AIMS Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 7(12): 20441–20460.
DOI: 10.3934/math.20221120
Received: 26 July 2022
Revised: 28 August 2022
Accepted: 08 September 2022
Published: 19 September 2022

Research article

An area-type nonmonotone filter method for nonlinear constrained optimization

Ke Su[1,2], **Wei Lu**[1,2*] **and Shaohua Liu**[1,2]

[1] College of Mathematics and Information Science, Hebei University, Baoding, China
[2] Key Laboratory of Machine Learning and Computational Intelligence, Hebei University, Baoding, China

* **Correspondence:** Email: wei_lu_22@163.com.

Abstract: In this paper, we define a new area-type filter algorithm based on the trust-region method. A relaxed trust-region quadratic correction subproblem is proposed to compute the trial direction at the current point. Consider the objective function and the constraint violation function at the current point as a point pair. We divide the point pairs into different partitions by the dominant region of the filter and calculate the contributions of the point pairs to the area of the filter separately. Different from the conventional filter, we define the contribution as the filter acceptance criterion for the trial point. The nonmonotone area-average form is also adopted in the filter mechanism. In this paper, monotone and nonmonotone methods are proposed and compared with the numerical values. Furthermore, the algorithm is proved to be convergent under some reasonable assumptions. The numerical experiment shows the effectiveness of the algorithm.

Keywords: area-type filter; trust-region; nonlinear programming; monotone; nonmonotone
Mathematics Subject Classification: 65K05, 90C30

1. Introduction

Currently, numerical methods for solving nonlinear optimization problems have been widely used in the military, transportation, engineering design, economic analysis, artificial intelligence and other fields. In this paper, we consider the constrained optimization problems, where the objective function and the nonlinear constraints are smooth. The numerical solution to the following problem is considered.

$$\begin{aligned} &\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) \\ &\text{subject to} \quad c_i(x) \leqslant 0, \quad i \in \mathcal{I}, \end{aligned} \tag{1.1}$$

where $\mathcal{I} = \{1, 2, 3, \ldots, m\}$. The real valued objective function $f(x) : \mathbb{R}^n \to \mathbb{R}$ and the inequality constants $c_i(x) = (c_1, c_2, \ldots, c_m)^T : \mathbb{R}^n \to \mathbb{R}^m$ are twice continuously differentiable.

In traditional optimization problems, the objective function and constraints are usually analytical models with deterministic parameters. The method of this paper is to solve the deterministic problem, but there are widely uncertain problems in practical problems. In fact, there are many optimization problems with uncertain information that have their own specific solutions for uncertain optimization problems, but eventually it all comes down to a deterministic method. Although the algorithm in this paper is for solving deterministic problems, it is still valid for uncertain problems. That is, the problem of uncertainty can also be applied to our algorithm. As we all know, there are two main approaches to uncertainty optimization, robust optimization (RO) [1,2] and reliability-based design optimization (RBDO) [3,4].

The common solutions to uncertain problems are probabilistic constraints and robust constraints methods. The robust constraint can be transformed into a semi-infinite problem by subjecting the uncertain set to certain inscriptions. The semi-infinite problem can be transformed into a finite problem in the form of (1.1) under discretization methods or other appropriate transformations. Probabilistic constraints can be transformed into deterministic optimization by replacing the objective function and nonlinear constraints with a Kriging model. The transformed deterministic optimization problem can be solved by the method in this paper. Therefore, the algorithm in this paper has some potential applications to the study of uncertain problems.

It is generally known that the sequential quadratic programming (SQP) method is one of the most effective methods to solve this problem. The SQP algorithm converts a complex nonlinear constrained optimization problem into a relatively simple quadratic programming problem (QP) solution. The objective function of a quadratic programming problem is quadratic, and the constraint function is linear. At the kth iterate, the search direction d_k is obtained by the following QP subproblem.

$$\begin{aligned} \underset{d}{\text{minimize}} \quad & \langle \nabla f(x_k), d \rangle + \frac{1}{2} \langle d, B_k d \rangle \\ \text{subject to} \quad & c_i(x_k) + \nabla c_i(x_k)^T d \leqslant 0, \end{aligned} \tag{1.2}$$

where $B_k \in \mathbb{R}^{n \times n}$ is an appropriate symmetric matrix of the the Hessian $\nabla^2 f(x_k)$.

Unlike the line search method, the trust-region method computes a trial step by solving a subproblem in which the model function is minimized in the trust-region. In general, the trust-region method is easier to establish global convergence than the line search method. In recent years, the trust region method has been widely used in constrained optimization [5–8], unconstrained optimization [9–12], nonlinear equations [13,14], least-squares problems [15,16] and other problems [17–19].

In order to ensure sufficient descent of sequential quadratic programming, a penalty function is introduced as a merit function to decide whether to accept test points. The estimation of penalty parameters may be difficult to choose. The filter method was proposed by Fletcher and Leyffer [20], and its convergence was proved in [21]. It avoids the difficulty of updating the parameter in the penalty function. The filter method adopts the idea of multi-objective constraints, and it balances the relationship between objective function and constraint violation function. The poor trial points can be rejected in the filter method. The global convergence of the filter method can be guaranteed by taking any point as the starting point.

In the traditional filter method, a pair (h_k, f_k) is dominated by (f_j, f_j) if and only if $h_k \leqslant h_j$ and $f_k \leqslant f_j$ for each $j \neq k$. h and f are constraint violation and objective functions, respectively. A filter set $\mathcal{F}$ is a set of pairs (h, f) such that no pair dominates any other pair. The trial point x_k is accepted by the filter set if and only if

$$h(x) \leq \beta h_j \quad \text{or} \quad f(x) \leqslant f_j - \gamma h_j \,, \quad \forall (h_j, f_j) \in \mathcal{F}, \tag{1.3}$$

where $0 < \gamma < \beta < 1$, and $h(x)$ and $f(x)$ are the constraint violation function and objective function. $\mathcal{F}$ represents the filter set. As the criteria are satisfied, the point is accepted by the current filter set. The filter technique has become important in recent years [22–28].

Su and Pu [29] proposed a nonmonotone method based on the traditional filter and obtained good numerical results for equality constrained optimization. Wang et al. [30] proposed a nonmonotone adaptive filter method for unconstrained optimization and adopted an adaptive strategy to fix that step size and update the trust-region radius. Xue and Liu [31] propose a multidimensional filter algorithm. The constraint is divided into several parts, and the filter structure is composed of them. The individual behavior of each part of the constraints is considered, but how to divide the constraints is still a difficult problem to be explored. [32] proposed an area-based filtering algorithm, but feasibility recovery was used in its algorithm. Although its convergence property was proved, no numerical calculation was performed to know the performance of its numerical results.

This paper presents an area-type filter algorithm based on the trust-region approach. The discriminant criterion is different from that of the traditional filter method and has the following advantages. 1) This paper proposes a relaxed trust-region quadratic correction subproblem by which the direction at the current point is calculated. The subproblem is guaranteed to be feasible. It avoids the feasibility recovery algorithm and makes the algorithm more efficient and concise. 2) In our algorithm, point pairs are divided into four partitions by the dominant region of the filter, and we calculate the contributions of the point pairs to the area of the filter separately. The area-type filter algorithm takes the contribution of the trial point to the area of the filter as the acceptance criterion. Our proposed algorithm no longer requires the notion of a "margin" around the filter, a device common to all theoretical approaches to filter methods so far. 3) Compared with traditional filter methods, the use of filter acceptance criteria in our new iteration is relaxed to allow rules that are dominated in some cases. 4) The approach is extended into monotone and nonmonotone methods. The nonmonotone method avoids the situation where the points fall into the "valley" and may be not convergent. In this paper, the numerical analysis is also performed for nonmonotone methods.

Our paper is organized as follows. An area-type nonmonotone filter algorithm for solving nonlinear programming problems is described in the second section. In the third section, the global convergence is established under some certain conditions. The preliminary numerical results of the algorithm are shown in the fourth section. Finally, the fifth section concludes this paper.

