Journal of Computational and Applied Mathematics 228 (2009) 313–325

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

A penalty-function-free line search SQP method for nonlinear programming[☆]

Wenjuan Xue[a,c,∗], Chungen Shen[b,c], Dingguo Pu[c]

[a] Department of Mathematics and Physics, Shanghai University of Electric Power, 200090, China

[b] Department of Applied Mathematics, Shanghai Finance University, 201209, China

[c] Department of Mathematics, Tongji University, 200092, China

ARTICLE INFO

Article history:
Received 13 November 2005
Received in revised form 13 September 2008

MSC:
90C30
65K10

Keywords:
Non-monotonicity
Line search
SQP
Global convergence
Local convergence
SOC

ABSTRACT

We propose a penalty-function-free non-monotone line search method for nonlinear optimization problems with equality and inequality constraints. This method yields global convergence without using a penalty function or a filter. Each step is required to satisfy a decrease condition for the constraint violation, as well as that for the objective function under some reasonable conditions. The proposed mechanism for accepting steps also combines the non-monotone technique on the decrease condition for the constraint violation, which leads to flexibility and an acceptance behavior comparable with filter based methods. Furthermore, it is shown that the proposed method can avoid the Maratos effect if the search directions are improved by second-order corrections (SOC). So locally superlinear convergence is achieved. We also present some numerical results which confirm the robustness and efficiency of our approach.

1. Introduction

In this paper, we consider the constrained optimization problem:

$$(\mathrm{P})\quad \begin{cases} \min & f(x) \\ \text{s.t.} & c_i(x) = 0, \quad i \in \mathcal{E}, \\ & c_i(x) \le 0, \quad i \in \mathcal{I}, \end{cases} \tag{1}$$

with twice continuously differentiable functions $f(x) : \mathbb{R}^n \to \mathbb{R}$, $c(x) = (c_1(x), \ldots, c_m(x)) : \mathbb{R}^n \to \mathbb{R}^m$. $\mathcal{E} = \{1, \ldots, m_e\}$, $\mathcal{I} = \{m_e + 1, \ldots, m\}$. The Lagrangian function associated with problem (P) is $L(x, \lambda) = f(x) + \lambda^{\mathrm{T}} c(x)$, where $\lambda \in \mathbb{R}^m$ is the corresponding Lagrangian multiplier.

The sequential quadratic programming (SQP) method has been widely used for solving problem (P) and has been investigated by many researchers [12,13,16,17]. However, traditional SQP methods may encounter the trouble of choosing problematic penalty parameters. Fletcher and Leyffer [6] proposed the filter method which is an alternative to traditional SQP methods to solve problem (P). The key point of the concept is that the trial point generated by solving a trust region SQP problem is accepted if there is a sufficient decrease of the objective function or the constraint violation. No penalty parameter needs to be chosen. So, we call it a penalty-function-free method. In addition, the computational results of the

☆ This research is supported by National Science Foundation of China (No. 10771162) and Shanghai Excellent Young Teacher Foundation.

∗ Corresponding author at: Department of Mathematics, Tongji University, 200092, China.
E-mail address: xuewenjuan@gmail.com (W. Xue).

0377-0427/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2008.09.031

algorithm in [6] are also very encouraging and subsequently the global convergence of the filter SQP methods was proved in [7,8]. There are also various methods using the filter strategy (see [1,4,5,15,22–24]).

Another penalty-function-free method is the non-monotone trust region SQP method proposed by Ulbrich and Ulbrich [21]. This algorithm is the only trust region algorithm without a penalty function or a filter. Therefore, it also does not require any penalty parameter. However, only equality constrained optimization problems are discussed in their algorithm.

In this paper, we propose a non-monotone line search SQP method, which interprets the optimization problem with general constraints as a bi-objective optimization problem. Our method uses neither a penalty function nor an augmented Lagrange function to test the acceptability of trial steps. Thus, it avoids the difficulty of choosing a proper penalty parameter. Since the filter acceptant criteria in the filter methods mentioned above require that the trial point compare with all the points in the filter, it implies that all the information about the points in the filter is needed. Storing the filter points results in a lot of memory being required in the optimization calculation. Therefore, the method that we investigate here does not use the concept of filters, either. Rather, the non-monotone line search technique is applied in our method. This strategy is comparable to filter methods with respect to the flexibility of accepting trial steps.

We also discuss the local convergence property of our method. The SQP method may suffer from the well-known Maratos effect [14]. As a remedy, Fletcher and Leyffer [6] proposed to improve the search direction by means of the second order correction (SOC) step which aims to further reduce the infeasibility, if the full step is rejected. In this paper, we show that this modification is indeed able to avoid the Maratos effect in our method.

Here, we make an extensive use of the symbols $o(\cdot)$ and $\mathcal{O}(\cdot)$. Denote $\{\eta_k\}$ and $\{\nu_k\}$ as two vanishing sequences, where $\eta_k, \nu_k \in \mathbb{R}$, k is a positive integer. If there exists a constant $C > 0$, such that $|\eta_k| \le C|\nu_k|$ for all k sufficiently large, we write $\eta_k = \mathcal{O}(\nu_k)$. If the sequence of ratios $\{\eta_k/\nu_k\}$ approaches zero when $k \to +\infty$, we write $\eta_k = o(\nu_k)$.

This paper is organized as follows. In Section 2 the algorithm is developed. We describe the decrease conditions for the objective function and the non-monotone decrease conditions for the constraint violation, which are the key ingredients of our algorithm. The global convergence of the algorithm is established in Section 3. In Section 4, we show that under some reasonable conditions the algorithm converges to a local minimum of problem (P) superlinearly. Some numerical results are reported in the final section.

2. Algorithm

Let x_k be the current iterate; the search direction is computed by solving the quadratic programming (QP) subproblem as follows:

$$\mathrm{QP}(x_k)\quad \begin{cases} \min & g_k^{\mathrm{T}} d + \dfrac{1}{2} d^{\mathrm{T}} B_k d \\ \text{s.t.} & A_i(x_k)^{\mathrm{T}} d + c_i(x_k) = 0, \quad i \in \mathcal{E}, \\ & A_i(x_k)^{\mathrm{T}} d + c_i(x_k) \le 0, \quad i \in \mathcal{I}, \end{cases} \tag{2}$$

where $g_k = g(x_k) = \nabla f(x_k)$, $A(x_k) = [A_1(x_k), \ldots, A_m(x_k)] = \nabla c(x_k)$, and $B_k \in \mathbb{R}^{n\times n}$ is symmetric positive definite.

There is a common difficulty in solving the QP subproblems. That is, the linearization of the nonlinear constraints may give rise to infeasibility of the QP subproblem. It means that $\mathrm{QP}(x_k)$ may be inconsistent. To overcome the difficulty, we define a relaxed feasible QP subproblem. We use the technique in [10] to deal with the infeasible problem (P) and the infeasible QP subproblem. Since $\mathrm{QP}(x_k)$ may be infeasible or the corresponding Lagrangian multipliers may be too large, we consider solving the following auxiliary problem:

$$\mathrm{P}(\gamma)\quad \begin{cases} \min & f(x) + \gamma e^{\mathrm{T}}(v + w) \\ \text{s.t.} & c_i(x) - v_i + w_i = 0, \quad i \in \mathcal{E}, \\ & c_i(x) - v_i + w_i \le 0, \quad i \in \mathcal{I}, \\ & v \ge 0, \qquad w \ge 0, \end{cases} \tag{3}$$

where γ is a non-negative penalty parameter and $v^{\mathrm{T}} = (v_1, \ldots, v_m)$, $w^{\mathrm{T}} = (w_1, \ldots, w_m)$, $e^{\mathrm{T}} = (1, \ldots, 1)$ with corresponding dimension. If problem (P) has a feasible solution and γ is sufficiently large, then the solutions to problem (P) and problem $\mathrm{P}(\gamma)$ are identical. Otherwise problem $\mathrm{P}(\gamma)$ tends to determine a "good" infeasible solution if γ is large enough. The choice of γ requires heuristics. We use the value $\gamma = 100\|\nabla f(\hat{x}_0)\|$, where $\hat{x}_0$ is the first iterate at which the inconsistent linearized constraints are detected. We do not stop increasing γ with the increasing factor 10 until $v = 0$, $w = 0$ or $\gamma = 10^{10}\|\nabla f(\hat{x}_0)\|$.

