SIAM J. OPTIM.
Vol. 16, No. 4, pp. 1137–1154

A TRUNCATED PROJECTED NEWTON-TYPE ALGORITHM FOR LARGE-SCALE SEMI-INFINITE PROGRAMMING*

QIN NI†, CHEN LING‡, LIQUN QI§, AND KOK LAY TEO¶

Abstract. In this paper, a truncated projected Newton-type algorithm is presented for solving large-scale semi-infinite programming problems. This is a hybrid method of a truncated projected Newton direction and a modified projected gradient direction. The truncated projected Newton method is used to solve the constrained nonlinear system. In order to guarantee global convergence, a robust loss function is chosen as the merit function, and the projected gradient method inserted is used to decrease the merit function. This algorithm is suitable for handling large-scale problems and possesses superlinear convergence rate. The global convergence of this algorithm is proved and the convergence rate is analyzed. The detailed implementation is discussed, and some numerical tests for solving large-scale semi-infinite programming problems, with examples up to 2000 decision variables, are reported.

Key words. semi-infinite programming, Karush–Kuhn–Tucker system, large-scale problem

AMS subject classifications. 90C34, 90C06, 90C90, 65K05, 49M05

DOI. 10.1137/040619867

1. Introduction.

We consider the semi-infinite programming (SIP) problem

$$\min\{f(x),\ x \in X\}, \tag{1.1}$$

where $X = \{x \in \Re^n :\ g(x,v) \le 0 \text{ for all } v \in \Omega\}$, $f(x) : \Re^n \to \Re$, and $g : \Re^n \times \Re^m \to \Re$ are twice continuously differentiable functions. In this paper, we assume that Ω is a nonempty compact box with

$$\Omega = \{v \in \Re^m :\ a \le v \le b\},$$

where $a \in \Re^m$, $b \in \Re^m$, and $a < b$.

Such an SIP problem has wide applications such as approximation theory, optimal control, eigenvalue computation, mechanical stress of materials, and statistical design. Many methods have been proposed for the SIP problem. We refer readers to [4, 6, 11, 12, 15, 17] for details.

Some large-scale SIP problems arise from the modeling of optimal control and approximation (see [5, 16, 19]). In order to increase the control precision in an optimal control problem, one should increase the number of switching points. That is, the larger the number of switching points set, the higher the control precision. If one sets

*Received by the editors November 30, 2004; accepted for publication (in revised form) November 7, 2005; published electronically February 15, 2006.
http://www.siam.org/journals/siopt/16-4/61986.html

†Department of Mathematics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, People's Republic of China (niqfs@nuaa.edu.cn). This author's work was supported by the National Natural Science Foundation of China (grant 10471062).

‡School of Information, Zhejiang University of Finance and Economics, Hangzhou, 310012, China (linghz@hzcnc.com). Current address: Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (01902146r@polyu.edu.hk).

§Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (maqilq@polyu.edu.hk). This author's work was supported by the Hong Kong Research Grant Council.

¶Department of Mathematics and Statistics, Curtin University of Technology, Bentley, WA 6102, Australia (K.L.Teo@curtin.edu.au).

a large number of switching points, the discretization of the control space will lead to large-scale SIP problems. In approximation theory, if a function $f(v)$ is approximated on the interval $[a, b]$ by a polynomial

$$f_N(v) = \sum_{j=1}^{N} x_j v^{j-1}$$

and the approximation is in the Chebyshev norm, then we get a SIP problem. It is clear that the larger the order of the polynomial, the higher the approximation precision. When a very high-order polynomial is used to approximate f on $[a, b]$, a large-scale SIP problem is generated. However, some efficient algorithms for small-scale SIP problems do not directly translate into algorithms for large-scale SIP problems.

In this paper, we extend a smoothing projected Newton-type algorithm proposed in [14] to solve large-scale SIP problems. The smoothing projected Newton-type algorithm proposed in [14] enjoys global and locally superlinear convergence. However, it is not suitable for large-scale SIP problems. We modify this algorithm in two aspects. First, a truncated solution of the system is determined by an iterative method, in which the computation of the matrix-vector product, instead of the matrix factorization, is used such that the implementation at each iteration is relatively simple and time-economic. Second, in order to guarantee the global convergence, a robust loss function [7] is chosen as the merit function and the projected gradient method inserted is used to decrease the merit function. This loss function uses a measure which does not weigh very large components of the variable heavily. Numerical results show that this loss function is a good merit function. This modified algorithm is called a truncated projected Newton-type algorithm and is suitable for handling large-scale problems. The global convergence of this algorithm is proved and the superlinear convergence rate is analyzed. The detailed implementation is discussed, and some numerical tests for solving large-scale SIP problems, with examples up to 2000 decision variables, are reported.

This paper is organized as follows. We present a truncated projected Newton-type algorithm in section 2. The convergence of the algorithm is analyzed in section 3 and numerical tests are given in section 4. We propose some comments in section 5.

For convenience, we denote $\nabla_x^T = (\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n})$ and $\nabla_x = (\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n})^T$ for $x \in \Re^n$. For a smoothing function $\Phi : \Re^n \to \Re^m$, we denote $\nabla_x^T \Phi = (\frac{\partial \Phi}{\partial x_1}, \ldots, \frac{\partial \Phi}{\partial x_n})$ and $\nabla_x \Phi \equiv \nabla_x \Phi^T = (\nabla_x \Phi_1, \ldots, \nabla_x \Phi_m)$. For $u \in \Re^n$ and $v \in \Re^m$, we denote by (u, v) the column vector $(u^T, v^T)^T$ in $\Re^{n+m}$.

2. A truncated projected Newton-type algorithm. In order to describe our algorithm, we recall some notation and definitions in [14].

Let

$$V(x) = \{v \in \Omega : \ g(x, v) = 0\}.$$

If there exists a vector $d \in \Re^n$ such that

$$\nabla_x g(x, v)^T d < 0$$

for all $v \in V(x)$, then we say that an extended Mangasarian–Fromovitz constraint qualification (EMFCQ) holds at x. It is well known that if x is a local minimizer

of the SIP problem (1.1), and EMFCQ holds at x, then the KKT system of the SIP problem (1.1) is as follows:

$$\begin{aligned} &\nabla f(x) + \sum_{j=1}^{p} u_j \nabla_x g(x, v^j) = 0, \\ &g(x, v) \le 0 \quad \forall\, v \in \Omega, \\ &u_j > 0, \quad g(x, v^j) = 0, \quad j = 1, \ldots, p, \end{aligned} \tag{2.1}$$

where $v^j \in V(x)$ for $j = 1, \ldots, p$ and $p \le n$.

