Journal of Computational Mathematics, Vol.20, No.3, 2002, 301–312.

SEQUENTIAL SYSTEMS OF LINEAR EQUATIONS ALGORITHM FOR NONLINEAR OPTIMIZATION PROBLEMS – INEQUALITY CONSTRAINED PROBLEMS*1)

Zi-you Gao
(*School of Traffic and Transportation, Northern Jiaotong University, Beijing 100044, China*)

Tian-de Guo
(*Institute for Loo-Keng Hua Applied Mathematics and Information Sciences, Graduate School of Chinese Academy of Sciences, Beijing 100039, China*)

Guo-ping He Fang Wu
(*Institute of Applied Mathematics, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, Beijing 100080, China*)

Abstract

In this paper, a new superlinearly convergent algorithm of sequential systems of linear equations (SSLE) for nonlinear optimization problems with inequality constraints is proposed. Since the new algorithm only needs to solve several systems of linear equations having a same coefficient matrix per iteration, the computation amount of the algorithm is much less than that of the existing SQP algorithms per iteration. Moreover, for the SQP type algorithms, there exist so-called inconsistent problems, i.e., quadratic programming subproblems of the SQP algorithms may not have a solution at some iterations, but this phenomenon will not occur with the SSLE algorithms because the related systems of linear equations always have solutions. Some numerical results are reported.

Key words: Optimization, Inequality constraints, Algorithms, Sequential systems of linear equations, Coefficient matrices, Superlinear convergence.

1. Introduction

In this paper, we consider the following nonlinear programming problem with inequality constraints:

$$(IP) \qquad \begin{array}{ll} min & f(x) \\ s.t. & g_j(x) \le 0, \quad j = 1, 2, \cdots, m. \end{array}$$

where $x = (x_1, \cdots, x_n)^T \in E^n$, the index set $I = \{1, 2, \cdots, m\}$ and $f : E^n \to E$ and $g_j : E^n \to E$ $(j = 1, \cdots, m)$ are all real-valued functions.

Since the algorithms of Sequential Quadratic Programming (i.e. SQP) generally have good superlinear convergence properties, they are currently considered to be one of the most effective approaches for solving nonlinear programming problems with nonlinear constraints, and have been widely studied by many authors. (See, e.g. [5-8]). However, most of the SQP algorithms have two serious shortcomings: (1) In order to obtain a search direction, one must solve one or more quadratic programming subproblems per iteration, general speaking, the computation amount of this type of algorithms is very large. In addition, it is difficult to use some good

* Received May 23, 1999; Final revised March 28, 2001.

1)This research was supported by the National Natural Science Foundation of China(19571001, 19971002, 79970014) and Cross-century Excellent Personnel Award and Teaching and Research Award Program for Outstanding Young Teachers in High Education, Ministry of Education, China.

sparse and symmetric properties in solving quadratic programming subproblems, this may restrict the application of the SQP type algorithms, especially for large scale problems; (2) The SQP algorithms require that the related quadratic programming subproblems must be solvable per iteration, however, it is difficult to be satisfied in general. So it is desirable to design some algorithms which can avoid these shortcomings for nonlinear optimization problems.

In Ref.[3], E.R. Panier, A,L.Tits and J.N. Herskovits gave a feasible algorithm which is two-step superlinear convergent. They try to replace quadratic programming problems by systems of linear equations for overcoming the difficulties encountered in the SQP methods. The algorithm in Ref.[3] needs to solve two linear systems and a quadratic subproblem at each iteration, and the initial point of the algorithm must be an interior point. Particularly, in order to obtain the global convergence, Ref.[3] needs two strong assumptions (i.e. an interior point is assumed, and the number of stationary points is finite.). In addition, the algorithm of Ref.[3] can not be used to deal with problems having equality constraints.

In order to overcome these shortcomings just mentioned for the SQP algorithms for solving constrained optimization problems, through improving the algorithm in Ref.[3], a new superlinearly convergent algorithm of sequential systems of linear equations (SSLE) for inequality constrained optimization problems is proposed in this paper. Comparing with the SQP algorithms recently suggested, the new algorithm has three main advantages: (1) The new algorithm is completely QP-free, more precisely, the algorithm only needs to solve four systems of linear equations having a same coefficient matrix per iteration, the computation amount of the algorithm is much less than that of the existing SQP algorithms per iteration; (2) The iterating points generated by the algorithm are feasible; (3) The rate of convergence is one-step superlinear under much milder assumptions than that in Ref.[3].

The proposed method is based on the following observation. Let $\{d_k^0\}$ be a sequence generated by the following linear system in $(d,\,\lambda)$:

$$H_k d_k^0 + \nabla f(x_k) + \sum_{j=1}^{m} \lambda_{k,j}^0 \nabla g_j(x_k) = 0, \tag{1.1}$$

$$\mu_{k,j} \nabla g_j(x_k)^T d_k^0 + \lambda_{k,j}^0 g_j(x_k) = 0, \qquad (j \in I), \tag{1.2}$$

where H_k is an estimate of the Hessian of $L(x,\,\lambda) = f(x) + \sum_{j=1}^{m} \lambda_j g_j(x)$, x_k the current estimate of a solution x^*, $x_k + d_k^0$ the next estimate, μ_k the current estimate of the Kuhn-Tucker multiplier vector associated with x^*, and λ_k^0 the next estimate of this vector. Locally, the system (1.1)-(1.2) is a higher order perturbation of the following quadratic programming (SQP):

$$\begin{array}{ll} min & \frac{1}{2} d^T H_k d + \nabla f(x_k)^T d \\ s.t. & g_j(x_k) + \nabla g_j(x_k)^T d \le 0, \qquad j = 1, \cdots, m, \end{array}$$

according to the theorem 4.6 of Ref.[3], we know that the sequence defined by $x_{k+1} = x_k + d_k^0$ have superlinear convergence. Thus, d_k^0 is obtained as first direction. However, d_k^0 is not entirely suitable as a search direction. Indeed, although d_k^0 is a descent direction, it may be zero at some iterates which are not K-T point of (IP). This effect can be avoided if one substitutes in each right-hand side of (1.2) a negative number v_j, $(j \in I)$, but v_j must tend to zero faster than d_k^0 in order for the convergence properties of $\{x_k\}$ to be preserved, we let v_j be $(\lambda_{k,j}^0)$ or $(-\lambda_{k,j}^0) g_j(x_k) ||d_k^0||^2$. Thus, after d_k^0 is obtained, a new direction d_k^1 would be computed by solving the linear system

$$\begin{aligned} & H_k d_k^1 + \nabla f(x_k) + \sum_{j=1}^{m} \lambda_{k,j}^1 \nabla g_j(x_k) = 0, \\ & \mu_{k,j} \nabla g_j(x_k)^T d_k^1 + \lambda_{k,j}^1 g_j(x_k) = \mu_{k,j} v_{k,j}, \qquad (j \in I), \end{aligned} \tag{1.3}$$

