OPTIMIZATION

A subspace inertial method for derivative-free nonlinear monotone equations

Morteza Kimiaei[a], Abdulkarim Hassan Ibrahim[b], Susan Ghaderi[c]

[a]Fakultät für Mathematik, Universität Wien, Oskar-Morgenstern-Platz 1, A-1090 Wien, Austria; [b]Interdisciplinary Research Center (IRC) for Smart Mobility and Logistics, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia; [c]Leuven.AI - KU Leuven Institute for AI, B-3000, Leuven, Belgium

ARTICLE HISTORY
Compiled September 11, 2023

ABSTRACT
We introduce a subspace inertial line search algorithm (`SILSA`), for finding solutions of nonlinear monotone equations (NME). At each iteration, a new point is generated in a subspace generated by the previous points. Of all finite points forming the subspace, a point with the largest residual norm is replaced by the new point to update the subspace. In this way, `SILSA` leaves regions far from the solution of NME and approaches regions near it, leading to a fast convergence to the solution. This study analyzes global convergence and complexity upper bounds on the number of iterations and the number of function evaluations required for `SILSA`. Numerical results show that `SILSA` is promising compared to the basic line search algorithm with several known derivative-free directions.

KEYWORDS
Monotone equations; derivative-free optimization; inertial technique; global convergence; complexity result

1. Introduction

This paper introduces an efficient derivative-free algorithm for solving *monotone equations*

$$F(x) = 0, \ x \in \mathbb{R}^n, \tag{1}$$

where x is a vector in $\mathbb{R}^n$ and $F : \mathbb{R}^n \to \mathbb{R}^n$ is a monotone function, possibly with large n. In finite precision arithmetic, for a given threshold $\varepsilon > 0$ and an initial point $x_0 \in \mathbb{R}^n$, the algorithm finds an ε-approximate solution x_ε of the problem (1), where the residual norm of x_ε is below $\min(\varepsilon, \|F(x_0)\|)$. Here $\|\cdot\|$ is the Euclidean norm.

CONTACT Morteza Kimiaei. Email: kimiaeim83@univie.ac.at

The problem (1) appears in various practical applications, including constrained neural networks [1], nonlinear compressed sensing [2], phase retrieval [3], and economic and chemical equilibrium problems [4].

Different algorithms [5–17] have been proposed and analyzed for finding an ε-approximate solution of the problem (1). However, these approaches either require the computation of the (real) Jacobian matrix, which can be computationally expensive and memory-intensive, making them unsuitable for high-dimensional problems, or use its approximation, which may require a large number of function evaluations in high dimensions. As a result, these methods are not ideal for tackling large scale nonlinear systems of equations.

To address these limitations, Solodov and Svaiter [18] proposed a basic line search algorithm (`BLSA`) augmented with a projected scheme for finding an ε-approximate solution of (1). Compared to methods that require the (real) Jacobian matrix or its approximation, derivative-free methods [19–38] have a simpler structure and lower memory requirements, making them suitable for solving large-scale problems. Nevertheless, it should be noted that `BLSA` does not provide a guarantee for reducing the residual norm at every accepted point. Instead, the residual norm can non-monotonically jump down or up until an ε-approximate solution of (1) is obtained. Hence, the convergence rate of `BLSA` is relatively slow.

To accelerate the convergence rate of `BLSA` and obtain an ε-approximate solution of (1), one potential approach is to integrate `BLSA` with inertial methods, such as those proposed in [39–45], which construct steps based on the two previous accepted points, as described in [29–31]. Despite using two previous points to generate new points, these methods still do not guarantee that the resulting points have the lowest residual norms in comparison to the prior accepted points. Another potential approach proposed in [46] is to augment the algorithm with an extrapolation step in the line search condition to enforce residual norm reduction at each accepted point. However, in ill-conditioned problems, this technique may face challenges in finding such points since it does not accept points whose residual norms have not been reduced.

The global convergence of `BLSA` with various derivative-free directions to find an ε-approximate solution of (1), has been established in [24,29–32,35]. However, based on our knowledge, no attempt has been made to find out the maximum number of iterations and the maximum number of function evaluations required to find an ε-approximate solution of (1) for `BLSA`. Thus, if complexity upper bounds on the number of function evaluations and the number of iterations are found, it would be interesting to know the cost of the algorithm before the algorithm is implemented and to know what parameters are appeared in such complexity bounds.

1.1. Contribution

This study proposes a new derivative-free line search algorithm, named subspace inertial line search algorithm (`SILSA`), which aims to find an ε-approximate solution of the problem (1) in Euclidean space. The underlying mapping of (1) is monotone and Lipschitz continuous. The proposed subspace inertial point uses the information of the previous evaluated points. It uses a procedure to replace a point with the largest residual norm in the subspace with a new evaluated point. Due to this replacement, `SILSA` moves from regions containing points with large residual norms to regions containing points with low residual norms, leading to a fast convergence to an ε-approximate solution of the problem (1). Additionally, the algorithm employs a spectral derivative-free direction based on the efficient direction of Liu and Storey [32], along with an improved version of `BLSA` by Solodov and Svaiter [18]. Moreover, we establish the global convergence property of `SILSA` under mild conditions and derive complexity upper bounds on the number of iterations and the number of function evaluations required by `SILSA` to find an ε-approximate solution of (1).

1.2. Organization of the paper

The paper is structured as follows. Section 2 provides preliminaries. Section 3 introduces `SILSA`, which comprises a new subspace inertial technique in Subsection 3.1 and a new derivative-free direction in Subsection 3.2. Section 4 investigates theoretical results for `SILSA`, which include auxiliary results in Subsection 4.1, global convergence in Subsection 4.2, and complexity results in Subsection 4.3. In Section 5, we compare `SILSA` with `BLSA` using several known derivative-free directions. Conclusion is given in Section 6.

2. Preliminaries

A mapping $F : \mathbb{R}^n \to \mathbb{R}^n$ is said to be *monotone* if the condition

$$(F(x) - F(y))^T(x - y) \geq 0, \quad \forall x, y \in \mathbb{R}^n \tag{2}$$

holds in a Euclidean space $\mathbb{R}^n$.

In this paper, we assume the following assumptions:

(A1) The function $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable and Lipschitz continuous with the Lipschitz constant $L > 0$.
(A2) F is monotone, i.e., the condition (2) holds.
(A3) The solution set $\mathbf{X}^*$ of the system (1) is nonempty.

In the following subsections, we review the important concepts (basic line search, least and most promising points, inertial point, and complexity bound) that we are using during our study.

2.1. Basic line search algorithm (`BLSA`)

In this subsection, we introduce the concept of `BLSA`, which generates a sequence of iterates, denoted as $\{y_k\}_{k\geq 0}$, to find an ε-approximate solution of (1). It enforces the condition that y_k must satisfy the line search condition

$$-F(y_k + \alpha_k d_k)^T d_k \geq \sigma\alpha_k \|F(y_k + \alpha_k d_k)\| \|d_k\|^2. \tag{3}$$

After the direction d_k satisfies the descent condition

$$F(y_k)^T d_k \le -c\|F(y_k)\|^2, \quad \text{with } 0 < c < 1 \tag{4}$$

and the step size α_k is found by satisfying the line search condition (3), `BLSA` accepts the new point

$$y_{k+1} := y_k + \alpha_k d_k, \quad \forall k \ge 0. \tag{5}$$

2.2. Least and most promising points

In this subsection, we define two important concepts, which are needed to clarify our algorithm: the *least promising point* (`LP` point), defined as the point with the highest residual norm among the evaluated points, and the *most promising point* (`MP` point), defined as the point with the lowest residual norm among the evaluated points. These definitions are motivated by the fact that there is no guarantee of a residual norm reduction in each step of `BLSA`.

2.3. Inertial step

In this subsection, the traditional inertial point

$$v_k = y_k + e_k(y_k - y_{k-1}) \tag{6}$$

is defined, where $y_{k-1}, y_k \in \mathbb{R}^n$ are the two distinct points generated by `BLSA` and $e_k \in [0, 1)$ is called *extrapolation step size*, which can be updated in various ways [29–31,39–43,45], one of which is

$$e_k = \min\left\{e_{\max}, k^{-2}\|y_k - y_{k-1}\|^{-2}\right\}, \tag{7}$$

where $0 < e_{\max} \le 1$ is a tuning parameter. This choice guarantees the global convergence for `BLSA` in combination with the initial step (7), e.g., see [29, Lemma 4.5].

Due to existing of no guarantee of producing y_{k-1} and y_k as `MP` points by applying `BLSA`, it may decrease the effectiveness of inertial point. However, employing a subspace inertial point based on the previous `MP` points, gives us this chance to generate a new `MP` point or a point close to two previous `MP` points. In this way, the new generated point would be far from the previous `LP` points.

2.4. Complexity bound

In this subsection, we define the complexity bound, i.e., the maximum number of iterations and the maximum number of function evaluations required to find an ε-

approximate solution x_ε of the problem (1) that satisfies the theoretical criterion

$$\|F(x_\varepsilon)\| \le \min(\varepsilon, \|F(x_0)\|). \tag{8}$$

Let us define $f(x) := \frac{1}{2}\|F(x)\|^2$ and its true gradient by $g(x) := J(x)^T F(x)$ of F at x, where $J(x)$ denotes the true Jacobian of F at x. Using the linear approximation of $F(x+d) \approx F(x) + J(x)^T d$, we have

$$f(x+d) \approx q(d) := f(x) + g(x)^T d + \frac{1}{2}\|J(x)d\|^2,$$

where the function $q(d)$ is a convex function. Assumptions (A1)–(A2) implies that for every $x, d \in \mathbb{R}^n$, we have

$$f(x+d) - f(x) = g(x)^T d + \frac{1}{2}\gamma^2\|d\|^2, \tag{9}$$

where γ depends on x and d and also satisfies

$$|\gamma| \le L \quad \text{(general case)}, \quad 0 \le \gamma \le L \quad \text{(convex case)}. \tag{10}$$

The following result is a variant of [47, Proposition 2], which is a crucial component to obtain the complexity bound for our algorithm. This result is independent of a particular derivative-free line search. Here we use the basic line search algorithm, which is different from the line search algorithm of [47].