By the definition of $V(x)$ and the first inequality of (2.1), $v^j \in V(x)$ $(j = 1, \ldots, p)$ imply that v^j $(j = 1, \ldots, p)$ are global minimizers of the following minimization problem:

$$\begin{aligned} \min \quad & -g(x, v) \\ \text{subject to} \quad & v \in \Omega. \end{aligned} \tag{2.2}$$

The KKT system of (2.2) can be written as

$$(v' - v)^T(-\nabla_v g(x, v)) \ge 0 \quad \forall\, v' \in \Omega,$$

and it can be reformulated as a system of nonsmooth equations as follows:

$$v - \operatorname{mid}(a, b, v + \nabla_v g(x, v)) = 0, \tag{2.3}$$

where the mid function is defined for all $i = 1, \ldots, m$ as

$$(\operatorname{mid}(c, d, w))_i = \begin{cases} c_i & \text{if } w_i < c_i, \\ w_i & \text{if } c_i \le w_i \le d_i, \\ d_i & \text{if } d_i < w_i. \end{cases}$$

The system of nonsmooth equations (2.3) can be approximated by

$$(\phi(t, x, v))_i = v_i - \varphi(t, a_i, b_i, v_i + (\nabla_v g(x, v))_i), \quad i = 1, \ldots, m, \tag{2.4}$$

where

$$\varphi(t, c, d, w) = \frac{c + \sqrt{(c - w)^2 + 4t^2}}{2} + \frac{d - \sqrt{(d - w)^2 + 4t^2}}{2}$$

is the Chen–Harker–Kanzow–Smale smoothing function for $\operatorname{mid}(c, d, w)$. It is clear that ϕ is smooth for $t \ne 0$. In order to handle the first constrained condition of (2.1), Teo, Rehbock, and Jennings [20] used a nonsmooth function

$$G(x) = \int_V [g(x, v)]_+ dv, \tag{2.5}$$

where $[x]_+ = \max\{0, x\}$. This is approximated by the smoothing function

$$G(t, x) = \int_\Omega \frac{\sqrt{g^2(x, v) + 4t^2} + g(x, v)}{2} dv, \tag{2.6}$$

which is defined in [14]. Hence (2.1) can be approximated by

$$\begin{array}{l} \nabla f(x) + \sum_{j=1}^{p} u_i \nabla_x g(x, v^j) = 0, \\ u_j > 0, \quad g(x, v^j) = 0, \quad j = 1, \ldots, p, \\ \phi(t, x, v^j) = 0, \quad j = 1, \ldots, p, \\ G(t, x) = 0. \end{array} \tag{2.7}$$

Now we define

$$L(x, u, V) = f(x) + \sum_{j=1}^{p} u_j g(x, v^j), \tag{2.8}$$

where $V = (v^1, v^2, \ldots, v^p) \in \Re^{mp}$ and $u = (u_1, \ldots, u_p)^T \in \Re^p$, and denote

$$\mathbf{g}(x, V) = \begin{bmatrix} g(x, v^1) \\ g(x, v^2) \\ \cdots \\ g(x, v^p) \end{bmatrix}, \quad \bar{\phi}(t, x, V) = \begin{bmatrix} \phi(t, x, v^1) \\ \phi(t, x, v^2) \\ \cdots \\ \phi(t, x, v^p) \end{bmatrix}. \tag{2.9}$$

In order to balance the number between equations and variables, we add an artificial variable y. By simple analysis, we can know that the KKT system of the SIP problem (1.1) can be reformulated as a equivalent system of constrained equations in the following:

$$\begin{array}{c} \Phi(w) = 0, \\ u \geq 0, \quad y \geq 0, \end{array} \tag{2.10}$$

where $w = (t, z) = (t, x, u, V, y) \in \Re \times \Re^n \times \Re^p \times \Re^{mp} \times \Re$, and

$$\Phi(w) = \begin{bmatrix} t \\ H(w) \end{bmatrix}, \quad H(w) = \begin{bmatrix} \nabla_x L(x, u, V) \\ \mathbf{g}(x, V) \\ \bar{\phi}(t, x, V) \\ G(t, x) + y \end{bmatrix}. \tag{2.11}$$

For convenience, we denote $w = (t, x, u, V, y) \in \Re^{\tilde{n}}$, $\tilde{n} = n+2+(m+1)p$. The function $\Phi(w)$ has the following property.

LEMMA 2.1 (see [14]). *$\Phi(w) = \Phi(t, z)$ is smooth at (t, z) with $t \neq 0$ and semismooth at $(0, z)$.*

For the meaning of semismoothness we refer readers to [10, 13].

The problem (2.10) was established and a smoothing projected Newton-type algorithm was proposed for solving this problem in [14]. In the smoothing projected Newton-type algorithm in [14], the Newton direction is obtained by solving the following linear system:

$$\Phi(w_k) + \nabla^T \Phi(w_k) \Delta w_k = \beta_k \bar{w}, \tag{2.12}$$

where $\Delta w_k = (\Delta t_k, \Delta x_k, \Delta u_k, \Delta V_k, \Delta y_k) \in \Re^{\tilde{n}}$, $\bar{w} = (\bar{t}, 0)$, $\bar{t} > 0$, and

(2.13)

$$\nabla^T \Phi(w_k) = \begin{bmatrix} 1 & 0_{1\times n} & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 0 \\ 0_{n\times 1} & \nabla_x^2 L(x,u,V) & \nabla_x \mathbf{g}^T(x,V) & u_1 \nabla_{v^1}^T(\nabla_x g(x,v^1)) & \cdots & u_p \nabla_{v^p}^T(\nabla_x g(x,v^p)) & 0_{n\times 1} \\ 0 & \nabla_x^T g(x,v^1) & 0_{1\times p} & \nabla_{v^1}^T g(x,v^1) & \cdots & 0_{1\times m} & 0 \\ 0 & \nabla_x^T g(x,v^2) & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \nabla_x^T g(x,v^p) & 0_{1\times p} & 0_{1\times m} & \cdots & \nabla_{v^p}^T g(x,v^p) & 0 \\ \nabla_t \phi(t,x,v^1) & \nabla_x^T \phi(t,x,v^1) & 0_{p\times p} & \nabla_{v^1}^T \phi(t,x,v^1) & \cdots & 0_{1\times m} & 0_{p\times 1} \\ \nabla_t \phi(t,x,v^2) & \nabla_x^T \phi(t,x,v^2) & 0_{p\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 0_{p\times 1} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \nabla_t \phi(t,x,v^p) & \nabla_x^T \phi(t,x,v^p) & 0_{p\times p} & 0_{1\times m} & \cdots & \nabla_{v^p}^T \phi(t,x,v^p) & 0_{p\times 1} \\ \nabla_t G(t,x) & \nabla_x^T G(t,x) & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 1 \end{bmatrix}.$$

We remark that from (2.13) we see that the last row of the matrix $\nabla^T \Phi(w_k)$ is independent of other rows, so the introduction of artificial variable y can reduce the possible degeneration generated by the function $G(t,x)$. In order to solve large-scale problems, we determine an inexact solution of (2.12) by using a restarted generalized minimum residual algorithm (GMRES($\tilde{m}$)) [3]. Here we use $\tilde{m}$ other than usual m because there is an another meaning for m in this paper. The vector Δw_k is called a truncated solution of (2.12) if

$$\|\Phi(w_k) + \nabla^T \Phi(w_k) \Delta w_k - \beta_k \bar{w}\| \le r_k \tag{2.14}$$

for $r_k > 0$.

In [14], a simple merit function

$$\Psi(w) = \frac{1}{2} \sum_{j=1}^{\tilde{n}} \Phi_j^2(w) \tag{2.15}$$

is chosen and its gradient is

$$\nabla \Psi(w) = \nabla \Phi(w) \Phi(w). \tag{2.16}$$

In order to solve the large-scale SIP problem, we consider the following function:

$$\Psi_h(w) = \sum_{j=1}^{\tilde{n}} \rho_{h_j}(\Phi_j(w)), \tag{2.17}$$

where

$$\rho_{h_j}(\xi) = \begin{cases} \xi^2/2 & \text{if } |\xi| \le h_j \\ h_j|\xi| - h_j^2/2 & \text{otherwise,} \end{cases}$$

h_j, $j = 1, \ldots, \tilde{n}$, are positive constants, and $\rho_{h_j}(\xi)$ is linear in ξ for $|\xi| > h_j$. This function was proposed by Huber and Dutter (see [7] and [2]) for solving the least squares problem. The measure $\rho(\xi)$ in this function does not weigh very large components of ξ heavily.