2. An area-type nonmonotone filter algorithm

2.1. The SQP step

The iterative approach of the algorithm in this paper is based on the Sequential Quadratic Programming (SQP) algorithm. In the trust-region method, the trial point $x_k + d_k$ of the next iteration is ensured to be within the trust-region centered at the point x_k. Thus, the traditional trust-region method can be used to solve the quadratic programming subproblem given by

$$\begin{aligned} \underset{d}{\text{minimize}} \quad & \langle \nabla f(x_k), d\rangle + \frac{1}{2}\langle d, B_k d\rangle \\ \text{subject to} \quad & c_i(x_k) + \nabla c_i(x_k)^T d \leqslant 0, \\ & \|d\|_\infty \leqslant \Delta_k \end{aligned} \tag{2.1}$$

where Δ_k is the trust-region radius. Su [7] proposes a convex problem to avoid the infeasibility of the trust-region subproblem.

$$\begin{aligned} \underset{d}{\text{minimize}} \quad & \langle \nabla f(x_k), d\rangle + \frac{1}{2}\langle d, B_k d\rangle \\ \text{subject to} \quad & c_i(x_k) + \nabla c_i(x_k)^T d \leqslant \psi^+(x_k, \Delta_k), \quad i \in \mathcal{I}, \\ & \|d\|_\infty \leqslant \Delta_k, \end{aligned} \tag{2.2}$$

where $\Delta_k > 0$, and

$$\begin{aligned} &\psi^+(x_k, \Delta_k) = \max\{\psi(x_k, \Delta_k),\ 0\}, \\ &\psi(x_k, \Delta_k) = \min\left\{\overline{\psi}(x_k, d_k) : \|d_k\| \leqslant \Delta_k\right\}, \\ &\overline{\psi}(x_k, d_k) = \max\left\{c_i(x_k) + \nabla c_i(x_k)^T d_k,\ \ i \in \mathcal{I}\right\}. \end{aligned} \tag{2.3}$$

Based on this, we introduce a relaxation variable τ in the quadratic subproblem to simplify the objective function and control constraints.

$$\begin{aligned} \text{(QP)}\ \underset{d,\tau}{\text{minimize}} \quad & m_k(d, \tau) = \tau + \frac{1}{2}\langle d, B_k d\rangle \\ \text{subject to} \quad & \nabla f(x_k)^T d \leqslant \tau, \\ & c_i(x_k) + \nabla c_i(x_k)^T d \leqslant \psi^+(x_k, \Delta_k),\ \ i \in \mathcal{I}, \\ & \|d\| \leqslant \Delta_k. \end{aligned} \tag{2.4}$$

The $\psi^+(x_k, \Delta_k)$ can be represented as

$$\psi^+(x_k, \Delta_k) = \max\left\{\min_{\|d\|\leqslant\Delta_k}\left\{\max_{j\in\mathcal{I}}\left\{c_j(x_k) + \nabla c_j(x_k)^T d_k\right\}\right\}, 0\right\}. \tag{2.5}$$

As a matter of fact, the (QP) is always feasible, as $(d_k, \tau_k) = (0, 0)$. Thus, at the kth iteration, the solution d_k of (QP) is used as the sequential quadratic programming step for the next iteration. So,

$$x_k^+ = x_k + d_k, \tag{2.6}$$

where x_k^+ is a new trial point.

2.2. The area-type filter criterion

At the current kth iteration, the first step is to consider whether the trial point x_k^+ satisfies the trust-region condition.

$$ared_k = f(x_k) - f(x_k + d_k), \quad pred_k = m_k(0, 0) - m_k(d_k, \tau_k). \tag{2.7}$$

The trust-region ratio is

$$\rho_k = \frac{ared_k}{pred_k} = \frac{f(x_k) - f(x_k + d_k)}{m_k(0,0) - m_k(d_k, \tau_k)}. \tag{2.8}$$

The trial point is accepted when the trust-region ratio is close to 1. As the ratio is close to 0, to decide whether the trial points are acceptable or not, the filter technique is adopted in this paper. Define a constraint violation function as an infeasibility measure.

$$\mathcal{H}(x) = \|c_i(x)^+\|_2 \ , \ i \in \mathcal{I}, \tag{2.9}$$

where

$$c_i(x)^+ = (\max\{c_1(x), 0\}, \max\{c_2(x), 0\}, \ldots, \max\{c_m(x), 0\})^T \ . \tag{2.10}$$

In the traditional filtering technique, a trial point is accepted if the objective function or the constraint violation function is decreasing compared to the result of the filter set. However, in this paper, we adopt an area-type filter.

An area-type filter is one in which the area dominated by the trial point is accepted, and then the point is accepted by the set of filters. Define

$$\mathcal{D}(\mathcal{F}) = \left\{(\mathcal{H}, f) | \mathcal{H} > \mathcal{H}_j \text{ and } f > f_j \text{ for some } j \in \mathcal{F}\right\}$$

such that the pair $(\mathcal{H}, f)$ is dominated by the filter. See Figure 1.

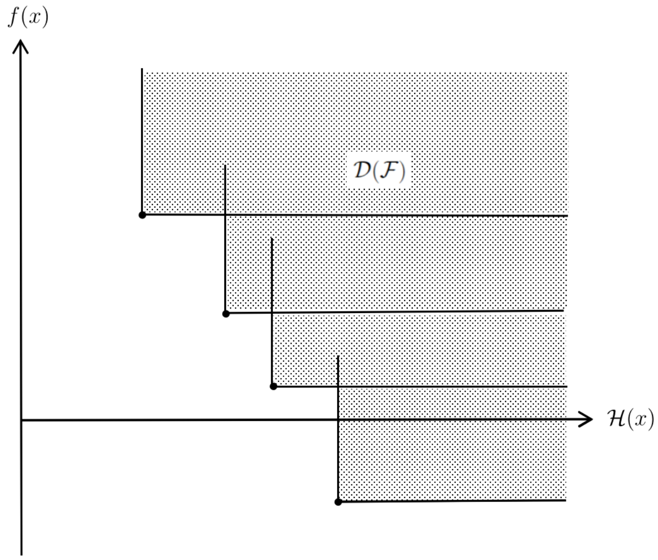

Figure 1. The domination of the filter.

In this paper, the constraint violation degree function and the objective function are put into the two-dimensional surface with $\mathcal{H}(x) \times f(x) \in [0, +\infty] \times [-\infty, +\infty]$. The plane is divided into four regions.

1) The upper left portion of the region, that not dominated by the filter $\mathcal{F}_k$.

$$\mathcal{R}_1(\mathcal{F}_k) \triangleq [0, \min_{j \in \mathcal{F}_k} \mathcal{H}(x_j)) \times (\max_{j \in \mathcal{F}_k} f(x_j), +\infty]. \tag{2.11}$$

2) The lower left portion of the region, that not dominated by the filter $\mathcal{F}_k$.

$$\mathcal{R}_2(\mathcal{F}_k) \triangleq \mathcal{D}(\mathcal{F}_k)^C \cap [0, \max_{j\in\mathcal{F}_k}\mathcal{H}(x_j)) \times (-\infty, \max_{j\in\mathcal{F}_k} f(x_j)], \tag{2.12}$$

where $\mathcal{D}(\mathcal{F}_k)^C$ is the complement of $\mathcal{D}(\mathcal{F}_k)$.

3) The lower right portion of the region, that not dominated by the filter $\mathcal{F}_k$.

$$\mathcal{R}_3(\mathcal{F}_k) \triangleq [\max_{j\in\mathcal{F}_k}\mathcal{H}(x_j), +\infty) \times (-\infty, \min_{j\in\mathcal{F}_k} f(x_j)]. \tag{2.13}$$

4) The region dominated by the filter.

$$\mathcal{R}_4(\mathcal{F}_k) \triangleq \left\{(\mathcal{H}, f) | \mathcal{H} > \mathcal{H}_j \text{ and } f > f_j \text{ for some } j \in \mathcal{F}_k\right\}. \tag{2.14}$$

See Figure 2 for the region division.