Proposition 2.1. *Consider $x, d \in \mathbb{R}^n$ and $\Delta_f \ge 0$, where Δ_f is a threshold on f. Then, we show that at least one of the following conditions is satisfied:*

(i) $f(x+d) < f(x) - \Delta_f$,
(ii) $f(x+d) > f(x) + \Delta_f$ *and* $f(x-d) < f(x) - \Delta_f$,
(iii) $|g(x)^T d| \le \Delta_f + \frac{1}{2}L^2\|d\|^2$.

Proof. We assume that (iii) does not satisfy. Hence, we have

$$|g(x)^T d| > \Delta_f + \frac{1}{2}L^2\|d\|^2.$$

Although the condition (4) holds, we cannot guarantee $g(x)^T d < 0$ because the true matrix $J(x)$ at x is not available in $g(x) = J(x)^T F(x)$. Hence, we consider the proof in the following two cases:
Case 1. If $g(x)^T d \le 0$, from (9) and (10), we have

$$f(x+d) - f(x) \le g(x)^T d + \frac{1}{2}L^2\|d\|^2 = -|g(x)^T d| + \frac{1}{2}L^2\|d\|^2 < -\Delta_f; \tag{11}$$

hence (i) holds.
CASE 2. If $g(x)^T d \geq 0$, from (9) and (10), we have

$$f(x-d) - f(x) \leq g(x)^T(-d) + \frac{1}{2}L^2\|d\|^2 = -|g(x)^T d| + \frac{1}{2}L^2\|d\|^2 < -\Delta_f; \qquad (12)$$

hence the second inequality of (ii) holds. The first inequality of (ii)

$$f(x+d) - f(x) \geq g(x)^T d - \frac{1}{2}L^2\|d\|^2 > \Delta_f$$

is obtained. □

In exact precision arithmetic, the aim is to obtain an exact solution of the problem (1). However, in the presence of finite precision arithmetic, the algorithm may get stuck before finding an approximate solution of (1), especially in nearly flat areas of the search space. For a finite termination, the theoretical criterion (8) is used to find an ε-approximate solution of (1).

2.5. Existing derivative-free directions

We here discuss several conjugate gradient (CG) type directions and their derivative-free variants.

Let us begin with a well-known CG method that aims to minimize an unconstrained smooth function $f : \mathbb{R}^n \to \mathbb{R}$, using the iterative formula:

$$x_{k+1} = x_k + \alpha_k d_k, \quad \forall k \geq 0, \qquad (13)$$

where α_k is a step size determined by a line search procedure. The search direction

$$d_0 := -g(x_0), \quad d_k := -g(x_k) + \beta_k d_{k-1}, \quad \forall k \geq 1 \qquad (14)$$

is computed, where $g(x_k)$ is the true gradient of $f(x)$ at x_k and $\beta_k \in \mathbb{R}$ is the CG parameter.

Some classical famous formulas of the CG parameter are

- **(i)** $\beta_k^{PRP} := \dfrac{g(x_k)^T v_{k-1}}{\|g(x_{k-1})\|^2}$ of Polak–Ribere–Polyak (PRP) [36,37], where $v_k := g(x_k) - g(x_{k-1})$;
- **(ii)** $\beta_k^{FR} := \dfrac{\|g(x_k)\|^2}{\|g(x_{k-1})\|^2}$ of Fletcher–Reeves (FR) [26];
- **(iii)** $\beta_k^{LS} := \dfrac{g(x_k)^T v_{k-1}}{-d_{k-1}^T g(x_{k-1})}$ of Liu–Storey (LS) [32];
- **(iv)** $\beta_k^{DY} := \dfrac{\|g(x_k)\|^2}{d_{k-1}^T v_{k-1}}$ of Dai–Yuan (DY) [25].

(v) $\beta_k^{DL} := \frac{g(x_k)^T v_{k-1}}{d_{k-1}^T v_{k-1}} - t\frac{g(x_k)^T s_{k-1}}{d_{k-1}^T v_{k-1}}$ of Dai–Liao (DL) [48], where $t \geq 0$ and $s_k := x_k - x_{k-1}$.

For the other CG type directions; see the survey [28].

To identify an ε-approximate solution of (1), BLSA generates the sequence $\{x_k\}_{k\geq 0}$ given by

$$x_{k+1} = x_k - \lambda_k F(z_k), \quad \lambda_k = \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2} \tag{15}$$

with the trial point $z_k = x_k + \alpha_k d_k$. As a cheap and useful choice for d_k, based on CG directions this paper focuses on the following three derivative-free search directions:

(a) Motivated by the PRP method, the derivative-free direction

$$d_0 = -F(x_0),\ d_k = -F(x_k) + \beta_k d_{k-1}$$

was proposed in [24], where $\beta_k = \frac{F(x_k)^T y_{k-1}}{\|F(x_{k-1})\|}$ and $y_k = F(x_k) - F(x_{k-1})$.

(b) Inspired by the FR method, the derivative-free direction

$$d_0 = -F(x_0),\ d_k = -F(x_k) + \beta_k^{FR} v_{k-1} - \theta_k F(x_k)$$

was proposed in [35] with $\beta_k^{FR} = \frac{\|F(x_k)\|^2}{\|F(x_{k-1})\|^2}$ and the three different choices

$$\theta_k^{(1)} = \frac{F(x_k)^T v_{k-1}}{\|F(x_{k-1})\|^2},\ \theta_k^{(2)} = \frac{\|F(x_k)\|^2\|v_{k-1}\|^2}{\|F(x_{k-1})\|^4},\ \theta_k^{(3)} = \theta_k^{(1)} + (\beta_k^{FR})^2,$$

where $v_k = z_k - x_k$.

(c) Motivated by the LS method, the derivative-free direction

$$d_0 = -F(x_0),\ d_k = -F(x_k) + \beta_k^{ELS} d_{k-1}$$

was proposed in [49], where $\beta_k^{ELS} = \frac{F(x_k)^T y_{k-1}}{F(x_{k-1})^T d_{k-1}} - t\frac{\|y_{k-1}\|^2 F(x_k)^T d_{k-1}}{(F(x_{k-1})^T d_{k-1})^2}$ and $t \geq \frac{1}{4}$.

These methods are particularly well-suited for tackling large-scale non-smooth problems, since they utilize only function values and require minimal memory. Furthermore, the stability of the search directions is independent of the type of line search employed. It has been demonstrated that the sequence $\{x_k\}_{k\geq 0}$ generated by these methods globally converges to the solution of (1), provided that the underlying mapping F is monotone and L-Lipschitz continuous [18]. In Section 5, we present and evaluate several derivative-free directions that are based on the well-known CG directions and compare them with our proposed method.

3. Modified derivative-free algorithm

As mentioned in the introduction, several iterative inertial methods have been proposed in [29–31] for obtaining an ε-approximate solution of nonlinear monotone equations (1) in Euclidean space. The authors established that the sequences generated by their methods converge globally to the solution of the problem under mild conditions. Their primary contribution is in achieving an ε-approximate solution to (1) at a faster rate.

As discussed in Section 2, the concepts of MP and LP points were defined. Since BLSA cannot guarantee that the two points used to construct the inertial method are the previous MP points, the traditional inertial point has a low probability of being an MP point. To address this issue, the inertial point must ideally be constructed based on the previous MP points, or at the very least, a point in close proximity to the previous MP points. Hence, SILSA reduces the oscillation intensity of the residual norm by moving from regions containing LP points to regions containing MP points.

3.1. Novel subspace inertial method

In this section, we present a novel subspace inertial method that chooses the ingredients of the subspace from the previous MP points, thereby accelerating convergence to an ε-approximate solution of (1).

Let $\{x_k\}_{k\geq 0}$ be the sequence generated by our method. At the kth iteration of SILSA, we save the points generated by SILSA as the columns of the matrix $X_{n\times m}$ and their residual norms as the components of the vector

$$\mathtt{NF}^k_{1\times m} := (\|F(X^k_{:1})\|, \ldots, \|F(X^k_{:m})\|)$$

Here the jth column of X^k is denoted by $X^k_{:j}$ and m is the subspace dimension. We now introduce our novel subspace inertial point

$$w_k := x_k + e_k \sum_{j=1}^{m-1} \lambda_j (X^k_{:j+1} - X^k_{:j}), \tag{16}$$

where the extrapolation step size

$$e_k := \min\left\{ e_{\max}, k^{-2} \Big\| \sum_{j=1}^{m-1} \lambda_j (X^k_{:j+1} - X^k_{:j}) \Big\|^{-2} \right\} \tag{17}$$

is computed, so that the condition

$$\sum_{k=1}^{\infty} \Big\| \sum_{j=1}^{m-1} \lambda_j (X^k_{:j+1} - X^k_{:j}) \Big\| < \infty \tag{18}$$

is satisfied. Here $0 < e_{\max} \leq 1$ is the maximum value for e_k and $0 < \lambda_j < 1$ for $j = 1, \cdots, m-1$ are called *subspace step sizes*, satisfying

$$\sum_{j=1}^{m-1} \lambda_j = 1. \tag{19}$$

These subspace step sizes will be chosen in Section 5 such that (19) holds. The condition (18) results in $\alpha_k \|d_k\| \to 0$ (see Lemma 4.1, below), where α_k satisfies the line search condition (3). This result guarantees the global convergence for `SILSA` (see Theorem 4.3, below).

