0)$. Remark 3.1 in [1] shows some particular cases where the set Ω_0 is easily obtainable.

We present now the new algorithm.

ALGORITHM 2.1.

Step 0. Choose a finite set $E_0 = \{s_1^0, \ldots, s_{m_0}^0\}$ such that $\Omega_0 \subset E_0$, and a small number $\eta > 0$. Solve $P(E_0)$ to obtain an optimum x^0. Set $k := 0$.

Step 1. Find an $s_{new}^k \in \Omega$ such that

$$g(x^k, s_{new}^k) > \eta. \tag{2.2}$$

If such an s_{new}^k does not exist, then STOP. Otherwise, let

$$\bar{E}_{k+1} := E_k \cup \{s_{new}^k\}.$$

Step 2. Solve $P(\bar{E}_{k+1})$ to obtain an optimum x^{k+1} and a set of associated multipliers $\{\nu_{k+1}(s) \mid s \in \bar{E}_{k+1}\}$.

Step 3. Let

$$E_{k+1} := \{s \in \bar{E}_{k+1} \mid s \in \Omega_0 \ \text{ or } \ \nu_{k+1}(s) > 0\}.$$

Set $k := k + 1$ and go to Step 1.

Note that at the kth iteration of the algorithm, we solve a subproblem $P(\bar{E}_k)$ and update E_k so that $E_{k+1} = \{s \in E_k \cup \{s_{new}^k\} | \ s \in \Omega_0 \ \text{ or } \ \nu_{k+1}(s) > 0\}$ with

s^k_{new} satisfying the criterion $g(x^k, s^k_{new}) > \eta$. Therefore we need not find a global ε-maximum, denoted by the following notation

$$s^k \in \varepsilon\text{-}arg\max\{g(x^k, s) \mid s \in \Omega\}, \tag{2.3}$$

which is required in a vast majority of solution methods (see, e.g., [14, 27] and references therein), and we only keep those active constraints with positive multipliers. Obviously, our adding-dropping rule seems more effective than (2.3) in saving computational time because the global ε-maximization problem (2.3) is not solved at each iteration, and the algorithm tends to keep the cardinality of E_k small by dropping not only all inactive but also some (possible) active constraints.

Next there are some detailed remarks on the main features of Algorithm 2.1:

- Steps 1 and 3 constitute the main difference with respect to exchange methods in the literature. The idea of looking for an index satisfying (2.2) in Step 1 and the dropping-rule (Step 3) are the main features of Algorithm 2.1.
- In Step 1, it is also possible to choose multiple elements satisfying (2.2). Although we merely deal with the single-point exchange scheme in the subsequent analysis, the obtained results are also applicable for multiple exchange.
- Observe also that thanks to the fact $\Omega_0 \subset \bar{E}_{k+1}$, $\forall k$, the existence of a minimum x^{k+1} is guaranteed.
- In Step 2, $P(\bar{E}_{k+1})$ can be solved by using any preexisting method for convex programming. Here we note from (2.1) that

$$\nabla f(x^{k+1}) + \sum_{s\in E_{k+1}} \nu_{k+1}(s)\nabla_x g(x^{k+1}, s) = 0, \tag{2.4}$$

$$\sum_{s\in E_{k+1}} \nu_{k+1}(s) g(x^{k+1}, s) = 0, \tag{2.5}$$

  and by Theorem 2.1, x^{k+1} also solves $P(E_{k+1})$.
- In Step 3, not only all inactive constraints of $P(\bar{E}_{k+1})$ at the optimum x^{k+1} are removed, but also some active constraints are possibly removed. Aside from the constraints associated with the indices in Ω_0, we only keep those active constraints with positive multipliers. This is perhaps the most crucial reason for saving computational time.
- Let f^* denote the optimal value of (P), then $f(x^k) \le f^*$. If x^k is feasible, then it will be an optimal solution of (P). However, x^k is in general unfeasible.
- Let $\mathcal{F}$ denote the feasible region of (P), and define for $\eta > 0$

$$\mathcal{F}_\eta := \{x \in \mathbb{R}^n \mid g(x, s) \le \eta \ \forall s \in \Omega\}. \tag{2.6}$$

  Obviously $\mathcal{F}_\eta \supset \mathcal{F}$. Observe that if Algorithm 2.1 stops at iteration k, then the final iterate x^k is in $\mathcal{F}_\eta$.

Let $v(P(E_k))$ denote the optimal value of $P(E_k)$. From Step 1 of Algorithm 2.1 we have the following proposition.

PROPOSITION 2.1. *The sequence of optimal values* $\{v(P(E_k))\}$ *of* $\{P(E_k)\}$ *is nondecreasing; i.e.,*

$$v(P(E_{k+1})) \ge v(P(E_k)) \ \text{ holds for } \ k = 1, 2, \ldots.$$

Proof. From Step 1, $E_k \subset \bar{E}_{k+1}$, which entails that the feasible region of $P(\bar{E}_{k+1})$ is contained in that of $P(E_k)$. Hence, from the above remarks one gets

$$v(P(E_{k+1})) = v(P(\bar{E}_{k+1})) \geq v(P(E_k)), \quad k = 1, 2, \dots . \qquad \square$$

Observe that the statement in Proposition 2.1 can be equivalently established as

$$f(x^{k+1}) \geq f(x^k) \text{ holds for } k = 1, 2, \dots .$$

PROPOSITION 2.2. *If there is an integer* $\bar{k}$ *such that*

$$f(x^{\bar{k}+1}) > f(x^{\bar{k}}),$$

then $s^{\bar{k}}_{new} \in E_{\bar{k}+1}$.

Proof. Suppose that $s^{\bar{k}}_{new} \notin E_{\bar{k}+1}$, then $\nu_{\bar{k}+1}(s^{\bar{k}}_{new}) = 0$. Now, from

$$\begin{aligned} &\nabla f(x^{\bar{k}+1}) + \sum_{s \in \bar{E}_{\bar{k}+1}} \nu_{\bar{k}+1}(s) \nabla_x g(x^{\bar{k}+1}, s) \\ &= \nabla f(x^{\bar{k}+1}) + \sum_{s \in E_{\bar{k}}} \nu_{\bar{k}+1}(s) \nabla_x g(x^{\bar{k}+1}, s) = 0, \\ &\sum_{s \in \bar{E}_{\bar{k}+1}} \nu_{\bar{k}+1}(s) g(x^{\bar{k}+1}, s) = \sum_{s \in E_{\bar{k}}} \nu_{\bar{k}+1}(s) g(x^{\bar{k}+1}, s) = 0, \end{aligned}$$

we observe, by Theorem 2.1, that $x^{\bar{k}+1}$ is also optimal for $P(E_{\bar{k}})$, which implies $f(x^{\bar{k}+1}) = f(x^{\bar{k}})$, and this contradicts the assumption. $\square$

We finish section 2 with the following easy proposition, which is crucial in what follows.