Lemma 3.4 below shows that d_k^1 is a strict descent direction of the objective function $f(x)$. Unfortunately, if any $g_j(x_k)$ becomes very close to 0, (1.3) forces d_k^1 to tend to a direction tangent to the feasible set $X = \{x \mid g_j(x) \leq 0 \quad (j \in I)\}$. Since feasibility of all iterates is required, this may result in a collapse of the step length, and convergence can not be guaranteed. We thus propose to use a search direction d_k, which is solution of the following linear system:

$$\begin{aligned} &H_k(d_k - d_k^1) + \sum_{j=1}^{m} (\lambda_{k,j} - \lambda_{k,j}^1) \nabla g_j(x_k) = 0, \\ &\mu_{k,j} \nabla g_j(x_k)^T (d_k - d_k^1) + (\lambda_{k,j} - \lambda_{k,j}^1) g_j(x_k) = -\rho_k \|d_k^1\|^\eta \mu_{k,j}, \ (j \in I), \end{aligned} \tag{1.4}$$

where ρ_k is a positive quantity carefully chosen and $\eta > 2$. It turns out that, with such a search direction, the basic convergence properties of the iteration sequence $\{x_k\}$ are preserved.

Maratos [4] pointed out that, in the case of the SQP iteration with line search based on the decrease of an exact penalty function, the unit step may not be accepted close to a solution. To prevent such occurrence, a new linear system as following has to be solved, its solution $\hat{d}_k$ can be viewed as a second order correction of d_k:

$$\begin{aligned} &H_k(\hat{d}_k - d_k) + \sum_{j=1}^{m} \hat{\lambda}_{k,j} \nabla g_j(x_k) = 0, \\ &\mu_{k,j} \nabla g_j(x_k)^T (\hat{d}_k - d_k) + \hat{\lambda}_{k,j} g_j(x_k) = -\phi_k + \mu_{k,j} h_{k,j}, \qquad (j \in I), \end{aligned} \tag{1.5}$$

where $h_{k,j} = g_j(x_k + d_k)$ or $h_{k,j} = 0$, and the positive quantity ϕ_k is chosen to make a further "bending" of the search direction $\hat{d}_k$ in order for obtaining the feasibility of $x_k + \hat{d}_k$.

A careful implementation of the ideas just put forth dose indeed allow achievement of the properties claimed. The proposed algorithm is stated in section 2. In section 3, the global convergence is established. The superlinear convergence of the new algorithm is analyzed in section 4. Implementation issues are discussed in section 5. Some numerical experiments are reported in section 6.

2. Algorithm A

Throughout this paper, we will use the following notations $X = \{x \mid g_j(x) \leq 0 \ (j = 1, 2, \cdots, m)\}$, $G = G(x) = \operatorname{diag}(g_1(x), \cdots, g_m(x))$, $A = A(x) = (\nabla g_1(x), \cdots, \nabla g_m(x))$, $e = (1, 1, \cdots, 1)^T \in E^m$. The set of active indices at a point x is defined as

$$I(x) = \{j \mid g_j(x) = 0, \qquad 1 \leq j \leq m\}. \tag{2.1}$$

The Euclidean norm of any vector v will be denoted by $\|v\|$, the corresponding induced norm of a matrix M by $\|M\|$, and the cardinality of any finite set I by $|I|$. A point $x \in X$ is said to be stationary for (IP) if it is feasible and the equalities

$$\begin{cases} \nabla f(x) + A(x)\lambda = 0 \\ G(x) \operatorname{diag}(\lambda_1, \cdots, \lambda_m) = 0 \end{cases}$$

hold for some multiplier vector $\lambda = (\lambda_1, \cdots, \lambda_m)^T$. If, moreover, the multipliers are all non-negative, the point $x \in X$ is said to be a Kuhn-Tucker (K-T) point of problem (IP) and the multiplier vector λ is called the Lagrange multiplier vector associated with the K-T point x.

Throughout this paper, the following two assumptions are assumed:

A1. The set X is not empty.

A2. The functions f and g_j $(j = 1, 2, \cdots, m)$ are all continuously differentiable.

Now, Algorithm for the solution of (IP) can be stated as follows.

ALGORITHM A.

Parameters. $\alpha \in (0, \frac{1}{2})$, $\beta \in (0, 1)$, $\theta \in (0, 1)$, $\eta > 2$, $\tau \in (2, 3)$, $\gamma \in (0, 1)$, $\overline{\mu} > 0$.

Data. $x_0 \in X$, $H_0 \in E^{n\times n}$ is a positive definite matrix; $0 < \mu_{0j} \le \overline{\mu}$ $(j = 1, 2, \cdots, m)$.

Step 0. Initializatin. Set $k = 0$.

Step 1. Computation of a search direction.

(1) Let (d_k^0, λ_k^0) be the unique solution of the following system of linear equations in (d, λ):

$$\text{(LS1)} \qquad F(x_k, H_k, \mu_k)\begin{pmatrix} d \\ \lambda \end{pmatrix} = -\begin{pmatrix} \nabla f(x) \\ 0 \end{pmatrix}$$

where $F(x, H, \mu) = \begin{pmatrix} H & A \\ MA^T & G \end{pmatrix}$, and $M = \operatorname{diag}(\mu_1, \cdots, \mu_m)$. if $d_k^0 = 0$ and $\lambda_k^0 \ge 0$, then stop.