We use the function (2.17) as the merit function. The gradient of this function $\Psi_h(w)$ is

$$\nabla\Psi_h(w) = \nabla\Phi(w)\Phi_h(w), \tag{2.18}$$

where

$$\Phi_h(w) = \sum_{j\in J_h} \Phi_j(w)e_j + \sum_{j\in K_h} \mathrm{sign}(\Phi_j(w))h_j e_j, \tag{2.19}$$

$$J_h = \{j:\ 1 \le j \le \tilde{n},\ |\Phi_j(w)| \le h_j\}, \quad K_h = \{1,\dots,\tilde{n}\}/J_h. \tag{2.20}$$

The problem (2.10) is equivalent to finding a global solution of the following minimization problem:

$$\begin{array}{ll} \min & \Psi_h(w) \\ \text{subject to} & u \ge 0,\ y \ge 0. \end{array} \tag{2.21}$$

We call w a stationary point of (2.21) if it satisfies

$$\|\bar{d}_G(1)\| = 0, \tag{2.22}$$

where

$$\bar{d}_G(1) = \Pi_W(w - \gamma\nabla\Psi_h(w)) - w = \left[\begin{array}{c} -\gamma\nabla_t\Psi_h(w) \\ \Pi_Z(z - \gamma\nabla_z\Psi_h(w)) - z \end{array}\right], \tag{2.23}$$

$\gamma > 0$ is a constant, and $\Pi_W(\cdot)$ is an orthogonal projection operator onto W,

$$W = \{w = (t,x,u,V,y) \in \Re^{\tilde{n}} :\ u \ge 0, y \ge 0\},$$
$$Z = \{z = (x,u,V,y) \in \Re^{\tilde{n}-1} :\ u \ge 0, y \ge 0\}.$$

Let $\alpha \in (0,1)$ be a constant. For a sequence $\{w_k\}_{k=0}^{\infty}$, we define

$$\beta_0 = \beta(w^0) = \alpha\min\{1, \|\bar{d}_G^0(1)\|^2\},$$

and

$$\beta_k = \beta(w^k) := \left\{\begin{array}{ll} \beta_{k-1} & \text{if } \alpha\min\{1, \|\bar{d}_G^k(1)\|^2\} > \beta_{k-1}, \\ \alpha\min\{1, \|\bar{d}_G^k(1)\|^2\} & \text{otherwise.} \end{array}\right. \tag{2.24}$$

Now we state our truncated projected Newton-type algorithm for solving (2.10) below.

Algorithm 2.1.

Step 0. (Initialization)

Choose constants $\eta, \rho, \sigma \in (0,1)$ with $\sigma\eta < 1$, $p_1 > 0$, $p_2 > 2$ and $\alpha \in (0,1)$, $\bar{t} > 0$ with $\alpha\bar{t} < 1$, $h_j \ge 1$, $j = 1,\dots,\tilde{n}$. Let $\bar{w} = (\bar{t},0,0,0,0)$, $t_0 = \bar{t}$, and $w^0 = (t_0, x^0, u^0, V^0, y^0)$ with $u_i^0 \ge 0$ $(i = 1,\dots,p)$, $y^0 \ge 0$. Set $k = 0$.

Step 1. (Termination Test)

Compute

$$\xi_k = \min\left\{1, \frac{t_k}{|t_k + \nabla_t H(w_k)H_h(w_k)|},\ \frac{\eta\|\Phi(w_k)\|}{\|\nabla\Psi_h(w_k)\|}\right\}, \tag{2.25}$$

where $H_h(w_k)$ is obtained by removing just the first element of $\Phi_h(w)$ (see (2.19)).

$$\gamma_k = \begin{cases} \min\left\{\xi_k, \frac{\eta\Psi_h(w_k)}{\|\nabla\Psi_h(w_k)\|^2}\right\} & \text{if } |\Phi_j(w_k)| \le h_j,\ j = 1, 2, \ldots, \tilde{n}, \\ \xi_k & \text{otherwise.} \end{cases} \tag{2.26}$$

If $\bar{d}_G^k(1) = 0$, then stop; otherwise compute β_k by (2.24) and go to Step 2.

Step 2. (Search Directions)

2.1. Compute the negative gradient direction. Compute

$$d_G^k = -\gamma_k \nabla\Psi_h(w_k) + \beta_k \bar{w}. \tag{2.27}$$

If

$$|\Phi_j(w_k)| \le h_j, \quad j = 1, 2, \ldots, \tilde{n}, \tag{2.28}$$

then go to Step 2.2; otherwise set $d_{tN}^k = d_G^k$ and go to Step 3.

2.2. Compute the truncated Newton direction. Determine Δw_k, which satisfies

$$\|\Phi(w_k) + \nabla^T\Phi(w_k)\Delta w_k - \beta_k\bar{w}\| = o(\Psi_h(w^k)), \tag{2.29}$$

and set $d_{tN}^k = \Delta w_k$.

Step 3. (Line Search)

Let m_k be the smallest nonnegative integer m satisfying

$$\Psi_h(w^k + \bar{d}^k((\rho)^m)) \le \Psi_h(w^k) + \sigma\nabla\Psi_h(w^k)^T\tilde{d}_G^k((\rho)^m), \tag{2.30}$$

where for any $\lambda \in [0, 1]$,

$$\bar{d}^k(\lambda) = \tau^*(\lambda)\tilde{d}_G^k(\lambda) + (1 - \tau^*(\lambda))\tilde{d}_{tN}^k(\lambda). \tag{2.31}$$

Here

$$\tilde{d}_G^k(\lambda) := \Pi_W(w_k + \lambda d_G^k) - w_k, \quad \tilde{d}_{tN}^k(\lambda) := \Pi_W(w_k + \lambda d_{tN}^k) - w_k, \tag{2.32}$$

and $\tau^*(\lambda)$ is a solution of the following minimization problem:

$$\min_{\tau\in[0,1]} \frac{1}{2}\|\Phi(w_k) + \Phi'(w_k)[\tau\tilde{d}_G^k(\lambda) + (1 - \tau)\tilde{d}_{tN}^k(\lambda)]\|^2.$$

Let $\lambda_k = (\rho)^{m_k}$ and $w^{k+1} = w^k + \bar{d}^k(\lambda_k)$.

Step 4. Set $k = k + 1$ and go to Step 1.

Remarks. (1) Algorithm 2.1 is able to handle sparse large-scale SIP problems. In Step 2.2, a truncated solution of the problem (2.12) is determined by using GMRES($\tilde{m}$) method. Hence, the matrix factorizations are avoided, because this iterative algorithm requires computing only matrix-vector products. If the SIP problem possesses the sparse data structure, the computation of the matrix, $\nabla^T\Phi(w_k)$, can take advantage of the sparsity of $\nabla_x^2 L(x_k, u_k, V_k)$. Therefore Algorithm 2.1 is applicable to the sparse large-scale SIP problem.

(2) If the condition (2.28) is not satisfied, then only the projected negative gradient direction is generated in the iteration; otherwise Step 2.2 is carried out and

mixed projected directions are generated. In addition, if (2.28) is satisfied, then $\Psi_h(w) = \Psi(w)$ holds.

(3) The condition (2.29) guarantees the convergence of Algorithm 2.1, which is discussed in the next section. In the implementation of the algorithm, one kind of choice of the right side in (2.29) is $\frac{1}{k+1}\min\{1, \Psi_h(w_k)\}$.

(4) $\tau^*(\lambda)$ is easily obtained, and we refer readers to [14].

(5) Another line search technique in Step 3 can be used if only the projected negative gradient is the search direction. Although it does not affect the convergence and its proof, it can decrease the number of inner iterations. In section 4 we give a detailed description.

(6) We remark that $G(t, x)$ in (2.11) and its derivative are not evaluated exactly. We use Newton–Cotes formulas (Simpson's rule) for approximating the integral and choose n_i equally spaced points in the interval $[a_i, b_i]$ such that $(b_i - a_i)/n_i \le 0.05$ is satisfied, where $i = 1, \ldots, m$. Numerical results show that this choice is proper.