For simplicity, we do not introduce the quadratic model of problem $\mathrm{P}(\gamma)$ in the following part of this paper, and just assume that $\mathrm{QP}(x_k)$ is always consistent.

Let d_k be the solution of $\mathrm{QP}(x_k)$. Then the KKT conditions of $\mathrm{QP}(x_k)$ are as follows:

$$\begin{cases} g_k + B_k d_k = -A(x_k)\lambda_k, \\ c_i(x_k) + A_i(x_k)^{\mathrm{T}} d_k = 0, \quad i \in \mathcal{E}, \\ \lambda_{k,i}\left(c_i(x_k) + A_i(x_k)^{\mathrm{T}} d_k\right) = 0, \quad i \in \mathcal{I}, \\ c_i(x_k) + A_i(x_k)^{\mathrm{T}} d_k \le 0, \qquad \lambda_{k,i} \ge 0, \quad i \in \mathcal{I}, \end{cases} \tag{4}$$

where $\lambda_k = (\lambda_{k,1}, \ldots, \lambda_{k,m})^{\mathrm{T}}$ is the associated Lagrangian multiplier.

After the search direction d_k is computed, a step size $\alpha_k \in (0, 1]$ is determined in order to obtain the next iterate $x_{k+1} := x_k + \alpha_k d_k$. We propose a backtracking line search procedure, where a decreasing sequence of step size α is tried until the acceptance criteria are satisfied. The basic idea in our approach is to interpret problem (P) as a bi-objective optimization problem with two goals, i.e., minimizing the objective function $f(x)$ and the constraint violation $h(c(x)) := \sum_{i\in\mathcal{E}} |c_i(x)| + \sum_{i\in\mathcal{I}} \max\{0, c_i(x)\}$.

First of all, we consider the decrease condition for the objective function. Define $\Delta f_k = f(x_k) - f(x_k + \alpha d_k)$ and $\Delta l_k = -g_k^{\mathrm{T}} d_k$. For simplicity, let $h(c_k) = h(c(x_k))$. If

$$g_k^{\mathrm{T}} d_k \le -\xi d_k^{\mathrm{T}} B_k d_k \quad \text{and} \quad h(c_k) \le \zeta_1 \|d_k\|^{\zeta_2}, \tag{5}$$

with constants $\xi \in (0, \frac{1}{2})$, $\zeta_1 > 0$, $\zeta_2 \in (2, 3)$, then the sufficient decrease condition

$$\Delta f_k \ge \sigma\alpha\Delta l_k \tag{6}$$

is required be satisfied, where $\sigma \in (0, \frac{1}{2})$ is a constant.

Now we consider the non-monotone decrease condition for the constraint violation $h(c(x))$, we use the technique inspired by [21]. The well-known KKT conditions for problem (P) are

$$\begin{cases} g(x) + A(x)\lambda = 0, \\ c_i(x) = 0, \quad i \in \mathcal{E}, \\ \lambda_i c_i(x) = 0, \quad i \in \mathcal{I}, \\ c_i(x) \le 0, \qquad \lambda_i \ge 0, \quad i \in \mathcal{I}, \end{cases} \tag{7}$$

where $\lambda \in \mathbb{R}^n$ is the Lagrangian multiplier. Denote $Nf_k = \|A_k\lambda_k + g_k\|$. If $Nf_k \to 0$ and $h(c_k) \to 0$, then the KKT conditions are satisfied at the limit points of $\{x_k\}$ if some reasonable conditions are satisfied. In order to reduce $h(c_k)$ and Nf_k evenly, we need a slack variable T_k.

The choice of T_k is an important issue in the design of the method. It is done in such a way that the feasibility requirement is relaxed if the feasibility is much better than the stationarity, i.e., if $h(c_k) \ll Nf_k$, then a value larger than $h(c_k)$ is chosen. In order to keep minimum control over the constraint violation, we choose T_k to be not larger than some upper bound b_{j_k}. Hereby, $\{b_{j_k}\}$ is a slowly decreasing sequence tending to zero and only when T_k yields the maximum in the first term of $\mathcal{R}_k$, j_k increases, i.e., $j_{k+1} = j_k + 1$. Let $\{b_j\}$ be a sequence with

$$b_0 > 0, \quad b_j = \frac{b_0}{j+1}\ (j \ge 1), \quad b_j \to 0 \quad (j \to +\infty), \quad \text{and} \quad \frac{1}{2} \le \frac{b_{j+1}}{b_j} < 1.$$

With a fixed positive integer $l > 1$, let

$$\mathcal{M}_{l,k} = \max_{k-l+1 \le i \le k-1} h(c_i), \tag{8}$$

and

$$\mathcal{R}_k = \max\{T_k, \mathcal{M}_{l,k}\}. \tag{9}$$

For the non-monotone decrease of the constraint violation, the trial step $x_k + \alpha d_k$ is accepted as a new iterate x_{k+1} if it satisfies

$$\mathcal{R}_k - h(c_{k+1}) \ge \alpha\eta\mathcal{R}_k, \quad \eta \in \left(0, \frac{1}{2}\right). \tag{10}$$

Now, we give a complete statement of our algorithm.

Algorithm A (*Update T_k*). Initiate parameters: $0 < \eta_1, \eta_2 < \frac{1}{2}$.

▷ If $h(c_k) < \min\{\eta_1 b_{j_k}, \eta_2 Nf_k\}$, set $T_k = \min\{b_{j_k}, Nf_k\}$,
- If $T_k \ge \mathcal{M}_{l,k}$, set $j_{k+1} = j_k + 1$;
- otherwise set $j_{k+1} = j_k$.

▷ else set $T_k = h(c_k)$, $j_{k+1} = j_k$.

Algorithm B. Initiate parameters: $x_0 \in \mathbb{R}^n$, $\sigma \in (0, \frac{1}{2})$, $\eta \in (0, \frac{1}{2})$, $\xi \in (0, \frac{1}{2})$, $B_0 \in \mathbb{R}^{n\times n}$, $t \in (0, 1)$, $\zeta_1 > 0$, $\zeta_2 \in (2, 3)$, $k := 1$.

Step 1 Compute the search direction.
Compute the search direction d_k and the corresponding Lagrangian multiplier λ_k from QP(x_k).
Set $\alpha = 1$.
Step 2 Evaluate functions at x_k.
Compute $f(x_k)$, $c(x_k)$, $g(x_k)$, $A(x_k)$.
Step 3 Check for termination.
If the KKT conditions of problem (P) are satisfied, stop.

Step 4 Update T_k.
Update T_k by calling Algorithm A.
Step 5 Backtracking line search.
If (5) holds, go to 5.1, else go to 5.2.
5.1 If $\mathcal{R}_k - h(c(x_k + \alpha d_k)) < \alpha\eta\mathcal{R}_k$ or $\Delta f_k < \sigma\alpha\Delta l_k$,
Case 1: $\alpha \neq 1$, then set $\alpha = t\alpha$, and go to Step 5.
Case 2: $\alpha = 1$, go to 5.3,
Otherwise set $\alpha_k = \alpha$, $x_{k+1} = x_k + \alpha_k d_k$, go to Step 6.
5.2 If $\mathcal{R}_k - h(c(x_k + \alpha d_k)) \geq \alpha\eta\mathcal{R}_k$, then set $\alpha_k = \alpha$, $x_{k+1} = x_k + \alpha_k d_k$, go to Step 6.
5.3 Compute the SOC step.
Solve the subproblem $\widetilde{\mathrm{QP}}(x_k)$ to obtain the SOC step $\tilde{d}_k$ and define $\tilde{x}_{k+1} = x_k + d_k + \tilde{d}_k$.

$$\widetilde{\mathrm{QP}}(x_k) \quad \begin{cases} \min & g(x_k)^{\mathrm{T}}(d_k + d) + \dfrac{1}{2}(d_k + d)^{\mathrm{T}} B_k (d_k + d) \\ \text{s.t.} & A_i(x_k)^{\mathrm{T}} d + c_i(x_k + d_k) = 0, \quad i \in \mathcal{E} \\ & A_i(x_k)^{\mathrm{T}} d + c_i(x_k + d_k) \leq 0, \quad i \in \mathcal{I} \end{cases} \tag{11}$$

5.3.1 If $\mathcal{R}_k - h(c(x_k + d_k + \tilde{d}_k)) \geq \alpha\eta\mathcal{R}_k$, go to 5.3.2, else set $\alpha = t\alpha$, go to Step 5.
5.3.2 If $\Delta\tilde{f}_k \geq \sigma\Delta l_k$, where $\Delta\tilde{f}_k = f(x_k) - f(\tilde{x}_{k+1})$, then set $x_{k+1} = \tilde{x}_{k+1}$, go to Step 6, else set $\alpha = t\alpha$, go to Step 1.
Step 6 Compute B_{k+1}, set $k := k + 1$, go to Step 1.

