A Generalized Reduced Gradient Approach for Solving a Class of Two-Stage Stochastic Nonlinear Programs

Fatmah Syarah¹, Herman Mawengkang², Anton Abdulbasah Kamil³, Sutarman²

¹ Graduate School of Mathematics, Universitas Sumatera Utara, Medan - Indonesia
² Department of Mathematics, Universitas Sumatera Utara, Medan - Indonesia
³ University of Telkom, Bandung, Indonesia

*hmawengkang@yahoo.com

Abstract. Two-stage stochastic programming problems arise in many practical situations, such as production and manpower planning, portfolio selections and so on. In general, the deterministic equivalences of these problems can be very large, and may not be solvable directly by general-purpose optimization approaches. Reduced gradient method (RGM) is a well known technique for nonlinear programming problems. After using scenario analysis technique, a direct search approach based on RGM for solving a class of two-stage stochastic nonlinear programs is proposed, which generates the search direction by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. By selecting the step-size to reduce an exact penalty function sufficiently, the algorithm terminates at an approximate optimal solution to the problem with any desirable accuracy.

1. Introducing

The stochastic nonlinear programming (SNLP) problems represent an important class of the optimization problems due to their omnipresence in real life situations. Many systems in nature are inherently nonlinear, necessitating nonlinear models for their representation, and consequently, nonlinear programming methods for optimization. Another important factor for consideration is uncertainty. Very rarely are the system details accurately known. Quite often, the parameters and variables are known only in terms of their ranges or, in some cases, in terms of their probability distributions. In such cases, the stochastic programming methods need to be resorted to for optimization. Two-stage stochastic programming problems can be found in many practical situations, such as production and manpower planning, portfolio selections and so on. Consider the following two-stage stochastic program with recourse:

\[
\min_{x \in X} \hat{c}_0(x) + E_{\xi_1} Q_1(x, \xi_1)
\]

where \( X = \{ x \mid c_0(x) = 0 \} \subseteq \mathbb{R}^n \), the recourse function

\[
Q_1(x, \xi_1) = \min_{y_1} q_1(x, y_1, \hat{\xi}_1) + E_{\xi_2} Q_2(x, y_1, \hat{\xi}_1, \xi_2)
\]

subject to \( c_1(x, y_1, \hat{\xi}_1) = 0 \)

and for \( t = 2, \ldots, T-1 \), recursively we have

\[
Q_t(x, y_1, \ldots, y_{t-1}, \hat{\xi}_1, \ldots, \hat{\xi}_{t-1}) = \min_{y_t} q_t(x, y_1, \ldots, y_{t-1}, y_t, \hat{\xi}_1, \ldots, \hat{\xi}_{t-1}, \hat{\xi}_t) + E_{\xi_{t+1}} Q_{t+1}(x, y_1, \ldots, y_t, \hat{\xi}_1, \ldots, \hat{\xi}_t, \hat{\xi}_{t+1})
\]
subject to \( c_i(x, y_1, ..., y_{t-1}, \xi_i, ..., \xi_T) = 0 \) \hspace{1cm} (4)

\( Q_T = 0 \). \( x \in \mathbb{R}^n \) is the deterministic vector, \( \xi_i \) is the realization of the random vector \( \xi_i \). \( y_i \in \mathbb{R}^n \) is the decision vector in the \( i \)-th stage, which is generated recursively by \( x, y_j, ..., y_{t-1} \) and \( \xi_{s1}, ..., \xi_{sT} \), hence represents \( y_i(x, y_1, ..., y_{t-1}, \xi_{s1}, ..., \xi_{sT}) \) actually.

\( \hat{c}_o \) and \( c_i \) are real-valued functions on \( \mathbb{R}^n \). \( c_i \) is random since it is related to \( \xi_{s1}, ..., \xi_{sT} \).

For the discrete random vector \( \xi = (\xi_1, ..., \xi_{T-1}) \) if \( c_i \) has finite realizations \( c_i(i = 1, ..., S_i) \), then all these \( c_i \) form the constraint functions on stage \( t \). The details on the formulation of multi-stage stochastic programs can be found, e.g. in [1].

Scenario analysis is an important factor when dealing with two-stage stochastic programs [2], which species the program into a finite number of scenarios for the considered time period. Let \( \xi = (\xi_1, ..., \xi_{T-1}) \) and assume that \((\Omega, \theta, P)\) is the associated probability space. Suppose that we have \( S \) scenarios

\[ \xi^{(s)} = (\xi^{(s)}_1, ..., \xi^{(s)}_{T-1}) \]

which has a fixed and known probability distribution

\[ \{(\xi^{(s)}, p_s) | s = 1, 2, ..., S\} \] . Then (1)-(4) can be reformulated as the following nonlinear programming problem:

\[
\min \sum_{s=1}^{S} f_s(z^{(s)})
\]

s.t. \( h_s(z^{(s)}) = 0, s = 1, 2, ..., S \)

\[ \sum_{s=1}^{S} A_s(z^{(s)}) = 0, \]

where \( z^{(s)} = (x^{(s)}, y_1^{(s)}, ..., y_{T-1}^{(s)}) \in \mathbb{R}^n, n = \sum_{i=0}^{T-1} n_i \)

\[ f_s(z^{(s)}) = p_s(c_0(x^{(s)}) + \sum_{t=1}^{T-1} q_t(x^{(s)}, y_1^{(s)}, ..., y_t^{(s)}, \xi^{(s)}_1, ..., \xi^{(s)}_t)) \]

Can be derived by letting

\[
h_s(z^{(s)}) = (c_0(x^{(s)}); c_1(x^{(s)}, y_1^{(s)}, \xi^{(s)}_1); ...; c_{T-1}(x^{(s)}, y_{T-1}^{(s)}, \xi_1^{(s)}, ..., \xi_{T-1}^{(s)}))
\]

Constraints (7) are the so-called nonanticipativity constraints, which reflect the fact that scenarios sharing a common history up to any moment of time must have a common decision up to that moment. Readers can refer to [2] for more details on this reformulation.

In this paper, we consider to solve (5)-(7). It is assumed that \( f_s : \mathbb{R}^n \rightarrow \mathbb{R} \) and \( h_s : \mathbb{R}^n \rightarrow \mathbb{R}^m \) are twice continuously differentiable functions,

\[ z^{(s)} \in \mathbb{R}^n, h_s : \mathbb{R}^n \rightarrow \mathbb{R} \]

and \( h_s(z^{(s)}) = (h_{s1}(z^{(s)}), ..., h_{sm}(z^{(s)}))^T \).

