AMO - Advanced Modeling and Optimization, Volume 14, Number3, 2012

A feasible directions method on combining feasibility with descent for nonlinear constrained optimization[1]

Chunyan Hu

College of Electronic Engineering and Automation,

Guilin University of Electronic Technology, Guilin, 541004, P. R. China

Abstract. In this paper, a modified gradient projection method is proposed to solve the nonlinear constrained optimization problems, where the search direction is obtained by combing feasibility with descent. In addition, it is pointed out that, for linear constrained optimization problems, this method may be simplified and viewed as the modified version to Rosen's method. The theoretical analysis shows that global convergence can be obtained under some suitable conditions.

Key words. Nonlinear constrained optimization, Feasible directions method, Gradient projection method, Global convergence

1. Introduction

Since Rosen has presented gradient projection method [1,2] from 1960, it becomes a basic technique for solving constrained nonlinear programming problems, as well, is studied and improved further by a lot of authors. For optimization with linear constraints, there existed some relevant references [3-11], which either had to modify Rosen's method to obtain global convergence, or relied mainly on the convergence of Rosen's method. While it was a long-standing problem as to whether Rosen's method is convergent or not. Later, Du and Zhang [12] proved global convergence of Rosen's method in 1989, obviously, the proof was complex. In addition, some above-mentioned methods required to solve two projection matrixes. On the other hand, for optimization with nonlinear constraints, gradient project direction didn't satisfy the requirement of feasibility at active constraints set. There were a lot of modified methods [13-16], but they either required some stronger assumptions, or were necessary to compute a more complex modified feasible directions of descent than the gradient project direction, thus computing cost was rather high.

In this paper, firstly, optimization with nonlinear constraints is studied and a new method is presented. A feasible direction is obtained by taking advantage of the descent gradient project direction, then, the search direction is obtained by making a convex combination with the descent direction and the feasible direction, thereby, in a single iteration, it is only necessary to solve one projection matrix. In the end, global convergence is proved under some general conditions. In addition, we point out that the method can be simplified for linear constrained optimizations. In comparison with Rosen's method, this method requires to solve only one projection matrix, in other words, if the gradient project direction is zero, then the corresponding

[1]This work was supported in part by NNSF(No.11061011) of China and Guangxi Fund for Distinguished Young Scholars (2012GXSFFA060003).

Corresponding author: E-mail: huchyel@hotmail.com

AMO -Advanced Modeling and Optimization. ISSN: 1841-4311

point must be a KKT point, moreover, analysis of convergence is simpler than that of Rosen's method.

2. Description of Algorithm

In this paper, we consider the following nonlinear programming (NLP):

$$\begin{array}{ll} \min & f_o(x) \\ s.t. & f_j(x) \leq 0, j = 1, 2, \ldots, m, \end{array} \tag{1}$$

where $f_j : R^n \to R, (j = 0, 1 \sim m)$ are smooth functions. For the sake of simplicity, we denote

$$I = \{1, 2, \ldots, m\}, X = \{x \in R^n | f_j(x) \leq 0, j \in I\}, I(x) = \{j \in I | f_j(x) = 0\},$$

Three basic assumptions are given as follows:

A_1**:** The feasible set is nonempty, i.e., $X \neq \phi$;

A_2**:** The functions $f_j(x)(j = 0, 1 \sim m)$are continuously differentiable;

A_3**:** For all $x \in X$, the vectors $\{g_j(x), j \in I(x)\}$are linearly independent.

For $x^k \in X$ and some set $J_k \subseteq I$, we define

$$g_j(x^k) = \nabla f_j(x^k), j = 0, 1 \sim m, A_k = A(x^k) = (g_j(x^k), j \in J_k,$$

$$Q_k = Q(x^k) = (A_k^T A_k)^{-1} A_k^T, P_k = P(x^k) = E_n - A_k Q_k, \tag{2}$$

$$u^k = u(x^k) = -Q_k g_o(x^k), d_o^k = d_o(x^k) = -P_k g_o(x^k) + Q_k^T v^k, \tag{3}$$

$$v^k = v(x^k) = (v_j^k, j \in J_k), v_j^k = \begin{cases} -f_j(x^k), & u_j^k > 0 \\ u_j^k, & \pi_j^k \leq 0 \end{cases}, \tag{4}$$