3. Convergence analysis. In this section we discuss the convergence property of Algorithm 2.1. From the definition of β_k, the following lemma is obvious.

LEMMA 3.1. *$\{\beta_k\}$ defined in (2.24) has the following properties:*

(i) *$\{\beta_k\}$ is a nonincreasing sequence.*

(ii) *For all k, β_k satisfies*

$$\beta_k \le \alpha \min\{1, \|\bar{d}_G^k(1)\|^2\}.$$

In the following we give the descent property of $\tilde{d}_G^k(\lambda)$ in Algorithm 2.1.

LEMMA 3.2. *Suppose that $w_k = (t^k, z^k) \in W$ with $t^k > 0$ is not a stationary point of (2.21). Then for any $\lambda \in (0, 1]$, it holds that*

$$\nabla\Psi_h(w_k)^T \tilde{d}_G^k(\lambda) \le -\frac{\lambda}{\xi_k}(1 - \alpha\bar{t}\,)\|\bar{d}_G^k(1)\|^2 < 0. \tag{3.1}$$

Proof. In this proof, for simplicity, we drop the superscript k. For any $w = (t, z) \in W$ with $t > 0$, suppose that w is not a stationary point of (2.21). Then

$$\nabla\Psi_h(w) = \nabla\Phi(w)\Phi_h(w) = \begin{bmatrix} \tilde{t} + \nabla_t H^T(w) H_h(w) \\ \nabla_z H(w) H_h(w) \end{bmatrix} \equiv \begin{bmatrix} \nabla_t \Psi_h(w) \\ \nabla_z \Psi_h(w) \end{bmatrix},$$

where $\tilde{t} = \min(t, h_1)$, $\nabla_t H^T(w)$ is the first row of $\nabla H(w)$, $\nabla_z H(w)$ is the submatrix of $\nabla H(w)$ obtained by removing just the first row of $\nabla H(w)$, and $H_h(w)$ is obtained by removing just the first element of $\Phi_h(w)$ (see (2.19)). From (2.32) and the definition of projection in (2.23), $\tilde{d}_G(\lambda)$ can be written as

$$\begin{aligned} \tilde{d}_G(\lambda) &\equiv \begin{bmatrix} (\tilde{d}_G(\lambda))_t \\ (\tilde{d}_G(\lambda))_z \end{bmatrix} = \Pi_W(w - \lambda\gamma\nabla\Psi_h(w) + \lambda\beta\bar{w}) - w \\ &= \begin{bmatrix} -\lambda\gamma(\tilde{t} + \nabla_t H^T(w) H_h(w)) + \lambda\beta\bar{t} \\ \Pi_Z(z - \lambda\gamma\nabla_z\Psi_h(w)) - z \end{bmatrix}. \end{aligned}$$

Then we have

$$
\begin{aligned}
&(\tilde{t} + \nabla_t H^T(w) H_h(w))[-\lambda\gamma(\tilde{t} + \nabla_t H(w) H_h(w)) + \lambda\beta\bar{t}] \\
&= -\lambda\gamma \|\tilde{t} + \nabla_t H(w) H_h(w)\|^2 + \lambda(\tilde{t} + \nabla_t H(w) H_h(w))\beta\bar{t} \\
&\le -\frac{\lambda}{\gamma}\|\gamma\nabla_t \Psi_h(w)\|^2 + \frac{\lambda}{\gamma}\|\gamma\nabla_t \Psi_h(w)\|\beta\bar{t} \qquad (3.2) \\
&\le -\frac{\lambda}{\gamma}\|\gamma\nabla_t \Psi_h(w)\|^2 + \frac{\lambda}{\gamma}\|\gamma\nabla_t \Psi_h(w)\|(\alpha\bar{t})\|\bar{d}_G(1)\| \\
&\le -\frac{\lambda}{\gamma}\|\gamma\nabla_t \Psi_h(w)\|^2 + \alpha\bar{t}\frac{\lambda}{\gamma}\|\bar{d}_G(1)\|^2,
\end{aligned}
$$

where the second inequality comes from Lemma 3.1(ii) and the fact that $\beta \le \alpha\|\bar{d}_G(1)\|$, the last inequality, is due to $\|\gamma\nabla_t \Psi_h(w)\| \le \|\bar{d}_G(1)\|$ (see (2.23)). In addition,

$$
\begin{aligned}
&\nabla_z \Psi_h(w)^T[\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - z] \\
&= -\frac{1}{\lambda\gamma}[z - \lambda\gamma\nabla_z \Psi_h(w) - z]^T[\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - z] \\
&= \frac{1}{\lambda\gamma}[\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - (z - \lambda\gamma\nabla_z \Psi_h(w))]^T[\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - z] \\
&-\frac{1}{\lambda\gamma}\|\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - z\|^2 \qquad (3.3) \\
&\le -\frac{1}{\lambda\gamma}\|\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - z\|^2 \\
&\le -\frac{\lambda}{\gamma}\|\Pi_Z(z - \gamma\nabla_z \Psi_h(w)) - z\|^2,
\end{aligned}
$$

where the first and second inequalities come from the property of projector (see [1]). It follows from (3.2) and (3.3) that

$$
\begin{aligned}
&\nabla\Psi_h(w)^T\tilde{d}_G(\lambda) \\
&= (\tilde{t} + \nabla_t H^T(w) H_h(w))[-\lambda\gamma(\tilde{t} + \nabla_t H(w) H_h(w)) + \lambda\beta\bar{t}] \\
&\quad + \nabla_z \Psi(w)^T[\Pi_Z(z - \lambda\gamma\nabla_z \Psi_h(w)) - z] \\
&\le -\frac{\lambda}{\gamma}\left[\|\gamma\nabla_t \Psi_h(w)\|^2 + \|\Pi_z(z - \gamma\nabla_z \Psi_h(w)) - z\|^2\right] + \alpha\bar{t}\frac{\lambda}{\gamma}\|\bar{d}_G(1)\|^2 \\
&= -\frac{\lambda}{\gamma}(1 - \alpha\bar{t})\|\bar{d}_G(1)\|^2 < 0.
\end{aligned}
$$

The proof is complete. ∎

Remark. If (2.28) is not satisfied and $\bar{d}_G^k(1) \ne 0$, then from Step 2.1 of Algorithm 2.1 we know that only the projected negative gradient is chosen as the search direction. Hence, Lemma 3.2 shows that this is a descent direction, which implies that after a finite number of iterations, (2.28) will be satisfied.

In order to establish the global convergence of Algorithm 2.1, we need the following lemma, which shows that Algorithm 2.1 can keep $t^k > 0$ at each iteration.

LEMMA 3.3. *Let $\{w^k\}$ be a sequence generated by Algorithm* 2.1. *Then for each k, $k = 0, 1, \ldots$, $w^k = (t^k, z^k)$ satisfies*

$$t^k \ge \beta_k\bar{t}. \qquad (3.4)$$

Furthermore, if w^k is not a stationary point of (2.21)*, then*

$$t^k > 0.$$

Proof. We prove this lemma by induction. From the choices of t^0 and β_0 in Algorithm 2.1, it is obvious that (3.4) holds for $k = 0$. Suppose that for any integer l, $w^l = (t^l, z^l)$ satisfies (3.4). Now we prove that $w^{l+1} = (t^{l+1}, z^{l+1})$ satisfies (3.4) as well.