\( A_s \in \mathbb{R}^{m \times n}, A = (A_1, A_2, ..., A_S) \in \mathbb{R}^{m \times n \times S} \) is a full row rank matrix and has a special structure, which is identified for the concrete examples in section 5.

When the scenario number \( S \) is large, program (5)-(7) can be very large, and may not be solvable directly by general-purpose optimization approaches. Thus, the development of techniques of decomposition has been taking as one of the important choices for solving the stochastic programming, see e.g. in [3]. Moreover, the parallelization of computers provides the feasibility for implementing the decomposition methods.

There are many references contributed to the decomposition methods in linear and nonlinear programming in literature, e.g. see in [4], [5], [6], [7] et al. Most of them are related to the well-known decomposition principle of [8], and to the duality theory based on the Lagrangian and augmented Lagrangian function.
The L-shaped decomposition method is efficient for solving multi-stage stochastic linear programs. In each cycle, sets of feasible cuts and optimal cuts are generated recursively, a sequence of decreasing feasible regions is derived. Some other methods for multi-stage stochastic linear programs can be found, e.g. in [9], [10] and their references. More recently, in [11], [12] proposed the logarithmic barrier methods for solving multi-stage stochastic linear programs. Since all these methods are based on the special structures and properties of stochastic linear programs, it is difficult to generalize them to solve the stochastic nonlinear programs.

Based on scenario analysis technique, in [2] proposed the progressive hedging method (PHA for short) for multi-stage stochastic programming, which is an iterative algorithm. In [13] applied the progressive hedging method to the stochastic generalized networks, and has achieved satisfactory numerical results. The additional works on PHA include in [14] and [15]. One of difficulties in implementing PHA is the selection of a suitable penalty parameter. In [14] showed that PHA is not the best candidate for the loosely-coupled scenario analysis problems, and the bundle-based decomposition method in [16] is more competitive than PHA. A new iterative method based on scenario analysis is proposed recently by [17], which relaxes the nonanticipativity constraints by the Lagrangian dual approach and combines with the logarithmic barrier methods. The implementation on this method is in progress.

Sequential quadratic programming (SQP for short) is an iterative method, and very effective for solving medium-size nonlinear programming, e.g. see in [18], [19], [20], and [21]. Recently, it has been applied to solve the complementarity problems, the variational inequality problems and the nonsmooth problems, for example, see in [22], [23], [24] and [25]. In this paper, SQP is applied to program (1.5)-(1.7), by combining with the Lagrangian dual approach, we present a decomposition method based on SQP, conjugate gradient methods can be introduced to derive the estimate of the dual multiplier associated with the nonanticipativity constraints (1.7), the search direction is generated by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. The global convergence of the algorithm is analyzed. The algorithm is also used to solve some stochastic nonlinear programs, and the preliminary numerical results are reported. Our method can be taken as one of examples of sequential quadratic programming in application to solving large-scale structural nonlinear optimization.

Although all notations can be identified easily from the context, we still list some of notations used in the paper for readers convenience: A letter with the superscript (s) for multi-stage stochastic programming in application to (1.5)

\[
\text{tucker equations of optimization problems.}
\]

programming problems. It is a

Sequential quadratic programming (SQP for short) is an iterative method, and very effective for solving medium-size nonlinear programming, e.g. see in [18], [19], [20], and [21]. Recently, it has been applied to solve the complementarity problems, the variational inequality problems and the nonsmooth problems, for example, see in [22], [23], [24] and [25]. In this paper, SQP is applied to program (1.5)-(1.7), by combining with the Lagrangian dual approach, we present a decomposition method based on SQP, conjugate gradient methods can be introduced to derive the estimate of the dual multiplier associated with the nonanticipativity constraints (1.7), the search direction is generated by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. The global convergence of the algorithm is analyzed. The algorithm is also used to solve some stochastic nonlinear programs, and the preliminary numerical results are reported. Our method can be taken as one of examples of sequential quadratic programming in application to solving large-scale structural nonlinear optimization.

Although all notations can be identified easily from the context, we still list some of notations used in the paper for readers convenience: A letter with the superscript (s) for multi-stage stochastic programming in application to (1.5)

\[
\text{tucker equations of optimization problems.}
\]

programming problems. It is a

Sequential quadratic programming (SQP for short) is an iterative method, and very effective for solving medium-size nonlinear programming, e.g. see in [18], [19], [20], and [21]. Recently, it has been applied to solve the complementarity problems, the variational inequality problems and the nonsmooth problems, for example, see in [22], [23], [24] and [25]. In this paper, SQP is applied to program (1.5)-(1.7), by combining with the Lagrangian dual approach, we present a decomposition method based on SQP, conjugate gradient methods can be introduced to derive the estimate of the dual multiplier associated with the nonanticipativity constraints (1.7), the search direction is generated by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. The global convergence of the algorithm is analyzed. The algorithm is also used to solve some stochastic nonlinear programs, and the preliminary numerical results are reported. Our method can be taken as one of examples of sequential quadratic programming in application to solving large-scale structural nonlinear optimization.

Although all notations can be identified easily from the context, we still list some of notations used in the paper for readers convenience: A letter with the superscript (s) for multi-stage stochastic programming in application to (1.5)

\[
\text{tucker equations of optimization problems.}
\]

programming problems. It is a

Sequential quadratic programming (SQP for short) is an iterative method, and very effective for solving medium-size nonlinear programming, e.g. see in [18], [19], [20], and [21]. Recently, it has been applied to solve the complementarity problems, the variational inequality problems and the nonsmooth problems, for example, see in [22], [23], [24] and [25]. In this paper, SQP is applied to program (1.5)-(1.7), by combining with the Lagrangian dual approach, we present a decomposition method based on SQP, conjugate gradient methods can be introduced to derive the estimate of the dual multiplier associated with the nonanticipativity constraints (1.7), the search direction is generated by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. The global convergence of the algorithm is analyzed. The algorithm is also used to solve some stochastic nonlinear programs, and the preliminary numerical results are reported. Our method can be taken as one of examples of sequential quadratic programming in application to solving large-scale structural nonlinear optimization.