(2) Let (d_k^1, λ_k^1) be the unique solution of the following system of linear equations in (d, λ):

$$\text{(LS2)} \qquad F(x_k, H_k, \mu_k)\begin{pmatrix} d \\ \lambda \end{pmatrix} = \begin{pmatrix} -\nabla f(x_k) \\ M_k v_k \end{pmatrix}$$

where $v_k = (v_{k1}, \cdots, v_{km})^T$ and $v_{kj} = \begin{cases} \lambda_{kj}^0, & \text{if } \lambda_{kj}^0 \le 0; \\ -\lambda_{kj}^0 g_j(x_k)\|d_k^0\|^2, & \text{if } \lambda_{kj}^0 > 0. \end{cases}$

(3) Let (d_k, λ_k) be the unique solution of the following system of linear equations in (d, λ):

$$\text{(LS3)} \qquad F(x_k, H_k, \mu_k)\begin{pmatrix} d - d_k^1 \\ \lambda - \lambda_k^1 \end{pmatrix} = -\rho_k \|d_k^1\|^\eta \begin{pmatrix} 0 \\ \mu_k \end{pmatrix}$$

where $\rho_k = \dfrac{(\theta - 1)\cdot \nabla f(x_k)^T d_k^1}{|\sum_{j=1}^m \lambda_{kj}^0| \cdot \|d_k^1\|^\eta + 1}$.

(4) Let $(\hat{d}_k, \hat{\lambda}_k)$ be the unique solution of the following system of linear equations in (d, λ):

$$\text{(LS4)} \qquad F(x_k, H_k, \mu_k)\begin{pmatrix} d - d_k \\ \lambda \end{pmatrix} = -\begin{pmatrix} 0 \\ \phi_k e + h_k \end{pmatrix}$$

where $h_k = (h_{k1}, \cdots, h_{km})^T$,

$$h_{kj} = \begin{cases} \mu_{kj} g_j(x_k + d_k), & \text{if } j \in I_k = \{j \mid -\lambda_{kj}^0 \le g_j(x_k)\} \\ 0, & \text{if } j \notin I_k \end{cases}$$

and $\phi_k = \max \left\{ \|d_k\|^\tau,\ \max_{\substack{j\in I_k \\ \lambda_{kj}\ne 0}} \left|\dfrac{\mu_{kj}}{\lambda_{kj}} - 1\right|^\gamma \cdot \|d_k\|^2 \right\}$. If $I_k = \phi$ or $\|\hat{d}_k - d_k\| > \|d_k\|$, set $\hat{d}_k = d_k$.

Step 2. Line Search. Set t_k to be the first number t of the sequence $\{1, \beta, \beta^2, \cdots\}$ satisfying $f(x_k + td_k + t^2(\hat{d}_k - d_k)) \le f(x_k) + \alpha t \nabla f(x_k)^T d_k$, $g_j(x_k + td_k + t^2(\hat{d}_k - d_k)) \le 0$, $(j = 1, 2, \cdots, m)$.

Step 3. Updates. Let H_{k+1} be a new positive definite approximation of the Hessian matrix (see Section 5). Set $\mu_{k+1,j} = \min\left\{\max\left(\lambda_{kj}^0, \|d_k\|\right), \overline{\mu}\right\}$ $(j = 1, 2, \cdots, m)$, $x_{k+1} = x_k + t_k d_k + t_k^2(\hat{d}_k - d_k)$, $k := k + 1$ then return back to Step 1.

3. Global Convergence

In this section, we will first give some lemmas about several matrices related to the common coefficient matrix $F(x_k, H_k, \mu_k)$ of (LS1) to (LS4), then we will show that Algorithm A is well defined, i.e., the line search can be completed successfully. Thirdly, we will give the global convergence of Algorithm A.

In order to establish the global convergence of Algorithm A, in addition to A1 and A2, we need the following three assumptions furthermore.

A3. The set $X \cap \{x \mid f(x) \le f(x_0)\}$ is compact.

A4. For any $x \in X$, the vectors $\{\nabla g_j(x) \mid j \in I(x)\}$ are linearly independent.

A5. There exist two positive constants $\kappa_2 \ge \kappa_1 > 0$ such that $\kappa_1\|y\|^2 \le y^T H_k y \le \kappa_2\|y\|^2$ holds for all k and for all $y \in E^n$.

From Algorithm A and the assumptions above, we can obtain:

Lemma 3.1. *Given any $x \in X$, any non-negative vector $\mu \in E^n$ such that $\mu_j > 0$ $(\forall j \in I(x))$. If H is positive definite on the subspace $\{y \mid y^T\nabla g_j(x) = 0\ (\forall j \in I(x))\}$, i.e., $y^THy > 0$ ($\forall$ nonzero $y \in \{y \mid y^T\nabla g_j(x) = 0\ (\forall j \in I(x))\}$) then $F(x, H, \mu)$ is non-singular.*

Lemma 3.2. *Let $x \in X$ be a feasible point, H be a positive definite matrix.*

(i) *If $\mu > 0$, then $M^{-1}D = A^TH^{-1}A - M^{-1}G$ is positive definite.*

(ii) *If $\mu > 0$, then $F(x, H, \mu)^{-1} = \begin{pmatrix} P & B \\ MB^T & -D^{-1} \end{pmatrix}$ where $D = MA^TH^{-1}A - G$, $B = H^{-1}AD^{-1}$, $P = H^{-1} - H^{-1}AMB^T$.*

(iii) *If $\mu \ge 0$ and satisfies lemma 3.1, then P is positive semi-definite, more precisely $y^TPy \ge \|H^{1/2}Py\|^2$, $(\forall y \in E^n)$.*

From the last three equations we have $d_k^0 = -P_k\nabla f(x_k)$, $\lambda_k^0 = -M_kB_k^T\ \nabla f(x_k)$, $d_k^1 = d_k^0 + B_kM_kv_k$, $\lambda_k^1 = \lambda_k^0 - D_k^{-1}M_kv_k$, $d_k = d_k^1 - \rho_k\|d_k^1\|^\eta B_k\mu_k$, $\lambda_k = \lambda_k^1 + \rho_k\|d_k^1\|^\eta D_k^{-1}\mu_k$, where ρ_k is given by (2.4) and $\eta > 2$ is a given parameter.

From (LS1) to (LS3) and lemma 3.1 and 3.2, we can get the following conclusions.