$$d_1^k = -||d_o^k|| Q_k^T e, e = (1, \ldots, 1)^T \in R^{|J_k|}, \tag{5}$$

$$q^k = (1 - \tau_k) d_o^k + \tau_k d_1^k, \tau_k = \begin{cases} 1, & g_o(x^k)^T d_1^k \leq \theta g_o(x^k)^T d_o^k \\ \frac{(1-\theta) g_o(x^k)^T d_o^k}{g_o(x^k)^T (d_o^k - d_1^k)}, & g_o(x^k)^T d_1^k > \theta g_o(x^k)^T d_o^k \end{cases}. \tag{6}$$

Now, the algorithm for the solution of (1) can be stated as follows:

Algorithm A

Step 0: (Initialization): Given a starting point $x^1 \in X$. Choose parameters $\varepsilon_o, \alpha, \theta \in (0, 1)$. Set $k = 1$;

Step 1: Let $i = 0, \varepsilon_{k,i} = \varepsilon_o$;

Step 2: If $\det(A_{k,i}^T A_{k,i}) \geq \varepsilon_{k,i}$, let $L_k = L_{k,i}, i_k = i$ and go to step 4. Otherwise, go to step 3, where

$$L_{k,i} = \{j \in I \mid -\varepsilon_{k,i} \leq f_j(x^k) \leq 0\}, A_{k,i} = (g_j(x^k), j \in L_{k,i}); \tag{7}$$

Step 3: Let $i = i + 1, \varepsilon_{k,i} = \frac{1}{2} \varepsilon_{k,i-1}$, and go to step 2;

Step 4: Compute d_o^k according to (3). If $d_o^k = 0$, STOP. Otherwise, compute q^k according to (6);

Step 5: Compute λ_k, the first number λ in the sequence $\{1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots\}$ satisfying

$$f_o(x^k + \lambda q^k) \le f_o(x^k) + \alpha\lambda g_o(x^k)^T q^k, \tag{8}$$

$$f_j(x^k + \lambda q^k) \le 0, \quad j \in I; \tag{9}$$

Step 6: Set $x^{k+1} = x^k + \lambda_k q^k$ and $k = k + 1$. Go back to step 1.

3. Global Convergence of Algorithm

In this section, we first show that the algorithm A given in section 2 is correctly stated, that is to say, it is possible to execute all the steps defined above. Then, we prove the global convergence of Algorithm A. Firstly, we make another assumption and let it hold in the remainder of this paper.

A_4**:** $\{x^k\}$ are bounded, which is a point range generated by the algorithm A.

Lemma 1 For any iteration, there is no infinite cycle between step 2 and step 3.

Proof The proof of this Lemma is similar to the proof of Lemma 1.1 in [17].

Theorem 1 (1). x^k is a KKT point of the problem (1) $\Longleftrightarrow d_o^k = 0$;

(2). If x^k is not a KKT point of the problem (1), then

$$g_o(x^k)^T d_o^k < 0, g_o(x^k)^T q^k < 0, g_j(x^k)^T q^k < 0, j \in I(x^k), \tag{10}$$

i.e., q^k is a feasible direction of descent;

Proof (1). If x^k is a KKT point of (1), then, from the definition of L_k, there exists a vector $\alpha = (\alpha_j, j \in L_k)$, such that

$$g_o(x^k) + A_k\alpha = 0, \alpha_j \ge 0, \alpha_j f_j(x^k) = 0, j \in L_k. \tag{11}$$

Obviously, $(A_k^T A_k)^{-1}$ is meaning according to step 2, so, it follows that

$$\alpha = -(A_k^T A_k)^{-1} A_k^T g_o(x^k) = u^k.$$

Thus, from (4) and (11), we get

$$v^k = 0, g_o(x^k) + A_k u^k = 0,$$

which shows that $d_o^k = 0$.

On the contrary, If $d_o^k = 0$, then, we obtain that

$$0 = A_k^T d_o^k = -A_k^T P_k g_o(x^k) + A_k Q_k^T v^k = v^k, P_k g_o(x^k) = 0,$$

thereby, from (3), (4), it is clear that

$$u_j^k \ge 0, u_j^k f_j(x^k) = 0, j \in L_k, g_o(x^k) + A_k u^k = 0. \tag{12}$$

From the definition of L_k, it is true that x^k is a KKT point of (1).