Remark 1. In step 6, update B_{k+1} by the modified BFGS method [16], or the modified Broyden methods [18,19]. We call Step 5 the inner loop, and call Step 1–Step 6 the outer loop.

3. Global convergence

In this section, we prove the global convergence of Algorithm B. Firstly, we give some assumptions:
A1 $\{x_k\}$, $\{x_k + d_k + \tilde{d}_k\}$ and $\{x_k + \alpha d_k\}$ are contained in a compact and convex set S of $\mathbb{R}^n$ for all $\alpha \in (0, 1]$.
A2 The functions $f(x)$, $c(x)$ are twice continuously differentiable on S.
A3 The matrix B_k is bounded and uniformly positive definite for all k. And the Lagrange multiplier λ_k is also bounded for all k.

Remark 2. A consequence of assumption A3 is that there exist two scalars $\delta > 0$ and $M > 0$, independent of k, such that $\delta\|y\|^2 \leq y^{\mathrm{T}} B_k y \leq M\|y\|^2$ for all $y \in \mathbb{R}^n$. By assumptions A1–A3, without loss of generality, we may also assume that $\|\lambda_k\|_\infty \leq M$, $\|\nabla^2 c_i(x)\| \leq M$, $i \in \mathcal{E} \cup \mathcal{I}$, $\|\nabla^2 f(x)\| \leq M$, $x \in S$.

Lemma 1. *Suppose assumptions* A1–A3 *hold, then for any* $x_k \in S$,

$$h(c(x_k + \alpha d_k)) \leq (1 - \alpha)h(c(x_k)) + \frac{1}{2}\alpha^2 mM\|d_k\|^2, \tag{12}$$

and

$$|\Delta f_k - \alpha\Delta l_k| \leq \frac{1}{2}\alpha^2 M\|d_k\|^2. \tag{13}$$

Proof. From the Taylor Expansion Theorem, (4) and Assumption A3, we have that for all $i \in \mathcal{E} \cup \mathcal{I}$,

$$\begin{aligned} c_i(x_k + \alpha d_k) &= (1 - \alpha)c_i(x_k) + \alpha c_i(x_k) + \alpha\nabla c_i(x_k)^{\mathrm{T}} d_k + \frac{1}{2}\alpha^2 d_k^{\mathrm{T}}\nabla^2 c_i(y_i) d_k \\ &\leq (1 - \alpha)c_i(x_k) + \frac{1}{2}\alpha^2 M\|d_k\|^2, \end{aligned}$$

where y_i denotes some point on the line segment from x_k to $x_k + \alpha d_k$. This, together with the definition of $h(c(x))$ implies (12). Similarly, we can also prove that (13) is true. □

Theorem 1. *Suppose assumptions* A1–A3 *hold, then the inner loop terminates in a finite number of iterations.*

Proof. If x_k is a KKT point of problem (P), then $d_k = 0$ solves $\mathrm{QP}(x_k)$ and Algorithm B terminates without going to the inner loop. Otherwise, we suppose $d_k \neq 0$. From Algorithm A, we have that $h(c_k) \leq T_k$. It follows with (12) and (9) that

$$\begin{aligned} h(c(x_k + \alpha d_k)) &\leq (1 - \alpha)h(c_k) + \frac{1}{2}\alpha^2 mM\|d_k\|^2 \\ &\leq (1 - \alpha)\mathcal{R}_k + \frac{1}{2}\alpha^2 mM\|d_k\|^2. \end{aligned} \tag{14}$$

Thus, (10) holds when $\alpha \le \frac{2(1-\eta)\mathcal{R}_k}{mM\|d_k\|^2}$. If (5) does not hold, then the inner loop terminates by Algorithm B. Otherwise, if (5) holds, two situations need to be considered.

Case 1. If $\triangle f_k < \sigma\alpha\triangle l_k$ and $\alpha = 1$, then SOC step $\tilde{x}_{k+1}$ is taken. If $\tilde{x}_{k+1}$ is accepted as the next iterate x_{k+1}, then the inner loop terminates.

Case 2. If $\alpha \ne 1$, we obtain from Remark 2 and (5) that

$$\Delta l_k \ge \xi\delta\|d_k\|^2.$$

This together with (13) implies

$$\left|\frac{\Delta f_k - \alpha\Delta l_k}{\alpha\Delta l_k}\right| \le \frac{\frac{\alpha^2 M\|d_k\|^2}{2}}{\alpha\xi\delta\|d_k\|^2} = \frac{\alpha M}{2\xi\delta}. \tag{15}$$

Therefore, $\Delta f_k \ge \sigma\alpha\Delta l_k$ when $\alpha \le \frac{2\delta\xi(1-\sigma)}{M}$. Define

$$C' = \frac{2\delta\xi(1-\sigma)}{M}, \qquad C_{k,2} = \frac{2(1-\eta)\mathcal{R}_k}{mM\|d_k\|^2}, \qquad C_k = \min\left\{C', C_{k,2}\right\}. \tag{16}$$

By (15) and (16), the inner loop terminates whenever $\alpha \le C_k$.

Therefore, the inner iteration terminates finitely in all cases. □

Lemma 2. *In Algorithm A, if the index k satisfies $j_{k+1} = j_k + 1$, then for all $k' \ge k$, $h(c_{k'}) \le b_{j_k}$. Furthermore, for any $k, j_k \ge 1$,*

$$h(c_k) \le 2b_{j_k}. \tag{17}$$

Proof. If $j_{k+1} = j_k + 1$, it follows from Algorithm A and (8) that $h(c_k) < \min\{\eta_1 b_{j_k}, \eta_2 N f_k\}$, and $b_{j_k} \ge T_k \ge \mathcal{M}_{l,k}$.

Thus for $k' = k - l + 1, \dots, k$, $h(c_{k'}) \le b_{j_k}$.

We use the inductive method to prove that for all $k' \ge k - l + 1$, $h(c_{k'}) \le b_{j_k}$.

For $k' = k - l + 1, \dots, k$, we have already proved that the claim is true. Now we assume that $h(c_{k'}) \le b_{j_k}$ holds for $k' = s - l + 1, \dots, s(\ge k)$. Next, we only have to prove that for $k' = s + 1$, it still holds.

From Algorithm A, the definition of b_{j_k}, for $s \ge k$,

$$T_s \le \max\{h(c_s), b_{j_s}\} \le \max\{h(c_s), b_{j_k}\} = b_{j_k}, \tag{18}$$

where the last equality holds because of the induction hypothesis.

By Theorem 1, there exists a step size $\alpha > 0$, such that

$$\mathcal{R}_s - h(c_{s+1}) \ge \alpha\eta\mathcal{R}_s > 0.$$

This together with the induction hypothesis and the definition of $\mathcal{M}_{l,k}$ implies

$$h(c_{s+1}) \le \mathcal{R}_s = \max\{T_s, \mathcal{M}_{l,s}\} \le b_{j_k}.$$

Therefore, for any $k' \ge k$, where $j_{k+1} = j_k + 1$, $h(c_{k'}) \le b_{j_k}$.

As to (17), if for some $k, j_{k+1} = j_k + 1$, then it follows from Algorithm A that (17) holds.