PROPOSITION 2.3. *If* $\{x^k\}$ *is the sequence generated by Algorithm* 2.1*, we have the following chain of relations:*

$$\begin{aligned} &f(x^{k+1}) - f(x^k) \geq \nabla f(x^k)^T (x^{k+1} - x^k) && (2.7)\\ &= -\sum_{s \in E_k} \nu_k(s) \nabla_x g(x^k, s)^T (x^{k+1} - x^k) && (2.8)\\ &\geq \sum_{s \in E_k} \nu_k(s) \{g(x^k, s) - g(x^{k+1}, s)\} && (2.9)\\ &= -\sum_{s \in E_k} \nu_k(s) g(x^{k+1}, s) && (2.10)\\ &\geq 0. && (2.11) \end{aligned}$$

Proof. Inequalities (2.7) and (2.9) are consequences of the convexity assumptions $A(i)$ and $A(ii)$. Equality (2.8) is nothing else but (2.4), and (2.10) is a consequence of (2.5). Inequality (2.11) comes from the fact that x^{k+1} satisfies all the inequalities in $\bar{E}_{k+1}$ and that $E_k \subset \bar{E}_{k+1}$. $\square$

Clearly, Proposition 2.3 also allows the recovery of the statement in Proposition 2.1.

3. Convergence analysis. In this section we show that, under reasonable conditions, Algorithm 2.1 terminates in a finite number of iterations, producing an approximate optimal solution for (P). Furthermore, we shall prove that this output tends to an optimal solution if the criterion value η tends to zero.

The following two lemmas are needed for the subsequent analysis.

LEMMA 3.1. *For each fixed $k \geq 1$, if Algorithm* 2.1 *does not stop at this iterate, the strict inequality $f(x^{k+1}) > f(x^k)$ holds if any one of the following conditions is satisfied:*

(i) *The subproblem $P(E_k)$ has a unique optimal solution. This happens, for instance, if f is strictly convex.*

(ii) *$g(\cdot, s)$ is strictly convex for every $s \in \Omega$ and $E_1 \backslash \Omega_0 \neq \emptyset$.*

Proof. Assume that condition (i) holds, and let $\mathcal{F}^k$ and $\bar{\mathcal{F}}^{k+1}$ be the feasible regions of $P(E_k)$ and $P(\bar{E}_{k+1})$, respectively. It is obvious that $\mathcal{F}^k \supseteq \bar{\mathcal{F}}^{k+1}$. Reasoning by contradiction, $f(x^{k+1}) = f(x^k)$ entails that x^{k+1} also solves $P(E_k)$. Therefore, condition (i) yields $x^{k+1} = x^k$, and this leads us to the contradiction

$$0 \geq g(x^{k+1}, s^k_{new}) = g(x^k, s^k_{new}) > \eta > 0. \tag{3.1}$$

If f is strictly convex, since it is level bounded on the feasible set of $P(E_k)$ (remember that $\Omega_0 \subset E_k$), this problem will have a unique optimal solution. In this case, we can also get the same conclusion by a straightforward argument: (2.7) is a strict inequality, and (2.11) implies the contradiction

$$0 = f(x^{k+1}) - f(x^k) > \nabla f(x^k)^T (x^{k+1} - x^k) \geq 0.$$

Assume now that condition (ii) holds. Then, for $k = 1$, (2.9) becomes a strict inequality, and hence

$$f(x^2) - f(x^1) > \sum_{s \in E_1} \nu_1(s) \left\{ g(x^1, s) - g(x^2, s) \right\} \geq 0. \tag{3.2}$$

We claim that

$$E_k \backslash \Omega_0 = \emptyset$$

will never happen for $k \geq 2$. Suppose not, then there exists $k_0 \geq 2$ such that $E_{k_0} \backslash \Omega_0 = \emptyset$. Hence, the point x^{k_0} satisfies

$$\nabla f(x^{k_0}) + \sum_{s \in \Omega_0} \nu_{k_0}(s) \nabla_x g(x^{k_0}, s) = 0,$$
$$\sum_{s \in \Omega_0} \nu_{k_0}(s) g(x^{k_0}, s) = 0,$$

and therefore, the point x^{k_0} is optimal for $P(\Omega_0)$ since the KKT conditions are held at this point.

Since $\Omega_0 \subset E_j,\ j = 0, 1, ...k_0 - 1$, the points x^j, $j = 0, 1, ...k_0 - 1$, are also feasible for $P(\Omega_0)$. From (3.2) and Proposition 2.3,

$$f(x^0) \leq f(x^1) < f(x^2) \leq \cdots \leq f(x^{k_0 - 1}) \leq f(x^{k_0}),$$

which produces a contradiction. Therefore, $E_k \backslash \Omega_0 \neq \emptyset$ for $k \geq 2$, (2.9) becomes a strict inequality and

$$f(x^{k+1}) - f(x^k) > \sum_{s \in E_k} \upsilon_k(s) \{ g(x^k, s) - g(x^{k+1}, s) \} \geq 0.$$

Then, we complete the proof. ☐

LEMMA 3.2. *Suppose that Algorithm* 2.1 *does not finitely terminate when it is applied to problem (P), generating the sequence* $\{x^k\}$. *Then we have the following statements:*

(a) *There exists an infinite subset* $\mathbb{K} \subset \{1, 2, ...\}$ *such that the subsequence* $\{x^k\}_{k\in\mathbb{K}}$ *is convergent.*

(b) *The sequence* $\{\sum_{s\in E_k} v_k(s)\}_{k\in\mathbb{K}}$ *is bounded away from zero, i.e., bounded from below by a positive scalar, if the sequence* $\{f(x^k)\}$ *is strictly increasing.*

Proof. (a) By proposition 2.1, we have

$$f(x^0) \le \cdots \le f(x^k) \le f(x^{k+1}) \le \cdots \le f^*.$$

The above inequalities entail

$$\{x^k\} \subset \{x \mid g(x,s) \le 0\ \forall s \in \Omega_0;\ f(x) \le f^*\}$$

which is a compact set, and $\{x^k\}$ will have an accumulation point, i.e., there will exist a subsequence $\{x^k\}_{k\in\mathbb{K}}$ converging to a certain point $\bar{x}$.

(b) Reasoning by contradiction, if $\{\sum_{s\in E_k} v_k(s)\}_{k\in\mathbb{K}}$ is not bounded away from zero, there will exist an infinite subset $\mathbb{K}' \subset \mathbb{K}$ such that

$$\lim_{k\to\infty,\ k\in\mathbb{K}'} \sum_{s\in E_k} v_k(s) = 0.$$

By Assumption A(ii), since $\{x^k\}$ is bounded, there exists a constant $M > 0$ such that

$$\max_{k\in\mathbb{K}', s\in\Omega} \left\|\nabla_x g(x^k, s)\right\| \le M,$$

and from (2.4) we have for any $k \in \mathbb{K}'$,

$$\left\|\nabla f(x^k)\right\| \le \left(\sum_{s\in E_k} v_k(s)\right) \max_{k\in\mathbb{K}', s\in\Omega} \left\|\nabla_x g(x^k, s)\right\| \le M\left(\sum_{s\in E_k} v_k(s)\right). \tag{3.3}$$

Taking limits in (3.3) for $k \to \infty, k \in \mathbb{K}'$, we obtain

$$\begin{aligned}\|\nabla f(\bar{x})\| &= \lim_{k\to\infty,\ k\in\mathbb{K}'} \left\|\nabla f(x^k)\right\| \\ &\le \lim_{k\to\infty,\ k\in\mathbb{K}'} M\left(\sum_{s\in E_k} v_k(s)\right) \\ &= 0;\end{aligned}$$

i.e., $\nabla f(\bar{x}) = 0$, and hence $\bar{x}$ is a global minimum of f in $\mathbb{R}^n$. Since the sequence $\{f(x^k)\}$ is strictly increasing,

$$f(x^1) < f(x^2) < \cdots < f(x^k) \le f(\bar{x}).$$

This contradicts the fact that $\bar{x}$ is a global minimum of f in $\mathbb{R}^n$. ∎

THEOREM 3.1. *If any one of the following conditions is satisfied:*

(i) f *is strictly convex,*

(ii) $g(\cdot, s)$ *is strictly convex for every* $s \in \Omega$ *and* $E_1\backslash\Omega_0 \neq \emptyset$,

then Algorithm 2.1 *terminates in a finite number of iterations.*

Proof. If either conditions (i) or (ii) holds, then from Lemma 3.1 and Proposition 2.2 we have $f(x^{k+1}) > f(x^k)$ and $s^k_{new} \in E_{k+1}$ for every k.