Lemma 3.3. *If Algorithm A stops at a point x_k with $d_k^0 = 0$ and $\lambda_k^0 \ge 0$, then x_k is a K-T point of (IP), and λ_k^0 is the Lagrange multiplier vector associated with x_k.*

Lemma 3.4. *If $x_k \in X$ and is not a K-T point of (IP), then d_k^1 satisfies*

(i) $\nabla g_j(x_k)^Td_k^1 \le 0 \qquad (\forall j \in I(x_k))$;

(ii) $\nabla f(x_k)^Td_k^1 \le -\|H_k^{1/2}d_k^0\|^2$

$$- \sum_{\substack{j\in I \\ \lambda_{kj}^0 \le 0}} (\lambda_{kj}^0)^2 - \sum_{\substack{j\in I \\ \lambda_{kj}^0 > 0}} \left(\lambda_{kj}^0\right)^2(-g_j(x_k))\|d_k^0\|^2 < 0; \tag{3.1}$$

(iii) $d_k^1 \ne 0$.

Lemma 3.5. *If $x_k \in X$ and is not a K-T point of (IP), then d_k satisfies* (i) $\nabla f(x_k)^Td_k \le \theta \cdot \nabla f(x_k)^Td_k^1 < 0$; (ii) $\nabla g_j(x_k)^Td_k < 0 \qquad (\forall j \in I(x_k))$; (iii) $d_k \ne 0$ *and* $\mu_{k+1} > 0$.

It follows that

Theorem 3.6. *Algorithm A is well defined.*

Lemma 3.7. (i) *The sequences $\{x_k\}$ and $\{d_k^0\}$ are bounded.* (ii) *If $\{x_k\}_K \to x^*$, then $\{\lambda_k^0\}_K$ is bounded.*

Lemma 3.8. *Let x^* be a limit point of the point sequence $\{x_k\}$ generated by Algorithm A and $\{x_k\}_K \to x^*$. If $\{d_k\}_K \to 0$, then x^* is a K-T point of (IP) and $\{\lambda_k^0\}_K \to \lambda^*$, where λ^* is the unique Lagrange multiplier vector associated with x^*. Moreover, if $\lambda_j^* \le \overline{\mu}$ for all j, then $\{\mu_{k+1}\}_K \to \lambda^*$.*

Lemma 3.9. *Let x^* be a limit point of the sequence $\{x_k\}$ generated by Algorithm A, and suppose $\{x_k\}_K \to x^*$. If $\inf\{\|d_{k-1}\|\}_K = 0$, then x^* is a K-T point of (IP).*

Lemma 3.10. *If $\{x_k\}_K \to x^*$, $\{\mu_k\}_K \to \mu^*$, and $\mu_j^* > 0$ $(\forall j \in I(x^*))$, then there exists a positive constant $C > 0$ such that for all $k \in K$,*

$$\|d_k^1 - d_k^0\| \le C\|v_k\|, \qquad \|\lambda_k^1 - \lambda_k^0\| \le C\|v_k\|, \tag{3.2}$$

$$\|d_k - d_k^1\| \le C\|d_k^1\|^\eta, \qquad \|\lambda_k - \lambda_k^1\| \le C\|d_k^1\|^\eta, \tag{3.3}$$

Lemma 3.11. *Let x^* be a limit point of the sequence $\{x_k\}$ generated by Algorithm A and suppose $\{x_k\}_K \to x^*$. If* $\inf\{\|d_{k-1}\|\}_K > 0$, *then* (i) $\{d_k\}_K \to 0$; (ii) x^* *is a K-T point of (IP).*

Proof. Suppose by contradiction that there exists a subset $K' \subseteq K$ such that $\{\|d_k\|\}_{K'} \to \zeta > 0$. From the boundness of $\|H_k\|$ and μ_k, we can suppose without loss of generality that $\{H_k\}_{K'} \to H^*$ and $\{\mu_k\}_{K'} \to \mu^*$. From the definition of μ_{k+1} and from the assumption of d_{k-1} in this lemma, we can see that all components of μ^* are strictly positive.

From (3.2), Lemma 3.7, $\{\|d_k^0\|\}_{K'}$, $\{\|d_k^1\|\}_{K'}$ and hence $\{\rho_k\}_{K'}$ are all bounded. Again, we can suppose without loss of generality that $\{\rho_k\}_{K'} \to \rho^*$.

Now, we are going to prove that the sequences $\{\nabla f(x_k)^T d_k^1\}_{K'}$ and $\{\|d_k^1\|\}_{K'}$ are bounded away from zero. In fact, if$\{\nabla f(x_k)^T d_k^1\}_{k''} \to 0$ for some subset $k'' \subseteq K'$, then from (3.1), we have $\{\|d_k^0\|\}_{K''} \to 0$ and $\left\{ \sum_{\substack{j\in I\\ \lambda_{kj}^0 \le 0}} (\lambda_{kj}^0)^2 \right\}_{K''} \to 0$ which implies that $\{v_k\}_{k''} \to 0$ and $\{d_k^1\}_{k''} \to 0$, and hence from Lemma 3.10, we have $\{d_k\}_{k''} \to 0$ which contradicts the hypothesis that $\{\|d_k\|\}_{k''} \to \zeta > 0$. Thus, there exists a positive number $\xi_1 > 0$ such that $\nabla f(x_k)^T d_k^1 \le -\xi_1 < 0$, $\|d_k^1\| \ge \xi_1 > 0$, $\rho_k \ge \xi_1 > 0$, hold for all $k \in K'$.

Since (d_k, λ_k) and (d_k^1, λ_k^1) are solutions of (LS3) and (LS2) respectively, we have

$$A_k^T d_k = A_k^T d_k^1 - M_k^1 G_k(\lambda_k - \lambda_k^1) - \rho_k \|d_k^1\|^\eta e = v_k - M_k^{-1} G_k \lambda_k - \rho_k \|d_k^1\|^\eta e\,,$$

and hence $\nabla g_j(x_k)^T d_k = v_{kj} - \mu_{kj}^{-1} g_j(x_k)\lambda_{kj} - \rho_k \|d_k^1\|^\eta$, $\forall j \in I$. From lemma 3.7 and the definition of v_k, we know $\{v_k\}_{K''}$ is bounded. Suppose that $\{v_k\}_{K''} \to v^*$,from the definition of v_k, it follows that $v_j^* \le 0$, $\forall i \in I(x^*)$. Thus, from the definition of v_k and ρ_k, there exists a positive number $\xi_2 > 0$ such that $\nabla f(x_k)^T d_k \le -\xi_2 < 0$, $\nabla g_j(x_k)^T d_k \le -\xi_2 < 0$ $(\forall j \in I(x^*))$, $g_j(x_k) \le -\xi_2 < 0$ $(\forall j \notin I(x^*))$ hold for all sufficiently large $k \in K'$.