Although all notations can be identified easily from the context, we still list some of notations used in the paper for readers convenience: A letter with the superscript (s) for multi-stage stochastic programming in application to (1.5)

\[
\text{tucker equations of optimization problems.}
\]

programming problems. It is a

Sequential quadratic programming (SQP for short) is an iterative method, and very effective for solving medium-size nonlinear programming, e.g. see in [18], [19], [20], and [21]. Recently, it has been applied to solve the complementarity problems, the variational inequality problems and the nonsmooth problems, for example, see in [22], [23], [24] and [25]. In this paper, SQP is applied to program (1.5)-(1.7), by combining with the Lagrangian dual approach, we present a decomposition method based on SQP, conjugate gradient methods can be introduced to derive the estimate of the dual multiplier associated with the nonanticipativity constraints (1.7), the search direction is generated by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. The global convergence of the algorithm is analyzed. The algorithm is also used to solve some stochastic nonlinear programs, and the preliminary numerical results are reported. Our method can be taken as one of examples of sequential quadratic programming in application to solving large-scale structural nonlinear optimization.

Although all notations can be identified easily from the context, we still list some of notations used in the paper for readers convenience: A letter with the superscript (s) for multi-stage stochastic programming in application to (1.5)

\[
\text{tucker equations of optimization problems.}
\]

programming problems. It is a

Sequential quadratic programming (SQP for short) is an iterative method, and very effective for solving medium-size nonlinear programming, e.g. see in [18], [19], [20], and [21]. Recently, it has been applied to solve the complementarity problems, the variational inequality problems and the nonsmooth problems, for example, see in [22], [23], [24] and [25]. In this paper, SQP is applied to program (1.5)-(1.7), by combining with the Lagrangian dual approach, we present a decomposition method based on SQP, conjugate gradient methods can be introduced to derive the estimate of the dual multiplier associated with the nonanticipativity constraints (1.7), the search direction is generated by solving parallelly a set of quadratic programming subproblems with size much less than the original problem at each iteration. The global convergence of the algorithm is analyzed. The algorithm is also used to solve some stochastic nonlinear programs, and the preliminary numerical results are reported. Our method can be taken as one of examples of sequential quadratic programming in application to solving large-scale structural nonlinear optimization.

Although all notations can be identified easily from the context, we still list some of notations used in the paper for readers convenience: A letter with the superscript (s) for multi-stage stochastic programming in application to (1.5)
Since (10)-(12) has the same size as the original problem (5)-(7), it may also be very difficult to solve, even not be solvable directly by optimization approaches. To overcome this difficulty, we will use a Lagrangian dual to exploit the separable structure of (11). What is more special and subtle, (11)-(12) may not be consistent for any given \( z \) since the coefficient vectors of (11)-(12) may be linearly dependent even if the coefficient vectors of (12) respectively are linearly independent. The problem becomes even more involved when the Lagrangian dual approach is used. We relax the constraint (12) and obtain the Lagrangian dual of (11)-(12) as follows:

\[
\max_{\mu} \varphi(\mu),
\]

Where

\[
\varphi(\mu) = \min_{d(\mu)} \sum_{s=1}^{S} \left( \nabla_{s} f_{s}^T d(\mu) + \frac{1}{2} d(\mu)^T H_{s} d(\mu) \right)
\]

subject to \( h_{s} + \nabla_{s} h_{s}^T d(\mu) = 0, s = 1, ..., S \)

and \( \mu \in \mathbb{R}^{m_S} \) is the multiplier vector corresponding to (12). It is easy to verify that \( \varphi(\mu) \) is a concave function. Denote by \( d(\mu) \) the optimal solution of (14)-(15), We have the following properties on \( \varphi(\mu) \).

**Lemma 2.1** If \( \nabla_{s} h_{s} (s = 1, ..., S) \) are full column rank, then \( \varphi(\mu) \) is continuously differentiable, and we have

\[
\nabla \varphi(\mu) = \sum_{s=1}^{S} A_{s} (z^{(s)} + d(\mu)^{(s)})
\]

\[
\nabla^{2} \varphi(\mu) = -\sum_{s=1}^{S} A_{s} (H_{s}^{-1} - H_{s} - \nabla_{s} h_{s} (\nabla_{s} h_{s}^T H_{s}^{-1} \nabla_{s} h_{s})^{-1} \nabla_{s} h_{s}^T H_{s}^{-1}) A_{s}^T
\]

Proof. The Kuhn-Tucker condition of (14)-(15) implies that there is a \( \lambda(\mu) \in \mathbb{R}^{n_S} \) such that for \( s = 1, 2, ..., S \),

\[
\nabla_{s} f_{s} + H_{s} d(\mu)^{(s)} + \nabla_{s} h_{s} \lambda(\mu)^{(s)} + A_{s}^T \mu = 0
\]

\[
h_{s} + \nabla_{s} h_{s}^T d(\mu)^{(s)} = 0
\]

By moving constants to the right-hand-side of the equation, (18)-(19) can be written as the following:

\[
\begin{bmatrix}
H_{s} & \nabla_{s} h_{s} \\
\nabla_{s} h_{s}^T & 0
\end{bmatrix}
\begin{bmatrix}
d(\mu)^{(s)} \\
\lambda(\mu)^{(s)}
\end{bmatrix} =
\begin{bmatrix}
-\nabla_{s} f_{s} - A_{s}^T \mu \\
-h_{s}
\end{bmatrix}
\]

Since \( H_{s} \) is positive definite and \( \nabla_{s} h_{s} \) is full column rank, the Jacobian of (20) is invertible. Thus \( d(\mu) \) is a linear function. The result follows from (13). (i) For convenience of statement, let \( f = (f_{1}, ..., f_{S})^T, A = (A_{1}, A_{2}, ..., A_{S}) \). Then for any \( \tilde{\mu} \),

\[
\varphi(\mu) \leq \nabla f (z)^T d(\mu) + \frac{1}{2} d(\mu)^T H d(\mu) + \tilde{\mu}^T A(z + d(\mu))
\]

\[
= \nabla f (z)^T d(\mu) + \frac{1}{2} d(\mu)^T H d(\mu) + \mu^T A(z + d(\mu)) + (\tilde{\mu} - \mu)^T A(z + d(\mu))
\]

\[
= \varphi(\mu) + (\tilde{\mu} - \mu)^T A(z + d(\mu))
\]
which implies the result. Differentiating (2.10)-(2.11) w.r.t \( \mu \) and doing some calculations, we have
\[
\nabla d(\mu)^{(s)} = (H_y^s \nabla_y H_y^s H_y^{-1} \nabla_y h^s) \left( \nabla_y h^s H_y^{-1} - H_y^{-1} \right) A_y^{(s)}
\]
By (i),
\[
\nabla^2 \varphi(\mu) = \sum_{s=1}^{S} A_s \nabla d(\mu)^{(s)},
\]
Thus, the result follows from (2.14).