Reasoning by contradiction, suppose that Algorithm 2.1 does not terminate in a finite number of iterations, generating a sequence $\{x^k\}$ such that

$$f(x^1) < \cdots < f(x^k) < f(x^{k+1}) < \cdots < f^*.$$

Hence

$$\lim_{k\to\infty} \{f(x^{k+1}) - f(x^k)\} = 0. \tag{3.4}$$

Since $\{x^k\}$ is bounded, and then $\{x^{k+1} - x^k\}$ is also bounded, and the sequence $\{s^k_{new}\}$ is contained in the compact Ω, there will exist $\bar{x} \in \mathbb{R}^n$, $\bar{d} \in \mathbb{R}^n$, and $\bar{s} \in \Omega$, and an infinite set $\mathbb{K} \subset \{1, 2, \ldots\}$ such that

$$\lim_{k\to\infty, k\in\mathbb{K}} (x^k, x^{k+1} - x^k, s^k_{new}) = (\bar{x}, \bar{d}, \bar{s}). \tag{3.5}$$

Let condition (i) hold. From (2.7) and (2.11),

$$f(x^{k+1}) - f(x^k) \geq \nabla f(x^k)^T (x^{k+1} - x^k) \geq 0,$$

and then taking limits for $k \to \infty, k \in \mathbb{K}$, we get

$$\begin{aligned} 0 &= \lim_{k\to\infty, k\in\mathbb{K}} \{f(x^{k+1}) - f(x^k)\} \\ &= f(\bar{x} + \bar{d}) - f(\bar{x}) \\ &\geq \nabla f(\bar{x})^T \bar{d} \\ &\geq 0. \end{aligned}$$

This yields

$$f(\bar{x} + \bar{d}) = f(\bar{x}) + \nabla f(\bar{x})^T \bar{d},$$

and condition (i) would entail $\bar{d} = 0$.

Consider now the case that condition (ii) holds. Take

$$s^k \in \operatorname{argmin}_{s\in\Omega} \{g(x^{k+1}, s) - g(x^k, s) - \nabla_x g(x^k, s)^T (x^{k+1} - x^k)\}.$$

Since $g(x^{k+1}, s) \leq 0$ holds for any $s \in E_k$, Assumption A(ii) and (2.8) imply

$$\begin{aligned} f(x^{k+1}) - f(x^k) &\geq \sum_{s\in E_k} v_k(s) \{g(x^{k+1}, s) - g(x^k, s) - \nabla_x g(x^k, s)^T (x^{k+1} - x^k)\} \\ &\geq \left(\sum_{s\in E_k} v_k(s)\right) \{g(x^{k+1}, s^k) - g(x^k, s^k) - \nabla_x g(x^k, s^k)^T (x^{k+1} - x^k)\} \\ &\geq 0. \end{aligned} \tag{3.6}$$

Since $\{s^k\} \subset \Omega$, there will exist $\widehat{s} \in \Omega$ and an infinite set $\mathbb{K}' \subset \mathbb{K}$ such that

$$\lim_{k\to\infty, k\in\mathbb{K}'} s^k = \widehat{s}.$$

Moreover, by Lemma 3.1 and condition (ii), the sequence $\{f(x^k)\}$ is strictly increasing, and hence Lemma 3.2 establishes that the sequence $\{\sum_{s\in E_k} v_k(s)\}_{k\in\mathbb{K}'}$ is bounded away from zero, and taking limits in (3.6) for $k\to\infty, k\in\mathbb{K}'$, one derives

$$\begin{aligned}0 &= \lim_{k\to\infty,k\in\mathbb{K}'}\left\{g(x^{k+1},s^k)-g(x^k,s^k)-\nabla_x g(x^k,s^k)^T(x^{k+1}-x^k)\right\}\\ &= g(\bar{x}+\bar{d},\widehat{s})-g(\bar{x},\widehat{s})-\nabla_x g(\bar{x},\widehat{s})^T\bar{d}.\end{aligned}$$

Once again we have to conclude that $\bar{d}=0$ due to condition (ii).

Under any one of conditions (i) and (ii) we concluded that $\bar{d}=0$ and so,

$$\lim_{k\to\infty,k\in\mathbb{K}} x^{k+1} = \lim_{k\to\infty,k\in\mathbb{K}}\left\{x^k+(x^{k+1}-x^k)\right\} = \bar{x}+\bar{d}=\bar{x}.$$

Since $g(x^{k+1},s^k_{new})\le 0\ \forall k$,

$$\lim_{k\to\infty,k\in\mathbb{K}} g(x^{k+1},s^k_{new}) = g(\bar{x},\bar{s})\le 0,$$

but at the same time $g(x^k,s^k_{new})>\eta\ \forall k$, and

$$\lim_{k\to\infty,k\in\mathbb{K}} g(x^k,s^k_{new}) = g(\bar{x},\bar{s})\ge\eta>0.$$

This is a contradiction. □

Up to now, we have shown the finite termination property of Algorithm 2.1 under certain assumptions. Actually, if there is no s^k_{new} such that $g(x^k,s^k_{new})\ge\eta>0$, then we could have finite termination property for Algorithm 2.1. And if during the unsuccessful search of s^k_{new}, we found a certain point $\bar{s}\in\Omega$ such that $0<g(x^k,\bar{s})<\eta$ then we could update η to $g(x^k,\bar{s})$, and the algorithm could continue with the new η to obtain the better solution. The following theorem guarantees that if $\eta>0$ tends to zero, then the solution of the last subproblem obtained by Algorithm 2.1 also tends to the optimal solution of (P); i.e., Algorithm 2.1 can yield an η-approximate optimal solution for (P) in a finite number of iterations.

THEOREM 3.2. *Given $\eta>0$, suppose that Algorithm 2.1 terminates in a finite number of iterations yielding as last iterate the point x^*_η. Then, the following statements hold:*

(a) *Every accumulation point of x^*_η as $\eta\to 0$ is an optimal solution of* (P).