To update the matrix X^k and the vector $\mathtt{NF}^k$ at the kth iteration of `SILSA`, we replace the `LP` point with a new `MP` point (if any). Therefore, we use the new subspace inertial point (16), which involves a weighted average of the m previous `MP` points. This increases the chance of discovering an `MP` point by `SILSA`, i.e., x_b with $b = \underset{i=1:m}{\operatorname{argmin}}(\mathtt{NF}^k_i)$.

It should be noted that when $m = 2$, the traditional inertial point (6) differs from our subspace inertial point (16), because (16) replaces the `LP` point among the previous `MP` points with a new point. This means that (16) is not restricted to using only the two previous points x_{k-1} and x_k, while the traditional inertial point (6) is limited to exactly these two points. If these two points are `LP` points, then `BLSA` using (6) cannot move quickly from regions with `LP` points to regions with `MP` points, while `SILSA` using (16) has a good chance of having more `MP` points in the subspace inertial, since the subspace inertial is updated by removing `LP` points as described above.

3.2. Novel derivative-free direction

Motivated by the `CG` method given in [32], we introduce the spectral derivative-free direction

$$d_k := \begin{cases} -\theta_0 F(w_k) & \text{if } k = 0, \\ -\theta_k F(w_k) + \beta_k^{DFLS} d_{k-1} & \text{if } k \geq 1, \end{cases} \tag{20}$$

with the scalar parameter

$$\beta_k^{DFLS} := -\frac{F(w_k)^T y_{k-1}}{F(w_{k-1})^T d_{k-1}} \tag{21}$$

and the spectral parameter

$$\theta_k := \begin{cases} c & \text{if } k = 0, \\ c + \beta_k^{DFLS} \dfrac{F(w_k)^T d_{k-1}}{\|F(w_k)\|^2} & \text{if } k \geq 1, \end{cases} \tag{22}$$

where $0 < c < 1$ is given and w_k comes from (16).

Lemma 3.1. *The search direction d_k computed by* (20) *for $k \geq 0$ satisfies the descent condition* (4)*.*

Proof. For the tuning parameter $0 < c < 1$, $d_0^T F(w_0) = -c\|F(w_0)\|^2$ and

$$\begin{aligned} d_k^T F(w_k) &= \Big(-\theta_k F(w_k) + \beta_k^{DFLS} d_{k-1} \Big)^T F(w_k) \\ &= -\Big(c + \beta_k^{DFLS} \frac{F(w_k)^T d_{k-1}}{\|F(w_k)\|^2} \Big) \|F(w_k)\|^2 + \beta_k^{DFLS} F(w_k)^T d_{k-1} \\ &= -c\|F(w_k)\|^2 \quad \text{for } k \geq 1; \end{aligned}$$

hence d_k satisfies (4) for all $k \geq 0$. □

3.3. Subspace inertial line search with `SILSA`

In this subsection, we introduce a detailed description of our subspace inertial derivative-free algorithm, which we call `SILSA`. This algorithm is designed to find an ε-approximate solution of (1) and is an improved version of `BLSA` that incorporates the subspace inertial method for faster convergence. In practice, the new subspace inertial method generates points that are, at worst, close to the previous `MP` points. Specifically, the new method replaces one of the previous `MP` points with the greatest residual norm. This substitution causes `SILSA` to move from regions with `LP` points to regions with `MP` points, and in practice quickly finds an approximate solution of (1).

Algorithm 1 `SILSA`, *subspace inertial line search algorithm for* (1)

Given above tuning parameters.
S0 (initialization)
choose $x_0 \in \mathbb{R}^n$ and $\lambda_j \in (0,1)$ for $j = 1, \cdots, m-1$;
Set $w_0 = x_0$, $d_0 = -F(w_0)$, $F(w_0) = F(x_0)$, $\mathtt{NF}^0_{:0} = \|F(x_0)\|$, $X^0_{:0} = x_0$, $e_0 = e_{\max}$, and $\delta_0 = \delta_{\max}$;
for $k = 0, 1, 2, \cdots$ **do** ▷ start of the main loop
  S1 (**performing the line search along** d_k)
  run $[z_k,\ F(z_k)] =$ `lineSearch`$(\delta_k,\ d_k,\ \sigma,\ r,\ w_k)$;
  if $\|F(z_k)\| < \varepsilon$ **then**, set $x_k = z_k$; **return**; ▷ the stopping test
  else
    S2 (**checking reduction in the residual norm**)
    run $[\delta_{k+1},\ e_{k+1}] =$ `checkDec`$(\overline{\gamma},\ \delta_{\max},\ e_{\max},\ \omega_d,\ \delta_k,\ e_k,\ F(w_k),\ F(z_k))$;
    S3 (**projecting** w_k **into Hyperplane** $H := \{w \in \mathbb{R}^n \mid F(z_k)^T(w - z_k) = 0\}$)
    run $[x_{k+1},\ F(x_{k+1})] =$ `projectPoint`$(z_k,\ F(z_k),\ w_k)$;
    if $\|F(x_{k+1})\| < \varepsilon$, **then**, **return**; **end if** ▷ the stopping test
    S4 (**update the subspace information**)
    run $[X^{k+1},\ \mathtt{NF}^{k+1}] =$ `updateSubspace`$(k,\ m,\ X^k,\ \mathtt{NF}^k,\ x_{k+1},\ F(x_{k+1}))$;
    S5 (**update the** $(k+1)$**th subspace inertial point**)
    compute w_{k+1} by (16);
    if $\|F(w_{k+1})\| < \varepsilon$, **then**, set $x_{k+1} = w_{k+1}$; **return end if** ▷ the stopping test
    if $\delta_{k+1} \le \delta_{\min}$, **then return**; **end if** ▷ the stopping test
    set $y_k = F(w_{k+1}) - F(w_k)$;
    S6 (**computing the** $(k+1)$**th search direction**)
    run $d_{k+1} =$ `searchDir`$(c,\ w_k,\ w_{k+1},\ y_k,\ d_k)$;
  end if
end for ▷ end of the main loop

The `SILSA` algorithm incorporates several tuning parameters: $\sigma > 0$ and $0 < \overline{\gamma} < 1$ (the line search parameters), $0 \leq \delta_{\min} < 1$ (the minimum threshold for δ_k), $0 < \delta_{\min} < \delta_{\max} \leq 1$ (the initial value for δ_k), $\omega_d > 1$ (the parameter for updating δ_k), $0 < c < 1$ (the parameter for computing d_k), $m \geq 2$ (the subspace dimension), $r \in (0, 1)$ (the parameter for reducing α_k), and $0 \leq e_{\max} < 1$ (the maximum value for e_k).

We now describe how `SILSA` algorithm works:

(S0) (**Initialization**) First, we choose an initial point $x_0 \in \mathbb{R}^n$. Next, we select the inertial weights $0 < \lambda_j < 1$, for $j = 1, \cdots, m-1$ such that the condition $\sum_{j=1}^{m-1} \lambda_j = 1$ is satisfied. We then choose the initial inertial point $w_0 = x_0$, and set the search direction to the negative residual vector at x_0. The initial parameter e_0 for adjusting the inertial point is set to the tuning parameter $0 < e_{\max} \leq 1$, while the initial step size $0 < \delta_0 < \infty$ is set to the tuning parameter $0 < \delta_{\max} \leq 1$.

(S1) (**Line search algorithm**) At the $k$the iteration, `SILSA` performs `lineSearch` (line 5) along the derivative-free direction d_k (d_k is computed by `searchDir` in line 16 for $k \geq 1$). Initially, `lineSearch` sets $j = 0$ and takes the initial step size

$$\alpha_{k,0} := \delta_k. \tag{23}$$

The other step sizes are then reduced by a given factor $0 < r < 1$ according to

$$\alpha_{k,j+1} := r\alpha_{k,j} \quad \text{for } j \geq 0, \tag{24}$$

and j is increased until the line search condition

$$-F(w_k + \alpha_{k,j} d_k)^T d_k \geq \sigma \alpha_{k,j} \|F(w_k + \alpha_{k,j} d_k)\| \|d_k\|^2 \tag{25}$$

is satisfied. Once this condition holds, we set $\alpha_k := \alpha_{k,j}$ and compute the accepted point

$$z_k := w_k + \alpha_k d_k \tag{26}$$

and its residual vector $F(z_k)$. If $\|F(z_k)\|$ is below a given threshold $\varepsilon > 0$, z_k is chosen as an approximate solution of (1), and `SILSA` terminates.

(S2) (**Checking reduction of the residual norm**) If $\|F(z_k)\| > \varepsilon$, then `checkDec` (line 8) checks whether or not the decrease condition

$$f(z_k) < f(w_k) - \overline{\gamma}\delta_k \tag{27}$$

holds, where $f(z_k) := \frac{1}{2}\|F(z_k)\|^2$ and $f(w_k) := \frac{1}{2}\|F(w_k)\|^2$. Accordingly, using the tuning parameter $\omega_d > 1$, it then either increases the step size δ_k of the decrease condition (27), i.e.,

$$\delta_{k+1} := \min(\omega_d \delta_k, \delta_{\max}) \tag{28}$$

or decreases it, i.e.,

$$\delta_{k+1} := \delta_k/\omega_d. \tag{29}$$

Moreover, the extrapolation step size e_k is computed by (17).

(S3) (**Projection of w_k into the hyperplane** $H := \{w \in \mathbb{R}^n \mid F(z_k)^T(w - z_k) = 0\}$) The point w_k is projected into H by `projectPoint` (line 9) and then the new point

$$x_{k+1} := w_k - \mu_k F(z_k) \tag{30}$$

is computed with the step size $\mu_k := \dfrac{F(z_k)^T(w_k - z_k)}{\|F(z_k)\|^2}$, and its residual norm $\|F(x_{k+1})\|$. If $\|F(x_{k+1})\|$ is below a given threshold $\varepsilon > 0$, x_{k+1} is chosen as an approximate solution of (1) and `SILSA` ends.