Now, consider any k' with $j_{k'} = j_{k+1}$, $k' \ge k + 1$. According to the definition of b_{j_k} and the previous result, we have that

$$h(c_{k'}) \le b_{j_k} = b_{j_{k'}-1} \le 2b_{j_{k'}}, \quad j_{k'} \ge 1.$$

Therefore, for any k ($j_{k'} \ge 1$), (17) holds. □

Lemma 3. *If $\mathcal{R}_k \ne \max\{h(c_k), \mathcal{M}_{l,k}\}$ holds for infinite iterations, then $j_k \to +\infty$, and $h(c_k) \to 0$.*

Proof. From the assumption of the lemma and Algorithm A, there exists an infinite sequence $\{k'\}$, such that

$$T_{k'} \ne h(c_{k'}) \quad \text{and} \quad T_{k'} > \mathcal{M}_{l,k'}.$$

Then

$$T_{k'} = \min\{b_{j_{k'}}, N f_{k'}\} > h(c_{k'}) \quad \text{and} \quad T_{k'} \ge \mathcal{M}_{l,k'}.$$

By Algorithm A we have that for any k', $j_{k'+1} = j_{k'} + 1$. Therefore, $j_k \to +\infty$, and $b_{j_k} = \frac{b_0}{j_k+1} \to 0$. This combining with (17) implies $h(c_k) \to 0$. □

Lemma 4. *If the outer loop of Algorithm B cannot terminate finitely, then $\lim_{k\to+\infty} h(c_k) = 0$.*

Proof. For the purpose of contradiction, we assume that $h(c_k) \nrightarrow 0$. By Lemma 3, there exists a sufficiently large integer $j_1 > 0$, such that for $k \geq j_1$,

$$\mathcal{R}_k = \max\{h(c_k), \mathcal{M}_{l,k}\}. \tag{19}$$

It implies with (8) that

$$\mathcal{R}_k = \max\{h(c_k), \mathcal{M}_{l,k}\} = \max_{k-l+1 \leq i \leq k} h(c_i). \tag{20}$$

According to Algorithm B, (10) and (20), we have that

$$\begin{cases} h(c_{k+1}) < (1 - \alpha_k \eta) \max\limits_{k-l+1 \leq i \leq k} h(c_i) \\ h(c_{k+2}) < (1 - \alpha_{k+1} \eta) \max\limits_{k-l+2 \leq i \leq k+1} h(c_i) \\ \cdots \\ h(c_{k+l}) < (1 - \alpha_{k+l-1} \eta) \max\limits_{k \leq i \leq k+l-1} h(c_i) \end{cases} \tag{21}$$

and for $k \geq j_1$,

$$\max_{k-l+1 \leq i \leq k} h(c_i) \geq \max_{k-l+2 \leq i \leq k+1} h(c_i) \geq \cdots \geq \max_{k \leq i \leq k+l-1} h(c_i). \tag{22}$$

Since $h(c_k) \nrightarrow 0$, it follows that there exists a positive constant $\epsilon_1 > 0$ and an infinite subsequence $\{h(c_{k_i})\}$, such that $h(c_{k_i}) \geq \epsilon_1$ for all $k_i > j_1$. Then, for any $k > j_1$, there exists a positive integer i_0, such that $k_{i_0} > k$. So it follows with (22) that

$$\max_{k \leq i \leq k+l-1} h(c_i) \geq \max_{k_{i_0} \leq i \leq k_{i_0}+l-1} h(c_i) \geq \epsilon_1. \tag{23}$$

According to (16) and (20), we have that there exists a step size $\alpha_{\min} > 0$, such that $\alpha_k > \alpha_{\min}$ for all $k > j_1$. Thus, it follows with (21) and (22) that

$$\max_{k \leq i \leq k+l-1} h(c_i) \leq (1 - \alpha_{\min} \eta)^{\lfloor \frac{k-k_1}{l} \rfloor} \max_{k_1-l+1 \leq i \leq k_1} h(c_i) \tag{24}$$

where $\lfloor a \rfloor$ denotes the maximal integer less than a.

So we have that

$$\lim_{k \to +\infty} \max_{k \leq i \leq k+l-1} h(c_i) = 0.$$

It implies that

$$\lim_{i \to +\infty} h(c_{k_i}) = 0,$$

which contradicts the assumption that $h(c_{k_i}) \geq \epsilon_1$ for all $k_i > j_1$. The conclusion follows. □

Lemma 5. *Suppose assumptions* A1–A3 *hold and* (5) *is satisfied. Then*

$$\alpha_k \begin{cases} = 1 & \text{if } C_k \geq 1 \text{ or } x_{k+1} = x_k + d_k + \tilde{d}_k; \\ \geq tC_k & \text{if } C_k < 1, \end{cases} \tag{25}$$

where t and C_k are from Algorithm B *and Theorem* 1.

Proof. If $x_{k+1} = x_k + d_k + \tilde{d}_k$, we easily obtain $\alpha_k = 1$. From the proof of Theorem 1, the inner loop terminates whenever $\alpha \leq C_k$. Since (5) holds, it follows with Step 5 of Algorithm B that $\alpha_k = 1$ if $C_k \geq 1$ and that $\alpha_k \geq tC_k$ if $C_k < 1$. □

Lemma 6. *Suppose assumptions* A1–A3 *hold, then*

$$\|d_k\| = \mathcal{O}(Nf_k).$$

Proof. It follows from (4) that

$$d_k = -B_k^{-1}(g_k + A_k \lambda_k). \tag{26}$$

Since $\{B_k\}$ is uniformly positive definite and uniformly bounded, we obtain that $\{B_k^{-1}\}$ is also positive definite and bounded for all k. Therefore $\|d_k\| = \mathcal{O}(\|g_k + A_k \lambda_k\|) = \mathcal{O}(Nf_k)$. □

Theorem 2. *Suppose assumptions* A1–A3 *hold. Then one of the following two situations occurs:*

(i) *Algorithm* B *terminates at a KKT point of problem* (P).

(ii) *There exists at least one accumulation point, which is a KKT point.*

Proof. We only need to consider the situation (ii). Since the inner loop is finite, we only need to consider that the outer loop is infinite.

Let $K_0 = \{k \mid g_k^T d_k > -\xi d_k^T B_k d_k \text{ or } h(c_k) > \zeta_1 \|d_k\|^{\zeta_2}\}$. There are two cases, depending on whether K_0 is finite or not.

(1) K_0 is an infinite set.

Since $\{x_k\}_{k\in K_0} \subset S$ is bounded, there exists an accumulation point denoted by $\bar{x}$, i.e.,

$$\lim_{k\in K_1, k\to+\infty} x_k = \bar{x},$$

where $K_1 \subset K_0$ is an infinite index set. It follows with Lemma 4 that $h(c_k) \to 0, k \to +\infty$. Thus $\bar{x}$ is a feasible point of problem (P).

If there exists a subset $\overline{K}_1 \subset K_1$, such that $\lim_{k\in\overline{K}_1,\ k\to+\infty} \|d_k\| = 0$, then $\bar{x}$ is a KKT point. Otherwise there exists a constant $\epsilon > 0$, such that $\|d_k\| > \epsilon$ for all $k \in K_1$. By Lemma 4, there exists a positive integer j_1, such that for all $k > j_1$, $k \in K_1$,

$$h(c_k) \le \min\left\{\frac{(1-\xi)\epsilon^2\delta}{M}, \zeta_1\|d_k\|^{\zeta_2}\right\}. \tag{27}$$

It implies that

$$h(c_k) \le \frac{(1-\xi)\delta\|d_k\|^2}{M} \le \frac{(1-\xi)d_k^T B_k d_k}{M}. \tag{28}$$

According to (4), (27) and (28), we have that for all $k \in K_1$, $k > j_1$,

$$\begin{aligned} g_k^T d_k &= -d^T A_k\lambda_k - d_k^T B_k d_k \\ &= \lambda_k^T c_k - d_k^T B_k d_k \\ &\le \|\lambda_k\|_\infty h(c_k) - d_k^T B_k d_k \\ &\le Mh(c_k) - d_k^T B_k d_k \\ &\le -\xi d_k^T B_k d_k \end{aligned}$$

and

$$h(c_k) \le \zeta_1\|d_k\|^{\zeta_2}.$$

That is a contradiction with the definition of K_0. So $\bar{x}$ is a KKT point.