(b) *There exists a positive constant M_1 such that*

$$0\le f^*-f(x^*_\eta)\le M_1\operatorname{dist}(\mathcal{F}\cap\operatorname{lev}_{\le\alpha}f,\mathcal{F}_\eta\cap\operatorname{lev}_{\le\alpha}f),$$

where

$$\operatorname{lev}_{\le\alpha}f=\{x\in\mathbb{R}^n:\ f(x)\le\alpha\},$$

with $\alpha\ge f^*$ and, if A and B are two compact sets such that $A\subset B$,

$$\operatorname{dist}(A,B)=\max_{x\in B}\min_{y\in A}\|x-y\|.$$

(c) *There exists a positive constant M_2 such that*

$$0\le f^*-f(x^*_\eta)\le\frac{\eta}{\eta+\rho}M_2\left\|x^*_\eta-\widehat{x}\right\|,$$

where $\widehat{x}$ is a point satisfying $g(\widehat{x}, s) < 0 \ \forall s \in \Omega$ (see Assumption A(iii)), *and*

$$\rho := -\max_{s\in\Omega} g(\widehat{x}, s).$$

Proof. (a) Since $x_\eta^* \in \{x \mid g(x,s) \leq 0 \ \forall s \in \Omega_0; \ f(x) \leq f^*\}$, which is compact, there will exist at least an accumulation point, namely x^*, of x_η^* as $\eta \to 0$.

Algorithm 2.1 stopped in x_η^* because

$$g(x_\eta^*, s) \leq \eta \quad \forall s \in \Omega. \tag{3.7}$$

Letting $\eta \to 0$ in (3.7), and using the continuity of g, we conclude that

$$g(x^*, s) \leq 0 \quad \forall s \in \Omega,$$

and x^* is feasible for (P); i.e. $x^* \in \mathcal{F}$.

Clearly, x_η^* is an optimal solution of the problem

$$\min\{f(x) \ \text{ s.t. } x \in \mathcal{F}_\eta\}.$$

Hence,

$$f(x_\eta^*) \leq f(x) \quad \forall x \in \mathcal{F}_\eta \supset \mathcal{F}. \tag{3.8}$$

Letting $\eta \to 0$ in (3.8), we get

$$f(x^*) \leq f(x) \quad \forall x \in \mathcal{F}.$$

That is, x^* is an optimal solution of (P), and then (a) is proved.

(b) Let $\widehat{x}_\eta^*$ the orthogonal projection of x_η^* onto $\mathcal{F}\cap\text{lev}_{\leq\alpha} f$. We have $f(\widehat{x}_\eta^*) \geq f^*$ and we write

$$\begin{aligned}
0 &\leq f^* - f(x_\eta^*) \\
&= f^* - f(\widehat{x}_\eta^*) + f(\widehat{x}_\eta^*) - f(x_\eta^*) \\
&\leq f(\widehat{x}_\eta^*) - f(x_\eta^*) \\
&= \nabla f(\widetilde{x}_\eta^*)^T(\widehat{x}_\eta^* - x_\eta^*) \\
&\leq \left\|\nabla f(\widetilde{x}_\eta^*)\right\| \left\|\widehat{x}_\eta^* - x_\eta^*\right\| \\
&\leq \left\|\nabla f(\widetilde{x}_\eta^*)\right\| \text{dist}(\mathcal{F}\cap \text{lev}_{\leq\alpha} f, \mathcal{F}_\eta \cap \text{lev}_{\leq\alpha} f),
\end{aligned}$$

where $\widetilde{x}_\eta^*$ is a point of the segment determined by $\widehat{x}_\eta^*$ and x_η^*, and so, $\widetilde{x}_\eta^* \in \mathcal{F}_\eta\cap\text{lev}_{\leq\alpha} f$, which is a compact set (all the nonempty sets $\mathcal{F}_\eta\cap\text{lev}_{\leq\alpha} f$ are compact). Then, we can take

$$M_1 := \max\left\{\|\nabla f(x)\| : \ x \in \mathcal{F}_\eta\cap\text{lev}_{\leq\alpha} f\right\},$$

which completes the proof of (b).

(c) Observe that

$$\begin{aligned}
g\left(\frac{\rho}{\eta+\rho}x_\eta^* + \frac{\eta}{\eta+\rho}\widehat{x}, s\right) &\leq \frac{\rho}{\eta+\rho}g(x_\eta^*, s) + \frac{\eta}{\eta+\rho}g(\widehat{x}, s) \\
&\leq \frac{\rho}{\eta+\rho}\eta + \frac{\eta}{\eta+\rho}(-\rho) \\
&= 0 \quad \forall s \in \Omega,
\end{aligned}$$

and hence

$$z_\eta^* := \frac{\rho}{\eta+\rho} x_\eta^* + \frac{\eta}{\eta+\rho}\widehat{x} \in \mathcal{F}.$$

Therefore,

$$\begin{aligned} 0 &\le f^* - f(x_\eta^*) \\ &= f^* - f(z_\eta^*) + f(z_\eta^*) - f(x_\eta^*) \\ &\le f(z_\eta^*) - f(x_\eta^*) \\ &= \nabla f(\widetilde{z}_\eta^*)^T (z_\eta^* - x_\eta^*) \\ &\le \frac{\eta}{\eta+\rho} \left\|\nabla f(\widetilde{z}_\eta^*)\right\| \left\|\widehat{x} - x_\eta^*\right\|, \end{aligned}$$

where $\widetilde{z}_\eta^*$ is a point of the segment determined by z_η^* and x_η^*, and so, $\widetilde{z}_\eta^* \in \mathcal{F}_\eta \cap \mathrm{lev}_{\le\alpha} f$, with

$$\alpha \ge \max\{f^*, f(\widehat{x})\}. \tag{3.9}$$

Since the set $\mathcal{F}_\eta \cap \mathrm{lev}_{\le\alpha} f$ is compact, M_2 is chosen as in (b); i.e.,

$$M_2 := \max\left\{\|\nabla f(x)\| : \ x \in \mathcal{F}_\eta \cap \mathrm{lev}_{\le\alpha} f\right\}$$

for any α satisfying (3.9), and hence (c) is proved. □

Remark 3.1. If (P) has a unique optimal solution, denoted by x^*, then by Theorem 3.2(a) we have $\lim_{\eta\to 0} x_\eta^* = x^*$. Hence, Algorithm 2.1 provides an approximate optimal solution to (P) after finitely many iterations.

Remark 3.2. Obviously,

$$\lim_{\eta\to 0} \mathrm{dist}(\mathcal{F}\cap \mathrm{lev}_{\le\alpha} f, \mathcal{F}_\eta \cap \mathrm{lev}_{\le\alpha} f) = 0 \quad \forall \alpha \ge f^*.$$

Hence, Theorem 3.2(b) and (c) provide error bounds for the approximate optimal solution x_η^* of (P).

Moreover, the last iteration point satisfies $x_\eta^* \in \mathcal{F}_\eta$ and $f(x_\eta^*) \le f^*$. If x_η^* is feasible for (P) then $f(x_\eta^*) \ge f^*$, and hence x_η^* is an optimal solution of (P).

Remark 3.3. In Step 1 of Algorithm 2.1, we may also simultaneously choose l different points $\{s_1^k, \ldots, s_l^k\}$ satisfying

$$g(x^k, s_i^k) > \eta \text{ for } i = 1, \ldots, l,$$

and let

$$\bar{E}_{k+1} := E_k \cup \{s_1^k, \ldots, s_l^k\}.$$

For such a multiple explicit exchange method, the aforementioned theorems also apply and can be proved by using analogous techniques.