(S4) (**Update of the subspace of the old points**) If the norm of the residual vector at the next point x_{k+1}, denoted by $\|F(x_{k+1})\|$, is greater than a given threshold $\varepsilon > 0$ then `updateSubspace` updates the information of the subspace inertial point, which includes the matrix X^k and the vector $\mathtt{NF}^k$ (defined in Section 3.1) in line 11. If the current iteration $k+1 \geq m$ holds, then a new point is added to the subspace. Specifically, we find the index i_w of the previous `MP` point with the largest residual norm and replace it with the new point x_{k+1}, i.e.,

$$i_w := \underset{i=1:m}{\operatorname{argmax}}\{\mathtt{NF}^{k+1}_{:i}\}, \quad X^{k+1}_{:i_w} = x_{k+1}, \quad \mathtt{NF}^{k+1}_{:i_w} = \|F(x_{k+1})\|.$$

On the other hand, if $k+1 < m$, we set $X^{k+1}_{:k+1} = x_{k+1}$ and $\mathtt{NF}^{k+1}_{:k+1} = \|F(x_{k+1})\|$. By adding more points with the lower residual norm to the subspace, this strategy increases the chance of finding an ε-approximate solution of (1).

(S5) (**Computation of a new inertial point**) After calculating the new subspace inertial point w_{k+1} by (16) and determining its residual norm, if the value of $\|F(w_{k+1})\|$ is found to be less than a certain specified threshold $\varepsilon > 0$, then w_{k+1} is considered as an approximate solution for (1), and `SILSA` terminates. Alternatively, if δ_k is found to be less than a given threshold $\delta_{\min} > 0$, then w_k is considered as an approximate solution for (1), and `SILSA` terminates.

(S6) (**Computation of the search direction**) If the value of $\|F(w_{k+1})\|$ is greater than ε, the difference between the residual at the inertial points w_{k+1} and w_k, denoted by $y_k := F(w_{k+1}) - F(w_k)$, is computed. Then the derivative-free direction d_{k+1} is computed by (20), whose step sizes β_k^{DFLS} and θ_{k+1} have been computed by (21) and (22), which depend on the values of y_k, d_k, w_k, and w_{k+1}.
The tuning parameters $\delta_{\max}$, $\delta_{\min}$, ω_d, and r appear in the complexity bound on the number of function evaluations, which is discussed in Section 4.3. It is worth noting that, based on the update rules for δ_k in (28) and (29), δ_k is always less than or equal to $\delta_{\max}$ for all values of k.

4. Convergence analysis and complexity

In this section, we first present several auxiliary results, that are necessary to establish global convergence and the complexity bounds, and then the main theoretical results.

4.1. Some auxiliary results

The following results have a key role in proving global convergence.

Lemma 4.1. *Let $\{w_k\}_{k\geq 0}$ and $\{x_k\}_{k\geq 0}$ be the two sequences generated by* `SILSA`, *assume that the assumptions (A1)–(A3) hold, and define $\Delta^k := \max_{j=1:m-1} |X^k_{:j+1} - X^k_{:j}|$. Then:*

(i) *The inequality*

$$\|x_{k+1} - x^*\|^2 \leq \|w_k - x^*\|^2 - \sigma^2\|w_k - z_k\|^4 \tag{31}$$

holds.

(ii) *The sequence $\{x_k\}_{k\geq 0}$ is bounded,*

$$\sum_{k=0}^{\infty} \|w_k - z_k\|^4 < \infty$$

and so

$$\lim_{k\to\infty} \|w_k - z_k\| = \lim_{k\to\infty} \alpha_k\|d_k\| = 0. \tag{32}$$

(iii) *Δ^k is finite and*

$$\sum_{j=1}^{m-1} \lambda_j\|X^k_{:j+1} - X^k_{:j}\| \leq (m-1)\Delta^k < \infty. \tag{33}$$

(iv) *There exists a positive constant $\overline{\Gamma}_w$ such that*

$$\|F(w_k)\| \leq \overline{\Gamma}_w, \quad \forall k \geq 0. \tag{34}$$

(v) *If the direction d_k is bounded, i.e., $\|d_k\| \leq \Gamma_d$ for a positive constant Γ_d, then there exists a positive constant Γ_z such that*

$$\|F(z_k)\| \leq \Gamma_z, \quad \forall k \geq 0. \tag{35}$$

Here, z_k is from (26).

(vi) *If* $F(w_k) \neq 0$ *for any* k*, then* $\|d_k\| \geq c\|F(w_k)\|$ *and* $d_k \neq 0$*.*

Proof. Let x^* be the solution of equation (1) and $x^* \in \mathbf{X}^* \subset \mathbb{R}^n$ be the set of feasible solutions.

(i-ii) The proof can be done like the proof of [29, Lemma 4.5], but with the difference that the extrapolation step size (17) and the condition (18) are used instead of the traditional extrapolation step size (6) and the condition (7), respectively.

(iii) There exist two positive integers $k > k'$ such that

$$\Delta^k = \max_{j=1:m-1} \|X^k_{:j+1} - X^k_{:j}\| = \|x_k - x_{k'}\| \leq \|x_k\| + \|x_{k'}\| < \infty$$

due to (ii). Then, the condition (33) holds, because $0 < m < \infty$ and $\sum_{j=1}^{m-1} \lambda_j = 1$.

(iv) From (ii), since $\{x_k\}_{k\geq 0}$ is bounded, there exists a positive constant Γ_0 such that $\|x_k\| \leq \Gamma_0$ for all $k \geq 0$. From (17) and since $0 < e_{\max} \leq 1$, we obtain $e_k \leq e_{\max} \leq 1$ for all k. Hence, by (16) and (33), the sequence $\{w_k\}_{k\geq 0}$ is bounded above, i.e.,

$$\begin{aligned}
\|w_k\| &= \Big\|x_k + e_k \sum_{j=1}^{m-1} \lambda_j (X_{:j+1} - X_{:j})\Big\| \\
&\leq \|x_k\| + e_k \sum_{j=1}^{m-1} \lambda_j \|X_{:j+1} - X_{:j}\| \\
&\leq \Gamma_0 + 2\Gamma_0(m-1) = (2m-1)\Gamma_0;
\end{aligned}$$

hence F is continuous from (A1) and therefore (34) is valid.

(v) From (23), (28), and since $0 < \delta_{\max} \leq 1$, we obtain $\alpha_k \leq \delta_{\max} \leq 1$ for all k. Hence, (26) and (iv) result in

$$\|z_k\| = \|w_k + \alpha_k d_k\| \leq \|w_k\| + \|d_k\| \leq (2m-1)\Gamma_0 + \Gamma_d.$$

Therefore the continuity of F implies that (35) is valid.

(vi) From (4) and the fact that $F(w_k) \neq 0$ for all k, we have

$$-\|d_k\| \leq \frac{F(w_k)^T d_k}{\|F(w_k)\|} \leq -c\|F(w_k)\| < 0,$$

resulting in $d_k \neq 0$.

□

In the following result, under the assumption that the residual norms are bounded below, upper and lower bounds for search directions and step sizes are restricted.

Proposition 4.2. *Let $\{x_k\}_{k\geq 0}$ and $\{w_k\}_{k\geq 0}$ be the two sequences generated by* ***SILSA*** *and assume that the assumptions (A1)–(A3) hold. If there is a positive constant $\underline{\Gamma}_w$ such that $\|F(w_k)\| \geq \underline{\Gamma}_w$ for all k, then the following two statements are valid:*

(i) *The search directions d_k are bounded above, i.e.,*

$$0 < \|d_k\| \leq \Gamma_d := c\overline{\Gamma}_w + \frac{4\overline{\Gamma}_w^2}{c\underline{\Gamma}_w^2\sigma} \quad \forall k, \tag{36}$$

where σ is from the line search condition (25), $\overline{\Gamma}_w$ is from (34), and c is from (4).

(ii) *If the line search condition (25) cannot be satisfied, then line search step sizes α_k are bounded, i.e.,*

$$\underline{\alpha} := \frac{rc\overline{\Gamma}_w^2}{(L+\sigma\Gamma_z)\Gamma_d^2} \leq \alpha_k \leq \delta_{\max} \leq 1. \tag{37}$$

where r is from the line search condition (25), L is from (A2), and $\delta_{\max}$ is a tuning parameter.