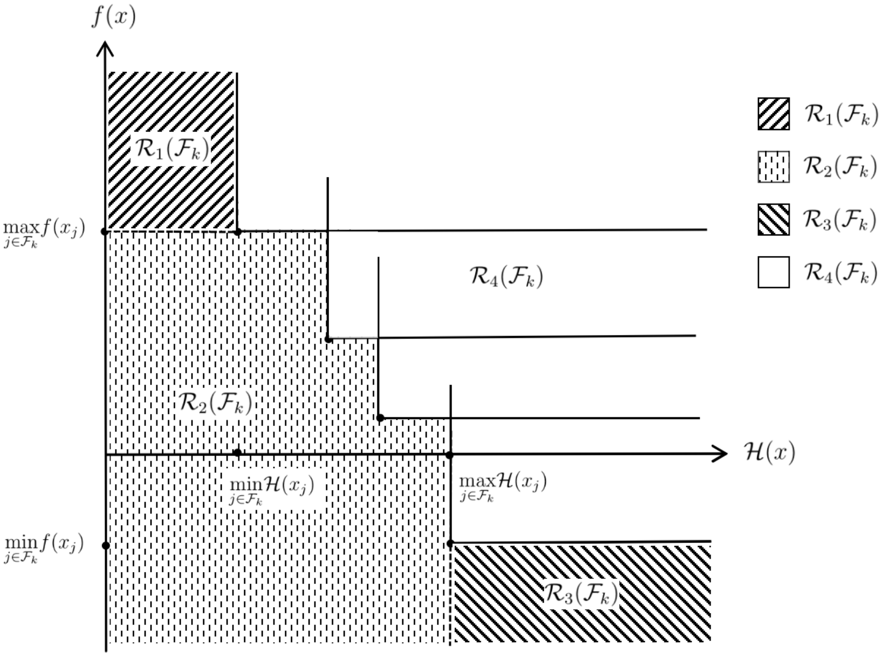

Figure 2. The partition of a filter $\mathcal{F}_k$ containing four $(\mathcal{H}(x), f(x))$ pairs.

Next, we calculate the contribution $\mathcal{A}$ of the corresponding point pairs $(\mathcal{H}(x_k^+), f(x_k^+))$ to the area of the filter for each of the test points x_k^+ , depending on the partition in which the trail point is located.

1) As the trial point x_k^+ is in the undominated upper left partition that is $(\mathcal{H}(x_k^+), f(x_k^+)) \in \mathcal{R}_1(\mathcal{F}_k)$,

$$\mathcal{A}(x_k^+, \mathcal{F}_k) \triangleq \lambda \left(\min_{j\in\mathcal{F}_k}\mathcal{H}(x_j) - \mathcal{H}(x_k^+)\right), \tag{2.15}$$

where λ is a positive constant.

2) As the trial point x_k^+ is in the undominated lower left partition, that is $(\mathcal{H}(x_k^+), f(x_k^+)) \in \mathcal{R}_2(\mathcal{F}_k)$,

$$\mathcal{A}(x_k^+, \mathcal{F}_k) \triangleq area\left(\mathcal{D}(\mathcal{F}_k)^C \cap \left[\mathcal{H}(x_k^+), \max_{j\in\mathcal{F}_k}\mathcal{H}(x_j)\right] \times \left[f(x_k^+), \max_{j\in\mathcal{F}_k} f(x_j)\right]\right). \tag{2.16}$$

3) As the trial point x_k^+ is in the undominated lower right partition, that is $(\mathcal{H}(x_k^+), f(x_k^+)) \in \mathcal{R}_3(\mathcal{F}_k)$,

$$\mathcal{A}(x_k^+, \mathcal{F}_k) \triangleq \lambda \left(\min_{j \in \mathcal{F}_k} f(x_j) - f(x_k^+) \right). \tag{2.17}$$

4) As the trial point x_k^+ is in the dominated partition, that is $(\mathcal{H}(x_k^+), f(x_k^+)) \in \mathcal{R}_4(\mathcal{F}_k)$,

$$\mathcal{A}(x_k^+, \mathcal{F}_k) \triangleq -area \left(\mathcal{D}(\mathcal{F}_k) \cap \left[\mathcal{H}(x_k^+) - \min_{j \in \mathcal{P}_k} \mathcal{H}(x_j) \right] \times \left[f(x_k^+) - \min_{j \in \mathcal{P}_k} f(x_j) \right] \right), \tag{2.18}$$

where

$$\mathcal{P}_k \triangleq \left\{ (\mathcal{H}(x_k), f(x_j)) \in \mathcal{F}_k | \mathcal{H}(x_j) < \mathcal{H}(x_k^+) \text{ and } f(x_j) < f(x_k^+) \right\}. \tag{2.19}$$

Figure 3 shows the contribution of the trial points in four different partitions, respectively, to the area of the filter $\mathcal{F}_k$. The contribution of the trial points in partitions $\mathcal{R}_1(\mathcal{F}_k), \mathcal{R}_2(\mathcal{F}_k)$ and $\mathcal{R}_3(\mathcal{F}_k)$ to the area of the filter is positive, while the contribution in partition $\mathcal{R}_4(\mathcal{F}_k)$ is negative.

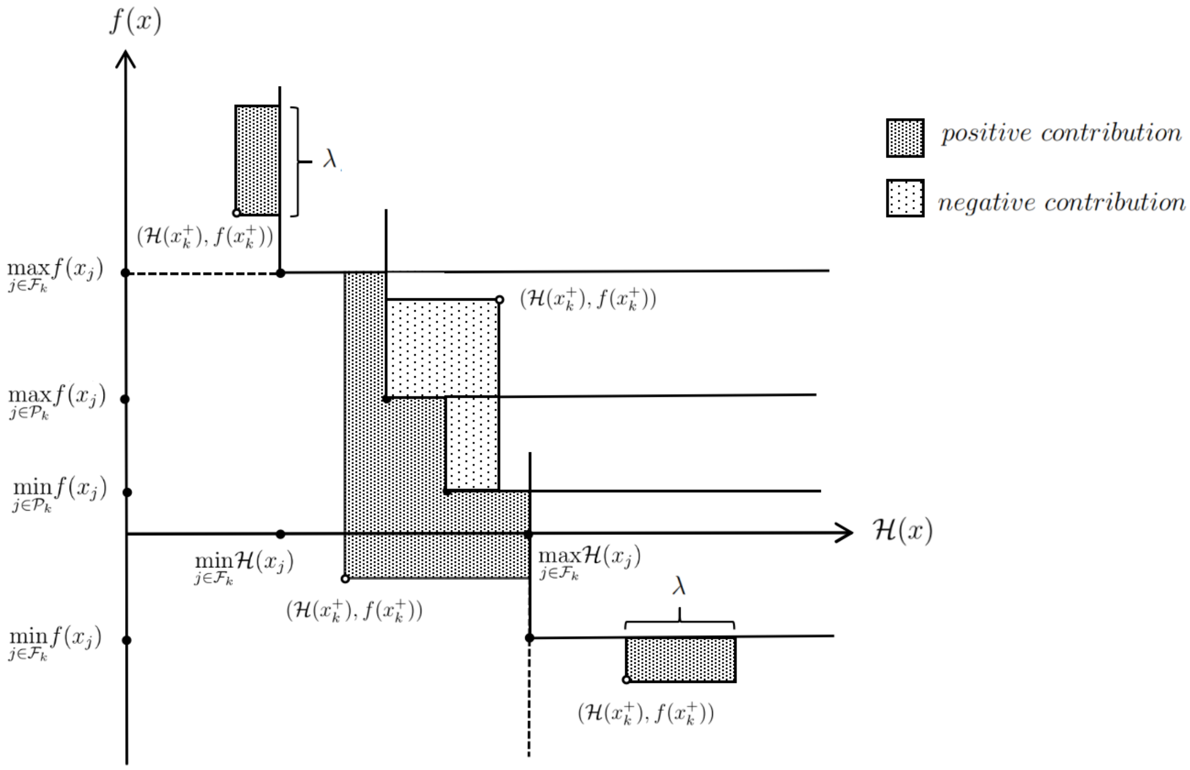

Figure 3. The contribution of four pairs $(\mathcal{H}(x_k^+), f(x_k^+))$ to the area of the filter $\mathcal{F}_k$.

After obtaining the contribution of the trial point to the filter, it can be determined whether the point is accepted by the filter.

In this paper, we give two methods to determine whether the trial point is acceptable for the filter, the monotone method and the nonmonotone method, respectively. x_k^+ satisfies

$$\mathcal{A}(x_k^+, \mathcal{F}_k) \geqslant \lambda \left(\mathcal{H}(x_k^+) \right)^2, \tag{2.20}$$

that is, the acceptance criterion for the monotone method. $\lambda \in (0,1)$ is a positive constant.

Define J as a set of non-negative integers.

$$j \in J = \{0, 1, 2, \dots\}, \tag{2.21}$$

where the maximum value in the set J is the number of updates of the filter, i.e. j_{max}.