(2) K_0 is a finite set.

It implies that there exists a positive integer $j_2 > 0$, such that (5) holds for any $k > j_2$. It follows with Theorem 1 that for any $k > j_2$, there exists an iterate $x_{k+1} = x_k + \alpha_k d_k$ or $x_{k+1} = x_k + d_k + \tilde{d}_k$, such that

$$f(x_k) - f(x_{k+1}) \ge \sigma\alpha_k\Delta l \ge \xi\sigma\alpha_k d_k^T B_k d_k \ge \xi\sigma\alpha_k\delta\|d_k\|^2. \tag{29}$$

Denote

$$\begin{aligned} K_2 &= \{k|\alpha_k = 1, k > j_2\}, \\ K_{3,1} &= \{k|C_k < 1 \text{ and } C' < C_{k,2}, k > j_2\}, \\ K_{3,2} &= \{k|C_k < 1 \text{ and } C' \ge C_{k,2}, k > j_2\}. \end{aligned}$$

According to (9) and (29) and Lemma 5, we have that

$$\sum_{k\in K_2}(f(x_k) - f(x_{k+1})) \ge \sum_{k\in K_2}\xi\sigma\delta\|d_k\|^2, \tag{30}$$

$$\sum_{k\in K_{3,1}}(f(x_k) - f(x_{k+1})) \ge \sum_{k\in K_{3,1}} tC'\xi\sigma\delta\|d_k\|^2, \tag{31}$$

$$\begin{aligned} \sum_{k\in K_{3,2}}(f(x_k) - f(x_{k+1})) &\ge \sum_{k\in K_{3,2}} tC_{k,2}\xi\sigma\delta\|d_k\|^2 \\ &= \sum_{k\in K_{3,2}} t\xi\sigma\delta\|d_k\|^2\frac{2(1-\eta)\mathcal{R}_k}{mM\|d_k\|^2}, \\ &\ge \sum_{k\in K_{3,2}}\frac{2(1-\eta)t\xi\sigma\delta T_k}{mM}, \\ &\ge \sum_{k\in K_{3,2}}\frac{2(1-\eta)t\xi\sigma\delta\min\{b_{j_k}, Nf_k\}}{mM}. \end{aligned} \tag{32}$$

By assumptions A1–A3, we have

$$+\infty > \sum_{k=j_2+1}^{+\infty}(f(x_k)-f(x_{k+1})) = \sum_{k\in K_2}(f(x_k)-f(x_{k+1})) + \sum_{k\in K_{3,1}}(f(x_k)-f(x_{k+1})) + \sum_{k\in K_{3,2}}(f(x_k)-f(x_{k+1})). \quad (33)$$

Since $\sum_{k=1}^{+\infty} b_{j_k} = +\infty$, it follows with (30), (31), (32) and Lemma 6 that

$$\sum_{k\to+\infty}\|d_k\|^2 < +\infty.$$

It implies that $\|d_k\| \to 0$, $k \to +\infty$.

The above two cases imply that the theorem holds. □

4. Local convergence

In this section, we prove the local convergence of Algorithm B. Let x^* be a local minimizer. We also need the following assumptions:

A4 The sequence $\{x_k\}$ converges to x^*. And the sequence $\{B_k\}$ converges to B^*. There exist constants $\bar{L} > 0$ and $1 \le \gamma < 2$, such that $\|d_k\|^\gamma \le \bar{L}b_{j_k}$.

A5 The point x^* is the KKT point of problem (P). The vectors $\{\nabla c_i(x^*), i \in J^*\}$ are linearly independent. The strict complementarity condition holds. Denote the matrix $A^c(x^*) = \{\nabla c_i(x^*), i \in J^*\}$, where $J^* = \mathcal{E}\bigcup\{i \in \mathcal{I} : c_i(x^*) = 0\}$.

A6 The Hessian matrix $\nabla_{xx}L(x^*, \lambda^*)$ is positive definite on the null space of $A^c(x^*)^{\mathrm{T}}$, i.e., there is a constant $\varrho > 0$, such that

$$d^{\mathrm{T}}\nabla_{xx}L(x^*, \lambda^*)d \ge \varrho\|d\|^2$$

for any $d \ne 0$ with $A^c(x^*)^{\mathrm{T}}d = 0$, where λ^* is the multiplier associated with x^*.

A7 Let $P_k = I - A^c(x_k)[A^c(x_k)^{\mathrm{T}}A^c(x_k)]^{-1}A^c(x_k)^{\mathrm{T}}$ and

$$\lim_{k\to+\infty}\frac{\|P_k(B_k - \nabla_{xx}L(x^*, \lambda^*))d_k\|}{\|d_k\|} = 0,$$

where I is an identity matrix with appropriate size.

Remark 3. The assumption $\|d_k\|^\gamma \le \bar{L}b_{j_k}$ is important to the proof of the local convergence of Algorithm B. From the global convergence analysis of Algorithm B, we know that the sequence $\{x_k\}$ converges to an optimal point. So d_k and Nf_k with the same order eventually converge to zero, which is proved by Lemmas 6 and 7. From Algorithm B, we know that the role of the sequence $\{b_{j_k}\}$ is to relax the infeasibility. Even if b_{j_k} converges to zero, it will not decrease to zero too fast.

Theoretical speaking, since d_k and Nf_k converge to zero with the same order simultaneously as x_k converges to an optimal point, and either Nf_k or $\mathcal{M}_{l,k}$ relaxes the infeasibility after sufficiently large k, the sequence $\{b_{j_k}\}$ may not decrease after some iterations from the mechanism of Algorithm A.

Furthermore, practical speaking, when x_k sufficiently approaches to an optimal point, the sequence $\{b_{j_k}\}$ never decreases, which is confirmed by the good numerical results in Section 5. Thus, assumption A4 is reasonable.

Lemma 7. *Suppose assumptions* A1–A7 *hold, then* $d_k \to 0$, $k \to +\infty$.

Proof. It follows from (4) and assumptions A1–A7 that $g_k \to g(x^*)$, $A_k \to A(x^*)$, $\lambda_k \to \lambda^*$, $B_k \to B^*$, $k \to +\infty$. Without loss of generality, let $d_k \to d^*$, $k \to +\infty$. assumptions A3–A4 imply that B^* is positive definite. Since x^* is the KKT point of problem (P), it follows that $g(x^*) + A(x^*)\lambda^* = 0$, $B^*d^* = 0$. Thus, $d^* = 0$, i.e., $d_k \to 0$, $k \to +\infty$. □

Lemma 8. *Suppose assumptions* A1–A7 *hold. Then there exists a neighborhood* U_1 *of* x^**, such that for* $x_k \in U_1$,

$$\tilde{d_k} = \mathcal{O}(\|d_k\|^2), \quad (34)$$

$$c_i(x_k + d_k + \tilde{d_k}) = o(\|d_k\|^2), \quad i \in J^*. \quad (35)$$

Proof. It follows from [20, Proposition 3.6] that $\tilde{d_k} = \mathcal{O}(\|d_k\|^2)$. According to assumptions A1–A7, there exists a neighborhood U_1 of x^*, such that for all $x_k \in U_1$, QP(x_k) is equivalent to the following subproblem

$$\mathrm{EQP}(x_k)\quad \begin{cases} \min & g(x_k)^{\mathrm{T}}d + \frac{1}{2}d^{\mathrm{T}}B_kd \\ \text{s.t.} & \nabla c_i(x_k)^{\mathrm{T}}d + c_i(x_k) = 0, \quad i \in J^*. \end{cases}$$

From the Taylor Expansion Theorem and the feasibility of $\widetilde{\mathrm{QP}}(x_k)$, we have

$$\begin{aligned} c_i(x_k + d_k + \tilde{d}_k) &= c_i(x_k + d_k) + \nabla c_i^{\mathrm{T}}(x_k + d_k)\tilde{d}_k + O(\|\tilde{d}_k\|^2) \\ &= o(\|d_k\|^2), \end{aligned}$$

for all $i \in J^*$. □

In order to prove the local convergence of Algorithm B, we need two conclusions in [3], which are applications of the SOC steps on the exact penalty function. We introduce a penalty function

$$\Phi_\rho(x) = f(x) + \rho h(c(x)) \tag{36}$$

and its quadratic approximation

$$q_\rho(x_k, d) = f(x_k) + g_k^{\mathrm{T}} d + \frac{1}{2} d^{\mathrm{T}} B_k d + \rho \left[\sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^{\mathrm{T}} d| + \sum_{i \in \mathcal{I}} \max\{c_i(x_k) + \nabla c_i(x_k)^{\mathrm{T}} d, 0\} \right] \tag{37}$$

just for proof. The following two lemmas from [3] are important to our proof.