Proof. By the Cauchy-Schwartz inequality and (25), we have

$$\|F(z_k)\|\|w_k - z_k\| \geq F(z_k)^T(w_k - z_k) \geq \sigma\alpha_{k,j}^2\|F(z_k)\|\|d_k\|^2 = \sigma\|F(z_k)\|\|w_k - z_k\|^2.$$

Thus, we obtain

$$\sigma\|w_k - z_k\| \leq 1, \quad \forall k \geq 0. \tag{38}$$

It follows from Lemma 4.1(iv), (4), (20), (34), and (38) that,

$$\begin{aligned} |\theta_k| &= \left| c - \frac{(F(w_k)^T y_{k-1})(F(w_k)^T d_{k-1})}{F(w_{k-1})^T d_{k-1}\|F(w_k)\|^2} \right| \\ &\leq c + \frac{\|F(w_k)\|\|y_{k-1}\|\|F(w_k)\|\|d_{k-1}\|}{|F(w_{k-1})^T d_{k-1}|\|F(w_k)\|^2} \end{aligned}$$

and $|\beta_k^{DFLS}| = \left|\dfrac{F(w_k)^T y_{k-1}}{F(w_{k-1})^T d_{k-1}}\right| \leq \dfrac{\|F(w_k)\|\|y_{k-1}\|}{|F(w_{k-1})^T d_{k-1}|}$, resulting in

$$\begin{aligned} 0 < \|d_k\| &= \left\| -\theta_k F(w_k) + \beta_k^{DFLS} d_{k-1} \right\| \leq c\|F(w_k)\| + 2\frac{\|F(w_k)\|\|y_{k-1}\|}{|F(w_{k-1})^T d_{k-1}|}\|d_{k-1}\| \\ &\leq c\|F(w_k)\| + 4\frac{\overline{\Gamma}_w^2}{c\|F(w_{k-1})\|^2}\|w_{k-1} - z_{k-1}\| \leq \Gamma_d = c\overline{\Gamma}_w + \frac{4\overline{\Gamma}_w^2}{c\underline{\Gamma}_w^2\sigma}. \end{aligned}$$

(ii) From Lemma 4.1(vi), $d_k \neq 0$. We show that `lineSearch` always terminates in a finite number of steps. From (23), we have $\alpha_{k,0} = \delta_k$. Then according to the role of updating $\alpha_{k,j}$ in (24) we have $\alpha_{k,j} = r^{-j}\delta_k$. If the condition (25) with $\alpha_{k,j} = r^{-j}\delta_k$ does not hold, i.e.,

$$-F(w_k + r^{-j}\delta_k d_k)^T d_k \quad < \quad \sigma r^{-j}\delta_k \|F(w_k + r^{-j}\delta_k d_k)\| \|d_k\|^2, \tag{39}$$

as j goes to infinity, we have $-F(w_k)^T d_k < 0$, which contradicts (4), since $\delta_k \leq \delta_{\max}$, $\|d_k\| \leq \Gamma_d$ (from (i)), and $\|F(w_k + r^{-j}\delta_k d_k)\| = \|F(w_k)\| \leq \Gamma_z$. Hence `lineSearch` terminates finitely; there is a positive integer j' such that

$$\alpha_k = \alpha_{k,j'} = r^{-j'}\delta_k,$$

satisfying (25). As long as (39) holds, applying (22) into (4) and using (A2), we have

$$\begin{aligned} c\|F(w_k)\|^2 &= -F(w_k)^T d_k \\ &= (F(w_k + r^{-(j'-1)}\delta_k, d_k)^T d_k - F(w_k)^T d_k) - F(w_k + r^{-(j'-1)}\delta_k d_k)^T d_k \\ &\leq Lr^{-(j'-1)}\delta_k \|d_k\|^2 + \sigma r^{-(j'-1)}\delta_k \|F(w_k + r^{-(j'-1)}\delta_k d_k)\| \|d_k\|^2, \end{aligned}$$

leading to

$$\delta_{\max} \geq \alpha_k = r^{-j'}\delta_k \geq \frac{rc\|F(w_k)\|^2}{(L + \sigma\|F(w_k + r^{-(j'-1)}\delta_k d_k)\|)\|d_k\|^2} \geq \underline{\alpha} = \frac{rc\overline{\Gamma}_w^2}{(L + \sigma\Gamma_z)\Gamma_d^2}$$

from Lemma 1(iv). □

4.2. Convergence analysis

The following result is the main global convergence of `SILSA`. The variants of this result can be found in [29–31], but with the different inertial point.

Theorem 4.3. *Suppose that (A1)–(A3) hold and $\{w_k\}_{k\geq 0}$, $\{z_k\}_{k\geq 0}$, $\{x_k\}_{k\geq 0}$ are the three sequences generated by* `SILSA`*. Let $\delta_{\min} = 0$. Then, at least one of*

$$\lim_{k\to\infty} \|F(z_k)\| = 0, \quad \lim_{k\to\infty} \|F(w_k)\| = 0, \quad \lim_{k\to\infty} \|F(x_k)\| = 0 \tag{40}$$

holds. Moreover, the sequences $\{x_k\}_{k\geq 0}$ and $\{w_k\}_{k\geq 0}$ converge to a solution of (1).

Proof. If $\|F(w_k)\| = 0$, then `SILSA` terminates and accepts w_k as a solution of (1) (see line 13 of `SILSA`). Otherwise, `SILSA` performs and therefore there is a positive constant $\underline{\Gamma}_w$ such that $\|F(w_k)\| > \underline{\Gamma}_w$, for all k, holds. Hence Proposition 4.2(i) results in that there is a positive constant Γ_d such that $0 < \|d_k\| \leq \Gamma_d$ for all k ($d_k \neq 0$ from Lemma 4.1(vi)). Moreover, if $\|F(w_k + r^{-j'}\delta_k d_k)\| = 0$, then `SILSA` terminates and accepts $z_k = w_k + r^{-j'}\delta_k d_k$ as a solution of (1) (here j' is a positive integer value such that $\alpha_k = \alpha_{k,j'} = r^{-j'}\delta_k$ satisfying (25)). Otherwise, from Lemma 4.1(iv), there is a positive constant Γ_z such that

$$0 < \|F(z_k)\| \leq \Gamma_z \quad \text{with } z_k = w_k + r^{-j'}\delta_k d_k.$$

As such, the assumptions of Proposition 4.2(ii) are verified and this proposition results in that there is a positive constant $\underline{\alpha}$ such that $\alpha_k \geq \underline{\alpha}$ for all k. Hence, from Lemma 4.1(vi), we obtain $\alpha_k\|d_k\| > \underline{\alpha}c\|F(w_k)\| > \underline{\alpha}c\underline{\Gamma}_w > 0$ for all k, which contradicts (32). Therefore, $\|F(w_k)\| = 0$ is obtained. From (32), $\lim_{k\to\infty} \|x_k - w_k\| = 0$ and the continuity of F results in

$$\lim_{k\to\infty} \|F(x_k)\| - \lim_{k\to\infty} \|F(w_k)\| \leq \lim_{k\to\infty} \|F(x_k) - F(w_k)\| \leq L \lim_{k\to\infty} \|x_k - w_k\| = 0,$$

which consequently implies

$$\lim_{k\to\infty} \|F(x_k)\| = 0. \tag{41}$$

From the continuity of F, the boundedness of $\{x_k\}_{k\geq 0}$ and (41), it implies that the sequence $\{x_k\}_{k\geq 0}$, generated by `SILSA`, has an accumulation point x^* such that $F(x^*) = 0$. On the other hand, the sequence $\{x_k - x^*\}_{k\geq 0}$ is convergent by Lemma 4.1, which means that the sequence $\{x_k\}_{k\geq 0}$ globally converges to the solution x^* of (1). □

4.3. Complexity results

This section concerns an investigation of the complexity of `SILSA`. Firstly, we establish an upper limit on the number of function evaluations required by `lineSearch`. Following that, we determine an upper bound on the number of iterations needed for `SILSA` to converge, with or without a reduction in residual norms. Consequently, we derive an upper threshold for the total number of function evaluations necessary to successfully find an approximate solution of (1) by `SILSA`.

Proposition 4.4. *Let $\{x_k\}_{k\geq 0}$ be the sequence generated by `SILSA` and assumes that (A1)–(A3) hold. Assuming that the initial step size is bounded by $0 < \delta_{\max} \leq 1$ and that the parameter $0 < r < 1$ is utilized to decrease the step size in `lineSearch`, the number `nf` of function evaluations used by `lineSearch` (line 5 of `SILSA`) can be constrained by*

$$\left\lceil \log_{r^{-1}} \frac{\delta_{\max}}{\underline{\alpha}} \right\rceil,$$

where $\underline{\alpha}$ is a positive constant derived from Proposition 4.2(ii).

Proof. From Proposition 4.2(ii), we have $\alpha_{k,nf} \geq \underline{\alpha}$ for all k. By (23) and (24), we have

$$r^{nf}\delta_{\max} \geq \alpha_{k,nf} = r^{nf}\delta_k \geq \underline{\alpha},$$

leading to

$$nf \leq \left\lceil \log_{r^{-1}} \frac{\delta_{\max}}{\underline{\alpha}} \right\rceil$$

since $r \in (0,1)$. □

By means of (27), we define the index set I_k as the set of all iterations k such that $f(z_k) < f(w_k) - \overline{\gamma}\delta_k$. This set encompasses iterations exhibiting at most $\overline{\gamma}\delta_k$ reductions in the residual norms, while the index set I_k^c is the complement of I_k.