It follows from Lemma 2.1 that \( \varphi(\mu) \) is a quadratic function, since its Hessian \( \nabla^2 \varphi(\mu) \) is not related to \( \mu \). The first-order coefficient vector is
\[
\varphi(0) = A(z + \hat{d}),
\]
where \( \hat{d} = (\hat{d}^{(1)}, \ldots, \hat{d}^{(S)}) \) is the solution of the problem
\[
\min_{(d^{(1)}, \ldots, d^{(S)})} \sum_{s=1}^{S} (\nabla \phi_y f_y^T d^{(s)} + \frac{1}{2} d^{(s)T} H_y d^{(s)})
\]
\[s.t. \quad h_s^y + \nabla_y h_s^y d^{(s)} = 0, s = 1, \ldots, S\]
which is the problem (2.6)-(2.7) with \( \mu = 0 \).

**Lemma 2.2** Suppose that there exists a \( \mu^+ \in \mathbb{R}^m \) which maximizes the concave quadratic function
\[
q(\mu) = \frac{1}{2} \mu^T \nabla^2 \varphi(0) \mu + (z + \hat{d})^T A \mu,
\]
where \( \nabla^2 \varphi(0) \) is the same as (2.9), \( d_s = d(\mu) \). Then:

1. \[
\sum_{s=1}^{S} A_s (z^{(s)} + d^{(s)}_s) = 0;
\]
2. \( d_s \) is the optimal solution of (2.2)-(2.4).

**Proof.** (1) It follows from Lemma 2.1 (i) that
\[
\sum_{s=1}^{S} A_s (z^{(s)} + d^{(s)}_s) = \nabla^2 \varphi(0) \mu + A(z + \hat{d})
\]
Thus, the result (1) follows immediately from the supposition of the lemma.

(2) \( d^+ = d(\mu^+) \) is the solution of (2.6)-(2.7), thus there is a \( \lambda_s \in \mathbb{R}^m \) such that
\[
\nabla_y f_y + H_y d^{(s)} + \nabla_y h_s^y \lambda_s^y + A_y^T \mu_s = 0,
\]
\[
h_s^y + \nabla_y h_s^y d^{(s)} = 0
\]
Hence, the result follows from (1).
By (2.19) and (2.6)-(2.7), we have
\[
\varphi(\mu) = q(\mu) + \varphi(0),
\]
where \( \varphi(0) = \sum_{s=1}^{S} (\nabla \phi_y f_y^T \hat{d}^{(s)} + \frac{1}{2} \hat{d}^{(s)T} H_y \hat{d}^{(s)}) \)

The following result shows us that the \( \mu^+ \) in Lemma 2.2 exists if some conditions hold.

**Lemma 2.3** Suppose that \( H_s (s = 1, \ldots, S) \) are positive definite. Then for any \( \mu, \nabla^2 \varphi(\mu) \) defined by (2.9) is negative semi-definite. Furthermore, if the matrix, \( (\nabla h_1(z) \nabla h_2(z) \ldots \nabla h_S(z) A^T) \) is full column rank, then \( \nabla^2 \varphi(\mu) \) is negative definite. Proof. The first part of the result is straightforward. In order to simplify the proof of the second part, we firstly prove that
\[ \tilde{B} = U(I - V(U^TV)^{-1}V^T)U^T \]  

(33)

is positive definite, if \((V, U^T)\) has full column rank. Since \((V, U^T)\) has full column rank, by QR decomposition, we have

\[ (V, U^T) = Q \begin{bmatrix} R & \tilde{Q} \\ 0 & 0 \end{bmatrix} = (Q_1, Q_2) \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}, \]

(34)

where \(Q\) is an unitary orthogonal matrix, \(R\) is an upper-triangle matrix with all main diagonal elements being nonzero, \(Q_1, Q_2\) have the same number of columns with \(V\) and \(U^T\) respectively. Thus, hence, therefrom, therefore,

\[ V = Q_1R_{11}, \quad U = R_{12}Q_1^T + R_{22}Q_2^T \]

(35)

\[ UV = (R_{12}Q_1^T + R_{22}Q_2^T)Q_1R_{11}, \]

(36)

\[ UU^T = R_{12}R_{12} + R_{22}R_{22}, \]

(37)

\[ \tilde{B} = UV(U^TV)^{-1}V^TU^T \]

(38)

\[ \tilde{B} = UU^T - UV(U^TV)^{-1}V^TU^T \]

(39)

It is known that \(I - R_{11}(R_{11}^T)^{-1}R_{11}\) is positive semi-definite, and \(R_{22}\) is a full rank square matrix. Thus, \(\tilde{B}\) is positive definite.

Now we prove the lemma. Since \(H_s\) is positive definite, let

\[ U = (A_1, A_2, \ldots, A_s) \begin{bmatrix} H_1^{-1/2} \\ H_2^{-1/2} \\ \vdots \\ H_s^{-1/2} \end{bmatrix}, \]

(40)

\[ V = \begin{bmatrix} H_1^{-1/2} \\ H_2^{-1/2} \\ \vdots \\ H_s^{-1/2} \end{bmatrix} \begin{bmatrix} \nabla_1 h_1 \\ \nabla_2 h_2 \\ \vdots \\ \nabla_s h_s \end{bmatrix}, \]

it follows from the supposition that \((V, U^T)\) is full column rank. Thus, the result follows from (2.9) and the rst part of the proof.

Although Lemma 2.3 implies that \(\phi(\mu)\) and \(q(\mu)\) may have strictly concavity, we note very unfortunately the assumption that \((\nabla h_1(z)\nabla h_2(z)\ldots\nabla h_s(z)A^T)\) is full column rank in Lemma 2.3 may not hold in many cases for multi-stage stochastic nonlinear programs, this observation can be verified easily from the examples in section 5.

The fact that the Jacobian in (2.3)-(2.4) is not full column rank may result in the inconsistency between the linearized constraints (2.3) and the constraints (2.4), which induces the corresponding dual multipliers \(\lambda^+\) and \(\mu^+\) to tend to infinity. The next lemma shows that if the current iterate \(z\) meets the nonanticipativity constraints (1.7), then (2.3) and (2.4) are consistent under suitable assumptions. Thus, the existence of \(\mu^+\) is guaranteed by the duality theory of convex programming.