4. Preliminary numerical experiments.

In this section we report some preliminary numerical test results based on numerical examples taken from the SIP literature and some medium size problems from complex approximation theory and FIR filter design. We implemented Algorithm 2.1 in MATLAB 7.0 and ran experiments on an IBM ThinkPad R52 personal computer, under Microsoft Windows XP operating system. For comparison purposes, the central cutting plane (CCP) algorithm for

CSIP [17], and the SIP solver *fseminf* in MATLAB toolbox, which is based on the discretization SQP method, were used for solving the same test problems. The CCP algorithm and the SIP solver *fseminf* were implemented in MATLAB 7.0 and we used the same computer.

Throughout the computational experiments, we set the details as follows. For the CCP algorithm, we set $\beta = 0.75$ in Deletion Rule 2 and $\sigma^k = \eta$ (where η is defined as in Algorithm 2.1) as its stopping criterion. For the SIP solver *fseminf*, we use 1000 points in the initial grid, and we set $TolCon = \eta$ and $MaxIte = 5000$. We implement Algorithm 2.1 with multiple exchange, and we apply the nonlinear programming solver *fmincon* and the linear programming solver *linprog* from MATLAB toolbox to solve each convex subproblem and linear subproblem, respectively. If we have, for instance, $\Omega = [a_1, b_1] \times \cdots \times [a_p, b_p] \subset \mathbb{R}^p$, and N is a natural number, we consider the uniform grids T_N (the cardinality of T_N is N^p) composed by those points $s \in \mathbb{R}^p$ whose ith coordinate ($i = 1, 2, \ldots, p$) is of the form

$$s_i = a_i + j\frac{b_i - a_i}{N-1} \quad \text{for some } j \in \{0, 1, \ldots, N-1\}.$$

To initialize the algorithm, we choose in Step 0 the initial set of indexes $E_0 = T_{10} \supset \Omega_0$.

The following implementable procedure is used in Step 1 of Algorithm 2.1 to find an $s^k_{new} \in \Omega$ satisfying $g(x^k, s^k_{new}) > \eta$.

Procedure A (at Step 1 of kth iteration of Algorithm 2.1).

Consider a set of natural numbers $N_0 < N_1 < \ldots < N_l$.

Step r ($r = 0, 1, \ldots, l$).

Check, point after point, if there is a point $\bar{s} \in T_{N_r}$ such that $g(x^k, \bar{s}) > \eta$.

```
(I). If such a point exists, we take $s^k_{new} = \bar{s}$ and STOP.
(II). If such a point does not exist:
    II(1). If $\max_{s \in T_{N_r}} g(x^k, s) < -\eta$, there are two cases: if $r < l$, go to Step $r+1$;
           if $r \geq l$, $s^k_{new}$ is not defined and Algorithm 2.1 stops.
    II(2). Otherwise, there exists $\tilde{s} \in T_{N_r}$ such that $g(x^k, \tilde{s}) = \max_{s \in T_{N_r}} g(x^k, s) \geq$
           $-\eta$ and $g(x^k, \tilde{s}) \leq \eta$. At this time we apply Newton method to the
           problem $\max\{g(x^k, s) |\ s \in \Omega\}$ starting from the point $\tilde{s}$.
        II(2)-1. If Newton method produces some point $\hat{s} \in \Omega$ such that $g(x^k, \hat{s}) > \eta$,
                 take $s^k_{new} = \hat{s}$ and STOP.
        II(2)-2. Otherwise, there are two cases: if $r < l$, go to Step $r+1$; if $r \geq l$,
                 $s^k_{new}$ is not defined and Algorithm 2.1 stops.
```

We implement Procedure A with $N_0 = 10, N_1 = 100, \ldots, N_l = 10^5$. We stop the iteration of Algorithm 2.1 once we have checked T_{10^5} without finding any index satisfying (2.2).

4.1. Numerical examples. The following four numerical problems are used for numerical experiments with Algorithm 2.1, the CCP algorithm, and the SIP solver *fseminf*. Here we set $\eta = 10^{-8}$ in Algorithm 2.1. In all cases we choose the vector of ones, i.e., $x^0 = (1, \ldots, 1)^T$, as the starting point, x^* denotes the point reached when the algorithm terminates, or the exact optimal solution if it is known.

Example 1.

$$\begin{array}{ll} \min & f(x) = (x_1 - 2)^2 + (x_2 - 0.2)^2 \\ \text{s.t.} & g(x, s) = 5x_1^2 \sin(\pi\sqrt{s})/(1 + s^2) - x_2 \leq 0 \ \forall s \in [0, 1], \\ & -1 \leq x_1 \leq 1, \quad 0 \leq x_2 \leq 0.2. \end{array}$$

This problem is taken from Tichatschke and Nebeling [32], and it was also tested by Kortanek and No [17]. Clearly, it is a convex SIP problem with strictly convex objective function and satisfies the SCQ, and the feasible set is bounded. Hence, it follows from Theorem 3.1 that Algorithm 2.1 finitely terminates for this problem. Actually, Algorithm 2.1 terminates after 4 iterations and provides

$$x^* = (0.292893216632121, 0.292893216632121)^T.$$

Example 2.

$$\begin{array}{ll} \min & f(x) = x_1^2 + x_2^2 \\ \text{s.t.} & g(x,s) = \cos(s)x_1 + \sin(s)x_2 - (1 + \cos(s) + \sin(s)) \le 0 \quad \forall s \in [\pi, 3\pi/2]. \end{array}$$

This problem is discussed in Goberna and López [10]. It has an objective function that is strictly convex, the feasible set is not bounded, but the objective function is level bounded on the feasible set, and hence Theorem 3.1 shows that Algorithm 2.1 finitely terminates for this problem. We can take $\Omega_0 = \{\pi, 3\pi/2\}$. Algorithm 2.1 terminates after 4 iterations and provides $x^* = (0.205236773567, 0.2)^T$.

Example 3.

$$\begin{array}{ll} \min & f(x) = x^T Hx - 2c^T x \\ \text{s.t.} & g_1(x,s) = |\phi(s)^T x - 1| - 0.05 \le 0 \quad \forall s \in \Omega^1 = [0, 0.05], \\ & g_2(x,s) = |\phi(s)^T x| - 0.01 \le 0 \quad \forall s \in \Omega^2 = [0.1, 0.5], \end{array}$$

where

$$H = \int_0^{0.05} \phi(s)\phi(s)^T \mathrm{d}s + 1000 \int_{0.1}^{0.5} \phi(s)\phi(s)^T \mathrm{d}s, \quad c = \int_0^{0.05} \phi(s)\mathrm{d}s,$$

$$\phi(s) = (2\cos(2\pi s(n-1)), 2\cos(2\pi s(n-2)), \ldots, 2\cos(2\pi s), 1)^T, \quad n = 18.$$

This problem comes from linear-phase FIR digital filter design with weighted peak-constrained least-square error [6]. Clearly, it can be regarded as a quadratically constrained convex quadratic SIP problem, its objective function is strictly convex, and the feasible set, with respect to $\Omega_0^1 = \{0\}$ and $\Omega_0^2 = \{0.5\}$, is bounded. Theorem 3.1 shows that Algorithm 2.1 finitely terminates for this problem. Algorithm 2.1 terminates after 10 iterations and provides

$$\begin{aligned} x^* =& (0.00527457, 0.00326932, 0.00066374, -0.00325896, -0.00834069, -0.01342369, \\ & -0.02101145, -0.02225248, -0.02198446, -0.01495435, -0.00071981, 0.01990574, \\ & 0.04471856, 0.07366002, 0.10022894, 0.12319959, 0.13823466, 0.14358151)^T. \end{aligned}$$