Now, from the identity $f(x_k + td_k + t^2(\hat{d}_k - d_k)) = f(x_k) + \int_0^1 t\nabla f(x_k + t\xi d_k + t^2\xi^2(\hat{d}_k - d_k))^T(d_k + 2t\xi(\hat{d}_k - d_k))d\xi$, it follows that, for $k \in K'$ large enough,

$$\begin{aligned}
&f(x_k + td_k + t^2(\hat{d}_k - d_k)) - f(x_k) - \alpha t \nabla f(x_k)^T d_k \\
=&t\left\{ \int_0^1 [\nabla f(x_k + t\xi d_k + t^2\xi^2(\hat{d}_k - d_k))^T (d_k + 2t\xi(\hat{d}_k - d_k)) \right.\\
&\left. - \nabla f(x_k)^T d_k] d\xi + (1-\alpha)\nabla f(x_k)^T d_k \right\} \\
\le& t\Big\{ \sup_{0\le\xi\le1} \big\|\nabla f(x_k + t\xi d_k + t^2\xi^2(\hat{d}_k - d_k)) - \nabla f(x_k)\big\| \cdot \|d_k\| \\
&+ 2t \sup_{0\le\xi\le1} \Big\|\nabla f(x_k + t\xi d_k + t^2\xi^2(\hat{d}_k - d_k))\Big\| \cdot \|\hat{d}_k - d_k\| - (1-\alpha)\xi_2 \Big\}.
\end{aligned}$$

Since d_k and $\hat{d}_k - d_k$ are bounded and $f \in C^1$, this ensures that there exists $t_0 > 0$, independent of k, such that for $t \in [0, t_0]$, $k \in K'$, k large enough,

$$f(x_k + td_k + t^2(\hat{d}_k - d_k)) - f(x_k) - \alpha t\nabla f(x_k)^T d_k \le 0.$$

Similarly, for $k \in K'$, k large enough, $t > 0$ and $j \in I(x^*)$, it holds

$$\begin{aligned}
&g_j(x_k + td_k + t^2(\hat{d}_k - d_k)) - g_j(x_k) \\
\le& t\{ \sup_{0\le t\le1} \|\nabla g_j(x_k + t\xi d_k + t^2\xi^2(\hat{d}_k - d_k)) - \nabla g_j(x_k)\| \cdot \|d_k\| \\
&+ 2t \sup_{0\le t\le1} \|\nabla g_j(x_k + t\xi d_k + t^2\xi^2(\hat{d}_k - d_k))\| \cdot \|\hat{d}_k - d_k\| - \xi_2 \},
\end{aligned}$$

so that there exists $t_j > 0$, independent of k, such that, for $t \in [0, t_j]$, $k \in K'$, k large enough, $g_j(x_k+td_k+t^2(\hat{d}_k-d_k)) \leq 0$. Also, since $g_j(x)$ is continuous, there exists $t_j > 0$, independent of k, such that, for $t \in [0, t_j]$, $k \in K'$, k large enough, and $j \notin I(x^*)$ $g_j(x_k+td_k+t^2(\hat{d}_k-d_k)) \leq -\xi_2/2 < 0$. Thus, if we let $\underline{t} = \min(t_0, t_1, \cdots, t_m)$, then step2 holds for $t_k \geq \beta\underline{t}$, $k \in K', k$ large enough. Now, for $k \in K'$, k large enough, we have $f(x_{k+1}) \leq f(x_k) + \alpha t_k \nabla f(x_k)^T d_k \leq f(x_k) - \alpha\underline{t}\xi_2$, which contradicts the fact that $\{f(x_k)\}_{K'} \to f(x^*)$ and our lemma is proved.

Now we come to the folowing global convergence theorem

Theorem 3.12. *Algorithm A either stops at a K-T point or generates an infinite sequence whose limit points are all K-T points of (IP).*

Proof. It is only a combination of Lemmas 3.3, 3.9 and 3.11.

4. Rate of Convergence

Now we turn our attention to the convergence rate of Algorithm A. For this purpose, we now strengthen the regularity assumptions on the functions involved.

A2$'$. The functions f, $g_j (j = 1, 2, \cdots, m)$ are twicely continuously differentiable.

A6. The sequence $\{x_k\}$ generated by Algorithm A possesses a limit point x^* (in view of Theorem 3.12, a K-T point) at which (i) second order sufficient conditions and strict complementary slackness hold, i.e., the Lagrange multiplier vector $\lambda^* \in E^m$ satisfies $\lambda_j^* > 0$, $(\forall j \in I(x^*))$ and the Hessian matrix $\nabla^2_{xx} L(x^*, \lambda^*)$ of the Lagrange function $L(x, \lambda) = f(x) + \sum_{j=1}^m \lambda_j g_j(x)$ is positive definite on the subspace $\{h \,|\, h^T \nabla g_j(x^*) = 0, (\forall j \in I(x^*))\}$; and (ii) the multiplier vector λ^* satisfies $\lambda_j^* \leq \overline{\mu} (j = 1, 2, \cdots, m)$.

Lemma 4.1. *The point x^* in A6 is an isolated K-T point, i.e., there is a positive constant $\varepsilon > 0$ such that there aren't any other K-T points in the ball $B(x^*, \varepsilon)$ except x^* itself.*

Then, similar to Ref. [3], we have

Lemma 4.2. *If there exists a subset of indices K such that $\{x_{k-1}\}_K \to x^*$ and $\{x_k\}_K \to x^*$, where x^* is defined as in A6, then $\{d_k\}_K \to 0$.*

Theorem 4.3. *We have* (i) *$\{x_k\} \to x^*$, i.e., the whole point sequence $\{x_k\}$ converges to the K-T point x^*;* (ii) *$\{d_k\} \to 0$, $\{\lambda_k^0\} \to \lambda^*$ and $\{\mu_k\} \to \lambda^*$;* (iii) *$I_k = I(x^*)$ for all sufficiently large k.*

Lemma 4.4. *We have* $\lambda_k^1 \to \lambda^*, \quad \lambda_k \to \lambda^*, \quad d_k^0 \to 0, \quad d_k^1 \to 0 \qquad (k \to \infty)$.