Lemma 2.4 Let \(W = \{z | Az = 0\}\). For any \(\bar{z} \in \mathbb{R}^n\), suppose that all \(\nabla c_i(t = 0, \ldots, T - 1; i = 1, \ldots, S)\) are linearly independent, where \(c_i\) is defined in (1.1)-(1.4). Then: (1) The linearized constraints (2.3) and (2.4) are consistent at any \(z \in W\); (2) There exists a \(\mu^+\) which maximizes (2.19).

Proof. (1) Firstly we prove that there exists a linear bijection \(\theta : W \to \mathbb{R}^n\), where \(n\) is the dimension of \(W\). Let the columns of \(E\) comprise a basis of the subspace \(W\), and let \(F\) be a matrix such that \(FE =
1. Then \( E \in \mathbb{R}^{n_S \times \pi} \), \( F \in \mathbb{R}^{n_S \times \pi} \), \( W = \{ E\bar{z} | \bar{z} \in \mathbb{R}^\pi \} \). Define \( \theta(z) = Fz \), it is easy to verify that \( \theta \) is a bijection, and \( \theta^{-1}(\bar{z}) = E\bar{z} \). Evidently, \( \nabla \theta^{-1}(\bar{z})^T = E \) for any \( \bar{z} \in \mathbb{R}^\pi \).

Let \( h(z) = (h_1(z^{(1)}); \ldots; h_3(z^{(3)})) \). For any \( \bar{z} \in \mathbb{R}^\pi \), \( \theta^{-1}(\bar{z}) \in W \). Define \( \tilde{h}(\bar{z}) = h(\theta^{-1}(\bar{z})) \) for any \( \bar{z} \in \mathbb{R}^\pi \). Then
\[
\nabla \tilde{h}(\bar{z}) = \nabla \theta^{-1}(\bar{z}) \nabla h(\theta^{-1}(\bar{z})) = E^T \nabla h(\theta^{-1}(\bar{z})) \tag{41}
\]

For any \( z \in W \) let \( \bar{z} = \theta(z) \in \mathbb{R}^\pi \). We consider the equation
\[
\tilde{h}(\bar{z}) + \nabla h(z)^T \bar{d} = 0 \tag{42}
\]

Let \( c = (c_0; c_{11}; \ldots; c_{T-1,S_T-1}) \) be the collection of all constraints in (1.1)-(1.3). By (1.9), definitions of \( h \) and \( \tilde{h} \), equation (2.34) is equivalent to
\[
c(\bar{z}) + \nabla c(\bar{z})^T \bar{d} \in \mathbb{R}^\pi = 0 \varphi(\mu) \nabla h_j c(\bar{z}) + \nabla c(\bar{z})^T \bar{d} = 0 \tag{43}
\]

(Actually, (2.35) can be obtained by deleting those repetitious constraints in (2.34).) It follows from the assumption that \( \nabla c(\bar{z}) \) is of full column rank that there is a \( \bar{d} \in \mathbb{R}^\pi \) such that (2.35) holds. This \( \bar{d} \) is also a solution of (2.34). Let \( d = Ed \). By using (2.33), we can write (2.34) as
\[
h(z) + \nabla h(z)^T d = 0 \tag{44}
\]

Moreover, \( Ed \in W \) since \( \bar{d} \in \mathbb{R}^\pi \), we have \( Ad = 0 \). The result follows directly from (2.36) and (2.37). (2) Since (2.5) is the dual of (2.2)-(2.4), and (2.3)-(2.4) has feasible solution, by the weak duality theorem, (2.5) is bounded. Furthermore, because \( \varphi(\mu) \) is a convex quadratic function, the boundedness of the unconstrained problem (2.5) implies the existence of optimal solutions of (2.5). By (2.24), we have the result. It is easy to note that the condition in Lemma 2.4 is based on problem (1.1)-(1.4). Under the condition of Lemma 2.4, by (1.9), we must have \( \nabla h_j \) to be of full column rank for all \( s = 1, \ldots, S \). However, the following example demonstrates the converse may not be true.

Example 2.5 Consider a two-stage problem with \( c_{0,t}(t = 0,1; S_0 = 1, S_1 = 2) \) defined by
\[
x_{1} - x_{2} = 0,
-x_{1} + x_{2} + y^{(1)} = 0,
x_{1} + y^{(2)} - 2 = 0,
-x_{1} + x_{2} + y^{(2)} = 0,
\]

where \((x_1, x_2)\) is to the rst stage, and \(y^{(1)}, y^{(2)}\) are corresponding to dierent realizations respectively. By notation (1.9), we have \( h_1(z^{(1)}) = 0 \) and \( h_2(z^{(2)}) = 0 \), which are below (2.43)-(2.45) and (2.46)-(2.48) respectively:

It is easy to verify that the Jacobian of (2.43)-(2.48) is of full column rank, but the Jacobian of (2.38)-(2.42) is not, which induces that the result of Lemma 2.4 does not hold.

\[
z^{(1)}_1 - z^{(1)}_2 = 0,
z^{(1)}_1 + z^{(1)}_3 = -1 = 0,
-z^{(1)}_1 + z^{(2)}_2 + z^{(2)}_3 = 0,
z^{(2)}_1 - z^{(2)}_1 = 0,
z^{(2)}_1 + z^{(2)}_3 - 2 = 0,
-z^{(2)}_1 + z^{(2)}_2 + z^{(2)}_3 = 0.
\]

Fortunately, under the condition of Lemma 2.4, since the simplicity of the nonanticipativity constraints, we can easily select an initial iteration point \( z_0 W \). By Lemma 2.2, by maximizing (2.19) and
then solving (2.6)-(2.7) to generate the search direction, we can ensure that the nonanticipativity constraints (1.7) hold at the new iterate. Thus, the algorithm can proceed. The details for the algorithm are stated in the next section.