(2). If x^k is not a KKT point of the problem (1), then $d_o^k \neq 0$, then, by (3), we obtain that

$$g_o(x^k)^T d_o^k = -g_o(x^k)^T P_k g_o(x^k) - (u^k)^T v^k$$
$$= -||P_k g_o(x^k)||^2 - \sum_{j\in L_k, \pi_j^k \leq 0} (\pi_j^k)^2 + \sum_{j\in L_k, \pi_j^k > 0} \pi_j^k f_j(x^k) < 0,$$

and

$$A_k^T d_o^k = v^k, g_j(x^k)^T d_o^k = v_j^k \leq 0, j \in I(x^k) \subseteq L_k.$$

In addition, from (5), we have

$$A_k^T d_1^k = -||d_o^k||e, g_j(x^k)^T d_1^k = -||d_o^k|| < 0, j \in I(x^k) \subseteq L_k.$$

While, from (6), it is obvious that $\tau_k \in (0, 1]$, and

$$g_j(x^k)^T q^k = g_j(x^k)^T [(1-\tau_k)d_o^k + \tau_k d_1^k] \leq \tau_k g_j(x^k)^T d_1^k < 0, j \in I(x^k).$$

Furthermore, when $g_o(x^k)^T d_1^k \leq \theta g_o(x^k)^T d_o^k$, i.e., $\tau = 1$, it can be seen that

$$g_o(x^k)^T q^k = g_o(x^k)^T d_1^k \leq \theta g_o(x^k)^T d_o^k < 0,$$

when $g_o(x^k)^T d_1^k > \theta g_o(x^k)^T d_o^k$, it follows that

$$g_o(x^k)^T q^k = g_o(x^k)^T [(1-\tau_k)d_o^k + \tau_k d_1^k] \leq \theta g_o(x^k)^T d_o^k < 0.$$

In the sequel, we'll prove that Algorithm A is globally convergent. Since there are only finitely many choices for sets $L_k \subseteq I, I(x^k) \subseteq I$ and x^k are bounded , we might as well assume that there exists a subsequence K, such that

$$x^k \to x^*, L_k \equiv L, I_k = I(x^k) \equiv I_*, k \in K,$$

where L and I_* are constant sets.

Lemma 2 If $x^k \to x^*, k \in K$, then there exists $\widetilde{\varepsilon}$, such that for $k \in K, k$ large enough, it holds that $\varepsilon_{k,i_k} \geq \widetilde{\varepsilon}$.

Proof The proof of this Lemma is similar to the proof of Lemma 2.8 in [17].

Denote $A_* = (g_j(x^*), j \in L)$, from Lemma 2, we have that $(A_*^T A_*)^{-1}$ is meaning. Furthermore, replacing A_* for $A(x^*)$ at x^*, we denote $d_o^*, u^*, Q_*, P_*, q^*$ are corresponding vectors of (2) $\sim$ (6). Obviously, by the assumption A_2, it is easy to prove that

$$d_o^k \to d_o^*, u^k \to u^*, q^k \to q^*, k \in K. \tag{13}$$

Lemma 3 If x^* is not a KKT point of (1), then $d_o^* \neq 0$.

Proof The proof is similar to that of Theorem 1.

Theorem 2 The algorithm A either stops at the KKT point x^k of the problem (1) in finite iteration, or generates an infinite sequence$\{x^k\}$ whose any accumulation point x^* is a KKT point of the problem (1).