Example 4.

$$\begin{array}{ll} \min & f(x) = x_1^2 + x_2^2 \\ \text{s.t.} & g(x,s) = s_1((x_1-2)^2 + (x_2-2)^2 - 4) + s_2(x_1^2 + x_2^2 - 4) \le 0 \\ & 0 \le x_1 \le 2, \quad 0 \le x_2 \le 2 \\ & \Omega = \{s \in R^2 |\, 0 \le s_1 \le 1,\, 0 \le s_2 \le 1\}. \end{array}$$

This problem was tested by Kortanek and No [17]. Clearly, it is a convex SIP problem with multidimensional index set. It has strictly convex objective function and satisfies the SCQ, and the feasible set is bounded. Hence, it follows from Theorem 3.1 that Algorithm 2.1 finitely terminates for this problem. Algorithm 2.1 terminates after 2 iterations and provides $x^* = (0.585786433298797, 0.585786440777682)^T$.

We compared the performance of our algorithm on these numerical examples with those of method CCP in [17] and the SIP solver *fseminf* in MATLAB toolbox. The detailed comparative results are summarized in Table 1, where *Iter* denotes the number of iterations, *CPU(s)* denotes the elapsed CPU time in seconds, *FuncCount* denotes the number of objective function evaluations, f^* is the objective function value $f(x^*)$, and g^* denotes the value of $\max\{g(x^*, s)|\ s \in T_{10^5}\}$. Clearly, g^* tests the feasibility of x^* on the final grid.

TABLE 1
The comparative results are for Examples 1–4.

Example	Algorithm	Iter	CPU (s)	FuncCount	f^*	g^*
1	Algorithm 2.1	4	0.25	25	3.2211750390	6.53e-13
	CCP	181	1.67	236	3.2211750390	-4.35e-11
	fseminf	6	2.09	21	3.2211750390	6.53e-13
2	Algorithm 2.1	4	0.05	45	0.1715728727	3.05e-9
	CCP	32	7.4	245	0.1715728727	-2.67e-8
	fseminf	53	1.95	288	0.199999954539671	3.83e-8
3	Algorithm 2.1	10	13.73	205	-0.1627692903	7.49e-9
	CCP	–	–	–	–	–
	fseminf	–	–	–	–	–
4	Algorithm 2.1	2	0.02	87	0.6862914996	1.14e-11
	CCP	9	0.30	129	0.6862915011	-8.27e-1
	fseminf	4	16.20	107	0.6862915006	9.16e-10

It can be seen from Table 1 that the performance of Algorithm 2.1 when used to solve Examples 1–4 is better than that of the CCP algorithm and the SIP solver *fseminf*. Actually, the SIP solver *fseminf* does not produce the optimal solution $(1 - 1/\sqrt{2}, 1 - 1/\sqrt{2})^T$ when it is applied to Example 2 and fails for Example 3 with the vector of ones x^0 as starting point. We had to give up trying the CCP algorithm for Example 3 since it took too much time to terminate.

4.2. Complex approximation. The linear Chebyshev complex approximation (CCA) theory problem is the following convex program: Given a set of functions u_j, $j = 1, \ldots, n$, and h, defined all on the complex plane $\mathbb{C}$ and complex valued, consider the residual function

$$r(x, s) = h(s) - \sum_{j=1}^{n} x_j u_j(s),$$

where $x, s \in \mathbb{C}$. If Ω is a compact set in the complex plane, to calculate the minimax error

$$\bar{\gamma} = \min_{x \in \mathbb{C}} \max\{|r(x, s)|\ s \in \Omega\} \tag{4.1}$$

is equivalent to the following SIP problem [2]:

$$\begin{array}{ll} \min & \gamma \\ \text{s.t.} & -\gamma + (Re(r(x, s)))^2 + (Im(r(x, s)))^2 \leq 0 \quad \forall s \in \Omega, \end{array} \tag{4.2}$$

where *Re* and *Im* denote the real and imaginary parts of a complex number, and the variables are x and γ.

Let us introduce vectors $u = (x^T, \gamma)^T \in \mathbb{R}^{n+1}$ and $v = (0, \ldots, 0, 1)^T \in \mathbb{R}^{n+1}$. Define the function $G(u, s) : \mathbb{R}^{n+1} \times \Omega$ as

$$G(u, s) := (Re(r(x, s)))^2 + (Im(r(x, s)))^2 - \gamma.$$

It follows (see, for instance, [18, p. 137]) that the function $G(\cdot, s)$ is strictly convex. Hence, the problem (4.2) can be reformulated as the following convex SIP problem:

$$\begin{array}{ll} \min & v^T u \\ \text{s.t.} & G(u,s) \le 0 \quad \forall s \in \Omega. \end{array}$$

Example 5.

$$h(s) = 1/(s-2), \quad u_j(s) = s^{j-1},\ j = 1, 2, \dots, n,$$

with $s = \cos t + i \sin t,\ t \in [0, 2\pi]$. All x_j are real and $|x_j| \le 3.1$ for $j = 1, \dots, n$.

This problem was tested in [2, 9, 17]. Obviously, $u_j(s) = e^{it(j-1)}, j = 1, 2, \dots, n$ are complex functions on $[0, 2\pi]$. Since the function $G(u, s)$ is strictly convex and the feasible set is bounded, it follows from Theorem 3.1 that Algorithm 2.1 finitely terminates for the CCA problem (4.1). We take the same stopping tolerance as in the CCP algorithm [17] and set $\eta = 10^{-12}$ in Algorithm 2.1. For instance, we implement Algorithm 2.1 for solving Example 5 with $n = 5$, and then the obtained approximate optimal solution u^* are:

$$\begin{aligned} (n = 5) \qquad u^* = &(-0.50000000, -0.25000000, -0.12500000, -0.06250000, \\ &-0.04166667, 4.340278e-4)^T. \end{aligned}$$

For comparison purposes, we solve Example 5 using Algorithm 2.1, the CCP algorithm, and the SIP solver *fseminf*. The obtained results are summarized in Table 2. We report here on the number of iterations, the CPU elapsed time, the minimax error $\bar{\gamma}$, and the feasibility criterion $G^* := \max\{G(u^*, s) \mid s \in T_{10^5}\}$.

TABLE 2
The comparative results are for Example 5.