Thus, the condition of the nonmonotone method is weaker than that of the monotone method

$$\mathcal{A}'(x_k, \mathcal{F}_k) + \mathcal{A}(x_k^+, \mathcal{F}_k) \geqslant \lambda \left(\mathcal{H}'(x_k)^2 + \mathcal{H}(x_k^+)^2 \right), \tag{2.22}$$

where $\lambda \in (0,1)$, and

$$\mathcal{A}'(x_k, \mathcal{F}_k) = \mathcal{A}'_{j+1}, \ \ \mathcal{H}'(x_k, \mathcal{F}_k) = \mathcal{H}'_{j+1}, \ \ j \in J, \tag{2.23}$$

$$W_0 = 1, \ \ W_{j+1} = \zeta_j W_j + 1, \ \ j \in J, \tag{2.24}$$

$$\mathcal{A}'_0 = \mathcal{A}_0, \ \ \mathcal{H}'_0 = \mathcal{H}_0, \tag{2.25}$$

$$\mathcal{A}'_{j+1} = \frac{\zeta_j W_j \mathcal{A}'_j + \mathcal{A}_{j+1}}{W_j}, \ \ \mathcal{H}'_{j+1} = \frac{\zeta_j W_j \mathcal{H}'_j + \mathcal{H}_{j+1}}{W_j}, \ \ j \in J, \tag{2.26}$$

$\zeta \in [\zeta_{min}, \zeta_{max}], \zeta_{max} < 1$. $\mathcal{A}_j$ is the contribution of the point to the filter at the jth successful iteration. $\mathcal{A}'(x_k, \mathcal{F}_k)$ is the contribution of the updated points to the filter area, which is nonmonotone. $\mathcal{H}'(x_k, \mathcal{F}_k)$ is the nonmonotone average of the constraint violation degree that corresponds to the point of $\mathcal{A}'(x_k, \mathcal{F}_k)$. If (2.22) is satisfied, the point is accepted as a new iteration point even though it may be dominated by the filter. Meanwhile, the set J and $\mathcal{F}$ need to be updated.

As (2.20) or (2.22) holds, the trial point is accepted, which means $x_{k+1} = x_k^+$.

Initialize the set of the filter at the beginning of the algorithm. Define $\mathcal{F}_0 = \varnothing$. If x_{k+1} is not dominated by the filter $x_{k+1} \in \mathcal{D}(\mathcal{F}_k)^C$, that is, x_k is in $\mathcal{R}_1(\mathcal{F}_k), \mathcal{R}_2(\mathcal{F}_k)$ or $\mathcal{R}_3(\mathcal{F}_k)$ partition, then

$$\mathcal{F}_{k+1} = \mathcal{F}_k \cup (\mathcal{H}(x_{k+1}), f(x_{k+1})) \setminus \mathcal{D}_{k+1}. \tag{2.27}$$

The other situation is that x_{k+1} is dominated by the filter, $x_{k+1} \in \mathcal{D}(\mathcal{F}_k)$. Thus, x_{k+1} is in the $\mathcal{R}_4(\mathcal{F}_k)$ partition.

$$\mathcal{F}_{k+1} = (\mathcal{F}_k \setminus \mathcal{P}_k) \cup \left(\min_{j \in \mathcal{P}_k} \mathcal{H}(x_j), f(x_{k+1}) \right) \cup \left(\mathcal{H}(x_{k+1}), \min_{j \in \mathcal{P}_k} f(x_j) \right). \tag{2.28}$$

Add a new iteration point to the filter, but remove the points that can dominate the new iteration points. See Figure 4.

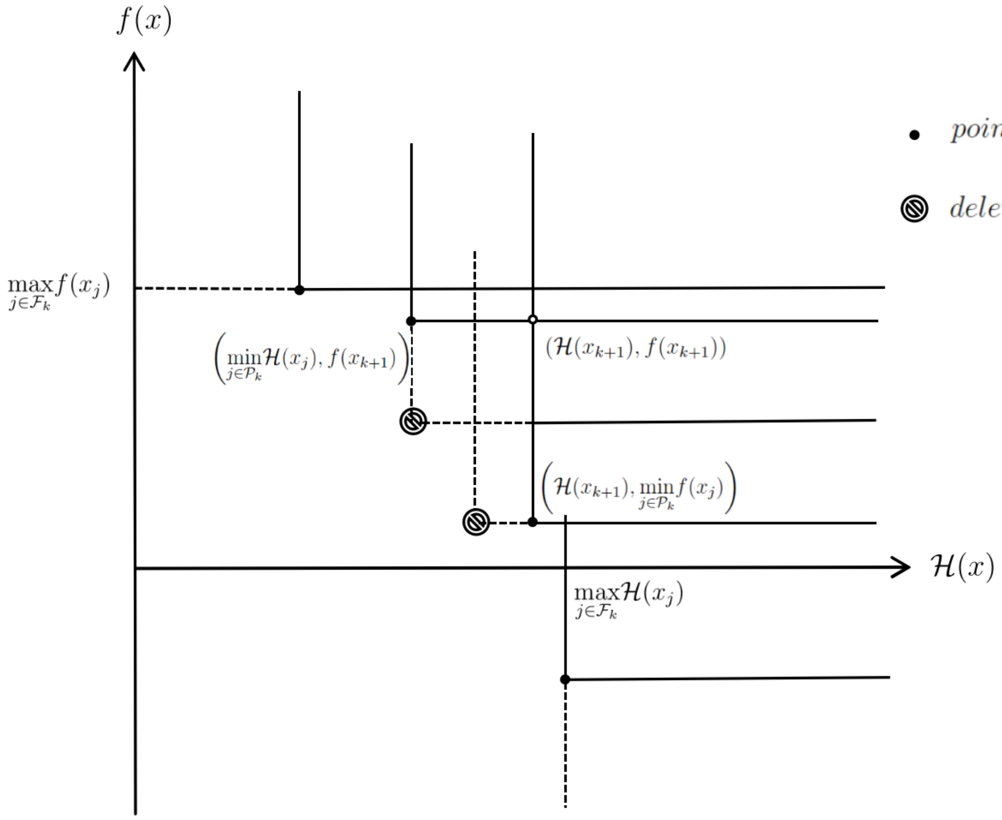

Figure 4. Update the filter $\mathcal{F}_{k+1}$ as $x_{k+1} \in \mathcal{D}(\mathcal{F}_k)$.

For convenience, define

$$\mathcal{U} = \{k |\ x_k^+ \text{ is updated by the filter}\}. \tag{2.29}$$

It is followed by the full statement of the algorithm.

Algorithm 1.

Step 0 Initialization. Given a start point x_0, an initial trust-region radius $\Delta_0 \geqslant 0$ and a symmetric matrix B_0, choose constants $1 > \eta_3 > \eta_2 > 0, \eta_1 \in (1, 2]$ and $\rho_1, \rho_2, \lambda, \zeta \in (0, 1)$. Let $\mathcal{F}_0 = \{(\mathcal{H}_0, f_0)\}$, $k = 0, j = 0$.

Step 1 Solve the modified subproblem (QP) to obtain τ_k and d_k.

Step 2 If $|\tau_k| \leqslant \epsilon$, stop.

Step 3 Set $x_k^+ = x_k + d_k$ and compute ρ_k by (2.8).

Step 4 If $\rho_k \geqslant \rho_1$, go to Step 6.

If $\rho_k \leqslant \rho_2$, set $x_{k+1} = x_k, \Delta_{k+1} = \eta_2 \Delta_k$ and $k = k + 1$. Go to Step 1.

Otherwise, determine the partition of the trial point x_k^+ and compute the contribution of the point to the area of the filter.

Step 5 If x_k^+ is not accepted by the filter $\mathcal{F}_k$, then set $x_{k+1} = x_k, \Delta_{k+1} = \eta_3 \Delta_k$ and $k = k + 1$. Go to Step 1. Otherwise, update the filter $\mathcal{F}_{k+1}$, $j = j + 1$, and go to Step 6.

Step 6 Set $x_{k+1} = x_k^+, \Delta_{k+1} = \eta_1 \Delta_k$. Update B_{k+1}, $k = k + 1$. Go to Step 1.

Remark 1. If the pair $(\mathcal{H}(x_k^+), f(x_k^+))$ of the trial point x_k^+ is on the boundary of the dominated region $\mathcal{D}(\mathcal{F}_k)$, then the contribution of this point to the area of the filter is zero.
Remark 2. j is the number of filter set updates.

3. Convergent properties

Assumption 1. The objective function $f(x)$ and the inequality constraints $c_i(x), i \in \mathcal{I}$ are twice continuous and differentiable. $\nabla f(x)$ is Lipschitz continuous.