If the condition (2.28) is not satisfied for $k = l$, we have

$$\bar{d}^l(\lambda_l) = \tilde{d}^l_G(\lambda_l) = \Pi_W(w^l + \lambda_l d^l_G) - w^l, \quad d^l_G = -\xi_l \nabla \Psi_h(w^l) + \beta_l \bar{w},$$

where λ_l is the accepted steplength at the lth iteration. It follows from Algorithm 2.1 that

$$\begin{aligned}(\bar{d}^l(\lambda_l))_t &= \lambda_l[-\xi_l(t^l + \nabla_t H(w) H_h(w)) + \beta(w^l)\bar{t}] \\ &\geq -\lambda_l t^l + \lambda_l \beta(w^l)\bar{t} \qquad (\text{ see (2.25) }),\end{aligned}$$

where $(\bar{d}^l(\lambda_l))_t$ is the first element of $\bar{d}^l(\lambda_l)$. Then we have

$$\begin{aligned}t^{l+1} - \beta(w^{l+1})\bar{t} &= t^l + (\bar{d}^l(\lambda_l))_t - \beta(w^{l+1})\bar{t} \\ &\geq (1 - \lambda_l)t^l + \lambda_l \beta(w^l)\bar{t} - \beta(w^{l+1})\bar{t} \\ &\geq (1 - \lambda_l)t^l + \lambda_l \beta(w^l)\bar{t} - \beta(w^l)\bar{t} \\ &= (1 - \lambda_l)(t^l - \beta(w^l)\bar{t}) \geq 0,\end{aligned}$$

where the second and third inequalities are due to the monotonicity property of $\beta(w^l)$ in Lemma 3.1 and $t^l \geq \beta(w^l)\bar{t}$.

If the condition (2.28) is satisfied for $k = l$, then we have

$$(\bar{d}^l(\lambda_l))_t = (\tau^*(\lambda_l)\bar{d}^l_G(\lambda_l) + (1 - \tau^*(\lambda_l))\tilde{d}^l_{tN}(\lambda_l))_t.$$

By a similar way, we can obtain that $t^{l+1} - \beta(w^{l+1})\bar{t} \geq 0$.

Therefore (3.4) holds for any nonnegative integer k. Furthermore, from (3.4) and the fact that w^k is not a stationary point of (2.21), $t^k > 0$ holds. We complete the proof. ∎

LEMMA 3.4. *Let $\{w^k\}$ be a sequence generated by Algorithm* 2.1*. Then any accumulation point of $\{w^k\}$ is a stationary point of* (2.21)*.*

Proof. Lemma 3.3 shows that if Algorithm 2.1 does not stop at a stationary point of (2.21), then $t^k > 0$ for any k. This implies that Ψ and Ψ_h are continuously differentiable at w_k. The remark after the proof of Lemma 3.2 implies that for k sufficiently large, the condition (2.28) is always satisfied and $\Psi_h(w) = \Psi(w)$ (see Remark (2) after Algorithm 2.1). Hence, by using a similar way to the proof of Theorem 4.1 in [18], we can prove that this theorem holds. ∎

In order to analyze the local convergence of Algorithm 2.1, we make the following standard assumption.

(A1) Let $w^* = (t^*, z^*) = (0, z^*)$ be an accumulation point of the sequence $\{w^k\}$ generated by Algorithm 2.1. Suppose $\lim_{k \in K} w_k = w^*$ for some subset $K \subset \{1, 2, \dots\}$, w^* is a solution of the system of equations (2.10), and Φ is BD-regular at w^* where the definition of BD-regularity refers to [13].

BD-regularity can be satisfied without special difficulty. Before giving a sufficient condition for BD-regularity to hold, we need the following assumptions:

(A2) The vectors $\nabla_x g(x, v^j), j = 1, \ldots, p$, are linearly independent.

(A3) The matrix $\nabla_x^2 L(x, u, V)$ is positive definite, and for every $j = 1, 2, \ldots, p$, the matrix $(\nabla_v^2 g(x, v^j))_M$ is negative definite whenever $J_M(x, v^j) \neq \emptyset$, where

$$J_M(x, v) = \{i \mid a_i < v_i + (\nabla_v g(x, v))_i < b_i\},$$

and $(\nabla_v^2 g(x, v^j))_M$ is a principal square submatrix of $\nabla_v^2 g(x, v)$, which is determined by the columns and rows with the index $i \in J_M(x, v)$.

(A4) For every $j = 1, 2, \ldots, p$, $\{i \mid v_i + (\nabla_v g(x, v))_i = a_i \text{ or } v_i + (\nabla_v g(x, v))_i = b_i\}$ is an empty set.

In addition, for any $(x, v) \in \Re^n \times \Re^m$, we denote

$$J_L(x, v) = \{i \mid v_i + (\nabla_v g(x, v))_i < a_i\}, \quad J_R(x, v) = \{i \mid b_i < v_i + (\nabla_v g(x, v))_i\}$$

and state a simple lemma without proof in the following.

LEMMA 3.5. *Let*

$$T = \begin{bmatrix} A & B & DC \\ B^T & 0 & 0 \\ D^T & 0 & F \end{bmatrix},$$

where $A \in \Re^{p \times p}$, $B \in \Re^{p \times q}$, $C \in \Re^{r \times r}$, $D \in \Re^{p \times r}$, and $F \in \Re^{r \times r}$. Suppose that A and $C^T F$ are positive definite and negative semidefinite, respectively. If the column rank of B and F are q and r, respectively, then T is nonsingular.

Proof. Let $Td = 0$, where $d = (d_1, d_2, d_3)$ is a suitable partitioned vector. Then

$$Ad_1 + Bd_2 + DCd_3 = 0, \tag{3.5}$$

$$B^T d_1 = 0, \tag{3.6}$$

$$D^T d_1 + Fd_3 = 0. \tag{3.7}$$

Multiplication (3.5) with d_1^T yields

$$d_1^T A d_1 + d_1^T B d_2 + d_1^T DC d_3 = 0,$$

which, together with (3.6) and (3.7), implies

$$d_1^T A d_1 + d_3^T (-C^T F) d_3 = 0.$$

From the property of A and $C^T F$, we have that $d_1 = 0$. Then it follows from (3.7) and the property of F that $d_3 = 0$. Because of (3.5) and the property of B, $d_2 = 0$ holds. The proof is complete. ∎

THEOREM 3.6. *Suppose that $w^* = (t^*, z^*) = (t^*, x^*, u^*, V^*, y^*)$ is a solution of* (2.10) *and satisfies* (A2)–(A4). *Then Φ is BD-regular at w^*.*

Proof. Without loss of generality, by (A4), we assume that

$$\begin{aligned} J_L(x^*, v^{j*}) &= \{1, 2, \ldots, k_1^j\}, \\ J_M(x^*, v^{j*}) &= \{k_1^j + 1, \ldots, k_2^j\}, \\ J_R(x^*, v^{j*}) &= \{k_2^j + 1, \ldots, m\}, \end{aligned}$$

where $1 \leq k_1^j \leq k_2^j \leq m$. Because $w^* = (t^*, z^*)$ is a solution of (2.10), $t^* = 0$. Moreover, we have, by $\phi(0, x^*, v^{j*}) = 0$, that

$$v^{j*} - \operatorname{mid}(a, b, v^{j*} + \nabla_v g(x^*, v^{j*})) = 0, \quad j = 1, \ldots, p. \tag{3.8}$$