Lemma 4.5. *We have* $\|d_k^1 - d_k^0\| = \circ(\|d_k^0\|), \qquad \|d_k - d_k^0\| = \circ(\|d_k^0\|)$.

Now set $R^* = (\nabla g_j(x^*) \,|\, j \in I(x^*))$, $\quad Q^* = I - R^*(R^{*T}R^*)^{-1}R^{*T}$, $R_k = (\nabla g_j(x_k) \,|\, j \in I(x^*))$, $\quad Q_k = I - R_k(R_k^T R_k)^{-1}R_k^T$, $\quad p_k = d_k^0 - Q_k d_k^0$,

Lemma 4.6. *We have*

$$\|p_k\| = O\left(\sum_{j \in I(x^*)} \left(\frac{\lambda_{kj}^0}{\mu_{kj}} g_j(x_k)\right)^2\right)^{1/2}.$$

Lemma 4.7. *We have*

$$\|\hat{d}_k - d_k\| = O\left(\max\left\{\|d_k\|^2, \max_{j \in I(x^*)} \left|\frac{\mu_{kj}}{\lambda_{kj}} - 1\right| \cdot \|d_k\|\right\}\right) = \circ(\|d_k\|),$$

$$\|\hat{\lambda}_k\| = O\left(\max\left\{\|d_k\|^2, \max_{j \in I(x^*)} \left|\frac{\mu_{kj}}{\lambda_{kj}} - 1\right| \cdot \|d_k\|\right\}\right) = \circ(\|d_k\|),$$

Lemma 4.8. *We have* $\|\hat{d}_k - d_k^0\| = \circ(\|d_k\|) = \circ(\|d_k^0\|)$.

We are now ready to show that the use of correction $\hat{d}_k - d_k$ on the search direction causes the line search to accept a unit step ($t_k = 1$) when k is sufficiently large. Thus any Maratos-like effect will not happen. For this purpose, as Ref.[3], we will make the follwing asssumption.

A7. The matrices H_k are positive definite and satisfy

$$\lim_{k\to\infty}\frac{\|Q_k(H_k-\nabla^2_{xx}L(x^*,\,\lambda^*))d^0_k\|}{\|d^0_k\|}=0,$$

where $L(x,\,\lambda)=f(x)+\sum_{j=1}^{m}\lambda_j g_j(x)$ is the Lagrangian function.

Lemma 4.9. *$t_k=1$ for all k large enough.*

Proof. Under the state of assumptions, the conlusion can be deduced from Theroem 4.3 and Lemma 4.4-4.8. The detail of the proof can be founded in Ref.[16].

Theorem 4.10. *Under the stated assumptions, the rate of convergence is one-step superlinear, i.e., the following relation holds*

$$\lim_{k\to\infty}\frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|}=0. \tag{4.1}$$

Proof. In view of Lemma 4.9, the line search gives a step $t_k=1$ for k large enough. Two successive iterates are thus related by $x_{k+1}-x_k=\hat{d}_k$. Obviously, we have from Lemma 4.6 $x_{k+1}-x_k=d^0_k+\circ(\|d^0_k\|)$. Let us consider the quadratic programming in d:

$$\begin{aligned}&\min\quad f(x_k)+\nabla f(x_k)^Td+\frac{1}{2}d^TH_kd\\&s.t.\quad \nabla g_j(x_k)^Td+g_j(x_k)\le 0\qquad j=1,\,2,\cdots,m.\end{aligned}\tag{4.2}$$

In Ref.[7], the one step superlinear convergence of sequence defined by $x_{k+1}-x_k=d^e_k$, where d^e_k solves (4.2) is proved under assumptions wholly similar to those of this theorem. In Ref.[3], it has been proved that, if $x_{k+1}-x_k=d^e_k+\circ(\|d^e_k\|)$, this result will still hold. Thus, it is enough to show that $d^e_k=d^0_k+\circ(\|d^0_k\|)$. But from Theorem 4.6 in Ref.[3], we can easily know $d^e_k=d^0_k+\circ(\|d^0_k\|)$. Thus, we have (4.1).

5. Implementation Issues

The first issue to be addressed is to select an updating procedure for H_k (in Step 3 of Algorithm A). The most widely choice is to use the BFGS updating formula with Powell's modification described in Ref.[6] to ensure positive definiteness. But, only two-step superlinear convergence has been proved for this procedure. An alternative version of the BFGS procedure which has some advantages over that described in Ref.[6] has been given in Ref.[8]. Here, we give a version of the BFGS procedure which is similar to that described in Ref.[8]. This procedure is simipler than that described in Ref.[8] on computation. Our procedure is

$$H_{k+1}=H_k-\frac{H_k\delta_k\delta_k^TH_k}{\delta_k^TH_k\delta_k}+\frac{y^1_ky^{1T}_k}{\delta_k^Ty^1_k}$$

where $\delta_k=x_{k+1}-x_k$, and

$$y_k=\bigtriangledown_xL(x_{k+1},\lambda_k)-\bigtriangledown_xL(x_k,\lambda_k);\quad y^1_k=y_k+a_k[e_k\delta_k+b_kR_kR_k^T\delta_k].$$

$$\begin{aligned}&e_k=\min\{||d_k||^2,\epsilon\},\quad \epsilon\in(0,1);\quad a_k=0,\,b_k=0,\quad if\quad \delta_k^Ty_k\ge e_k||\delta_k||^2;\\&a_k=1,\,b_k=0,\quad if\quad 0\le\delta_k^Ty_k<e_k||\delta_k||^2;\\&a_k=l+1,\,b_k=0,\quad if\quad 0>\delta_k^Ty_k\ge -le_k||\delta_k||^2,(l\ \text{ is a positive constant.})\\&a_k=\frac{||\delta_k||^2-\delta_k^Ty_k}{e_k||\delta_k||^2+\delta_k^TR_kR_k^T\delta_k},\ b_k=1,otherwise.\end{aligned}$$

From the procedure described above, we can see that $\delta_k^Ty^1_k>0$ at all iterations, hence positive definiteness is preserved. Moreover, we have the following important theorem.

Theorem 5.1. *Let the previous assumptions (A1-A6) be satisfied and there exists a matrix H^* such that $H_k \to H^*$, as $k \to \infty$. Then the matrix sequence $\{H_k\}$ generated by the procedure described above satisfies Assumption A7.*

The next question is that how we can efficiently solve (LS1)-(LS4) in Algorithm A. A key is given by Theorem 5.2 below.