Proof The first statement is obvious, the only stopping point being in step 4. Thus, suppose that $\{x^k\}$ is generated by the algorithm A, and $\{x^k\}_{k\in K} \to x^*$. Suppose that the desired conclusion is not true, then, according to Lemma 3, it holds that $d_o^* \neq 0$. Thereby, by imitating the proof of theorem 1, it follows that

$$g_o(x^*)^T d_o^* < 0, g_o(x^*)^T q^* < 0, g_j(x^*)^T q^* < 0, j \in I_*.$$

In addition, because of

$$g_o(x^k)^T q^k \to g_o(x^*)^T q^*, g_j(x^k)^T q^k \to g_j(x^*)^T q^*, j \in I_k, k \in K,$$

for $k \in K, k$ large enough, we have that

$$g_o(x^k)^T q^k \leq \frac{1}{2} g_o(x^*)^T q^* < 0, g_j(x^k)^T q^k \leq \frac{1}{2} g_j(x^*)^T q^* < 0, j \in I_*. \tag{14}$$

We first show that, in this case, the step λ_k is bounded away from zero on K, i.e.,

$$\lambda_k \geq \lambda_* = \inf\{\lambda_k, k \in K\} > 0, k \in K. \tag{15}$$

Analyze(8). Denote a_k as follows:

$$\begin{aligned} a_k &\stackrel{\triangle}{=} f_o(x^k + \lambda q^k) - f_o(x^k) - \alpha \lambda g_o(x^k)^T q^k \\ &= (1-\alpha)\lambda g_o(x^k)^T q^k + o(\lambda) \leq \tfrac{1}{2}(1-\alpha)\lambda g_o(x^*)^T q^* + o(\lambda). \end{aligned}$$

For $k \in K, k$ large enough and $\lambda > 0$ sufficiently small, it is clear to see that $a_k \leq 0$.

Analyze(9). If $j \in I \setminus I_k$, then $f_j(x^k) < 0$, so, it is obvious that $f_j(x^k + \lambda q^k) \leq 0$ (for $\lambda > 0$ small enough); If $j \in L_k$, then $f_j(x^k) = 0$. By (14), we have

$$b_k \stackrel{\triangle}{=} f_j(x^k + \lambda q^k) = \lambda g_j(x^k)^T q^k + o(\lambda) \leq \frac{1}{2} \lambda g_j(x^*)^T q^* + o(\lambda).$$

For $k \in K, k$ large enough and $\lambda > 0$ small enough, it follows that $b_k \leq 0$.

Thus, by above-mentioned analysis, it holds that $\lambda_k \geq \lambda_* = \inf\{\lambda_k, k \in K\} > 0, k \in K$. In addition, from (8), (10), it is evident that $\{f_o(x^k)\}$ is monotonous decreasing. Hence, considering $\{x^k\}_K \to x^*$ and assumption A_2, there holds

$$f_o(x^k) \to f_o(x^*), k \to \infty. \tag{16}$$

So, from (8), (16), we get

$$0 = \lim_{k \in K} (f_o(x^{k+1}) - f_o(x^k)) \leq \lim_{k \in K} (\alpha \lambda_k g_o(x^k)^T q^k) \leq \frac{1}{2} \alpha \lambda_* g_o(x^*)^T q^* < 0.$$

It is a contradiction, which shows that x^* is a KKT point of the problem (1).

4. Linear Constraints

In this section, we consider the following linear constrained optimization:

$$\begin{array}{ll} \min & f(x) \\ s.t. & a_j^T x \leq b_j, j \in I = \{1, 2, \ldots, m\}, \end{array} \tag{17}$$

To solve (17), Algorithm A may be simplified as follows:

Algorithm B

Step 0: Given a feasible point $x^o, \alpha \in (0,1), k = 0$;

Step 1: Compute

$$g(x^k) = \nabla f(x^k), J_k = \{j \in I \mid a_j^T x^k = b_j\}, A_k = (a_j, j \in J_k),$$
$$Q_k = (A_k^T A_k)^{-1} A_k^T, P_k = E_n - A_k Q_k, u^k = -Q_k g(x^k),$$
$$d^k = -P_k g(x^k) + Q_k^T v^k, v^k = (v_j^k, j \in J_k), v_j^k = \begin{cases} b_j - a_j^T x^k, & u_j^k > 0, \\ u_j^T, & u_j^k \leq 0. \end{cases}$$

If $d^k = 0$, STOP (in this case, x^k is a KKT point of (17)), otherwise, CONTINUE.