Then, there exists a positive constant L, such that

$$\|\nabla f(x) - \nabla f(y)\| \leqslant L\|x - y\|.$$

Assumption 2. Sequence $\{x_k\}$ is a compact convex subset of $\mathbb{R}^n$, generated by Algorithm 1.
Assumption 3. The matrix sequence $\{B_k\}$ and sequence $\{d_k\}$ are uniformly bounded. That is, there exist positive constants $M_1, M_2 > 0$, such that $\|B_k\| \leqslant M_1$ and $\|d_k\| \leqslant M_2$.
Assumption 4. The Mangasarian Fromovitz Constraint Qualification (MFCQ) holds at feasible point $x \in \mathbb{R}^n$. There exists a vector p such that $\{d|\nabla c(x)^T p < 0\} \neq \emptyset$.
Lemma 1. Suppose that Assumptions 1 and 4 hold; the (QP) has an optimal solution.
Proof. First, we prove the feasible region of the (QP) is not an empty set. As $(d_k, \tau_k) = (0, 0)$, the constraints

$$\nabla f(x_k)^T d_k \leqslant \tau_k, \ \ c_i(x_k) + \nabla c_i(x_k)^T d_k \leqslant \psi^+(x_k, \Delta_k), \ \ \|d_k\| \leqslant \Delta_k, \ \ i \in \mathcal{I},$$

hold. Therefore, the feasible region of the subproblem is not an empty set. For every possible solution to the modified (QP), we have $g_k^T d_k \leqslant \tau_k$. Thus,

$$\tau_k + \frac{1}{2} d_k^T B_k d_k \geqslant \nabla f(x_k)^T d_k + \frac{1}{2} d_k^T B_k d_k.$$

Because B_k is a positive symmetry, the objective function of the (QP) has a lower bound. Hence, there exists a constant $V \in (-\infty, +\infty)$, such that

$$V = \inf \left\{ \tau_k + \frac{1}{2} d_k^T B_k d_k : (d_k, \tau_k) \in X_k \right\},$$

$$X_k = \left\{ (d_k, \tau_k) : \nabla f(x_k)^T d_k \leqslant \tau_k, \ c_i(x_k) + \nabla c_i(x_k)^T d_k \leqslant \psi^+(x_k, \Delta_k), \|d_k\| \leqslant \Delta_k, i \in \mathcal{I} \right\},$$

where X_k is the feasible region. From the Weierstrass theorem, there exists a subsequence $\left\{ d_{k_j}, \tau_{k_j} \right\}$ such that

$$\tau_{k_j} + \frac{1}{2} d_{k_j}^T B_{k_j} d_{k_j} \to v, \ \ j \to \infty.$$

Since j is sufficiently large, there is

$$\nabla f(x_{k_j})^T d_{k_j} + \frac{1}{2} d_{k_j}^T B_{k_j} d_{k_j} \leqslant \tau_{k_j} + \frac{1}{2} d_{k_j}^T B_{k_j} d_{k_j} \leqslant V.$$

According to the boundedness of $\left\{ d_{k_j} \right\}_j, \left\{ \tau_{k_j} \right\}_j$, there exists a subset $J \subseteq [0, \infty)$,

$$\lim_{j \to J} (d_{k_j}, \tau_{k_j}) = (d_k, \tau_k) \in X_k, \ \ V = \tau_k + \frac{1}{2} d_k^T B_k d_k.$$

The optimal solution is unique, because the (QP) is a convex programming problem.

Thus, the (QP) has an optimal solution.

Lemma 2. Suppose that Assumptions 1–4 hold, and (d_k, τ_k) is the optimal solution to the (QP); then
(1) (d_k, τ_k) is the Karush-Kuhn-Tucker (KKT) point of the (QP).
(2) $\tau_k = 0$ if and only if $d_k = 0$.

Proof. (1) If (d_k, τ_k) is the optimal solution of (QP), and the constraints are linear functions, then (d_k, τ_k) is the Karush-Kuhn-Tucker point of the modified quadratic problem.
(2) According to the assumptions, $(\tau_k, d_k) = (0, 0)$ is a feasible solution to the (QP). From hypothesis, we know that (τ_k, d_k) is the optimal solution to the (QP). It means that $\tau_k + \frac{1}{2} d_k^T B_k d_k \leqslant 0$.

(i) If $\tau_k = 0$, then $\frac{1}{2} d_k^T B_k d_k \leqslant 0$. Because B_k is a positive definite matrix, $d_k = 0$.

(ii) If $d_k = 0$, according to the positive quality of B_k, we have $\tau_k \leqslant 0$. By the constraint condition of the modified (QP), we have $\nabla f(x_k)^T d_k \leqslant \tau_k$, and then $\tau_k \geqslant 0$. Therefore, $\tau_k = 0$.

The proof is completed.

$\tau_k = 0$ can serve as the termination condition of Algorithm 1 for the following lemma.

Lemma 3. Suppose that Assumptions hold, and $\tau_k = 0$ is the optimal solution of the modified (QP); then, x_k is the (KKT) point of the problem (1.1).

Proof. From Lemmas 1 and 2, from $\tau_k = 0$, we get $d_k = 0$.

First we need to prove $\psi^+(x_k, \Delta_k) = 0$. Suppose $\psi^+(x_k, \Delta_k) > 0$, and there exists

$$x \in \{x : \max\{c_i(x_k), i \in \mathcal{I}\}\}.$$

Let

$$\mathcal{M}(x) = \{i : c_i(x) \geqslant 0, i \in \mathcal{I}\}.$$

From Mangasarian-Fromovitz constraint qualification, there exits $d \in \mathbb{R}^n$ and $\|d\| \leqslant \Delta_k$ such that

$$c_i(x_k) + \nabla c_i(x_k) d < c_i(x_k), i \in \mathcal{M}(x_k),$$

$$c_i(x_k) + \nabla c_i(x_k) d < 0, i \in \mathcal{I} \backslash \mathcal{M}(x_k).$$

Thus,

$$\max\left\{c_i(x_k) + \nabla c_i(x_k)^T d_k, i \in \mathcal{I}\right\} < \max\{c_i(x_k), i \in \mathcal{I}\},$$

that is,

$$\overline{\psi}(x_k, d_k) < \max\{c_i(x_k), i \in \mathcal{I}\}.$$

So,

$$\psi(x_k, \Delta_k) < \max\{c_i(x_k), i \in \mathcal{I}\}.$$

We have $0 \notin \left\{d : c_i(x_k) + \nabla c_i(x_k)^T d_k \leqslant \psi^+(x_k, \Delta_k), \|d\| \leqslant \Delta_k, i \in \mathcal{I}\right\}$. This contradicts with $d_k = 0$. Hence, $\psi^+(x_k, \Delta_k) = 0$.

Next, we prove $u_i \neq 0$. According to (2.4), there exist Lagrange multipliers $U = (u_1, u_2, \ldots, u_n)^T$

$,V = (v_1, v_2, \ldots, v_m)^T, W = (w_1, w_2, \ldots, w_n)^T$ and $L = (l_1, l_2, \ldots, l_n)^T$, such that

$$\begin{aligned}
&B_k d + \nabla f(x_k)^T U + \nabla c_i(x_k)^T V + W - L = 0, \\
&1 - \sum_{i=1}^{n} u_i = 0, \\
&\sum_{i=1}^{n} u_i (\nabla f(x_k)^T d - \tau) = 0, \\
&\sum_{i=1}^{m} v_i (c_i(x_k) + \nabla c_i(x_k)^T d - \psi^+) = 0, \\
&W^T (d - \Delta_k e) = 0, L^T(-d - \Delta_k e) = 0, \\
&\nabla f(x_k)^T d - \tau \leqslant 0, c_i(x_k) + \nabla c_i(x_k)^T d - \psi^+ \leqslant 0, \|d\| - \Delta_k \leqslant 0, \\
&U \geqslant 0, V \geqslant 0, W \geqslant 0, L \geqslant 0,
\end{aligned} \tag{3.1}$$

where $e = (1, 1, \cdots, 1)^T_{n \times 1}$.