Theorem 4.5. *Let $\{w_k\}_{k\geq 0}$, $\{z_k\}_{k\geq 0}$, $\{x_k\}_{k\geq 0}$ be the sequences generated by* `SILSA`*, let x_ε be an ε-approximate solution of* (1) *found by* `SILSA`*, and assume that (A1)–(A3) hold. Moreover, the tuning parameters $0 < \overline{\gamma} < 1$ (parameter for line search), $0 < \delta_{\min} < \delta_{\max} \leq 1$ (initial and minimal threshold for the step size δ_k), $0 < r < 1$ (parameter for reducing step size by* `lineSearch`*), $1 \leq \omega_d < \infty$ (parameter for updating the step size δ_k) are given. Then the following statements are valid:*

(i) *The number of iterations of* `SILSA` *with reductions in the residual norms is bounded by*

$$|I_k| \leq \frac{f(x_0) - f(x_\varepsilon)}{\overline{\gamma}\delta_{\min}}. \tag{42}$$

(ii) *The number of iterations of* `SILSA` *without reductions in the residual norms is bounded by*

$$|I_k^c| \leq \log_{\omega_d} \frac{\delta_{\max}}{\delta_{\min}}. \tag{43}$$

(iii) *The number of iterations of* `SILSA` *is bounded by*

$$N = |I_k| + |I_k^c| \leq \frac{f(x_0) - f(x_\varepsilon)}{\overline{\gamma}\delta_{\min}} + \log_{\omega_d} \frac{\delta_{\max}}{\delta_{\min}} = \mathcal{O}(\delta_{\min}^{-1}).$$

(iv) *The number of function evaluations of* `SILSA` *is bounded by*

$$nf_{total} \leq N \left\lceil \log_{r^{-1}} \frac{\delta_{\max}}{\underline{\alpha}} \right\rceil.$$

(v) *If there is a positive constant M_0 such that*

$$\|g(w_k)\| \leq M_0 \|F(w_k)\| \tag{44}$$

for all k and `SILSA` *has no iteration with a reduction in the residual norm,* `SILSA` *finds at least a point w_k with at most $\mathcal{O}(\varepsilon^{-2})$ function evaluations satisfying $\|F(w_k)\| = \mathcal{O}(\varepsilon)$. Here $g(w_k) = J(w_k)^T F(w_k)$ comes from Section 2.*

Proof. **(i)** The index set I_k is defined as $\{k \mid f(z_k) < f(w_k) - \overline{\gamma}\delta_k\}$. By the definition of I_k, we have:

$$f(x_0) - f(x_\varepsilon) \geq \sum_{j \in I_k} (f(w_j) - f(z_j)) \geq \overline{\gamma} \sum_{j \in I_k} \delta_k \geq \overline{\gamma} \sum_{j \in I_k} \delta_{\min} = |I_k| \overline{\gamma} \delta_{\min},$$

which yields the result in (42).

(ii) The set I_k^c is defined as $\{1, 2, \cdots, k\} \setminus I_k$. Updating $\delta_k = \delta_{k-1}/\omega_d$ guarantees that $\delta_{\min} \leq \delta_k \leq \delta_{\max}$, which leads to the derivation of (43).

(iii) Combining the results from (i) and (ii) yields the desired outcome.

(iv) The result is obtained from (iii) and Proposition 4.4.

(v) Consider any $j \in \mathbb{N} \cup \{0\}$ with $j < \infty$. During the execution of `lineSearch`, the trial points $w_k + \alpha_{k,j} d_k$ are generated, the last of which satisfies the line search condition (25) and is accepted as z_k. However, the condition (27) along $\pm d_k$ may not be satisfied. In the worst case, we assume that (27) is not satisfied. Then, by applying Proposition 2.1(iii), we have:

$$|g(w_k)^T(\alpha_k d_k)| \leq \overline{\gamma}\delta_k + \frac{L}{2}\|\alpha_k d_k\|^2.$$

We now consider the following two cases:

Case 1: If $\|F(w_k)\| \leq \varepsilon := \sqrt{\delta_{\min}}$, then $x_k = w_k$ is a solution of (1). Hence `SILSA` finds a point x_k whose residual norm is less than ε with at most $\mathcal{O}(\varepsilon^{-2})$ function evaluations.

Case 2: Assuming that $\|F(w_k)\| > \varepsilon$ for all k, we can apply Proposition 4.2(i) to obtain the condition (36), which ensures that $\|d_k\| \leq \Gamma_d$ for all k. Additionally, Lemma 4.1(iv) and Proposition 4.2(ii) guarantee that the condition (37) holds, i.e., $\alpha_k \geq \underline{\alpha}$ for all k. Consequently, after a finite number of iterations, `SILSA` terminates due to the role of updating δ_k in (29), which implies the existence of a positive integer k_0 such that $\alpha_k \leq \delta_k \leq \delta_{\min}$ for $k \geq k_0$. Considering the worst-case scenario where there is no reduction of the residual norm at z_k for all k (i.e., I_k is empty), we can use Proposition 2.1, (36), (37), and (44) to obtain

$$M_0|F(w_k)^T(\alpha_k d_k)| \leq |g(w_k)^T(\alpha_k d_k)| \leq \overline{\gamma}\delta_k + \frac{L}{2}\|\alpha_k d_k\|^2 \leq \overline{\gamma}\delta_{\min} + \frac{L}{2}\alpha_k^2\Gamma_d^2$$

for all $k \geq k_0$, i.e.,

$$c\|F(w_k)\|^2 = |F(w_k)^T d_k| \leq \frac{\overline{\gamma}\delta_{\min}}{M_0\alpha_k} + \frac{L}{2M_0}\Gamma_d^2\alpha_k \leq \Big(\frac{\overline{\gamma}}{M_0\underline{\alpha}} + \frac{L\Gamma_d^2}{2M_0}\Big)\delta_{\min}.$$

Combining the results of the two cases, we obtain $\|F(w_k)\| = \mathcal{O}(\varepsilon) = \mathcal{O}(\sqrt{\delta_{\min}})$, which completes the proof.

□

As stated in the introduction, our complexity bound is of the same order as the bounds obtained by Cartis et al. [50], Curtis et al. [51], Dodangeh and Vicente [52], Dodangeh et al. [53], Kimiaei and Neumaier [47], and Vicente [54] for other optimization methods.

5. Numerical experiments

In this section, we present a comparative analysis of our algorithm, `SILSA`, with 10 well-known algorithms (discussed below) on a set of 18 test problems with the dimensions

$$n \in \{10, 50, 300, 500, 1000, 5000\}.$$

This results in a total of 108 test functions. The Matlab codes of these 18 problems are available in the Section 7. To ensure that all test problems used are monotone in finite precision arithmetic, we randomly generated 10^6 distinct points x and y and verified that the condition

$$\frac{(x-y)^T(F(x)-F(y))}{|x-y|^T(|F(x)|+|F(y)|)+1} \leq -0.1$$

was fulfilled.

We compare `SILSA` with the following algorithms:

(1) `BLSA-DY`, `BLSA` with the derivative-free direction using the `CG` direction of Dai and Yuan [25].
(2) `BLSA-HZ`, `BLSA` with the derivative-free direction using the `CG` direction of Hager and Zhang [27].
(3) `BLSA-PR`, `BLSA` with the derivative-free direction using the `CG` direction of Polak-Ribere-Polyak `(PRP)` [36,37].
(4) `BLSA-FR`, `BLSA` with the derivative-free direction using the `CG` direction of Fletcher and Reeves `(FR)` [26].
(5) `BLSA-3PR`, `BLSA` with the derivative-free direction using the `CG` direction of Zhang et al. [38].
(6) `BLSA-3A`, `BLSA` with the derivative-free direction using the `CG` direction of Andrei [22].
(7) `BLSA-IM`, `BLSA` with the derivative-free direction of Ivanov et al. [55].
(8) `BLSA-AK`, `BLSA` with the derivative-free direction of Abubakar and Kumam [56].
(9) `BLSA-HD`, `BLSA` with the derivative-free direction of Huang et al. [57].
(10) `BLSA-SS`, `BLSA` with the derivative-free direction of Sabi'u et al. [58].

In our comparison, the line search parameters for all algorithms were set to $\sigma = 0.01$ and $r = 0.5$. We performed a tuning process to choose these two values for the selected test problems. The values of the other tuning parameters of the proposed algorithms are default values. For `SILSA`, the default values of the tuning parameters are as follows: $\delta_{\max} = 0.5$ (the initial step size δ_k), $\delta_{\min} = 0$ (the minimum threshold for the step size δ_k), $\omega_d = 2$ (the parameter for updating the step size δ_k), $c = 0.5$ (the direction parameter), $e_{\max} = 10^{-4}$ (maximum value for e_k), $\overline{\gamma} = 10^{-20}$ (the line search parameter), $\lambda_i^0 = \ln(\mu + \frac{1}{2}) - \ln i$ (the initial values for weights), and $m = 10$ (the subspace inertial dimension). Here $\mu = 4 + \lfloor 3 \ln n \rfloor$ was chosen and the normalized version λ_i^0 of $\lambda_i := \lambda_i^0 / \sum_{j=1}^{m-1} \lambda_j^0$ for $i = 1, 2, \cdots, m-1$ was computed.

Following the data profile of Mor'e and Wild [59] and the performance profile of Dolan and Mor'e [60],
• the data profile $\delta_s(\kappa)$ of the solver s for a positive value of κ measures the fraction of problems that the solver s can solve with at most $\kappa(n+1)$ function evaluations, where n is the dimension of problems,
• the performance profile $\rho_s(\tau)$ of the solver s for a positive value of τ measures the relative efficiency of the solver s in solving the set of problems.

In particular, the fraction of problems that the solver s wins compared to the other solvers is $\rho_s(1)$ and the fraction of problems for sufficiently large τ (or κ) that the solver s can solve is $\rho_s(\tau)$ (or $\delta_s(\kappa)$). The measure for efficiency considered in this paper is the number `nf` of function evaluations. The efficiency with respect to `nf` is called `nf` efficiency. All algorithms terminated when exactly one of the conditions $\|F(x_\varepsilon)\| \le 10^{-5}$, `nf` $\le$ `nfmax` $= 10000$, and `sec` $\le$ `secmax` $= 360$ sec was satisfied. Here `sec` denotes time in seconds.

To evaluate their robustness and efficiency, we plot the data and performance profiles of all algorithms. From Figure 1, we conclude that `SILSA` is competitive with the other algorithms. Of the 112 test functions, `SILSA` is able to solve 95% of them, while also having the lowest number of function evaluations on 45% of these problems.