By (3.8) and the definition of the mid function, we have that for $j = 1, \ldots, p$ and $i \in J_M(x^*, v^{j*})$,

$$(\nabla_{v^j} g(x^*, v^{j*}))_i = 0. \tag{3.9}$$

By direct computation, we obtain that for any $Q \in \partial_B \Phi(w^*)$,

(3.10)

$$Q = \begin{bmatrix} 1 & 0_{1\times n} & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 0 \\ 0_{n\times 1} & \nabla_x^2 L(x^*, u^*, V^*) & \nabla_x \mathbf{g}^T(x^*, V^*) & u_1^* D_1 & \cdots & u_p^* D_p & 0_{n\times 1} \\ 0 & \nabla_x^T g(x^*, v^{1*}) & 0_{1\times p} & \nabla_{v^1}^T g(x^*, v^{1*}) & \cdots & 0_{1\times m} & 0 \\ 0 & \nabla_x^T g(x^*, v^{2*}) & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \nabla_x^T g(x^*, v^{p*}) & 0_{1\times p} & 0_{1\times m} & \cdots & \nabla_{v^p}^T g(x^*, v^{p*}) & 0 \\ Q_1 & C_1 D_1^T & 0_{m\times p} & E_1 + C_1 F_1 & \cdots & 0_{m\times m} & 0_{p\times 1} \\ Q_2 & C_2 D_2^T & 0_{m\times p} & 0_{m\times m} & \cdots & 0_{m\times m} & 0_{p\times 1} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ Q_p & C_p D_p^T & 0_{m\times p} & 0_{m\times m} & \cdots & E_p + C_p F_p & 0_{p\times 1} \\ U_1 & U_2 & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} & 1 \end{bmatrix},$$

where $U_1 \in \partial_t G(0, x^*)$, $U_2 \in \partial_x G(0, x^*)$, and for $j = 1, \ldots, p$,

$$Q_j \in \partial_t \phi(0, x^*, v^{j*}),\ D_j = \nabla_{v^j}^T (\nabla_x g(x^*, v^{j*})),\ F_j = \nabla_{v^j}^T (\nabla_{v^j} g(x^*, v^{j*})),$$

$$C_j = \operatorname{diag}(0_{j_1}, -I_{j_2}, 0_{j_3}), \quad E_j = \operatorname{diag}(I_{j_1}, 0_{j_2}, I_{j_3}), \tag{3.11}$$

where $0_{j_1}, 0_{j_2}$, and 0_{j_3} are zero square matrices with k_1^j, $(k_2^j - k_1^j)$, and $(m - k_2^j)$ order, respectively, and I_{j_1}, I_{j_2}, and I_{j_3} are identity matrices with k_1^j, $(k_2^j - k_1^j)$, and $(m - k_2^j)$ order, respectively. By (3.10), it is easy to see that the matrix Q is also nonsingular as the matrix

$$\tilde{Q} = \begin{bmatrix} \nabla_x^2 L(x^*, u^*, V^*) & \nabla_x \mathbf{g}^T(x^*, V^*) & u_1^* D_1 & \cdots & u_p^* D_p \\ \nabla_x^T g(x^*, v^{1*}) & 0_{1\times p} & \nabla_{v^1}^T g(x^*, v^{1*}) & \cdots & 0_{1\times m} \\ \nabla_x^T g(x^*, v^{2*}) & 0_{1\times p} & 0_{1\times m} & \cdots & 0_{1\times m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \nabla_x^T g(x^*, v^{p*}) & 0_{1\times p} & 0_{1\times m} & \cdots & \nabla_{v^p}^T g(x^*, v^{p*}) \\ C_1 D_1^T & 0_{m\times p} & E_1 + C_1 F_1 & \cdots & 0_{m\times m} \\ C_2 D_2^T & 0_{m\times p} & 0_{m\times m} & \cdots & 0_{m\times m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_p D_p^T & 0_{m\times p} & 0_{m\times m} & \cdots & E_p + C_p F_p \end{bmatrix}.$$

We denote by $(D_j)_{ML}$ a submatrix of D_j constituted by the columns with the index $i \in J_M(x, v^j)$ and by $(F_j)_M$ a principal square submatrix of F_j, which is determined by the columns and rows with the index $i \in J_M(x, v^j)$. Then from special forms of C_j and E_j we have

$$C_j D_j^T = \begin{bmatrix} 0 \\ -(D_j)_{ML}^T \\ 0 \end{bmatrix}, \quad E_j + C_j F_j = \begin{bmatrix} I_{j_1} & 0 & 0 \\ * & -(F_j)_M & * \\ 0 & 0 & I_{j_2} \end{bmatrix},$$

where two $*$ are some proper partitioned matrices. Hence the nonzero elements of $\nabla_{v^j}^T g(x^*, v^{j*})$ and the matrix $*$ are deleted by the some proper row transformations. Hence it is not difficult to know that the matrix $\tilde{Q}$ is also nonsingular as the matrix

$$Q^* = \begin{bmatrix} \nabla_x^2 L(x^*, u^*, V^*) & \nabla_x \mathbf{g}^T(x^*, V^*) & DU \\ \nabla_x^T \mathbf{g}(x^*, V^*) & 0 & 0 \\ D^T & 0 & F \end{bmatrix}, \tag{3.12}$$

where $D = ((D_1)_{ML}, \ldots, (D_p)_{ML})$, $F = \mathrm{diag}((F_1)_M, \ldots, (F_p)_M)$, $U = \mathrm{diag}(u_1^* I_1, \ldots, u_p^* I_p)$, and I_j, $j = 1, \ldots, p$, are some proper identity matrices. It is clear that $U^T F$ is negative definite, and from (A2) and (A3) it follows that all other conditions in Lemma 3.5 are satisfied. Hence from Lemma 3.5 we know that Q^* is nonsingular and we complete the proof. □

The following lemma is the same as Lemma 4.1 in [14]; its proof is omitted.

LEMMA 3.7. *There exist positive constants κ and ϵ such that for every w_k satisfying $\|w_k - w^*\| \le \epsilon$,*

(i) *$\nabla^T \Phi(w_k)$ is nonsingular and satisfies*

$$\|\nabla^T \Phi(w_k)\| \le \kappa,$$

(ii)

$$\|\Phi(w_k)\| = \sqrt{2}\Psi(w_k)^{\frac{1}{2}} = O(\|w_k - w^*\|).$$

LEMMA 3.8. *Let $\{w^k\}$ be a sequence generated by Algorithm 2.1. Then for all $k \in K$ sufficiently large, we have*

$$\beta(w^k) = O(\Psi(w^k)) = O(\|w^k - w^*\|^2); \tag{3.13}$$

and

$$w^k + \lambda d_{tN}^k = (1 - \lambda) w^k + \lambda w^* + \lambda o(\Psi(w^k)^{\frac{1}{2}}) \tag{3.14}$$

for any $\lambda \in (0, 1]$.

Proof. From the definition of $\beta(w_k)$ (see (2.24)), the choice of γ_k (see (2.26)), the projection property, and Lemma 3.7, it follows that for w^k sufficiently close to w^*, $\Psi_h(w_k) = \Psi(w_k)$,

$$\beta(w^k) \le \alpha \|\bar{d}_G^k(1)\|^2 \le \alpha \gamma_k^2 \|\nabla \Psi(w^k)\|^2 \le \alpha \eta \Psi(w^k) = \frac{\alpha \eta}{2} \|\Phi(w^k)\|^2 = O(\|w^k - w^*\|^2).$$

This shows that (i) holds. Let

$$\theta_k = \Phi(w_k) - \beta_k \bar{w} + \nabla^T \Phi(w_k) d_{tN}^k.$$

Then from (2.29), we have that for w^k sufficiently close to w^*,

$$\|\theta_k\| = o(\Psi_h(w_k)) = o(\Psi(w_k)), \tag{3.15}$$

which implies that

$$\begin{aligned} w^k + \lambda d_{tN}^k &= w^k + \lambda \nabla^T \Phi(w^k)^{-1} [-\Phi(w^k) + \beta(w^k)\bar{w} + \theta_k] \\ &= w^k - \lambda \nabla^T \Phi(w^k)^{-1} [\Phi(w^k) - \Phi(w^*) - \nabla^T \Phi(w^k)(w^k - w^*)] \\ &\quad -\lambda(w^k - w^*) + \lambda \nabla^T \Phi(w^k)^{-1} (\beta(w^k)\bar{w} + \theta_k) \\ &= (1 - \lambda) w^k + \lambda w^* + \lambda o(\|w^k - w^*\|) + \lambda o(\Psi(w^k)) \\ &= (1 - \lambda) w^k + \lambda w^* + \lambda o(\Psi(w^k)^{\frac{1}{2}}), \end{aligned}$$

where the third equality is due to the semismoothness of Φ, (i), and (3.15). The proof is complete. ∎

Now we obtain the following convergence theorem.