With $\tau_k = 0$ and $d_k = 0$, the KKT condition of the (QP) can be reworded as

$$\begin{aligned}
&g_k^T U + \nabla c_i(x_k)^T V + W - L = 0, \\
&1 - \sum_{i=1}^{n} u_i = 0, \\
&\sum_{i=1}^{m} v_i (c_i(x_k) - \psi^+) = 0, \\
&- W^T (\Delta_k e) = 0, -L^T(\Delta_k e) = 0, \\
&c_i(x_k) - \psi^+ \leqslant 0, \\
&U \geqslant 0, V \geqslant 0, W \geqslant 0, L \geqslant 0.
\end{aligned} \tag{3.2}$$

From the above, we have $\sum_{i=1}^{n} u_i = 1$. Thus, $u_i \neq 0$. By the KKT condition, we have $W = L = 0$. Let

$$\lambda_i = \frac{v_i}{u_i}.$$

The KKT condition of the problem (1.1) is

$$\begin{aligned}
&\nabla f(x_k) + \sum_{i=1}^{m} \lambda_i \nabla c_i(x_k) = 0, \\
&c_i(x_k) \leqslant 0, \\
&\sum_{i=1}^{m} \lambda_i c_i(x_k) = 0, \\
&\lambda_i \geqslant 0.
\end{aligned} \tag{3.3}$$

Thus, there exist Lagrange multipliers $\lambda \in \mathbb{R}^n$, $\lambda_i = \frac{v_i}{u_i}, \sum_{i=1}^{n} u_i = 1$ and $W = L = 0$ such that x_k is a KKT point of the inequality constraint problem (1.1).

Lemma 4. Suppose that Assumptions 1-4 hold, and then for all k, we have

$$|(f(x_k) - f(x_k + d)) - (m_k(0,0) - m_k(d_k, \tau_k))| \leqslant O(\|d^2\|). \tag{3.4}$$

Proof. From Lemma 1, the subproblem is always feasible. So, $\nabla f(x_k)^T d_k \leqslant \tau_k$ and $\tau_k \geqslant 0$.

According to Taylor's theorem, we obtain

$$\begin{aligned}
&|(f(x_k) - f(x_k + d)) - (m_k(0,0) - m_k(d_k, \tau_k))| \\
&= |f_{x_k} - \left(f(x_x) + \nabla f(x_k)^T d_k + \int_0^1 d_k^T \left(\nabla f(x_k)\,(x_k + \xi d_k) - \nabla f(x_k)\right)\, dx\right)| \\
&= |f(x_k) - \left(f(x_k) + \nabla f(x_k)^T d_k + \int_0^1 d_k^T \left(\nabla f(x_k)\,(x_k + \xi d_k) - \nabla f(x_k)\right)\, d\xi\right) + \tau_k + \frac{1}{2} d_k B_k^T d_k| \\
&\leqslant | - \nabla f(x_k)^T d_k + \int_0^1 \|d_k\| \|\nabla f(x_k)\,(x_k + \xi d_k - \nabla f(x_k))\|\, d\xi + \tau_k + \frac{1}{2} d_k B_k^T d_k| \\
&\leqslant \frac{1}{2}\|d_k\|^2 M_1 + L \int \|d_k\|\, d\xi + \tau_k - \nabla f(x_k)^T d_k \\
&\leqslant \frac{1}{2}\|d_k\|^2 M_1 + L \int \|d_k\|^2 + \frac{1}{2}\|d_k\|^2 \tau_k \\
&= O(\|d_k\|^2).
\end{aligned} \tag{3.5}$$

The proof is completed.

x_k is generated by Algorithm 1. We consider that there exists a constant $\epsilon > 0$, such that $\|\nabla f(x_k)\| > \epsilon$. Then, for every k, x_{k+1} is a successful iteration point.

Lemma 5. Suppose that Assumptions hold. Then, the inner loop of the Algorithm 1 terminates finitely.

Proof. There are two inner loops in the algorithm, but their proofs are the same, as follows. Assume that the algorithm does not terminate at finite iterations. That is, in kth, x_{k+1} is not the successful iteration point.

From Algorithm 1, we get $\rho_k < \rho_1, \Delta_k \to 0$ and $d_k \to 0$.

$$\begin{aligned}
|\rho_k - 1| &= |\frac{ared}{pred} - 1| = |\frac{ared - pred}{pred}| \\
&= |\frac{(f(x_k) - f(x_k + d_k)) - (m_k(0,0) - m_k(d_k, \tau_k))}{m_k(0,0) - m_k(d_k, \tau_k)}|.
\end{aligned} \tag{3.6}$$

From the last lemma,

$$|(f(x_k) - f(x_k + d)) - (m_k(0,0) - m_k(d_k, \tau_k))| \leqslant O(\|d^2\|). \tag{3.7}$$

We obtain

$$\begin{aligned}
|\rho_k - 1| &\leqslant \frac{O(\|d^2\|)}{m_k(0,0) - m_k(d_k, \tau_k)} = \frac{O(\|d^2\|)}{\tau_k + \frac{1}{2} d_k B_k^T d_k} \\
&\leqslant \frac{O(\|d^2\|)}{\nabla f(x_k)^T d_k + \frac{1}{2} d_k B_k^T d_k} \\
&\to 0.
\end{aligned} \tag{3.8}$$

This means $\rho_k \geqslant \rho_1$, which contradicts $\rho_k < \rho_1$.

Thus the inner loop of Algorithm 1 terminates at a finite number of iterations.

Lemma 6. Suppose that Assumptions hold. In the iteration, for each k,

$$area\,(\mathcal{D}\,(\mathcal{F}_k)) \geqslant \lambda \sum_{i\in\mathcal{U}} \mathcal{H}(x_i)^2. \tag{3.9}$$

Proof. The proof is similar to Lemma 3.3 in Gould et al. [32].

Lemma 7. Suppose that Assumptions hold, and the filter set is updated infinitely. That is, $|\mathcal{U}| = +\infty$. Then, there exists a subsequence $\left\{k_j\right\}$ such that

$$\lim_{j\to\infty} \mathcal{H}(x_{k_j}) = 0. \tag{3.10}$$

Proof. Suppose that there exists an infinite subsequence $k_i \subseteq \mathcal{U}$ such that $\mathcal{H}(x_{k_i}) \geqslant \epsilon$ for $\epsilon > 0$. According to Lemma 6, we deduce that

$$area\,(\mathcal{D}\,(\mathcal{F}_{k_i+1})) \geqslant i\lambda\epsilon^2. \tag{3.11}$$

From Assumptions 1 and 2 can be directly derived that, for all k,

$$f^{min} \leqslant f(x_k) \leqslant f^{max}, \tag{3.12}$$

and

$$0 \leqslant \mathcal{H}(x_k) \leqslant \mathcal{H}^{max}, \tag{3.13}$$

for some constants $f^{min} \leqslant f^{max}$ and $0 \leqslant \mathcal{H}^{max}$. This means that the domain of the $(f, \mathcal{H})-$points is $\left[f^{min}, f^{max}\right] \times [0, \mathcal{H}^{max}]$.

Hence, for any k, $area\,(\mathcal{D}\,(\mathcal{F}_{k_i+1}))$ is bounded above by a constant $\sigma^{max} \geqslant 0$ independent of k. Thus, we have

$$i \leqslant \frac{\sigma^{max}}{\lambda\epsilon^2}. \tag{3.14}$$

Therefore, i is finite. This contradicts that the subsequence k_i is infinite.

$$\lim_{j\to\infty,\ k\in\mathcal{U}} \mathcal{H}(x_k) = 0. \tag{3.15}$$

So, the conclusion is valid.

Lemma 8. Suppose that Assumptions hold, and the filter is updated finitely. This means that $|\mathcal{U}| < \infty$. Then, $\mathcal{H}(x_k) = 0$.

Proof. According to the termination condition of Algorithm 1, if the algorithm is finitely terminated, we obtain $\tau_k = 0$ and $d_k = 0$. Thus, $\mathcal{H}(x_k) = 0$. The proof is completed.

Theorem 1. Suppose that Assumptions hold, and sequence $\{x_k\}$ is obtained by Algorithm 1. There exists a subsequence $\left\{x_{k_j}\right\}$ and

$$\lim_{j\to\infty} x_{k_j} = x^*.$$

The cluster point x^* is the KKT point for the problem (1.1).

Proof. It is easy to prove by the above hypothesis and lemma.

4. Numerical results

We have implemented the preliminary numerical result of the Algorithm 1 in Matlab R2020a. All experiments are run on a laptop with Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz 2.30 GHz and 16GB RAM. The performance of our area-type filter method is compared with other methods for solving nonlinear programming problems. SNOPT is an SQP algorithm that uses a smooth augmented Lagrangian merit function [33]. IPOPT is an interior-point filter line-search algorithm for nonlinear programming [34]. In this paper, the monotone and the nonmonotone area-type methods are denoted by Monotone and Nonmonotone, respectively.