Doing line search is necessary for the global convergence of SQP methods. Generally, a step-size is selected such that the chosen merit function is reduced sufficiently along the search direction \( d \). The \( l_1 \) exact penalty function (e.g. [26], [18] and [22]) can be taken as the merit function. In [19] proposed a SQP algorithm that used the differentiable exact penalty function as the merit function. In [21] proved the global convergence of their algorithm that used a penalty function defined by general convex norm. In this paper, we define the following \( l_1 \) exact penalty function as the merit function:

\[
M(z, \rho) = \sum_{s=1}^{S} f_s(z^{(s)}) + \rho \left( \sum_{s=1}^{S} \left( \| h_s(z^{(s)}) \|_1 + \| \sum_{s=1}^{S} A_s z^{(s)} \|_1 \right) \right)
\]

(45)

where \( \rho > 0 \) is the penalty parameter. However, since (1.7) holds for all iteration points, we just consider the function

\[
M(z, \rho) = \sum_{s=1}^{S} f_s(z^{(s)}) + \rho \left( \sum_{s=1}^{S} \| h_s(z^{(s)}) \|_1 \right)
\]

(46)

The following lemma is a natural extension of a fundamental result in convex analysis (e.g. see in [27]).

**Lemma 2.6**

Let \( N(z) = \sum_{s=1}^{S} \| h_s(z^{(s)}) \|_1 \).

Then for any \( z \in \mathbb{R}^n \) and \( d^{(s)} \in \mathbb{R}^n, d = (d^{(1)T}, \ldots, d^{(S)T})^T \in \mathbb{R}^{nS} \) the directional derivative of function \( N(z) \) along \( d \), defined by

\[
N'(z; d) = \lim_{\alpha \downarrow 0} \frac{N(z + \alpha d) - N(z)}{\alpha},
\]

exists, and we have

\[
N'(z; d) \leq \sum_{s=1}^{S} (\| h_s + \nabla_s h_s^T d^{(s)} \|_1 - \| h_s \|_1)
\]

(48)

Proof. The existence of \( N'(z; d) \) follows from the convexity of \( N(z) \) (see in [27]). By the properties of the norm, for \( 0 < \alpha < 1 \), we have

\[
\| h_s(z^{(s)} + \alpha d^{(s)}) \|_1 - \| h_s(z^{(s)}) \|_1 = \| h_s + \nabla_s h_s^T d^{(s)} + o(\alpha) \|_1 - \| h_s \|_1 \leq \alpha (\| h_s + \nabla_s h_s^T d^{(s)} \|_1 - \| h_s \|_1) + o(\alpha),
\]

(49)

Then the result follows from (2.51). It follows from Lemma 2.6 and the definition of the directional derivative (2.51) that

\[
M'(z, \rho; d) = \sum_{s=1}^{S} \nabla_s f_s^T d^{(s)} + \rho N'(z; d)
\]

(50)

(2.54) implies that if the right-hand-side of (2.52) is negative, then the penalty parameter \( \rho \) can be increased such that \( M'(z, \rho; d) < 0 \). Thus, the \( d \) such that the right-hand-side of (2.52) is negative can be a descent direction of the merit function \( M(z, \rho) \) for large \( \rho \). On the other hand, if the right-hand-side of (2.52) is zero, then \( M'(z, \rho; d) < 0 \) if \( \nabla_s f_s^T d^{(s)} < 0 \) for \( s = 1, \ldots, S \).
The next result has a little difference from the common one for general sequential quadratic programming methods (e.g. see Fukushima[8]).

**Lemma 2.7**
If $(z^*, \lambda^*, \mu^*)$ is a Kuhn-Tucker triple of problem (1.5)-(1.7) with $z^* = (z^{(1)};...,z^{(S)}) \in \mathbb{R}^{n_S}$, $\lambda^* = (\lambda^{(1)};...,\lambda^{(S)}) \in \mathbb{R}^{n_S}$ and $\mu^* \in \mathbb{R}^m$, $\rho > \max(||\lambda^*||_\infty;||\mu^*||_\infty)$ then $M'(z^*, \rho; d) \geq 0$ for any $d \in \{\tilde{d} \in \mathbb{R}^{n_S} : \tilde{A}d = 0\}$

Proof. $(z^*, \lambda^*, \mu^*)$ is a Kuhn-Tucker triple of problem (1.5)-(1.7), which implies that

$$
\sum_{s=1}^{S} \nabla_s f_s(z^{(s)}) d^{(s)} = -\sum_{s=1}^{S} (\nabla_s f_s(z^{(s)})^T \lambda^{(s)} - (\sum_{s=1}^{S} A_s d^{(s)})^T \mu^*) 
$$

$$
\geq - \max ||\lambda^{(s)}||_\infty \sum_{s=1}^{S} ||(\nabla_s f_s(z^{(s)})^T ||_{\infty} - ||\mu^*||_\infty \sum_{s=1}^{S} A_s d^{(s)} ||_{1} 
$$

By the first equation of (2.54), (2.51) and $h_s(z) = 0(s = 1, S)$,

$$
M'(z^*, \rho; d) = \sum_{s=1}^{S} \nabla_s f_s(z^{(s)})^T d^{(s)} + \rho \sum_{s=1}^{S} D_s d^{(s)} 
$$

hence the result follows immediately from $Ad=0$

An appropriate stopping criterion should be designed to guarantee that the algorithm terminates finitely at the desirable point of the problem. Generally, the approximate Kuhn-Tucker condition of the original problem is one of the selections. Similar to in [22], a stopping criterion is introduced into our algorithm, which is a natural combination with our technique for the search direction.

For convenience of statement, we need the following definition:

**Definition 2.8** For any $\varepsilon > 0$, we call $z$ an -optimal solution to the program (1.5)-(1.7), if there is a $\lambda \in \mathbb{R}^{n_S}$ and a $\mu \in \mathbb{R}^m$ such that

$$
||\nabla_s f_s(z^{(s)}) + \nabla_s h_s(z^{(s)}) \lambda^{(s)} + A_s^T \mu|| \leq \varepsilon, \\
|h_s(z^{(s)})|| \leq \varepsilon, s = 1, ..., S \\
||\sum_{s=1}^{S} A_s z^{(s)}|| \leq \varepsilon. 
$$

if $\varepsilon=0$, then $z$ is a Kuhn-Tucker point of (1.5)-(1.7)

3. Results and Discussion

We present the algorithm for solving problem (1.5)-(1.7) in this section. In the algorithm, Hks, for $s = 1,2,...,S$, is supposed to be the approximate Hessian of the Lagrangian $L_s(z^{(s)}), \lambda^{(s)}) = f_s(z^{(s)}) + \lambda^{(s)^T} h_s(z^{(s)})$ at iteration $(z^{(s)}_k, \lambda^{(s)}_k)$ where $\lambda^{(s)}_k$ is an estimate of the multiplier associated with $h_s$.

$$
B_k = \sum_{s=1}^{S} A_s^T (H_s - H^{(-)}_s \nabla h_s \nabla h_s^T H^{(-)}_s \nabla h_s) H^{(-)}_s \nabla h_s A_s^T 
$$

is the value of $f_s(z_0^{(s)}), h_s(z_0^{(s)}), \nabla_s f_s(z_0^{(s)}), \nabla_s h_s(z_0^{(s)})$ at $z_k$.