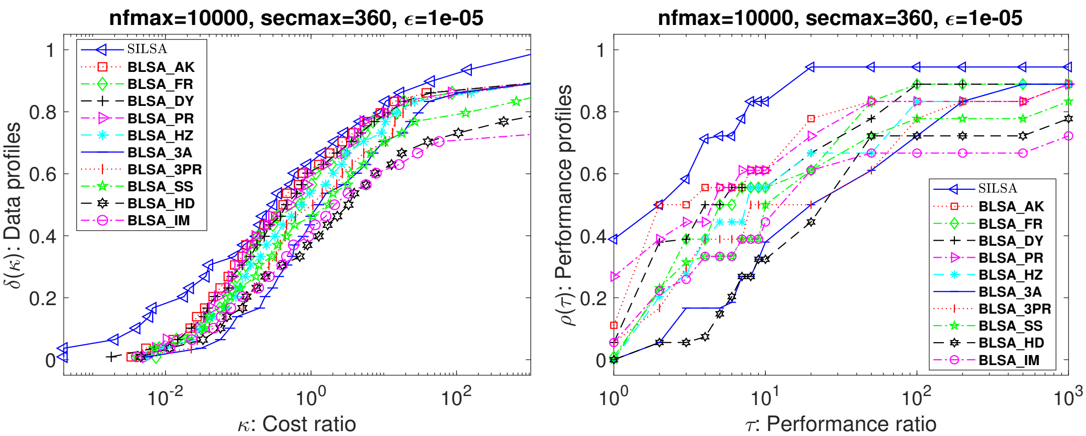

Figure 1.: Results for problems with $n \in \{10, 50, 300, 500, 1000, 5000\}$, the maximum number of function reevaluations (`nfmax` $= 10000$), the maximum time in seconds (`secmax` $= 360$ sec), and $\varepsilon = 10^{-5}$: Data profiles $\delta(\kappa)$ (left) in dependence of a bound κ on the cost ratio, performance profiles $\rho(\tau)$ (right) in dependence of a bound τ on the performance ratio, in terms of `nf`. Problems solved by no solver are ignored.

6. Conclusion

The paper discusses an improved derivative-free line search method for nonlinear monotone equations. Our line search is a combination of the basic line search algorithm proposed by Solodov and Svaiter [18] and a novel subspace inertial point whose goal is to speed up reaching an approximate solution of nonlinear monotone equations. The subspace is generated based on a finite number of the previous `MP` points such that a point with largest residual norm among the previous `MP` points is replaced by a new evaluated point. The global convergence and worst case complexity results are proved. Numerical results show that our improved line search method is competitive with the state-of-the-art derivative-free methods.

7. Test problems in Matlab

The Matlab codes of all 18 test problems are as follows:

```
Problem 1
i=2:(n-1);
F(1)=2*x(1)+sin(x(1))-1;
F(i)=-x(i-1)+2*(x(i))+sin(x(i))-1;
F(n)=2*x(n)+sin(x(n))-1;

Problem 2
i=1:(n);F(i)=2*x(i)-sin(abs(x(i)));

Problem 3
i=1:(n);F(i)=exp(x(i))-1;

Problem 4
h=1/(n+1);i=2:(n-1);
F(1)=x(1)-exp(cos(h*(x(1)+x(2))));
F(i)=x(i)-exp(cos(h.*(x(i-1)+x(i)+x(i+1))));
F(n)=x(n)-exp(cos(h*(x(n-1)+x(n))));

Problem 5
i=2:(n-1);
F(1)=x(1)*(x(1)^2+2*x(2)^2)-1;
F(i)=x(i).*(x(i-1).^2+2*x(i).^2+x(i+1).^2)-1;
F(n)=x(n).*(x(n-1).^2+x(n).^2);


Problem 6
i=2:(n-1);
F(1)=2.5*x(1)+x(2)-1;
F(i)=x(i-1)+2.5*(x(i))+x(i+1)-1;
F(n)=x(n-1)+2.5*(x(n))-1;
```

```
Problem 7
i=2:(n);F(1)=exp(x(1))-1;F(i)=exp(x(i))+x(i)-1;

Problem 8
i=1:n;F(i)=min(min((x(i)),x(i).^2),max((x(i)),x(i).^3));

Problem 9
i=1:(n);F(i)=(i/n)*exp(x(i))-1;

Problem 10
i=1:(n);F(i)=x(i)-sin(abs(x(i)-1));

Problem 11
i=1:(n-1);F(i)=-4+4*x(i).*((x(i).^2)+x(n).^2);
F(n)=4*x(n).*sum(x(i).^2+x(n).^2);

Problem 12
i=1:(n);F(i)=(exp(x(i))).^2+3*sin(x(i)).*cos(x(i))-1;


Problem 13
i=1:(n);F(i)=((8)^0.5)*x(i)-1;

Problem 14
F(1)=x(1);for i=2:n,F(i)=cos(x(i-1))+x(i)-1;end

Problem 15
h=1/(n+1);F(1)=2*x(1)+h*2*(x(1)+sin (x(1)))-x(2);
for i=2:n-1,
    F(i)=2*x(i)+h*2*(x(i)+sin(x(i)))-x(i-1)-x(i+1);
end
F(n)=2*x(n)+h*2*(x(n)+sin(x(n)))-x(n-1);

Problem 16
mu=1e-5;m=n/2;s=x(1:m);y=x(m+1:n);i=1:m;
f(i)=min(min((y(i)),y(i).^2),max((y(i)),y(i).^3));f=f(:);
F(1:m)=s-f;F(m+1:n)=y+s-sqrt((y-s).^2+4*mu);

Problem 17
mu=1e-5;m=n/2;s=x(1:m);y=x(m+1:n);i=1:m;
f(i)=2*y(i)-sin(abs(y(i)));
f=f(:);F(1:m)=s-f; F(m+1:n)=y+s-sqrt((y-s).^2+4*mu);

Problem 18
mu=1e-5;m=n/2;s=x(1:m);y=x(m+1:n);f(1)=y(1);
for i=2:m,f(i)=cos(y(i-1))+y(i)-1;end
f=f(:);F(1:m)=s-f;F(m+1:n)=y+s-sqrt((y-s).^2+4
```

The problems 1-4, 6-8, 10, 13 are from [46], the problem 5 is from [14], the problems 9, 14, 15 are from [61], the problems 12 is from [62], and the problems 16-18 are from the present paper. To obtain the problems 16-18, the nonlinear complementarity problem $(x, s) \geq 0,\ s = F(x),\ x^T s = 0$ is converted to the nonsmooth equations

$$\begin{pmatrix} s - F(x) \\ \min\{x, s\} \end{pmatrix} = 0, \tag{45}$$

where F is a monotone operator. Following [63], by defining

$$\phi(\mu, a, b) := a + b - \sqrt{(a-b)^2 + 4\mu} \ \text{ for all } (\mu, a, b) \in \mathbb{R}^3,$$

the problem (45) is transformed into

$$\begin{pmatrix} s - F(x) \\ \phi(\mu, x_1, s_1) \\ \phi(\mu, x_2, s_2) \\ \\ \phi(\mu, x_n, s_n) \end{pmatrix} = 0.$$

For all test problems, the initial points were chosen to be $x_i = i/(i+2)$ for $i = 1, \ldots, n$.

Disclosure statement

Nopotential conflict of interest was reported by the author(s)

Funding

The first author acknowledges financial support of the Austrian Science Foundation under Project No. P 34317. The second author is grateful to King Fahd University of Petroleum and Minerals for providing excellent research facilities. The third author is supported by an FWO junior postdoctoral fellowship [12AK924N]. In addition she received funding from the Flemish Government (AI Research Program). Susan Ghaderi is affiliated with Leuven.AI - KU Leuven Institute for AI, B-3000, Leuven, Belgium.

References

[1] Chorowski J, Zurada JM. Learning understandable neural networks with nonnegative weight constraints. IEEE Trans Neural Netw Learn Syst. 2015;26(1):62–69.
[2] Blumensath T. Compressed sensing with nonlinear observations and related nonlinear optimization problems. IEEE Trans. Inf. Theory. 2013;59:3466–3474.
[3] Candes EJ, Li X, Soltanolkotabi M. Phase retrieval via wirtinger flow: Theory and algorithms. IEEE Trans. Inf. Theory. 2015 Apr;61(4):1985–2007. Available from: https://doi.org/10.1109/tit.2015.2399924.
[4] Dirkse SP, Ferris MC. Mcplib: a collection of nonlinear mixed complementarity problems. Optim Methods Softw. 1995 Jan;5(4):319–345. Available from: https://doi.org/10.1080/10556789508805619.
[5] Ahookhosh M, Amini K, Bahrami S. Two derivative-free projection approaches for systems of large-scale nonlinear monotone equations. Numer Algorithms. 2012 Oct;64(1):21–42. Available from: https://doi.org/10.1007/s11075-012-9653-z.