Step 2: Compute

$$\overline{\lambda}_k = \begin{cases} 1, & \text{if} \quad a_j^T d^k \leq 0, j \in I, \\ \min\{\frac{b_j - a_j^T x^k}{a_j^T d^k} : a_j^T d^k > 0\}, & \text{otherwise.} \end{cases}$$

Choose $\lambda_k = (\frac{1}{2})^{i^*} \overline{\lambda}_k$ so that i^* is the smallest non-negative integer i satisfying the inequality

$$f(x^k + (\frac{1}{2})^i \overline{\lambda}_k d^k) \leq f(x^k) + \alpha (\frac{1}{2})^i \overline{\lambda}_k g(x^k)^T d^k;$$

Step 3: Let $x^{k+1} = x^k + \lambda_k d^k, k = k+1$. Go back to step 1.

In comparison with Rosen's method, the search direction d^k is more complex slightly than that of Rosen's method, but it requires to compute only one projection matrix, in addition, analysis of this method is fairly simpler than that of Rosen's method.

Analysis of the algorithm B is similar to that of Algorithm A.

Lemma 4 1). x^k is a KKT point of (1)$\Longleftrightarrow d^k = 0$;

2). If $d^k \neq 0$, then

$$g(x^k)^T d^k < 0, a_j^T d^k \leq 0, j \in J_k,$$

i.e., d^k is a feasible direction of descent.

Theorem 3 The algorithm B either stops at the KKT point x^k of the problem (17) in finite iteration, or generates an infinite sequence$\{x^k\}$ whose any accumulation point x^* is a KKT point of the problem (17).

5. Numerical experiments

In this section, we carry out some limited numerical experiments based on the algorithm. The code of the proposed algorithm is written by using Matlab programming language, and run on Windows XP.

In the implementations, the parameters are chosen as $\varepsilon_0 = 1e-06, \theta = 0.3, \alpha = 0.35$. The stopping criterion is $\|d_o^k\| \leq 10^{-6}$. This algorithm has been tested on some problems from [19]. The results are summarized in the following table. For each test problem, No. is the number of the test problem in [19], NIT the number of iterations, NF the number of evaluations of the objective functions, NG the number of evaluations of scalar constraint functions, FV the final value of the objective function. In this paper, we obtain the conclusion that the algorithm stops when $d_o^k = 0$. In fact, if $d_0^k = 0$, we can prove that $u^k \geq 0$ (see Theorem 1). The numerical results show that this fact is true (see the value of u_{min}^k in the following table which means the minimal component of the multipliers u^k for the final iteration).

Problem 1 See HS03 in Ref.[19]

$$\begin{aligned} &min\ \ x_2 + 0.00001 * (x_2 - x_1)^2 \\ &s.t.\ \ x_2 \geq 0. \end{aligned}$$

The optimal solution and value in [19]:

$$x^* = (0,0)^T, f_0(x^*) = 0.$$

Problem 2 See HS10 in Ref.[19]

$$\begin{aligned} &min\ \ x_1 - x_2 \\ &s.t.\ \ -3 * x_1^2 + 2 * x_1 * x_2 + x_2^2 - 1 \geq -1. \end{aligned}$$

The optimal solution and value in [19]:

$$x^* = (1,0)^T, f_0(x^*) = -1.$$

Problem 3 See HS22 in Ref.[19]

$$\begin{aligned} &min\ \ (x_1 - 2)^2 + (x_2 - 1)^2 \\ &s.t.\ \ x_1 + x_2 - 2 \leq 0, \\ &\qquad x_1^2 - x_2 \leq 0. \end{aligned}$$

The optimal solution and value in [19]:

$$x^* = (1,1)^T, f_0(x^*) = 1.$$

Problem 4 See HS29 in Ref.[19]

$$\begin{aligned} &min\ \ -x_1 * x_2 * x_3 \\ &s.t.\ \ x_1^2 + 2 * x_2^2 + 4 * x_3^2 \leq 48 \end{aligned}$$

The optimal solution and value in [19]:

$$x^* = (4, 2.82843, 2)^T, f_0(x^*) = -16 * \sqrt{2}.$$

Problem 4 See HS43 in Ref.[19]

$$\begin{aligned} &min\ \ x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4 \\ &s.t.\ \ x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_1 - x_2 + x_3 - x_4 - 8 \leq 0, \\ &\qquad x_1^2 + 2x_2^2 + x_3^2 + 2x_4^2 - x_1 - x_4 - 10 \leq 0, \\ &\qquad 2x_1^2 + x_2^2 + x_3^2 + 2x_1 - x_2 - x_4 - 5 \leq 0. \end{aligned}$$