The following values are exploited: $B_0 = I \in R^n \times R^n, \rho_1 = 0.75, \rho_2 = 0.01, \lambda = 0.0001, \eta_1 = 2, \eta_2 = 0.1, \eta_3 = 0.5$and $\zeta = 0.85$. In addition, B_{k+1} is updated by the BFGS formula [35]:

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

where

$$y_k = \delta_k y_k' + (1 - \delta_k) H_k p_k,$$

$$y_k' = \nabla f(x_{k+1}) - \nabla f(x_k), p_k = x_{k+1} - x_k,$$

and

$$\delta_k = \begin{cases} 1, & p_k^T y_k' \geq 0.2 p_k^T H_k p_k, \\ \frac{0.8 p_k^T H_k p_k}{p_k^T H_k p_k - p_k^T y_k'}, & \text{otherwise.} \end{cases}$$

$\epsilon = 10^{-4}$ is the convergence tolerance of the algorithm.

The test problems are from [36,37]. In Table 1, the number of iterations calculated by monotone and nonmonotone area-type filter algorithms are compared with the traditional filter method (FILTER) [20], SNOPT and IPOPT. n,m are the numbers of problem variables and general constraints. NIT represents the number of iterations to solve the nonlinear programming problem.

For simplicity of comparison, we have compared the efficiency of the number of iterations in Figure 5 by using the performance profile [38]. Define the performance profile by

$$\Theta_s(T) = \frac{\text{The number of problems where } log_2(R_{p,s})}{\text{Total number of problems}}, \tag{4.1}$$

where

$$R_{p,s} = \frac{NIT_{p,s}}{\min\limits_{s \in S} NIT_{p,s}}. \tag{4.2}$$

$NIT_{p,s}$ means the number of iterations calculated by solver S for problem P. $\min\limits_{s \in S} NIT_{p,s}$ means the minimum of the number of iterations to solve problem P for all solvers compared.

Table 1. Comparison of NIT.

Problem	n	m	monotone	nonmonotone	FILTER	SNOPT	IPOPT
			NIT	NIT	NIT	NIT	NIT
HS03	2	1	8	5	11	2	5
HS04	2	2	2	2	2	4	6
HS07	2	2	8	8	23	30	28
HS09	2	2	17	19	8	10	6
HS10	2	1	11	11	-	31	13
HS13	2	3	16	16	-	17	79
HS14	2	3	8	8	6	10	8
HS15	2	3	4	13	4	11	21
HS16	2	5	10	10	3	5	23
HS17	2	5	7	7	14	19	18
HS18	2	6	15	15	28	32	27
HS19	2	6	10	10	6	9	16
HS21	2	5	2	2	4	1	9
HS22	2	2	8	8	5	7	7
HS24	2	5	5	5	5	8	13
HS27	3	2	6	6	7	21	143
HS30	3	7	5	5	32	5	16
HS31	3	7	8	8	-	11	8
HS32	3	6	11	14	8	5	20
HS33	3	6	7	7	12	9	16
HS34	3	8	12	12	8	7	10
HS35	3	4	5	5	8	5	8
HS39	4	4	22	16	-	28	14
HS40	4	6	8	11	57	9	4
HS41	4	10	6	6	7	7	12
HS44	4	10	18	18	6	2	20
HS45	5	10	8	8	9	2	48
HS46	5	4	13	13	-	32	20
HS48	5	4	8	8	8	6	2
HS49	5	4	9	9	12	34	20

When the solver S cannot solve the problem P, the ratio R goes to infinity. If $T \to \infty$, $\Theta_s(T)$ is the percentage of the number of problems that can be solved by solver S. The performance of the considered algorithm is best in the range of the optimal T.

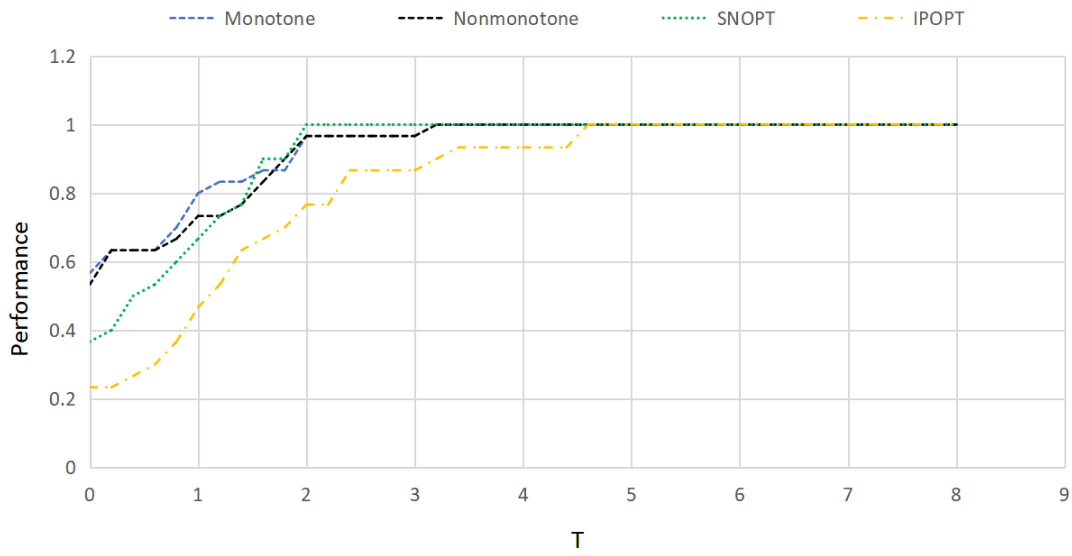

Figure 5. Performance profile.

Next, the different results of monotone and nonmonotone are taken to have a comparison in detail. The number of iterations, CPU runtime and optimal values are compared, and the corresponding results are shown in Table 2.

Table 2. Comparison of Monotone and Nonmonotone.

Problem	n	m	Monotone			Nonmonotone		
			NIT	CPU TIME	f(x*)	NIT	CPU TIME	f(x*)
HS03	2	1	8	0.994	-0.0009	5	0.982	0.0009
HS09	2	2	17	0.866	-0.4388	19	0.961	-0.4999
HS15	2	3	4	0.723	306.500	13	1.289	306.499
HS32	3	6	11	0.775	1.0000	14	2.029	0.9033
HS39	4	4	22	4.367	-1.0000	16	3.978	-0.9992
HS40	4	6	8	1.124	-0.2513	11	1.865	-0.2498

As shown in Table 1 and Figure 5, most of the area-type filter algorithm is better than FILTER, SNOPT, and IPOPT. Our algorithm usually requires a small number of iterations. As a whole, our algorithm can be applied to this problem generally. There are slightly worse cases for individual problems. However, on the scale of the problem, it can be negligible. In most of the problems, Algorithm 1 requires fewer iterations, such as HS17, HS18, HS33, HS49 and so on. As for HS10, HS13, HS31, HS39 and HS46, the traditional filter method can not obtain the optimal solution as the subproblem is infeasible, but this paper can avoid this problem. Moreover, Algorithm 1 is a good solver. It can be observed from Table 2 that the monotone method is slightly better than the nonmonotone method in terms of both the number of iterations and CPU time. The nonmonotone method avoids the situation where the points fall into the "valley" and cannot get out, making the algorithm more efficient. As can be seen from the above table, our algorithm outperforms the traditional filter method for most problems. The result indicates the use of the area-type filter method provides a fast, convergent mechanism that reduces the number of iterations.

5. Conclusions

In this paper, a new area-type filter algorithm is proposed based on the trust-region method. A relaxed trust-region quadratic correction subproblem is proposed to compute the direction. The subproblem is guaranteed to be feasible. It avoids the feasibility recovery phase and makes the algorithm more efficient and concise. The discriminant criterion is different from that of the traditional filter method. The point pairs are divided into four partitions by the dominant region of the filter. The area-type filter algorithm takes the contribution of the trial point to the area of the filter as the acceptance criterion. The criterion is extended into monotone and nonmonotone methods. Both of them are compared, and better numerical results are obtained. The global convergence of the area-type filter method is also demonstrated. This algorithm can reduce the computational effort to a certain extent, as shown in the numerical results.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (No. 61572011) and the Natural Science Foundation of Hebei Province (No. A2022201002).

Conflict of interest

All authors disclosed no relevant relationships.