Algorithm 3.1

Step 0. Given $z_0 \in \mathbb{R}^{n_S}$ such that (1.7) holds, $H_{bs} \in \mathbb{R}^{n_n}$ $(s = 1,2,...,S)$, $\rho_0 > 0$ and positive constants $\delta < \frac{1}{2}, \varepsilon, \beta < 1$ and $\sigma_0 > 0$. Evaluate $f_s(z_0^{(s)}), h_s(z_0^{(s)}), \nabla_s f_s(z_0^{(s)}), \nabla_s h_s(z_0^{(s)})$ for $s = 1,2,...,S$ and $B_0$. Let $\mu_0 = 0, k = 0$;

Step 1. For $s = 1,2,...,S$ solve subproblems
\[ \min \phi_k^{(s)}(d) = \nabla_s f_s(z_k^{(s)})^T d + \frac{1}{2} d^T H_k d \]  
(56)

s.t. \[ h_j(z_k^{(s)}) + \nabla_s h_j(z_k^{(s)})^T d = 0 \]  
(57)

Let \[ \hat{d}_k^{(s)}, s=1,\ldots,S \] be the solutions. Set \[ v_0 = \sigma_k, \mu_0 = \mu_k, j=0; \]
(58)

Step 2. Computing \[ \mu_{j+1} = \mu_j + d_{\mu} \], where \[ d_{\mu} \] is the solution to the unconstrained quadratic programming subproblem

\[ \max \bar{q}_k(d_{\mu}) = (A\hat{d}_k - B_k \mu_j)^T d_{\mu} - \frac{1}{2} d_{\mu}^T (B_k + v_j I) d_{\mu} \]  
(59)

and can be derived by conjugate gradient methods. If \[ B_k \mu_{j+1} = A\hat{d}_k B_k \mu_{j+1} = A\hat{d}_k \] then \[ \mu_{k+1} = \mu_{j+1}, \sigma_k = v_j \] and go to Step 3; Else compute \[ r = \text{Ared}_j / \text{Pred}_j \], where \[ \text{Ared}_j = q_k(\mu_{j+1}) - q_k(\mu_j) \] and \[ \text{Pred}_j = q_k(d_{\mu}) - \bar{q}_k(0). \] \[ v_j \] is updated as the following:

\[ v_{j+1} = \begin{cases} 0.5v_j, & \text{if } r > 0.75; \\ 4v_j, & \text{if } r < 0.25; \\ v_j, & \text{otherwise} \end{cases} \]  
(60)

Let \[ j = j + 1 \] and go to Step 2.

Step 3. For \[ s = 1,2,\ldots,S \], solve the subproblems

\[ \min \psi_k^{(s)}(d) = (\nabla_s f_s(z_k^{(s)}) + A_k^T \mu_{k+1})^T d + \frac{1}{2} d^T H_k d \]  
(61)

s.t. \[ h_j(z_k^{(s)}) + \nabla_s h_j(z_k^{(s)})^T d = 0 \]  
(62)

to generate \[ \hat{d}_k^{(s)}, s=1,\ldots,S; \]

Step 4. Check if the stopping criterion

\[ |M(z_k, \rho_k) - \sum_{s=1}^S f_s(z_k^{(s)}) - \sum_{s=1}^S \psi_k^{(s)}(d_{k}^{(s)})| < \varepsilon \]  
(63)

is satisfied. If yes, stop; otherwise, go to step 5;

Step 5. Update the penalty parameter \[ \rho \]. If

\[ \rho_{k+1} = \rho_k; \] otherwise,

\[ \rho_{k+1} = \max \left\{ \frac{\sum_{s=1}^S (\nabla_s f_s(z_k^{(s)})^T d_k + \frac{1}{2} d_k^T H_k d_k)}{\sum_{s=1}^S \| h_j(z_k^{(s)}) \|_1}, 2\rho_k \right\} \]  
(64)

Step 6. Select the least positive integer \[ r \] such that

\[ f_s(z_{k+1}^{(s)}), h_j(z_{k+1}^{(s)}), \nabla_s f_s(z_{k+1}^{(s)}), \nabla_s h_j(z_{k+1}^{(s)}) \] for \( s = 1,2,\ldots,S \) and

\[ B_{k+1}. \]

let \[ \alpha_k = \beta^r \] and \[ z_{k+1} = z_k + \alpha_k d_k; \]

Step 7. update \[ H_k \] to \[ H_{k+1} \], calculate

\[ f_s(z_{k+1}^{(s)}), h_j(z_{k+1}^{(s)}), \nabla_s f_s(z_{k+1}^{(s)}), \nabla_s h_j(z_{k+1}^{(s)}) \] for \( s = 1,2,\ldots,S \) and \[ B_{k+1}. \]

set \[ k=k+1 \] and go to step 1.

Note 1. One of the key difficulties for an iterative method is how to generate the search direction, by which the new approximate to the solution is generated. In Algorithm 3.1 we generate the search direction by solving a set of quadratic programming subproblems (3.3)-(3.9), where (3.3)-(3.4) is the decomposition of the problem (2.17)-(2.18). It is noted that (3.3)-(3.4) and (3.8)-(3.9) are quadratic.
programming subproblems with dimension $n$. Thus, they can be solved by standard algorithms for quadratic programming, such as the dual active set method proposed by Goldfarb and Idinani[10] and Powell[20]. On the other hand, for $s = 1, 2, \ldots, S$, (3.3)-(3.4) and (3.8)-(3.9) can be solved parallelly.

Note 2. (3.5) is a strictly concave unconstrained quadratic programming with dimension $m_0$, which may be very large. If we apply Newton method to this problem, it needs only one iteration to get the optimal solution. The large memory, however, is required for the inverse matrix of $B_k + v_j I$. We suggest to use conjugate gradient methods with exact line search procedure, which do not need the information on $(B_k + v_j I)$ and the optimal solution will be derived in finite iterations (e.g. see Bazaraa and Shetty[1]).