	Algorithm	Iter	CPU (s)	Error $\bar{\gamma}$	G^*
$n = 5$	Algorithm 2.1	6	2.48	2.08333333e-2	5.59e-18
	CCP	14	20.78	2.08333334e-2	-8.63e-4
	fseminf	10	11.53	2.08499864e-2	1.83e-13
$n = 7$	Algorithm 2.1	22	10.45	5.20833333e-3	6.16e-15
	CCP	459	76.56	5.20833334e-3	-1.65e-5
	fseminf	117	880.56	5.26993446e-3	3.51e-12
$n = 10$	Algorithm 2.1	24	16.20	6.51041667e-4	2.34e-16
	CCP	576	131.38	6.51063431e-4	-1.86e-4
	fseminf	–	–	–	–
$n = 20$	Algorithm 2.1	66	77.03	5.71265742e-7	9.00e-13
	CCP	789	656.70	5.83404094e-5	-7.11e-3
	fseminf	–	–	–	–

As Table 2 shows, the performance of the CCP algorithm and the SIP solver *fseminf* are acceptable for small values of n. However, those two algorithms are not comparable with our algorithm when n is large. Actually, the CCP algorithm took too much time to stop for $n = 20$, and we had to give up trying *fseminf* for $n \ge 10$, since it did not converge and took too much time to stop.

4.3. FIR filter design. The design of FIR filters is widely performed by means of Chebyshev approximation. Then an optimal FIR filter in this sense is obtained by the solution of the following complex Chebyshev-approximation problem [26]:

$$\ell^* = \min \max_{s \in \Omega} W(s) \left| D(s) - H(x, s) \right| \quad \text{over all } x \in \mathbb{R}^n,$$

where $x := (x_1, \dots, x_n)^T$ is a vector of FIR filter coefficients in $\mathbb{R}^n$,

$$H(x, s) := \sum_{l=1}^{n} x_l \, e^{-is(l-1)}, \quad s \in [0, \pi]$$

is the system function of a FIR filter with coefficients of the unit impulse response, $W(\cdot)$ is a continuous positive real-valued weight function on Ω, and $D(\cdot)$ is the desired frequency response of the frequency s on

$$\begin{aligned} \Omega &= [\alpha_1, \beta_1] \cup [\alpha_2, \beta_2] \cup \cdots \cup [\alpha_{\bar{m}}, \beta_{\bar{m}}], \\ 0 &\le \alpha_1 < \beta_1 < \alpha_2 < \beta_2, \cdots < \alpha_{\bar{m}} < \beta_{\bar{m}} \le \pi. \end{aligned}$$

We approach this problem in its equivalent formulation as a convex SIP problem:

$$\begin{array}{lll} \text{(4.3)} & \min & \theta \\ & \text{s.t.} & \psi_j(\chi, s) \le 0 \ \ \forall s \in \Omega_j = [\alpha_j, \beta_j], \quad j = 1, \dots, \bar{m}, \end{array}$$

where $\chi = (x, \theta)^T \in \mathbb{R}^{n+1}$ is the variables vector, and the function $\psi_j : \mathbb{R}^{n+1} \times \Omega_j \to \mathbb{R}$ is defined as

$$\psi_j(\chi, s) := W(s)^2 \left| D(s) - H(x, s) \right|^2 - \theta, \quad s \in \Omega_j.$$

Example 6. Nonlinear phase lowpass filter: $\bar{m} = 2$, $n = 160$,

$$\Omega_1 = [0, 0.12\pi], \quad \Omega_2 = [0.15\pi, \pi],$$

$$D(s) = \begin{cases} e^{-i55s}, & s \in \Omega_1, \\ 0, & s \in \Omega_2, \end{cases} \quad W(s) = \begin{cases} 1, & s \in \Omega_1, \\ 5, & s \in \Omega_2. \end{cases}$$

We set $\eta = 10^{-5}$ in Algorithm 2.1 for the filter design problem (4.3) with data in Example 6. Let ψ^* denote the value of $\max\{\psi_j(\chi^*, s) | s \in T^j_{10^5}, j = 1, 2\}$. By Theorem 2 in [18, p. 137], we know that the constrained functions $\psi_1(\cdot, s)$, $s \in \Omega_1$ and $\psi_2(\cdot, s)$, $s \in \Omega_2$ are strictly convex, and $\{x^k\}$ is bounded if we include in E_k the lower and upper bound constraints. Hence, it follows from Theorem 3.1 that Algorithm 2.1 finitely terminates for the FIR filter design problem with data in Example 6. In fact, Algorithm 2.1 terminated after 7 iterations, elapsing 119.57 seconds of CPU time, and provided the modulus of complex error ℓ^*=1.2756e-02 and the feasibility criterion ψ^*=9.95e-06. Figure 1 shows the modulus function of the complex approximate errors $W(s)|D(s) - H(x^*, s)|$, $s \in \Omega_j$, for $j = 1, 2$, at the obtained solution x^*. Figure 2 shows the magnitude $|H(x^*, s)|$ for $s \in [0, \pi]$.

It should be pointed out that for this problem the CCP algorithm and the SIP solver *fseminf* failed because they took too much time to stop. However, our algorithm was able to solve the FIR filter design problem.

We also compare the performance of Algorithm 2.1 with that of the algorithm in [26, pp. 131–132] that was specifically proposed for solving FIR filter design problems. We wrote the MATLAB code for the algorithm in [26] and ran it on the same computer to solve Example 6. When we used the same stopping tolerance as 10^{-5}, the algorithm took too much time to stop, and we had to give up the experiment. When we set the stopping tolerance as 5×10^{-3}, the algorithm stopped after 11 iterations, elapsing 63.74 seconds of CPU time and provided the modulus of complex error ℓ^*=1.5546e-02 and $\psi^* = \max\{\psi_j(\chi^*, , s) | s \in T^j_{10^5}, j = 1, 2\} = 2.7364\text{e-}03$.

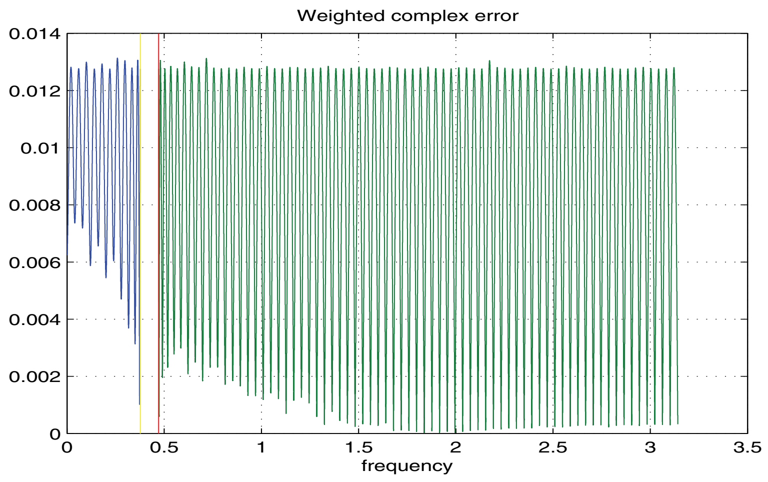

FIG. 1. *Modulus of complex error.*

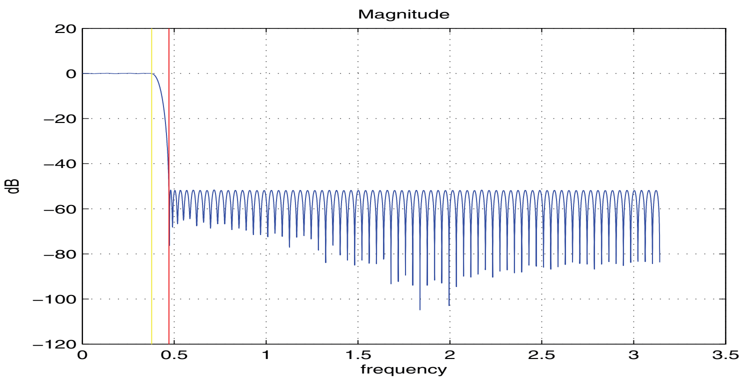

FIG. 2. *Magnitude.*

5. Concluding remarks. We proposed an exchange algorithm (Algorithm 2.1) for solving convex SIP problems and proved the finite termination under reasonable assumptions. In particular, we introduced a new exchange rule. We also proved that if Algorithm 2.1 terminates after a finite number of iterations at the point x^*_η, then x^*_η is approximate optimal solution of (P) as η tends to zero. Additionally, if (P) has a unique optimal solution, x^*_η converges to the optimal solution when η tends to zero, provided that x^*_η lives in a bounded set. We also provided error bounds for the approximate solution in Theorem 3.2.