References

1. J. C. Tang, C. M. Fu, C. J. Mi, H. B. Liu, An interval sequential linear programming for nonlinear robust optimization problems, *Appl. Math. Model.*, **107** (2022), 256–274. https://doi.org/10.1016/j.apm.2022.02.037
2. M. D. Yang, D. Q. Zhang, X. Han, New efficient and robust method for structural reliability analysis and its application in reliability-based design optimization, *Comput. Method. Appl. Mech. Eng.*, **366** (2020), 113018. https://doi.org/10.1016/j.cma.2020.113018
3. M. D. Yang, D. Q. Zhang, C. Jiang, X. Han, Q. Li, A hybrid adaptive Kriging-based single loop approach for complex reliability-based design optimization problems, *Reliab. Eng. Syst. Safe.*, **215** (2022), 107736. https://doi.org/10.1016/j.ress.2021.107736
4. N. C. Xiao, K. Yuan, C. N. Zhou, Adaptive kriging-based efficient reliability method for structural systems with multiple failure modes and mixed variables, *Comput. Method. Appl. Mech. Eng.*, **359** (2020), 112649. https://doi.org/10.1016/j.cma.2019.112649

5. F. E. Curtis, N. I. M. Gould, D. P. Robinson, P. L. Toint, An interior-point trust-funnel algorithm for nonlinear optimization, *Math. Program.*, **161** (2017), 73–134. https://doi.org/10.1007/s10107-016-1003-9
6. C. Gu, D. T. Zhu, Global and local convergence of a new affine scaling trust region algorithm for linearly constrained optimization, *Acta Math. Sin.-English Ser.*, **32** (2016), 1203–1213. https://doi.org/10.1007/s10114-016-4513-8
7. K. Su, X. C. Li, R. Y. Hou, A nonmonotone flexible filter method for nonlinear constrained optimization, *J. Math. Industry*, **6** (2016), 8. https://doi.org/10.1186/s13362-016-0029-1
8. X. J. Zhu, On a globally convergent trust region algorithm with infeasibility control for equality constrained optimization, *J. Appl. Math. Comput.*, **50** (2016), 275–298. https://doi.org/10.1007/s12190-015-0870-1
9. M. J. D. Powell, On the convergence of trust region algorithms for unconstrained minimization without derivatives, *Comput. Optim. Appl.*, **53** (2012), 527–555. https://doi.org/10.1007/s10589-012-9483-x
10. X. F. Ding, Q. Qu, X. Y. Wang, A modified filter nonmonotone adaptive retrospective trust region method, *PLoS ONE*, **16** (2021), e0253016. https://doi.org/10.1371/journal.pone.0253016
11. A. Kamandi, K. Amini, A new nonmonotone adaptive trust region algorithm, *Appl. Math.*, **67** (2022), 233–250. https://doi.org/10.21136/AM.2021.0122-20
12. J. J. Liu, X. M. Xu, X. H. Cui, An accelerated nonmonotone trust region method with adaptive trust region for unconstrained optimization, *Comput. Optim. Appl.*, **69** (2018), 77–97. https://doi.org/10.1007/s10589-017-9941-6
13. I. S. Duff, J. Nocedal, J. K. Reid, The use of linear programming for the solution of sparse sets of nonlinear equations, *SIAM J. Sci. Stat. Comput.*, **8** (1987), 99–108. https://doi.org/10.1137/0908024
14. J. Y. Fan, J. Y. Pan, An improved trust region algorithm for nonlinear equations, *Comput. Optim. Appli.*, **48** (2011), 59–70. https://doi.org/10.1007/s10589-009-9236-7
15. H. C. Zhang, A. R. Conn, K. Scheinberg, A derivative-free algorithm for least-squares minimization, *SIAM J. Optim.*, **20** (2010), 3555–3576. https://doi.org/10.1137/09075531X
16. C. Cartis, N. I. Gould, P. L.Toint, Trust-region and other regularisations of linear least-squares problems, *Bit. Numer. Math.*, **49** (2009), 21–53. https://doi.org/10.1007/s10543-008-0206-8
17. S. A. Karbasy, M. Salahi, On the branch and bound algorithm for the extended trust-region subproblem, *J. Glob. Optim.*, **83** (2022), 221–233. https://doi.org/10.1007/s10898-021-01104-0
18. C. Kanzow, A. Klug, An interior-point affine-scaling trust-region method for semismooth equations with box constraints, *Comput. Optim. Appl.*, **37** (2007), 329–353. https://doi.org/10.1007/s10589-007-9029-9
19. Y. X. Yuan, Recent advances in trust region algorithms, *Math. Program.*, **151** (2015), 249–281. https://doi.org/10.1007/s10107-015-0893-2
20. R. Fletcher, S. Leyffer, Nonlinear programming without a penalty function, *Math. Program.*, **91** (2002), 239–269. https://doi.org/10.1007/s101070100244
21. R. Fletcher, S. Leyffer, P. L. Toint, On the global convergence of a filter-SQP algorithm, *SIAM J. Optim.*, **13** (2002), 44–59. https://doi.org/10.1137/S105262340038081X

22. C. Gu, D. T. Zhu, A dwindling filter algorithm with a modified subproblem for nonlinear inequality constrained optimization, *Chin. Ann. Math. Ser. B*, **35** (2014), 209–224. https://doi.org/10.1007/s11401-014-0826-z
23. J. Y. Wang, C. Gu, G. Q. Wang, Some results on the filter method for nonlinear complementary problems, *J. Inequal. Appl.*, **2021** (2021), 30. https://doi.org/10.1186/s13660-021-02558-2
24. X. J. Zhu, D. G. Pu, A restoration-free filter SQP algorithm for equality constrained optimization, *Appl. Math. Comput.*, **219** (2013), 6016–6029. https://doi.org/10.1016/j.amc.2012.12.002
25. K. Su, A globally and superlinearly convergent modified SQP-filter method, *J. Glob. Optim.*, **41** (2008), 203–217. https://doi.org/10.1007/s10898-007-9219-0
26. H. Ahmadzadeh, N. Mahdavi-Amiri, A competitive inexact nonmonotone filter SQP method: Convergence analysis and numerical results, *Optim. Method. Softw.*, 2021, 1–34. https://doi.org/10.1080/10556788.2021.1913155
27. A. Q. Huang, A filter-type method for solving nonlinear semidefinite programming, *Appl. Numer. Math.*, **158** (2020), 415–424. https://doi.org/10.1016/j.apnum.2020.08.012
28. S. Leyffer, C. Vanaret, An augmented Lagrangian filter method, *Math. Meth. Oper. Res.*, **92** (2020), 343–376. https://doi.org/10.1007/s00186-020-00713-x
29. K. Su, D. G. Pu, A nonmonotone filter trust region method for nonlinear constrained optimization, *J. Comput. Appl. Math.*, **223** (2009), 230–239. https://doi.org/10.1016/j.cam.2008.01.013
30. X. Y. Wang, X. F. Ding, Q. Qu, A new filter nonmonotone adaptive trust region method for unconstrained optimization, *Symmetr*y, **12** (2020), 208. https://doi.org/10.3390/sym12020208
31. W. J. Xue, W. L. Liu A multidimensional filter SQP algorithm for nonlinear programming, *J. Comput. Math.*, **38** (2020), 683–704. DOI: 10.4208/jcm.1903-m2018-0072
32. N. I. M. Gould, P. L.Toint, Global convergence of a non-monotone trust-region filter algorithm for nonlinear programming, In: *Multiscale optimization methods and applications*, Boston, MA: Springer, 2006, 125–150. https://doi.org/10.1007/0-387-29550-X_5
33. P. E. Gill, W. Murray, M. A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, *SIAM Rev.*, **47** (2005), 99–131. https://doi.org/10.1137/S0036144504446096
34. A. Wächter, L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Math. Program.*, **106** (2006), 25–57. https://doi.org/10.1007/s10107-004-0559-y
35. J. Nocedal, S. Wright, *Numerical optimization*, New York: Springer, 1999. https://doi.org/10.1007/b98874
36. W. Hock, K. Schittkowski, Test examples for nonlinear programming codes, *J. Optim. Theory Appl.*, **30** (1980), 127–129. https://doi.org/10.1007/BF00934594
37. K. Schittkowski, *More test examples for nonlinear programming codes*, Berlin, Heidelberg: Springer, 1987. https://doi.org/10.1007/978-3-642-61582-5
38. E. D. Dolan, J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.*, **91** (2002), 201–213. https://doi.org/10.1007/s101070100263

AIMS Press