Note 3. Step 2 is designed for maximizing function $q_k(\mu)$, which is defined by (2.19). Since the Hessian of $q_k(\mu)$ may be singular, our algorithm for maximizing (2.19) is similar to the well-known Levenberg-Marquardt method for the linear least squares problems. On the other hand, in our implementation, instead of (3.6), we use

$$v_k^{(s)} = \left\{ \begin{array}{ll}
\hat{v}_k^{(s)} + \nu_k^{(s)} u_k^{(s)} & \geq 0, 2u_k^{(s)^T} H_{ks} u_k^{(s)} \\
\theta_k \hat{v}_k^{(s)} + (1-\theta_k) H_{ks} u_k^{(s)} & , otherwise
\end{array} \right. \quad (68)$$

where $\varepsilon_0$ is a small tolerance scalar.

Note 4. $H_{ks}$ is updated by the damped BFGS formulae:

$$H_{(k+1)s} = H_{ks} - \frac{H_{ks} u_k^{(s)^T} H_{ks}}{u_k^{(s)^T} H_{ks} u_k^{(s)}} + \frac{v_k^{(s)} v_k^{(s)^T}}{u_k^{(s)^T} v_k^{(s)}} \quad (69)$$

where

$$v_k^{(s)} = \left\{ \begin{array}{ll}
\hat{v}_k^{(s)} + \nu_k^{(s)} u_k^{(s)} & \geq 0, 2u_k^{(s)^T} H_{ks} u_k^{(s)} \\
\theta_k \hat{v}_k^{(s)} + (1-\theta_k) H_{ks} u_k^{(s)} & , otherwise
\end{array} \right. \quad (70)$$

$$\hat{v}_k^{(s)} = \nabla_s L (\bar{\zeta}_k^{(s)}, \bar{\lambda}_k^{(s)}) - \nabla_s L (\bar{\zeta}_k^{(s)}, \bar{\lambda}_k^{(s)}) - \bar{\nu}_k^{(s)} = \bar{\zeta}_k^{(s+1)} - \bar{\zeta}_k^{(s)} \quad (71)$$

and $\lambda_k^{(s+1)}$ is multiplier associated with $h_k$. It can be proved that $H_{(k+1)s}$ is positive definite if $H_{ks}$ is positive definite in [26].

4. Conclusion

This paper presents an approach to solve two-stage nonlinear stochastic program. The approach is based on reduced gradient (RG) method which normally is used to solve nonlinear programming problem. We extend the direct search method involved in RG and the concept of superbasic variables.

References

[1] P. Kall and S.W. Wallace, 1994. Stochastic Programming, John Wiley & Sons, NY.
[2] R.T. Rockafellar and R.J-B. Wets, 2001. Scenarios and policy aggregation in optimization under uncertainty. Math. Oper. Res., 16, 119-147.
[3] A. Ruszczyński, 1997. Decomposition methods in stochastic programming, Math. Prog., 79, 333-353.
[4] L.S. Lasdon, 1970. Optimization theory for large systems. Macmillan, New York.
[5] B. Feinberg, 1989. Coercion functions and decentralized linear programming. Math. Oper. Res., 14, 177-187.
[6] S.P. Han, 1989. A decomposition method and its application to convex programming. Math. Oper. Res., 14, 237-248.
[7] A. Ruszczyński, 1995. On convergence of an augmented Lagrangian decomposition method for sparse convex optimization. Math. Oper. Res., 20, 634-656.
[8] G.B. Dantzig and P. Wolfe, 1960. Decomposition principle for linear programs. Oper. Res., 8,
101-111.

[9] J.R. Birge, 1995. Current Trends in Stochastic Programming Computation And Applications. Department of Industrial and Operations Engineering, University of Michigan, Technique Report 95-15.

[10] J.R. Birge and F. Louveaux, 1997. Introduction to Stochastic programming, Springer-Verlag.

[11] G. Zhao, 1999. Interior-Point Methods With Decomposition For Solving Large-Scale Linear Programs, JOTA, 102, 169-192.

[12] G. Zhao, A Log-Barrier Method With Benders Decomposition For Solving Two-Stage Stochastic Programs,

[13] J.M. Mulvey and H. Vladimirous, 1991. Applying The Progressive Hedging Algorithm To Stochastic Generalized Networks. Ann. Oper. Res., 31, 399-424.

[14] B.J. Chun and S.M. Robinson, 1995. Scenario Analysis Via Bundle Decomposition. Ann. Oper. Res., 56, 39-63.

[15] T. Helgason and S.W. Wallace, 1991. Approximate Scenario Solutions In The Progressive Hedging Algorithm. Ann. Oper. Res., 31, 425-444.

[16] S.M. Robinson, 1991. Extended Scenario Analysis. Ann. Oper. Res., 31, 385-398.

[17] G. Zhao, 1999. Lagrangian Dual Method With Self-Concordant Barriers For Multi-Stage Stochastic Nonlinear Programming. Report.

[18] M. Fukushima, 1986. A Successive Quadratic Programming Algorithm With Global And Super-Linear Convergence Properties. Math. Prog., 35, 253-264.

[19] M.J.D. Powell and Y. Yuan, 1986. A Recursive Quadratic Programming Algorithm That Using Differentiable Exact Penalty Function. Math. Prog., 35, 265-278.

[20] P.T. Boggs and J.W. Tolle, 1995. Sequential Quadratic Programming. in Acta Numerica 95, Cambridge University Press, Cambridge, 1-51.

[21] X.W. Liu and Y. Yuan, 2000. A Robust Algorithm For Optimization With General Equality And Inequality Constraints. SIAM J. Sci. Comput., 22, 517-534.

[22] M. Fukushima, 1991. A successive quadratic programming method for a class of con-strained nonsmooth optimization problems. Math. Prog., 49, 231-251.

[23] J.S. Pang, S.P. Han and N. Rangaraj, 1991. Minimization of locally Lipschitzian functions. SIAM J. Opt., 1, 57-82.

[24] S.P. Han, J.S. Pang and N. Rangaraj, 1992. Globally Convergent Newton Methods For Nonsmooth Equations, Math. Oper. Res., 17, 586-607.

[25] L. Qi, 1994. Superlinearly Convergent Approximate Newton Methods For L C 1 Optimization Problems. Math. Prog., 64, 277-294.

[26] M.J.D. Powell, 1978. A Fast Algorithm For Nonlinearly Constrained Optimization Calculations, in proc. 1977 Dundee Biennial Conference on Numerical Analysis, G.A.Watson, ed., Springer-Verlag, Berlin, 144-157.
R.T. Rockafellar, 1970. Convex analysis. Princeton University Press, Princeton.