THEOREM 3.9. *Suppose that $\{w^k\}$ is a sequence generated by Algorithm* 2.1 *and w^* is a point satisfying* (A1). *Then the whole sequence $\{w^k\}$ superlinearly converges to w^*.*

Proof. At first we know that for w^k sufficiently close to w^*, $\Psi_h(w_k) = \Psi(w_k)$.

In a similar way to the proof of Theorem 3.2 in [18] and Lemma 3.8, we have that for sufficiently large $k \in K$,

$$\|w^k + \bar{d}^k(1) - w^*\| = o(\Psi(w^k)^{\frac{1}{2}}) = o(\|\Phi(w^k)\|) = o(\|w^k - w^*\|), \tag{3.16}$$

and

$$\begin{aligned} \Psi(w^k + \bar{d}^k(1)) &= \frac{1}{2}\|\Phi(w^k + \bar{d}^k(1))\|^2 \\ &= \frac{1}{2}\|\Phi(w^k + \bar{d}^k(1)) - \Phi(w^*)\|^2 \\ &= O(\|w^k + \bar{d}^k(1) - w^*\|^2) \\ &= o(\Psi(w^k)), \end{aligned} \tag{3.17}$$

where the last equality is due to (3.16). Thus,

$$\begin{aligned} -\nabla\Psi(w^k)^T \tilde{d}_G^k(1) &\le \|\nabla\Psi(w^k)\|\|\tilde{d}_G^k(1)\| \\ &= \|\nabla\Psi(w^k)\|\|\Pi_W(w^k - \gamma_k\nabla\Psi(w^k) + \beta(w^k)\bar{w}) - w^k\| \\ &\le \|\nabla\Psi(w^k)\|[\|\gamma_k\nabla\Psi(w^k)\| + O(\Psi(w^k))] \\ &\le \eta\Psi(w^k) + o(\Psi(w^k)), \end{aligned} \tag{3.18}$$

where the second inequality is due to the property of $\beta(w^k)$ and the projection property, and the last inequality comes from the choice of γ_k. It follows from (3.17) and (3.18) that

$$\begin{aligned} \Psi(w^k) + \sigma\nabla\Psi(w^k)^T \tilde{d}_G^k(1) &\ge (1 - \sigma\eta)\Psi(w^k) + o(\Psi(w^k)) \\ &\ge o(\Psi(w^k)) = \Psi(w^k + \bar{d}^k(1)), \end{aligned} \tag{3.19}$$

which implies that

$$w^{k+1} = w^k + \bar{d}^k(1),$$

for k sufficiently large. Moreover, from (3.16) we conclude that w^k converges to w^* superlinearly. We complete the proof. ∎

4. Implementation and numerical tests. In this section, we discuss some detailed implementation of Algorithm 2.1 and give some numerical results for medium-sized and large-scale SIP problems.

4.1. Implementation of Algorithm 2.1. In order to decrease the number of inner iterations, we use another line search technique if only the projected gradient direction is the search direction. In this case, the initial value of λ is set to

$$\min\left\{1, \frac{1}{\|d_G^k\|}, \frac{0.2\Psi_h(w_k)}{-\nabla\Psi_h(w_k)^T d_G^k}, \frac{t_k}{|t_k + \nabla_t H(w_k)H_h(w_k)|}\right\},$$

and λ is updated by quadratic interpolation technique.

In Algorithm 2.1, we choose the values of parameters (see Step 0) as

$$\eta = 0.9,\ \rho = 0.5,\ \sigma = 0.0005,\ \alpha = 0.5,\ \bar{t} = 0.9,\ p_1 = 10^{-10},\ p_2 = 2.1,$$

and

$$h_j = \max\{2.5, 10^{-3} * |\Psi(w^0)|,\ j = 1, 2, \ldots, \tilde{n}\},$$

where the choice of h_j, $j = 1, 2, \ldots, \tilde{n}$, is similar to that in [9]. The starting points u^0 and y^0 for all problems are set to $t^0 = \bar{t}$, $u^0 = 0.05e$, $y^0 = 0.5$, where e is the vector of ones. For GMRES($\tilde{m}$), we choose $\tilde{m} = 10$ when $n < 100$ and $\tilde{m} = 20$ when $n \geq 100$.

In some test problems, we have tried using a simple left preconditioning matrix

$$M = \mathrm{diag}(1, L_{ssor}, I_{(m+1)p+1}),$$

where L_{ssor} is an SSOR preconditioning matrix defined by

$$L_{ssor} = (D - \omega E)D^{-1}(D - \omega F),$$

D is the diagonal part of $\nabla^2_{xx}L(x, u, V)$, and $-F$ and $-E$ are the strict upper and lower parts of $\nabla^2_{xx}L(x, u, V)$. Numerical results show that GMRES without preconditioning is better than that with preconditioning for $\omega = 1$ and $\omega = 0.5$. Hence we give the numerical results without preconditioning in the following.

4.2. Numerical results. Now we discuss the implementation of Algorithm 2.1, which has been implemented in FORTRAN 77. All calculation within the driving programs, test problems, and optimization code are carried out in double precision. The problem is solved on a personal computer (Pentium III 1133 MHz, 256 MB memory).

Although a lot of large SIP-type problems arise from optimal control and approximation theory, it is difficult to find large-scale SIP problems in the literature suitable for using as test problems. In order to evaluate Algorithm 2.1 for large-scale SIP problems, we enlarge three test problems, where two problems are from [14] and [8] and another is generated from an optimal control problem. We list the three SIP problems in the following.

Problem I.

$$f(x) = \frac{1}{2}x^T x, \quad g(x, v) = 3 + 4.5 \sin\left(\frac{4.7\pi(v - 1.23)}{8}\right) - \sum_{i=1}^{n} x_i v^{i-1},$$

$V = [1, b], p = 1$ if $n \leq 60, b = 100$; otherwise $b = 1$.

Problem II.

$$f(x) = \int_0^1 \left(\sum_{i=1}^{n} x_i t^{i-1} - \tan t\right)^2 dt,\ g(x, v) = \tan v - \sum_{i=1}^{n} x_i v^{i-1},\ V = [0, 1], p = 1.$$

Problem III.

$$\begin{aligned} \min\ & p(g)h^T h \\ \text{subject to}\ & g^T A(v_1, v_2)h \leq r(v_1, v_2), \end{aligned}$$

TABLE 1
Test results of Problem I.

n	ITK	iITK	$\|\bar{d}_G^k(1)\|$	$\Psi(w_k)$	$f(x_k)$
10	8	65	5.52E-12	1.60E-19	0.07412
20	8	50	4.96E-12	4.29E-19	0.08319
40	67	393	9.86E-7	2.06E-11	3.1788
60	98	489	8.84E-7	2.91E-11	4.8860
100	60	286	9.66E-7	5.10E-8	2.3862
400	67	580	2.70E-7	3.58E-13	4.580
1000	78	630	8.29E-6	2.97E-10	8.519
2000	52	603	6.96E-6	6.69E-7	18.07

where $v_1 \in [-\pi, \pi]$, $v_2 \in [0, 2\pi]$, $p(g) = g^T B g$, $h \in \Re^{n_1}$, $g \in \Re^{n_2}$, $B \in \Re^{n_2 \times n_2}$; $A(v_1, v_2) \in \Re^{n_2 \times n_1}$, $n_2 = n_1$, and

$$B = \begin{bmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{bmatrix},$$

$$A(v_1, v_2) = \begin{bmatrix} 1 & \sin(bv_2) & \cos(cv_1) & & & & & & & \\ \sin(av_1) & 1 & \sin(bv_2) & \cos(cv_1) & & & & & & \\ \cos(dv_2) & \sin(av_1) & 1 & \sin(bv_2) & \cos(cv_1) & & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & & \cos(dv_2) & \sin(av_1) & 1 & \sin(bv_2) & \cos(cv_1) \\ & & & & & \cos(dv_2) & \sin(av_1) & 1 & \sin(bv_2) \\ & & & & & & \cos(dv_2) & \sin(av_1) & 1 \end{bmatrix}.$$