[6] Ahookhosh M, Artacho FJA, Fleming RMT, et al. Local convergence of the Levenberg–Marquardt method under hölder metric subregularity. Adv Comput Math. 2019 Jun; 45(5-6):2771–2806. Available from: https://doi.org/10.1007/s10444-019-09708-7.
[7] Ahookhosh M, Fleming RMT, Vuong PT. Finding zeros of hölder metrically subregular mappings via globally convergent Levenberg–Marquardt methods. Optim Methods Softw. 2020 Jan;37(1):113–149. Available from: https://doi.org/10.1080/10556788.2020.1712602.
[8] Amini K, Shiker MA, Kimiaei M. A line search trust-region algorithm with nonmonotone adaptive radius for a system of nonlinear equations. 4OR. 2016;14(2):133–152.
[9] Brown PN, Saad Y. Convergence theory of nonlinear Newton–Krylov algorithms. SIAM J Optim. 1994;4(2):297–330.
[10] Dennis JE, Moré JJ. A characterization of superlinear convergence and its application to quasi-Newton methods. Math Comput. 1974;28(126):549–560.
[11] Esmaeili H, Kimiaei M. A trust-region method with improved adaptive radius for systems of nonlinear equations. Math Methods Oper Res. 2016;83(1):109–125.
[12] Kimiaei M. Nonmonotone self-adaptive Levenberg–Marquardt approach for solving systems of nonlinear equations. Numer Funct Anal Optim. 2018;39(1):47–66.
[13] Kimiaei M, Esmaeili H. A trust-region approach with novel filter adaptive radius for system of nonlinear equations. Numer Algorithms. 2016;73(4):999–1016.
[14] Li D, Fukushima M. A globally and superlinearly convergent Gauss–Newton-based bfgs method for symmetric nonlinear equations. SIAM J Numer Anal. 1999;37(1):152–172.
[15] Yamashita N, Fukushima M. On the rate of convergence of the Levenberg-Marquardt method. In: Topics in numerical analysis. Springer; 2001. p. 239–249.
[16] Yuan Y. Recent advances in numerical methods for nonlinear equations and nonlinear least squares. NACO. 2011;1(1):15–34. Available from: https://doi.org/10.3934/naco.2011.1.15.
[17] Zhou G, Toh KC. Superlinear convergence of a Newton-type algorithm for monotone equations. J Optim Theory Appl. 2005;125(1):205–221.
[18] Solodov MV, Svaiter BF. A globally convergent inexact Newton method for systems of monotone equations. In: Reformulation: Nonsmooth, piecewise smooth, semismooth and smoothing methods. Springer; 1998. p. 355–369.
[19] Al-Baali M, Narushima Y, Yabe H. A family of three-term conjugate gradient methods with sufficient descent property for unconstrained optimization. Comput Optim Appl. 2014 May;60(1):89–110. Available from: https://doi.org/10.1007/s10589-014-9662-z.
[20] Aminifard Z, Babaie-Kafaki S. A modified descent Polak–Ribiére–Polyak conjugate gradient method with global convergence property for nonconvex functions. Calcolo. 2019; 56(2):1–11.
[21] Aminifard Z, Hosseini A, Babaie-Kafaki S. Modified conjugate gradient method for solving sparse recovery problem with nonconvex penalty. Signal Process.. 2022;193:108424.
[22] Andrei N. A simple three-term conjugate gradient algorithm for unconstrained optimization. Comput Appl Math. 2013;241:19–29.
[23] Andrei N. Nonlinear conjugate gradient methods for unconstrained optimization. Springer International Publishing; 2020. Available from: https://doi.org/10.1007/978-3-030-42950-8.
[24] Cheng W. A PRP type method for systems of monotone equations. MCM. 2009;50(1-2):15–20.
[25] Dai YH, Yuan Y. A nonlinear conjugate gradient method with a strong global convergence property. SIAM J Optim. 1999;10(1):177–182.
[26] Fletcher R, Reeves CM. Function minimization by conjugate gradients. Comput j. 1964; 7(2):149–154.
[27] Hager WW, Zhang H. A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J Optim. 2005;16(1):170–192.
[28] Hager WW, Zhang H. A survey of nonlinear conjugate gradient methods. Pacific J Optim.

2006;2(1):35–58.
[29] Ibrahim A, Kumam P, Abubakar AB, et al. A method with inertial extrapolation step for convex constrained monotone equations. J Inequal Appl. 2021;2021(1):1–25.
[30] Ibrahim AH, Kumam P, Abubakar AB, et al. Accelerated derivative-free method for nonlinear monotone equations with an application. Numer Linear Algebra Appl. 2021 Nov;29(3). Available from: https://doi.org/10.1002/nla.2424.
[31] Ibrahim AH, Kumam P, Sun M, et al. Projection method with inertial step for nonlinear equations: Application to signal recovery. J Ind Manag. 2021;.
[32] Liu Y, Storey C. Efficient generalized conjugate gradient algorithms, part 1: theory. J Optim Theory Appl. 1991;69(1):129–137.
[33] Liu Y, Zhu Z, Zhang B. Two sufficient descent three-term conjugate gradient methods for unconstrained optimization problems with applications in compressive sensing. J Appl Math Comput. 2021;:1–30.
[34] Lotfi M, Mohammad Hosseini S. An efficient hybrid conjugate gradient method with sufficient descent property for unconstrained optimization. Optim Methods Softw. 2021; :1–15.
[35] Papp Z, Rapajić S. FR type methods for systems of large-scale nonlinear monotone equations. Appl Math Comput. 2015;269:816–823.
[36] Polak E, Ribiere G. Note sur la convergence de méthodes de directions conjuguées. ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique. 1969;3(R1):35–43.
[37] Polyak BT. The conjugate gradient method in extremal problems. USSR comput math math phys. 1969;9(4):94–112.
[38] Zhang L, Zhou W, Li DH. A descent modified Polak–Ribière–Polyak conjugate gradient method and its global convergence. IMA J Numer. 2006;26(4):629–640.
[39] Alvarez F. Weak convergence of a relaxed and inertial hybrid projection-proximal point algorithm for maximal monotone operators in hilbert space. SIAM J Optim. 2004 Jan; 14(3):773–782. Available from: https://doi.org/10.1137/s1052623403427859.
[40] Alvarez F, Attouch H. An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping. Set-Valued Var Anal. 2001;9(1):3–11.
[41] Attouch H, Peypouquet J, Redont P. A dynamical approach to an inertial forward-backward algorithm for convex minimization. SIAM J Optim. 2014;24(1):232–256.
[42] Beck A, Teboulle M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J Imaging Sci. 2009;2(1):183–202.
[43] Boţ RI, Grad SM. Inertial forward–backward methods for solving vector optimization problems. Optimization. 2018 Feb;67(7):959–974. Available from: https://doi.org/10.1080/02331934.2018.1440553.
[44] Boţ RI, Nguyen DK. A forward–backward penalty scheme with inertial effects for monotone inclusions. applications to convex bilevel programming. Optimization. 2018 Dec; 68(10):1855–1880. Available from: https://doi.org/10.1080/02331934.2018.1556662.
[45] Boţ R, Sedlmayer M, Vuong PT. A relaxed inertial forward-backward-forward algorithm for solving monotone inclusions with application to GANs. arXiv e-prints. 2020 Mar; :arXiv:2003.07886.
[46] Ibrahim AH, Kimiaei M, Kumam P. A new black box method for monotone nonlinear equations. Optimization. 2021 Nov;:1–19Available from: https://doi.org/10.1080/02331934.2021.2002326.
[47] Kimiaei M, Neumaier A. Efficient global unconstrained black box optimization. Math Program Comput. 2022 feb;Available from: https://link.springer.com/article/10.1007%2Fs12532-021-00215-9.
[48] Dai YH, Liao LZ. New conjugacy conditions and related nonlinear conjugate gradient methods. Appl Math Optim. 2001 Jan;43(1):87–101. Available from: https://doi.org/10.1007/s002450010019.
[49] Li M. An Liu-Storey-type method for solving large-scale nonlinear monotone equations.

Numer Funct Anal Optim. 2014;35(3):310–322.
[50] Cartis C, Sampaio P, Toint P. Worst-case evaluation complexity of non-monotone gradient-related algorithms for unconstrained optimization. Optimization. 2014 Jan; 64(5):1349–1361. Available from: https://doi.org/10.1080/02331934.2013.869809.
[51] Curtis FE, Lubberts Z, Robinson DP. Concise complexity analyses for trust region methods. Optim. Lett. 2018 Jun;12(8):1713–1724. Available from: https://doi.org/10.1007/s11590-018-1286-2.
[52] Dodangeh M, Vicente LN. Worst case complexity of direct search under convexity. Math Program. 2014 Nov;155(1-2):307–332. Available from: https://doi.org/10.1007/s10107-014-0847-0.
[53] Dodangeh M, Vicente LN, Zhang Z. On the optimal order of worst case complexity of direct search. Optim. Lett. 2015 Jun;10(4):699–708. Available from: https://doi.org/10.1007/s11590-015-0908-1.
[54] Vicente L. Worst case complexity of direct search. EURO J Comput Optim. 2013 May; 1(1-2):143–153. Available from: https://doi.org/10.1007/s13675-012-0003-7.
[55] Ivanov B, Milovanović GV, Stanimirović PS. Accelerated dai-liao projection method for solving systems of monotone nonlinear equations with application to image deblurring. J Glob Optim. 2022 Jul;85(2):377–420. Available from: https://doi.org/10.1007/s10898-022-01213-4.
[56] Abubakar AB, Kumam P. A descent Dai-Liao conjugate gradient method for nonlinear equations. Numer Algorithms. 2018 May;81(1):197–210. Available from: https://doi.org/10.1007/s11075-018-0541-z.
[57] Huang F, Deng S, Tang J. A derivative-free memoryless BFGS hyperplane projection method for solving large-scale nonlinear monotone equations. Soft Comput. 2022 Oct; 27(7):3805–3815. Available from: https://doi.org/10.1007/s00500-022-07536-4.
[58] Sabi'u J, Shah A, Stanimirović PS, et al. Modified optimal Perry conjugate gradient method for solving system of monotone equations with applications. Appl Numer Math. 2023 Feb;184:431–445. Available from: https://doi.org/10.1016/j.apnum.2022.10.016.
[59] Moré JJ, Wild SM. Benchmarking derivative-free optimization algorithms. SIAM J Optim. 2009 Jan;20(1):172–191. Available from: https://doi.org/10.1137/080724083.
[60] Dolan ED, Moré JJ. Benchmarking optimization software with performance profiles. Math Program. 2002 Jan;91(2):201–213. Available from: https://doi.org/10.1007/s101070100263.
[61] La Cruz W, Martínez J, Raydan M. Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. Math Comput. 2006; 75(255):1429–1448.
[62] Gao P, He C. An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints. Calcolo. 2018 Nov;55(4). Available from: https://doi.org/10.1007/s10092-018-0291-2.
[63] Geiger C, Kanzow C. On the resolution of monotone complementarity problems. Comput Optim Appl. 1996 Mar;5(2):155–173. Available from: https://doi.org/10.1007/bf00249054.