To show the efficiency of the proposed algorithm, we solved some numerical examples and some problems from complex approximation theory and FIR filter design and compared with the CCP algorithm [17] for convex SIP and the SIP solver *fseminf*. We saw that our algorithm solved the test problems faster than these other methods and that using regular grids in our search turned out to contribute to its efficiency. Specially, we find that our algorithm has better performance on the FIR filter design problem than do some technical methods.

Acknowledgments. The authors would like to thank the Associate Editor and two anonymous referees for their very helpful comments and suggestions that helped us to significantly improve the preliminary version of this paper.

REFERENCES

[1] A. Auslender, M. A. Goberna, and M. A. López, *Penalty and smoothing methods for convex semi-infinite programming*, Math. Oper. Res., 34 (2009), pp. 303–319.
[2] I. Barrodale, L. M. Delves, and J. C. Mason, *Linear Chebyshev approximation of complex-valued functions*, Math. Comp., 32 (1978), pp. 853–863.
[3] B. Betrò, *An accelerated central cutting plane algorithm for linear semi-infinite programming*, Math. Program., 101 (2004), pp. 479–495.
[4] E. W. Cheney, *Introduction to Approximation Theory*, 2nd ed., Chelsea, New York, 1982.
[5] I. D. Coope and G. A. Watson, *A projected Lagrangian algorithm for semi-infinite programming*, Math. Program., 32 (1985), pp. 337–356.
[6] H. H. Dam, K. L. Teo, S. Nordebo, and A. Cantoni, *The dual parameterization approach to optimal least square* FIR *filter design subject to maximum error constraints*, IEEE Trans. Signal Process., 48 (2000), pp. 2314–2320.
[7] A. V. Fiacco and K. O. Kortanek, *Semi-Infinite Programming and Applications, Lecture Notes in Econ. and Math. Systems* 215, Springer, New York, 1983.
[8] K. Glashoff and S. A. Gustafon, *Linear Optimization and Application*, Springer, Berlin, 1983.
[9] K. Glashoff and K. Roleff, *A new method for Chebyshev approximation of complex-valued functions*, Math. Comp., 36 (1981), pp. 233–239.
[10] M. A. Goberna and M. A. López, *Linear Semi-Infinite Optimization*, Wiley, New-York, 1998.
[11] M. A. Goberna and M. A. López, *Linear semi-infinite programming theory: An updated survey*, European J. Oper. Res., 143 (2002), pp. 390–405.
[12] C. Gonzaga, E. Polak, and R. Trahan, *An improved algorithm for optimization problems with functional inequality constraints*, IEEE Trans. Automat. Control, 25 (1980), pp. 49–54.
[13] R. Hettich and K. O. Kortanek, *Semi-infinite programming: Theory, methods and applications*, SIAM Rev., 35 (1993), pp. 380–429.
[14] R. Hettich, *An implementation of a discretization method for semi-infinite programming*, Math. Program., 34 (1986), pp. 354–361.
[15] R. Hettich and G. Gramlich, *A note on an implementation of a method for quadratic semi-infinite programming*, Math. Program., 46 (1990), pp. 249–254.
[16] D. Klatte and R. Henrion, *Regularity and stability in nonlinear semi-infinite optimization*, in Semi-Infinite Programming, R. Reemsten and J. Rückmann, eds., Kluwer Academic Publishers, Boston, 1998, pp. 69–99.
[17] K. O. Kortanek and H. No, *A central cutting plane algorithm for convex semi-infinite programming problems*, SIAM J. Optim., 3 (1993), pp. 901–918.
[18] R. T. Kreyszig, *Advanced Engineering Mathematics*, Wiley, New York, 1993.
[19] H.-C. Lai and S.-Y. Wu, *On linear semi-infinite programming problems–an algorithm*, Numer. Funct. Anal. Optim., 13 (1992), pp. 287–304.
[20] P. J. Laurent and C. Carasso, *An algorithm of successive minimization in convex programming*, Numer. Anal., 12 (1978), pp. 377–400.
[21] M. A. López and G. Still, *Semi-infinite programming*, European J. Oper. Res., 80 (2007), pp. 491–518.
[22] D. G. Luenberger, *Linear and Nonlinear Programming*, 2nd ed., Kluwer Academic Publishers, Boston/Dordrecht/London, 2004.
[23] H. Mine, M. Fukushima, and Y. Tanaka, *On the use of ε-most-active constraints in an exact penalty function method for nonlinear optimization*, IEEE Trans. Automat. Control, 29 (1984), pp. 1040–1042.
[24] C. J. Price and I. D. Coope, *Numerical experiments in semi-infinite programming*, Comput. Optim. Appl., 6 (1996), pp. 169–189.
[25] E. Polak, *On the use of consistent approximations in the solution of semi-infinite optimization and optimal control problems*, Math. Program., 62 (1993), pp. 385–414.
[26] A. Potchinkov and R. Reemsten, *The design of* FIR *filter in the complex plane by convex optimization*, Signal Process., 46 (1995), pp. 127–146.

[27] R. Reemsten and S. Görner, *Numerical methods for semi-infinite programming: A survey*, in Semi-Infinite Programming, R. Reemsten and J. Rückmann, eds., Kluwer Academic Publishers, Boston, 1998, pp. 195–275.

[28] O. Stein and G. Still, *Solving semi-infinite optimization problems with interior point techniques*, SIAM J. Control Optim., 42 (2003), pp. 769–788.

[29] G. Still, *Discretization in semi-infinite programming: The rate of convergence*, Math. Program., 91 (2001), pp. 53–69.

[30] K. L. Teo, X. Q. Yang, and L. S. Jennings, *Computational discretization algorithms for functional inequality constrained optimization*, Ann. Oper. Res., 28 (2000), pp. 215–234.

[31] Y. Liu and K. L. Teo, *An adaptive dual parametrization algorithm for quadratic semi-infinite programming problems*, J. Global Optim., 24 (2002), pp. 205–217.

[32] R. Tichatschke and V. Nebeling, *A cutting plane method for quadratic semi-infinite programming problems*, Optimization, 19 (1988), pp. 803–817.

[33] G. A. Watson, *Numerical experiments with globally convergent methods for semi-infinite programming problems*, in Semi-infinite Programming and Applications, Lecture Notes in Econ. and Math. Systems 215, A.V. Fiacco, K. O. Kortanek, eds., Springer, New York, 1983, pp. 193–205.