We use Algorithm 2.1 to solve these problems. The dimensions (n) of these problems are chosen by 10, 20, 40, 60, 80, 100, 200, 400, 1000, and 2000. The termination condition is that the l_2 norm of the projected gradient, $\|\bar{d}_G^k(1)\|$, is reduced below 10^{-6} when $n < 100$ (10^{-5} when $n \geq 100$). The results of the test are given in Tables 1, 2, and 3. The number of outer iterations (ITK), the total number of inner iterations for solving subproblems (iITK), the norm of projected gradient ($\|\bar{d}_G^k(1)\|$), the merit function value $\Psi(w_k)$, and the objective function value $f(x_k)$ are shown in these tables.

Table 1 shows that Algorithm 2.1 performs very well for solving Problem I with the different dimensions. There is some difference among different dimensions. When $n \geq 40$, there is a slight increase in the iteration number.

Problem II is dense; i.e., its Hessian of Lagrangian function $\nabla_x^2 L(x, u, V)$ is not sparse. Although the Hessian can be stored according to its special structure, the computation in each iteration cannot be decreased. Here Algorithm 2.1 is used for solving Problem II, whose dimensions range from 10 to 200. Table 2 shows that Algorithm 2.1 performs well for solving some medium dense SIP problems.

TABLE 2
Test results of Problem II.

n	ITK	iITK	$\|\bar{d}_G^k(1)\|$	$\Psi(w_k)$	$f(x_k)$
10	31	140	1.14E-7	4.86E-12	0.3147
20	46	235	9.68E-7	7.98E-10	0.6717
40	50	306	1.48E-7	3.48E-12	0. 5803
80	65	476	4.48E-7	3.93E-11	1.424
100	58	528	2.87E-7	1.86E-11	1.069
200	71	768	2.86E-6	1.46E-9	1.323

TABLE 3
Test results of Problem III.

n	ITK	iITK	$\|\bar{d}_G^k(1)\|$	$\Psi(w_k)$	$f(x_k)$
20	26	694	9.04E-7	7.80E-12	18.23
60	28	973	5.72E-7	3.03E-12	21.82
100	27	963	9.63E-6	8.98E-12	20.36
200	24	605	6.97E-6	4.56E-10	16.84
600	20	509	9.21E-6	7.75E-10	13.72
1000	25	494	9.32E-6	7.59E-10	13.82
2000	22	488	8.39E-6	8.06E-10	13.81

Problem III is a somewhat complicated SIP problem which often arises from the optimal control field. In this problem, $\Omega \subset \Re^2$, while in Problems I and II, $\Omega \subset \Re$. Its Hessian of the Lagrangian function is sparse; however, the computation of elements is not simple due to some trigonometric functions. Numerical results of this problem are given in Table 3, which shows that Algorithm 2.1 can solve some large-scale sparse SIP problems. It is interesting that the outer iteration number does not increase and inner iteration numbers decrease as the dimensions increase.

5. Comments. Although the development of the code for Algorithm 2.1 is still at its primary stage, the numerical results have indicated that Algorithm 2.1 is capable of processing large-scale SIP problems. However, there are some issues which may be addressed in further research.

Because "large scale" here refers only to the decision variables, it is hoped that an improved version of Algorithm 2.1 may also be capable of handling high-dimensional index sets. In addition, our method works on the KKT system of SIP; i.e., it does not minimize the original objective function f. Sometimes this may limit the applicability of this method to a special class of SIP problems.

By Algorithm 2.1 we can obtain stationary points of (2.21). It is possible that some of them may not be stationary points of (1.1). If Ω in (1.1) is a nonpolyhedral index set, then our method cannot be used directly.

We hope that with further research more efficient methods can be obtained for solving general SIP problem with many decision variables and high-dimensional index sets.

Acknowledgments. We are grateful to Professor Fukushima and two anonymous referees for their detailed comments. The approach for solving nonsymmetric linear system suggested by a referee improved our method.

REFERENCES

[1] P. H. Calamai and J. J. Moré, *Projected gradient methods for linear constrained problems*, Math. Programming, 39 (1987), pp. 93–116.

[2] J. E. Dennis, *Nonlinear least squares and equations*, in The State of the Art in Numerical Analysis, D. A. H. Jacobs, ed., Academic Press, New York, 1975, pp. 269–312.

[3] V. Fraysse, L. Giraud, S. Gratton, and J. Langou, *A Set of GMRES Routines for Real and Complex Arithmetics on High Performance Computers*, CERFACS Technical report TR/PA/03/3, Toulouse Cedex, France, 2003.

[4] M. A. Goberna and M. A. López, *Semi-infinite Programming: Recent Advances*, Kluwer Academic Publishers, Boston, 2001.

[5] P. R. Gribik, *Selected applications of semi-infinite programming*, in Constructive Approaches to Mathematical Models, Academic Press, New York, 1979, pp. 171–187.

[6] R. Hettich and K. O. Kortanek, *Semi-infinite programming: Theory, methods, and applications*, SIAM Rev., 35 (1993), pp. 380–429.

[7] P. J. Huber, *Robust regression: Asympotics, conjectures, and Monte Carlo*, Ann. Statist., 1 (1973), pp. 799–821.

[8] S. Ito, Y. Liu, and K. L. Teo, *A dual parametrization method for convex semi-infinite programming*, Ann. Oper. Res., 98 (2000), pp. 189–214.

[9] Q. Ni, *Global convergence and implementation of NGTN method for solving large scale nonlinear sparse nonlinear programming problems*, J. Comput. Math., 19 (2001), pp. 337–346.

[10] J.-S. Pang and L. Qi, *Nonsmooth equations: Motivation and algorithms*, SIAM J. Optim., 3 (1993), pp. 443–465.

[11] E. Polak, *On the mathematical foundations of nondifferentiable optimization in engineering design*, SIAM Rev., 29 (1987), pp. 21–89.

[12] E. Polak, *Optimization: Algorithms and Consistent Approximation*, Springer-Verlag, New York, 1997.

[13] L. Qi, *Convergence analysis of some algorithms for solving nonsmooth equations*, Math. Oper. Res., 18 (1993), pp. 227–244.

[14] L. Qi, C. Ling, X. J. Tong, and G. L. Zhou, *A Smoothing Projected Newton-Type Algorithm for Semi-infinite Programming*, Technical report, Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hum, Kowloon, Hong Kong, 2004.

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

[16] E. W. Sachs, *Semi-infinite programming in control*, in Semi-infinite Programming, R. Reemtsen and J. Rükmann, eds., Kluwer Academic Publishers, Boston, 1998, pp. 389–411.

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

[18] D. Sun, R. S. Womersley, and H. Qi, *A feasible semismooth asymptotically Newton method for mixed complementarity problems*, Math. Program., 94 (2002), pp. 167–187.

[19] K. L. Teo, C. J. Goh, and K. H. Wong, *A Unified Computational Approach to Optimal Control Problems*, Longman Scientific and Technical, New York, 1991.

[20] K. L. Teo, V. Rehbock, and L. S. Jennings, *A new computational algorithm for functional inequality constrained optimization problems*, Automatica, 29 (1993), pp. 789–792.