Lemma 9 *([3, Theorem 15.3.7]). Suppose assumptions* A1–A7 *hold. The exact penalty function Φ_ρ and q_ρ are described by (36) and (37), where $\rho > \|\lambda^*\|_\infty$, then*

$$\lim_{k\to\infty} \frac{\Phi_\rho(x_k) - \Phi_\rho(x_k + d_k + \tilde{d}_k)}{q_\rho(x_k, 0) - q_\rho(x_k, d_k)} = 1. \quad \square \tag{38}$$

Lemma 10 *([3, Theorem 15.3.2]). Suppose assumptions* A1–A7 *hold. Let d_k be the solution of $QP(x_k)$, λ_k is the associated multiplier, and $\rho > \|\lambda^*\|_\infty$, then*

$$q_\rho(x_k, 0) - q_\rho(x_k, d_k) \ge 0. \quad \square \tag{39}$$

Lemma 11. *Suppose assumptions* A1–A7 *hold, then there exists a neighborhood U_2 ($\subset U_1$) of x^*, such that for $x_k \in U_2$, $\Delta f_k \ge \sigma \Delta l_k$ or $\Delta \tilde{f}_k \ge \sigma \Delta l_k$ whenever (5) holds.*

Proof. If $\Delta f_k \ge \sigma \Delta l_k$, then the theorem holds. Otherwise the SOC step is computed. We only need to prove that $\Delta \tilde{f}_k \ge \sigma \alpha \Delta l_k$ holds. Since (5) holds, it follows

$$h(c_k) = o(\|d_k\|^2). \tag{40}$$

By Lemmas 8 and 9, for all sufficiently large k,

$$\Phi_\rho(x_k) - \Phi_\rho(x_k + d_k + \tilde{d}_k) \ge \left(\frac{1}{2} + \sigma\right)(q_\rho(x_k, 0) - q_\rho(x_k, d_k)), \tag{41}$$

where $\rho > \|\lambda_k\|_\infty$, independent of k. Then

$$\begin{aligned} f(x_k) - f(x_k + d_k + \tilde{d}_k) &= \Phi_\rho(x_k) - \Phi_\rho(x_k + d_k + \tilde{d}_k) - \rho(h(c(x_k)) - h(c(x_k + d_k + \tilde{d}_k))) \quad \text{(by (36))} \\ &\ge \left(\frac{1}{2} + \sigma\right)(q_\rho(x_k, 0) - q_\rho(x_k, d_k)) + o(\|d_k\|^2) \quad \text{(by (35), (40) and (41))} \\ &= -\left(\frac{1}{2} + \sigma\right)(g_k^{\mathrm{T}} d_k + \frac{1}{2} d_k^{\mathrm{T}} B_k d_k) + o(\|d_k\|^2) \quad \text{(by (35) and (37))} \end{aligned} \tag{42}$$

and

$$\begin{aligned} f(x_k) + \sigma g_k^{\mathrm{T}} d_k - f(x_k + d_k + \tilde{d}_k) &\ge -\frac{1}{2} g_k^{\mathrm{T}} d_k - \left(\frac{1}{4} + \frac{\sigma}{2}\right) d_k^{\mathrm{T}} B_k d_k + o(\|d_k\|^2) \quad \text{(by (42))} \\ &= \frac{1}{2}(d_k^{\mathrm{T}} B_k d_k - c(x_k)^{\mathrm{T}} \lambda_k) - \left(\frac{1}{4} + \frac{\sigma}{2}\right) d_k^{\mathrm{T}} B_k d_k + o(\|d_k\|^2) \quad \text{(by (4))} \\ &\ge \left(\frac{1}{4} - \frac{\sigma}{2}\right) d_k^{\mathrm{T}} B_k d_k - \|\lambda_k\|_\infty h(c(x_k)) + o(\|d_k\|^2) \quad \text{(by (4))} \\ &= \left(\frac{1}{4} - \frac{\sigma}{2}\right) d_k^{\mathrm{T}} B_k d_k + o(\|d_k\|^2), \text{(by the boundedness of } \|\lambda_k\|_\infty \text{)}. \end{aligned} \tag{43}$$

This, together with Lemma 7 implies $\Delta \tilde{f}_k \ge \sigma \Delta l_k$. □

Table 1
Numerical results.

Problem	n	m	Algorithm B					LANCELOT		
			NIT	NF	NG	NC	NA	NIT	NF/NC	NG/NA
HS002	2	1	13	24	14	21	14	4	4	5
HS004	2	2	1	2	2	2	2	1	1	2
HS008	2	2	5	6	6	6	6	8	8	8
HS010	2	1	11	12	12	12	12	27	27	22
HS022	2	2	1	2	2	2	2	12	12	13
HS026	3	1	31	46	32	46	32	32	32	32
HS029	3	1	13	18	14	18	14	43	43	33
HS035	3	4	5	8	6	7	6	9	9	10
HS038	4	8	63	101	81	101	81	45	45	42
HS047	5	3	21	23	22	23	22	23	23	23
HS049	5	2	17	24	18	21	18	23	23	24
HS065	3	7	8	10	9	10	9	43	43	39
HS067	3	20	22	23	23	23	23	69	69	64
HS072	4	10	25	26	26	26	26	110	110	111
HS088	2	1	21	24	22	24	22	93	93	78
HS090	4	1	25	27	26	27	26	97	97	79
HS092	6	1	32	34	33	34	33	102	102	86
HS100	7	4	18	44	19	44	19	31	31	30
HS101	7	20	53	73	54	77	5	–	–	–
HS113	10	8	14	22	15	22	15	48	48	43

Theorem 3. *Suppose assumptions* A1–A7 *hold, then for all sufficiently large k, either $x_k + d_k$ or $x_k + d_k + \tilde{d}_k$ is accepted and the sequence $\{x_k\}$ converges to x^* superlinearly.*

Proof. If x_k is not a KKT point of problem (P), then $d_k \neq 0$. By the mechanism of Algorithm A, two cases may occur.

(1) If $h(c(x_k)) < \min\{\eta_1, b_{j_k}\eta_2 Nf_k\}$,

$$T_k = \min\{b_{j_k}, Nf_k\} \geq \min\{\eta_1 b_{j_k}, \eta_2 Nf_k\}.$$

(2) If $h(c(x_k)) \geq \min\{\eta_1 b_{j_k}, \eta_2 Nf_k\}$,

$$T_k = h(c_k) \geq \min\{\eta_1 b_{j_k}, \eta_2 Nf_k\}.$$

Therefore,

$$T_k \geq \min\{\eta_1 b_{j_k}, \eta_2 Nf_k\}, \tag{44}$$

holds in both cases. From A4 and Lemma 6, we have

$$\|d_k\|^\gamma \leq \bar{L} b_{j_k}, \qquad Nf_k = O(\|d_k\|), \quad 1 \leq \gamma < 2. \tag{45}$$

Combining this with (35), we have $h(c(x_k + d_k + \tilde{d}_k)) \leq (1-\eta)T_k \leq (1-\eta)\mathcal{R}_k$ for all $k > k_1$, where $k_1 > 0$ is some sufficiently large integer. Now for $k > k_1$, if (5) does not hold, then either $x_k + d_k$ or $x_k + d_k + \tilde{d}_k$ is accepted as a new iterate by Algorithm B. If (5) holds, by Lemma 11, it follows $\Delta f_k \geq \sigma\alpha\Delta l_k$ or $\Delta \tilde{f}_k \geq \sigma\alpha\Delta l_k$. So $x_k + d_k + \tilde{d}_k$ is accepted as a new iterate. Therefore, by [9, Theorem 5.2], the sequence $\{x_k\}$ converges to x^* superlinearly. □

5. Numerical results

In this section, we give some numerical results for a set of problems from Hock and Schittkowski's problems [11]. The results are obtained by a preliminary Matlab implementation of Algorithm B. At each iteration, we use matlab command `quadprog` to solve the QP subproblem. Details about the implementation are described as follows.