Theorem 5.2. *$D_k = (M_k A_k^T H_k^{-1} A_k - G_k)$ is invertible and the solutions of (LS1)-(LS4) can be expressed as follows.*

$$\begin{aligned}
&\lambda_k^0 = -D_k^{-1} M_k A_k^T H_k^{-1} \bigtriangledown f(x_k);\ d_k^0 = -H_k^{-1}(\bigtriangledown f(x_k) + A_k \lambda_k^0)\\
&\lambda_k^1 = \lambda_k^0 - D_k^{-1} M_k v_k;\ d_k^1 = d_k^0 - H_k^{-1} A_k(\lambda_k^1 - \lambda_k^0)\\
&\lambda_k = \lambda_k^1 + ||d_k^1||^\eta \rho_k D_k^{-1} \mu_k;\ d_k = d_k^1 - H_k^{-1} A_k(\lambda_k - \lambda_k^1)\\
&\hat{\lambda}_k = D_k^{-1}[\phi_k e + h_k];\ \hat{d}_k = d_k^0 - H_k^{-1} A_k \hat{\lambda}_k
\end{aligned}$$

From Theorem 5.2, we know that, for the computation of the search direction d_k and correction $\hat{d}_k$, the main task is to compute the matrix D_k and its inverse. Since, from the BFGS update, the Cholesky factor associated with H_k is known, matrix D_k can be evaluated. Its inverse is not computed explicitly, or rather, a LU decomposition is performed. Thus, the total work per iteration (in addtion to function evaluations) is essentially that associated with one Cholesky factorization of size m, the number of constraints. Therfore, the total work per iteration of our algorithm is much less than that of the algorithm proposed in Ref.[3] which essentially associated with two Cholesky factorizations of size m.

6. Numerical Experiments

In order to test the given algorithm, an efficient implementation of the algorithm has been completed. In this implementation, we select $\alpha = 0.2,\ \beta = 0.5,\ \theta = 0.99,\ \eta = 2.0001,\ \tau = 2.99,\ \gamma = 0.9$ *and* $H_0 = I$ (an $n \times n$ unit matrix). H_k is updated by means of the $BFGS$ formula with some modifications as described in section 5. The following minor modifications with respect to the algorithm as described in section 2 were found to be beneficial and were implemented.

(i) Concerning the line search, similar to Ref.[13], the order in which the functions (objective and constraints) are evaluated will be alternated according to the computation results at the previous trial step. For $t = 1$, the constraints are evaluated before the objective and in the order of their indices, except that all constraints for which the multiplier in (LS_1) is nonzero are evaluated first (and the tests are terminated as soon as a violation is detected); For $t < 1$, the same rule is used, except that the function whose violation led to test termination at the previous trial step is evaluated first.

(ii) The stopping criterion in Step 1 (1) is unsuitable for implementation. Instead, execution is terminated if the norm of d_k^0 is less than a constant $\epsilon > 0$ and $\lambda_{k,j}^0 \geq 0,\ j = 1, 2, \cdots, m$. Generally, we take $\epsilon = 1.0D - 8$. In addition, if the norm of gradient of the Lagrange function at the current point with the multipliers obtained in solving (LS1) is less than ϵ, the execution is also terminated.

(iii) Since the systems of linear equations (LS1)-(LS4) have the same coefficient matrix $F(x, H, \mu)$, only one LU-decomposition is required. In the computation of L matrix and U matrix, we use the technique of column principal element decomposition. In addition, when we compute the solutions of the systems of linear equations, some correction steps will be used. That is, for a system of linear equations, if we denote its first approximation solution by y_1, then we can compute the residual of y_1 by $r_1 = b - F(x, H, \mu) y_1$, where b is the right side of the system of linear equations .

Using r_1, we compute the solution of the system of linear equations $F(x, H, \mu) z = r_1$. Denote its solution by z_1, and set $y_2 = y_1 + z_1$, then, generally, y_2 is more accurate than y_1. Similarly, we can compute the residual of y_2 by $r_2 = b - F(x, H, \mu)y_2$, and compute a more accurate solution $y_3 = y_2 + z_2$, where z_2 is the solution of the system of linear equations $F(x, H, \mu)z = r_2$.

The above correction step should be performed in at most 5 times, then we can obtain a satisfactory solution. Considering that these linear systems have the same decomposed coefficient matrix, the total computation amount is not large.

Table 1.

No	*Code*	*Size*	*NF*	*NG*	*NIT*	*FV*	*EPS*	*VC*
12	VF02AD	2,1	11	11	9	-0.300000000E+02	0.10E-05	0.0
	A		7	18	7	-0.300000000E+02	0.10E-05	0.0
	FSQP		7	14	7	-0.300000000E+02	0.10E-05	0.0
	SSLE		10	12	7	-0.299976845E+02	0.10E-05	0.0
29	VF02AD	3,1	12	12	11	-0.226274177E+02	0.10E-04	0.19E-09
	A		14	24	10	-0.226274177E+02	0.10E-04	0.0
	FSQP		11	20	10	-0.226274177E+02	0.10E-04	0.0
	SSLE		16	18	11	-0.226272698E+02	0.10E-06	0.0
30	VF02AD	3,1(6)	13	13	13	0.1000000000E+01	0.10E-06	0.73E-11
	A		14	44	13	0.1000000000E+01	0.10E-06	0.0
	FSQP		13	25	13	0.1000000000E+01	0.10E-06	0.0
	SSLE		23	31	20	0.1000000000E+01	0.10E-06	0.0
33	VF02AD	3,2(4)	5	10	5	-0.400000000E+01	0.10E-07	0.0
	A		4	17	4	-0.400000000E+01	0.10E-07	0.0
	FSQP		4	11	4	-0.400000000E+01	0.10E-07	0.0
	SSLE		12	23	11	-0.400000000E+01	0.10E-07	0.0
43	VF02AD	4,3	13	39	9	-0.440000000E+02	0.10E-04	0.15E-08
	A		10	67	10	-0.440000000E+02	0.10E-04	0.0
	FSQP		11	51	9	-0.440000000E+02	0.10E-04	0.0
	SSLE		34	54	12	-0.449998623E+02	0.10E-05	0.0
100	VF02AD	7,4	20	80	13	0.680630057E+03	0.10E-03	0.58E-10
	A		43	240	15	0.680630057E+03	0.10E-03	0.0
	FSQP		23	114	16	0.680630057E+03	0.10E-03	0.0
	SSLE		58	74	16	0.680630099E+03	0.10E-04	0.0
113	VF02AD	10,8	15	120	12	0.243065532E+02	0.10E-02	0.78E-08
	A		15	324	14	0.243065532E+02	0.10E-02	0.0
	FSQP		12	108	12	0.243065532E+02	0.10E-02	0.0
	SSLE		21	105	13	0.243064487E+02	0.10E-06	0.0
117	VF02AD	15,5(15)	16	89	16	0.323468790E+02	0.10E-03	0.45E-04
	A		28	741	22	0.323468790E+02	0.10E-03	0.0
	FSQP		20	219	19	0.323468790E+02	0.10E-03	0.0
	SSLE		71	179	29	0.323486790E+02	0.10E-05	0.0
278	NLPQL	6,6(6)	15	90	5	0.745414E+01	0.10E-06	0.4
	SSLE		17	40	10	0.784127E+01	0.10E-06	0.0
285	NLPQL	15,10	55	550	20	-0.825200E+04	0.10E-06	0.4E-07
	SSLE		11	277	11	-0.825056E+04	0.10E-06	0.0
386	NLPQL	15,11	75	825	34	-0.816437E+04	0.10E-06	0.9E-08
	SSLE		39	684	37	-0.816437E+04	0.10E-06	0.0