The optimal solution and value in [19]:

$$x^* = (0,1,2,-1)^T, f_0(x^*) = -44.$$

Table: The detailed information of the results

$No.$	NIT	NF	NG	FV	$\|d_0^k\|$ / u_{min}^k
03	10	18	19	0.040353623273424	$\|d_o^k\| = 8.064125836437674e-006$ $u_{min}^k = 1.000000806412674$
10	38	76	76	-0.999999447862117	$\|d_o^k\| = 9.504722714043553e-006$ $u_{min}^k = 0.499997039731185$
22	39	76	115	1.000001836160267	$\|d_o^k\| = 1.143437043146816e-006$ $u_{min}^k = 0.666667232355497$
29	37	90	101	-22.627416993190010	$\|d_o^k\| = 6.451489138418337e-007$ $u_{min}^k = 0.707120788665736$
43	71	200	734	-43.999992576356789	$\|d_o^k\| = 7.085351411397026e-006$ $u_{min}^k = 0.000005260336491$

Acknowledgments

The author would like to thank the editors, whose constructive comments led to a considerable revision of the original paper.

References

[1] J.B.Rosen. The gradient projection method for nonlinear programming, Part 1: Linear constraints. SIAM Journal on Applied Mathematics, 8(1960):181-217.

[2] J.B.Rosen. The gradient projection method for nonlinear programming, Part 2: Nonlinear constraints. SIAM Journal on Applied Mathematics, 9(1961):514-553.

[3] J.Denel. Extensions of the continuity of point-to set maps: applications to fixed point algorithms. Mathematical Programming Study, 10(1979):48-68.

[4] D.Goldfarb. Extension of Davidson's variable metric method to maximization under linear inequality and equality constraints. SIAM Journal on Applied Mathematics, 17(1969):739-764.

[5] D.Goldfarb and L.Lapidus. Conjugate gradient method for nonlinear programming problems with linear constraints. Industrial and Engineering Chemistry Fundamentals, 7(1968):142-151.

[6] D.Z.Du. A family of gradient projection algorithms. Acta Mathematicae Applicatae Sinica (English Series) 2(1985):1-13.

[7] P.Huard. Optimization algorithms and point-to-set maps. Mathematical Programming, 8(1975):308-331.

[8] M.Yue and J.Han. A unified approach to the feasible direction methods for nonlinear programming with linear constraints. Acta Mathematidae Applicatae Sinica (English Series),1(1984):63-75.

[9] W.I.Zhangwill. Convergence conditions for nonlinear programming algorithms. Management Science, 16(1961):1-13.

[10] X.S.Zhang. An improved Rosen-Polak method. Acta Mathematicae Applicatae Sinica(in Chinese),2(1979):257-267.

[11] D.Z.Du. A modification of Rosen-Polak's algorithm. Kexue Tonbao,28(1983):301-305.

[12] D.Z.Du and X.S.Zhang. A convergence theorem for Rosen's gradient projection method. Mathematical Programming,36(1986):135-144.

[13] X.S.Zhang. Discussion on Polak's algorithm for nonlinear programming. Acta Mathematicae Applicatae Sinica(in Chinese), 4(1981):1-13.

[14] E.Polak. Computational Methods in Optimization.Academic Press,New York,1971:185-205.

[15] Mordecai Avriel. Nonlinear Programming Analysis and Methods. Englewood Cliffs, Prentice-Hall,1976:449-454.

[16] D.Z.Du. A gradient projection algorithm for nonlinear constraints. Acta Mathematicae Applicatae sinica(in Chinese),1(1985):7-16.

[17] Z.Y.Gao, G.P.He, and F.Wu. The algorithm of sequential system of linear equation for arbitrary initial point. Science in China (Series A), 27,1(1997):24-33.

[18] G.L.Zhou. A modified SQP method and its global convergence. Jouunal of Global optimization, 11(1997):193-205.

[19] W. Hock, and K. Schittkowski. *Test Examples for Nonlinear Programming Codes.* Lecture Notes in Economics and Mathematical Systems 187, Springer, Berlin, 1981.