(a) Termination criteria. Algorithm B stops if $h(c_k) \leq \varepsilon\sqrt{m}$ and $Nf_k \leq \varepsilon\sqrt{n}$.
(b) Update B_k. For each test problems, we chose $B_0 = I$, where I denotes the identity matrix with an appropriate size, as the initial guess of the Lagrangian Hessian. At each step, the matrix B_k was updated by the BFGS formula from Powell's modifications [16]. Specifically, we set

$$B_{k+1} = B_k - \frac{B_k s_k s_k^{\mathrm{T}} B_k}{s_k^{\mathrm{T}} B_k s_k} + \frac{y_k y_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k},$$

where

$$y_k = \begin{cases} \tilde{y}_k, & \tilde{y}_k^{\mathrm{T}} s_k \geq 0.2 s_k^{\mathrm{T}} B_k s_k, \\ \theta^k \tilde{y}_k + (1-\theta^k) B_k s_k, & \text{otherwise}, \end{cases}$$

Table 2
Numerical results.

Problem	n	m	Algorithm B					SNOPT		
			NIT	NF	NG	NC	NA	NIT	NF/NC	NG/NA
HS001	2	1	38	56	39	48	39	71	49	48
HS002	2	1	13	24	14	21	14	18	15	14
HS003	2	1	9	10	10	10	10	2	2	2
HS004	2	2	1	2	2	2	2	2	3	2
HS005	2	4	7	12	8	10	8	8	9	8
HS006	2	1	9	13	10	13	10	5	8	7
HS007	2	1	11	12	12	12	12	18	31	30
HS008	2	2	5	6	6	6	6	2	7	6
HS009	2	1	6	7	7	7	7	9	9	8
HS010	2	1	11	12	12	12	12	20	32	31
HS011	2	1	7	8	8	8	8	11	18	17
HS012	2	1	9	10	10	10	10	12	12	11
HS013	2	3	7	8	8	8	8	1	7	6
HS014	2	2	5	6	6	6	6	2	10	9
HS015	2	3	2	3	3	3	3	5	11	10
HS016	2	5	4	5	5	5	5	1	5	4
HS017	2	5	6	9	7	8	7	13	19	18
HS018	2	6	8	9	9	9	9	19	31	30
HS019	2	6	5	6	6	6	6	2	9	8
HS020	2	5	3	4	4	4	4	1	5	4
HS021	2	5	1	4	2	3	2	1	1	1
HS022	2	2	1	2	2	2	2	2	6	5
HS023	2	9	5	6	6	6	6	1	7	6
HS024	2	5	4	5	5	5	5	4	6	5
HS025	3	6	0	1	1	1	1	0	2	1
HS026	3	1	31	46	32	46	32	25	25	24
HS027	3	1	18	26	19	28	19	22	24	23
HS028	3	1	3	6	4	5	4	4	4	4
HS029	3	1	13	18	14	18	14	17	19	18
HS030	3	7	11	12	12	12	12	7	5	4
HS031	3	7	9	17	10	17	10	9	11	10
HS032	3	5	2	3	3	3	3	5	5	4
HS033	3	6	3	4	4	4	4	3	9	8
HS034	3	8	7	8	8	8	8	5	8	7
HS035	3	4	5	8	6	7	6	5	5	5
HS036	3	7	1	2	2	2	2	10	9	8
HS037	3	8	9	17	10	15	10	12	10	11
HS038	4	8	63	101	81	101	81	160	119	118
HS039	4	2	12	13	13	13	13	20	31	30
HS040	4	3	6	7	7	7	7	7	8	7
HS041	4	9	7	8	8	8	8	12	9	8
HS042	4	2	8	11	9	11	9	8	9	8
HS043	4	3	11	15	12	15	12	14	10	9
HS044	4	10	5	6	6	6	6	12	12	12
HS045	5	10	7	8	8	8	8	Fail	Fail	Fail
HS046	5	2	34	40	35	40	35	28	27	26
HS047	5	3	21	23	22	23	22	24	32	31
HS048	5	2	3	10	4	7	4	6	6	6
HS049	5	2	17	24	18	21	18	Fail	Fail	Fail
HS050	5	3	11	14	12	13	12	33	22	21
HS051	5	3	2	7	3	5	3	6	6	6
HS052	5	3	6	10	7	10	7	5	5	5
HS053	5	13	8	10	9	10	9	2	2	2
HS054	6	13	1	2	2	2	2	5	5	5
HS055	6	14	1	2	2	2	2	3	3	3
HS056	7	4	16	22	17	22	17	13	15	14
HS057	2	3	15	23	16	22	16	5	6	5
HS059	2	7	16	22	17	20	17	20	20	19
HS060	3	7	7	11	8	11	8	11	13	12
HS061	3	2	9	16	10	16	10	17	24	23
HS062	3	7	9	18	10	14	10	13	16	15
HS063	3	5	8	9	9	10	9	13	14	13
HS064	3	4	42	64	43	65	43	33	26	25
HS065	3	7	8	10	9	10	9	15	11	10
HS066	3	8	7	8	8	8	8	7	6	5
HS067	3	20	22	23	23	23	23	38	28	27
HS068	4	10	37	50	38	50	38	Fail	Fail	Fail
HS069	4	10	12	18	13	18	13	Fail	Fail	Fail

(*continued on next page*)

Table 2 (*continued*)

Problem	n	m	Algorithm B					SNOPT		
			NIT	NF	NG	NC	NA	NIT	NF/NC	NG/NA
HS070	4	9	34	37	35	36	35	14	12	11
HS071	4	10	5	6	6	6	6	9	8	7
HS072	4	10	25	26	26	26	26	36	41	40
HS073	4	7	4	5	5	5	5	11	8	7
HS074	4	13	11	12	12	12	12	12	15	14
HS075	4	13	8	9	9	9	9	5	12	11
HS076	4	7	6	7	7	7	7	4	4	4
HS077	5	2	14	19	15	19	15	15	15	14
HS078	5	3	8	9	9	9	9	9	8	7
HS079	5	3	9	12	10	12	10	14	15	14
HS080	5	13	6	7	7	7	7	14	22	21
HS081	5	13	7	8	8	8	8	10	10	9
HS083	5	16	4	5	5	5	5	9	8	7
HS084	5	16	8	9	9	9	9	11	10	9
HS085	5	48	35	50	36	50	36	17	17	16
HS086	5	15	5	8	6	7	6	23	23	23
HS088	2	1	21	24	22	24	22	32	55	54
HS089	3	1	27	33	28	33	28	41	47	46
HS090	4	1	25	27	26	27	26	Fail	Fail	Fail
HS091	5	1	36	42	37	45	37	37	62	61
HS092	6	1	32	34	33	34	33	37	54	53
HS093	6	8	14	20	15	19	15	41	34	33
HS095	6	16	1	2	2	2	2	1	3	2
HS096	6	16	1	2	2	2	2	1	2	3
HS097	6	16	6	7	7	7	7	13	37	36
HS098	6	16	6	7	7	7	7	13	37	36
HS099	7	16	15	35	16	38	16	32	23	22
HS100	7	4	18	44	19	44	19	21	17	16
HS101	7	20	53	73	54	77	5	165	463	462
HS102	7	20	46	59	47	63	47	117	323	322
HS103	7	20	31	40	32	43	32	84	179	178
HS104	8	22	17	18	18	18	18	38	25	24
HS105	8	17	49	50	50	50	50	127	107	106
HS106	8	22	35	37	36	37	36	32	14	13
HS107	9	14	7	10	8	10	8	22	14	13
HS108	9	14	12	13	13	13	13	36	14	13
HS109	9	26	27	28	28	28	28	39	29	28
HS110	10	20	7	12	8	10	8	23	8	7
HS111	10	23	77	81	78	81	78	60	78	77
HS112	10	13	46	87	47	72	47	52	30	29
HS113	10	8	14	22	15	22	15	37	19	18
HS114	10	31	29	30	30	30	30	59	42	41
HS116	13	41	44	45	45	45	45	91	28	27
HS117	15	20	17	18	18	18	18	49	19	18
HS118	15	59	18	19	19	19	19	13	13	13
HS119	16	40	7	8	8	8	8	63	17	16

and

$$\begin{cases} s_k = x_{k+1} - x_k, \\ \tilde{y}_k = \nabla f(x_{k+1}) - \nabla f(x_k) + (A_{k+1} - A_k)\lambda_k, \\ \theta^k = 0.8 s_k^{\mathrm{T}} B_k s_k / (s_k^{\mathrm{T}} B_k s_k - s_k^{\mathrm{T}} \tilde{y}_k). \end{cases}$$

(c) Compute T_k. Let $b_0 = \min\{0.1\max(1, h(c_0)), Nf_0 + h(c_0)\}$, $b_j = \frac{b_0}{j+1}$, $j \geq 1$. If $h(c_k) \ll Nf_k$, then set $T_k = \min\{b_{j_k}, Nf_k\}$, else set $T_k = h(c_k)$.