Experiments were conducted on all test problems from Ref.[11-12], where a feasible initial point is provided for each problem except Problem 278 and no equality constraints are present.

We compared five algorithms on those problems, namely: VF02AD of [15], A of [1], FSQP of [13], NLPQL of [14], and Algorithm A as implemented in SSLE. The results are summarized in Table 1. In that table, No is the number of the test problem in Ref.[11-12], $Size$ represents the number of variables and the number of constraints, where number in the bracket is the number of all upper and lower bounds of the variables, NF is the number of evaluations of the objective function, NG the number of evaluations of scalar constraint functions, NIT the number of iterations, FV the final value of the objective function, VC the final constraint violation and EPS the stopping criterion threshold. Some datums for Algorithms VF02AD, A and FSQP are quoted from Ref.[13], and some datums for Algorithm NLPQL are quoted from Ref.[12].

From the table, we can see that the number of iterations for the algorithms are about the same except for Problems 30, 33. For most of problems, the number of nonlinear constraint evaluations in SSLE is a little less, especially for larger size problems, but the number of objective evaluations in SSLE is a little more than other algorithms.

The computation work per iteration in SSLE is much lower than other algorithms, since only four systems of linear equations having the same coefficient matrix need to be solved. We only have to make a LU factorization per iteration. Obviously, for large scale problems, SSLE algorithm will be one of the best available "feasible iterate" algorithm. Finally, SSLE is very stable. It can efficiently solve some "ill-condition" problems, for example, Problem 278. The final constraint violation value achieved by VF02AD is larger than 0.4. Such undersirable phenomena will not occur with SSLE.

References

[1] E.R. Panier, A.L. Tits, A superlinear convergent feasible method for the solution of inequality constrained optimization problems, *SIAM J. Control and Optimization,* **25**:4 (1987), 934-950.

[2] J.N. Herskovits, A two-stage feasible directions algorithm for nonlinear constrained optimization, *Math. Prog.,* **36**:1 (1986) 19-38.

[3] E.R. Panier, A.L. Tits, J.N. Herskovits, A QP-free, globally convergent, locally superlinearly convergent algorithm for inequality constrained optimization, *SIAM J. Control and Optimization,* **26**:4 (1988), 788-811.

[4] M. Maratos, Exact penalty function algorithms for finite dimensional and optimization problems, Ph.D thesis, Imperial College of Science and Technology, London, 1978.

[5] E. Polak, D.Q. Mayne, A superlinearly convergent algorithm for constrained optimization problems, *Math. Prog. Stud.,* **16** (1982), 45-61.

[6] M.J.D. Powell, A fast algorithm for nonlinear constrained optimization calculations, in Numerical Analysis, Proceedings, Biennial conference, Dundee 1977, Lecture Notes In Math. 630, G. A. Waston, ed., Springer-Verlag, Berlin, 1978, 144-157.

[7] M.J.D. Powell, Variable metric methods for constrained optimization, in Math. Prog., A. Bachem, M. Grotschel and B. Korte, eds, Springer-Verlag, Berlin, 1983, 288-311.

[8] J.F.A. De, O. Pantoja, D.Q. Mayne, Exact penalty function algorithm with simple updating of the penalty parameter, *JOTA,* **69**:3 (1991), 441-467.

[9] M.J.D. Powell, The convergence of variable metric methods for nonlinearly constrained optimization calculations, in O.L. Mangasarian, R.R. Meyer and S.M. Robinson, eds, Nonlinear Programming 3, 1978, 27-64.

[10] Lai Yanlian, Gao Ziyou, He Guoping, A generalized gradient projection algorithm for nonlinearly optimization, *Scientia Sinica,* **9** (1992), 916-924 (in Chinese).

[11] W. Hock, K. Schittkowski, Test examples for nonlinear programming codes, in Lecture Notes in Econom. and Math. Systems, 187, Springer-Verlag, Berlin, 1981.

[12] K. Schttfkowski, More test examples for nonlinear programming codes, in Lecture Notes in Econom. and Math. Systems, 282, Springer-Verlag, Berlin, 1987.

[13] E.R. Panier, A.L. Tits, On combining feasibility, descent and superlinear convergence in inequality constrained optimization, *Math. Prog.,* **59** (1993), 261-276.

[14] W. Hock, K. Schittkowski, NLPQL: A Fortran-Subroutine solving constrained nonlinear programming, *Annals of Operations Research,* **5** (1985), 485-500.

[15] "Harwell Subrotining Library", Library Reference Manual, Harwell, England, 1985.

[16] Z.Y. Gao, G.P. He, F. Wu, Algorithm of sequential systems of linear equations for nonlinear optimization problems, Part I–Inequality constrained problems, Technical Report 94-31, Institute of Applied Mathematics, Academia Sinica, Beijing 100080, China, 1994.