(d) The parameters are chosen as follows:

$$\sigma = 0.1, \quad l = 5, \quad \eta = 0.1, \quad \zeta_1 = 1, \quad \zeta_2 = 2.2, \varepsilon = 10^{-6}, \quad t = 0.6, \quad \eta_1 = \eta_2 = 0.2.$$

In the tables :

Problem: the problem number given in [11],
n: the number of variables,
m: the number of constraints,
NIT = the number of iterations,
NF = the number of evaluations for $f(x)$,
NG = the number of evaluations for $\nabla f(x)$,

NC = the number of evaluations for $c(x)$,
NA = the number of evaluations for $\nabla c(x)$.
'–' denotes the iteration number is greater than 1000.
'Fail' denotes some errors occur when the solver solves the problem.

Since we use a quasi-Newton method to update B_k in Algorithm B, the second order information of the Lagrangian function is not used. Therefore, it is not proper for us to compare Algorithm B with the algorithm in [6]. Therefore, for comparison, we include the corresponding results obtained by the well-known optimization solvers LANCELOT [2] (column "LANCELOT" in Table 1). In LANCELOT solver, (1) The exact Cauchy step is computed; (2) Accurate solution of quadratic problems with bound constraints (BQP) is computed; (3) Bandsolver preconditioned Conjugate Gradient (CG) is used (semi-bandwidth = 5); (4) Infinity-norm trust region is used; (6) SR.1 approximation to second derivatives is used.

From Table 1, we can see that our algorithm is better than LANCELOT in these problems.

Furthermore, for more comparison, we also compare Algorithm B with SNOPT [10] (column "SNOPT" in Table 2). We run on a NEOS Server [25] with default options, where feasibility tolerance and optimality tolerance are 10^{-6}. A total of 114 problems are selected from [11]. HS087 is the only problem left out. Since it is a discontinuous optimization problem, it is not suitable for our algorithm.

The numerical results on these problems are summarized in Table 2. The initial point of HS025 is a stationary point, then our algorithm terminates at the first iteration which is an undesired result because of its non-minimal property. From Table 2, we find that Algorithm B succeeds in solving all the test problems, and for all these problems the number of iterations is small. From Table 2, Algorithm B is more efficient for 89 problems in terms of NIT, 74 problems in terms of NF and 86 problems in terms of NG.

Therefore, all the computational results illustrate that our algorithm is competitive with those in [2,10].

As expected from our local convergence theory, we also observed a transition to fast local convergence for all problems tested, i.e., unit steps are accepted in last several iterations for all problems tested here. The results indicate that our new non-monotone line search method without a penalty function is an interesting and competitive alternative to algorithms with penalty functions and deserves further consideration. Considering the lower memory requirement, it is also competitive with filter methods. So, numerical tests confirm the robustness and efficiency of our approach.

References

[1] C. Audet, J.E. Dennis, A pattern search filter method for nonlinear programming without derivatives, SIAM J. Optim. 14 (2004) 980–1010.
[2] A.R. Conn, N.I.M. Gould, Ph.L. Toint, LANCELOT: A fortran package for large-scale nonlinear optimization, in: Numerical Optimization (Release A), in: Springer Ser. Comput. Math., Springer-Verlag, Berlin, 1992.
[3] A.R. Conn, N.I.M. Gould, Ph.L. Toint, Trust Region Methods, SIAM, Philadelphia, PA, USA, 2000.
[4] H.Y. Benson, D.F. Shanno, R. Vanderbei, Interion-point methods for nonconvex nonlinear programming jamming and numercical test, Math. Program. 99 (2004) 35–48.
[5] R. Fletcher, S. Leyffer, A bundle filter method for nonsmooth nonlinear optimization, Technical Report NA/195, Department of Mathematics, University of Dundee, Scotland, December 1999.
[6] R. Fletcher, S. Leyffer, Nonlinear programming without a penalty function, Math. Program. 91 (2002) 239–269.
[7] R. Fletcher, N.I.M. Gould, S. Leyffer, Ph.L. Toint, A. Wächter, Global convergence of a trust region SQP-filter algorithms for general nonlinear programming, SIAM J. Optim. 13 (2002) 635–659.
[8] R. Fletcher, S. Leyffer, Ph.L. Toint, On the global convergence of a filter-SQP algorithm, SIAM J. Optim. 13 (2002) 44–59.
[9] F. Facchinei, S. Lucidi, Quadratically and superlinearly convergent algorithm for the solution of inequality constrained minimization problems, J. Optim. Theory Appl. 85 (1995) 265–289.
[10] Ph.E. Gill, W. Murray, M.A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, SIAM Rev. 47 (2005) 99–131.
[11] W. Hock, K. Schittkowski, Test Examples for Nonlinear Programming Codes, Springer-Verlag, 1981.
[12] S.P. Han, Superlinearly convergent variable metric algorithms for general nonlinear programming problems, Math. Program. 11 (1976) 263–282.
[13] S.P. Han, A globally convergent method for nonlinear programming, J. Optim. Theory Appl. 22 (1977) 297–309.
[14] N. Maratos, Exact penalty function algorithms for finite dimensional and control optimization problems, Ph.D. Thesis, University of London, London, UK, 1978.
[15] P.Y. Nie, A filter method for solving nonlinear complementarity problems, Appl. Math. Comput. 167 (2005) 677–694.
[16] M.J.D. Powell, A fast algorithm for nonlinearly constrained optimization calculations, in: Numerical Analysis, Proceedings, Biennial conference, Dundee 1977, in: G.A. Waston (Ed.), Lecture Notes in Math., vol. 630, Springer-Verlag, Berlin, New York, 1978, pp. 144–157.
[17] M.J.D. Powell, Variable metric methods for constrained optimization, in: A. Bachem, M. Grotschel, B. Korte (Eds.), Math. Programming: The State of Art, Bonn, 1982.
[18] D. Pu, W. Tian, A class of modified Broyden algorithms, J. Comput. Math. 8 (1994) 366–379.
[19] D. Pu, W. Tian, A class of modified Broyden algorithms without exact line search, Appl. Math. - A J. Chinese Univ. 10 (1995) 313–322.
[20] E.R. Panier, A.L. Tits, Avoiding the Maratos effect by means of a nonmonotone line search I: General constrained problems, SIAM J. Numer. Anal. 28 (1991) 1183–1195.
[21] M. Ulbrich, S. Ulbrich, Nonmonotone trust-region methods for nonlinear equality constrained optimization without a penalty function, Math. Program. 95 (2003) 103–135.
[22] M. Ulbrich, S. Ulbrich, L.N. Vicente, A globally convergent primal-dual interior filter method for nonconvex nonlinear programming, Math. Program. 100 (2004) 379–410.
[23] A. Wächter, L.T. Biegler, Line search filter methods for nonlinear programming: Motivation and global convergence, SIAM J. Optim. 16 (2005) 1–31.
[24] A. Wächter, L.T. Biegler, Line search filter methods for nonlinear programming: Local convergence, SIAM J. Optim. 16 (2005) 32–48.
[25] NEOS Server: http://neos.mcs.anl.gov/neos/solvers/nco:SNOPT/GAMS.html, 2006.