RAIRO-Oper. Res. 58 (2024) 2887–2905
https://doi.org/10.1051/ro/2024102

RAIRO Operations Research
www.rairo-ro.org

ON DIAGONALLY STRUCTURED SCHEME FOR NONLINEAR LEAST SQUARES AND DATA-FITTING PROBLEMS

Mahmoud Muhammad Yahaya[1,2], Poom Kumam[1,2,*], Parin Chaipunya[1,2], Aliyu Muhammed Awwal[1,3] and Lin Wang[4]

Abstract. Recently, structured nonlinear least-squares (NLS) based algorithms gained considerable emphasis from researchers; this attention may result from increasingly applicable areas of these algorithms in different science and engineering domains. In this article, we coined a new efficient structured-based NLS algorithm. We developed a diagonal Hessian-based formulation for solving NLS problems. We derived the quasi-Newton update based on a diagonal matrix scheme subject to a modified structured secant condition. Also, we show that the algorithm's search direction satisfies a sufficient descent condition under some standard assumptions. Subsequently, we also prove the global convergence of the algorithm and then eventually show its linear convergence rate for strongly convex functions. Furthermore, to show case the proposed algorithm's performance, we experimented numerically by comparing it with other approaches on some benchmark test functions available in the literature. Finally, the introduced scheme is applied to solve some data-fitting problems

Mathematics Subject Classification. 93E24, 90C53, 65D10.

Received October 1, 2022. Accepted May 6, 2024.

1. Introduction

In this article, we consider a notion of finding a minimizer, x, of the following mathematical model or formulation

$$\text{minimize}_{x\in\mathbb{R}^n} f(x), \; \textit{where } f(x) = \frac{1}{2}\sum_{i=1}^{m}(r_i(x))^2 = \frac{1}{2}\mathbf{r}(x)^T\mathbf{r}(x) = \frac{1}{2}\|\mathbf{r}(x)\|^2. \tag{1.1}$$

Keywords. Data fitting, diagonal update, nonlinear least squares, secant condition, convergence rate.

[1] Center of Excellence in Theoretical and Computational Science (TaCS-CoE) and KMUTTFixed Point, Research Laboratory, Room SCL 802 Fixed Point Laboratory Science Laboratory Building, Department of Mathematics, Faculty of Science, King Mongkut's University of Technology Thonburi (KMUTT), 126 Pracha-Uthit Road, Bang Mod, Thung Khru, Bangkok 10140, Thailand.

[2] NCAO Research Center, Fixed Point Theory and Applications Research Group, Center of Excellence in Theoretical and Computational Science (TaCSCoE), Faculty of Science, King Mongkut's University of Technology, Thonburi (KMUTT), 126 Pracha-Uthit Rd, Bang Mod, Thung Khru, Bangkok 10140, Thailand.

[3] Department of Mathematics, Faculty of Science, Gombe State University, Gombe 760214, Nigeria.

[4] Office of Science and Research, Yunnan University of Finance and Economics, Kunming, P.R. China.

*Corresponding author: poom.kum@kmutt.ac.th

The model (1.1) is one of the standard formulations of *nonlinear least squares (NLS)*, $r_i : \mathbb{R}^n \to \mathbb{R}$ is a smooth function for each $i = 1, 2, \ldots, m$, $\mathbf{r} : \mathbb{R}^n \to \mathbb{R}^m (m \geq n)$ is twice continuously differentiable and $\|\cdot\|$ is Euclidean norm. The equation (1.1) possesses a special structure about how its *gradient* and *Hessian* are evaluated. These are computed using the following:

$$g(x) = \nabla f(x) = \sum_{i=1}^{m} r_i(x) \nabla r_i(x) = J(x)^T \mathbf{r}(x), \tag{1.2}$$

similarly,

$$H(x) = \nabla^2 f(x) = \sum_{i=1}^{m} \nabla r_i(x) \nabla r_i(x)^T + \sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x) = J(x)^T J(x) + P(x), \tag{1.3}$$

where, $J(x)$ is a Jacobian matrix, i.e a derivative of $\mathbf{r}$ at x and $P(x) = \sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x)$. There are mainly two classes of algorithms for solving (1.1). These are *line-search* and *trust-region* based approaches, respectively. Both of these generate a sequence of iterates, say $\{x_k\}$ using the recurrence formulation below:

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, 2, \ldots \tag{1.4}$$

in which α_k is a *step-size* and d_k is a direction. However, unlike in the line-search-based approach where the algorithm would first generate a descent direction d_k and take a step α_k in that direction, the trust-region-based approach first defines a model of the function in the region around current x_{k+1} and then choose a step that would lead to an approximate minimizer of the model in that region. This article focuses on developing a *line-search based* efficient and robust algorithm.

One of the approaches used for choosing a suitable *step-size* is known as a non-monotone Armijo rule. The step-length, $\alpha_k := \bar{\alpha}_k \delta^{j_k}$, where $\bar{\alpha}_k > 0$ initial trial point and j_k is the largest integer for which the following

$$f(x_k + \alpha d_k) \leq R_k + \sigma \alpha g_k^T d_k \tag{1.5}$$

holds. The line search rule (1.5) coupled with a backtracking strategy is described below:

Algorithm 1: The Zhang and Hager [21] line search.

Input: $f(x_k)$, descent direction, d_k at iterate, x_k, and $0 \leq \vartheta_{\min} \leq \vartheta_{\max} \leq 1$, $\sigma \in (0, 1)$, $f(x_0) = R(x_0)$, $W_0 = 1$, and $\vartheta_k \in [\vartheta_{\min}, \vartheta_{\max}]$.

Initialize $\alpha = 1$;

while $f(x_k + \alpha d_k) > R_k + \sigma \alpha g_k^T d_k$ **do**

$\alpha = \alpha/2$;

Choose $\vartheta_k \in [\vartheta_{\min}, \vartheta_{\max}]$ and update as follows

$$R_{k+1} = \frac{\vartheta_k W_k R_k + f(x_{k+1})}{W_{k+1}}, \quad \textit{and} \quad W_{k+1} = \vartheta_k W_k + 1. \tag{1.6}$$

end

Remark 1.1. The non-negative parameter, ϑ, determines the degree of monotonicity of the line search. If for each, k, we have $\vartheta_k = 0$, then the line search is termed a monotone Armijo-type line search; else, it is called non-monotone.

Remark 1.2. Moreover, if the parameter $\vartheta_k = 1$, for all k, then $R_k = \varphi_k$ with φ_k defined as

$$\varphi_k = \frac{1}{k+1} \sum_{i=1}^{k} f(x_i). \tag{1.7}$$

Algorithms designed for solving NLS problems are essential due to their wide applicability in solving numerous important problems. Some of the useful problems that can often be modelled into (1.1) include data-fitting tasks, robotic motion control, parameter estimation, and imaging problems. In addition, if problem (1.1) is slightly modified, the NLS algorithm can be used to train a feed-forward neural network and so on (interested readers may wish to check the following references [1, 2, 11, 16]). Some classical *line-search based* algorithms for NLS problems, include Gauss-Newton (GN), Lavenberg Marquart (LM) and some of their variants. However, these classical algorithms may fall short, particularly for high-dimensional problems. Recently, researchers turned their attention towards structured-based approaches, where the idea is to approximate the second-order term in the Hessian with an action of a vector, $s_k := x_{k+1} - x_k$.

Algorithms derived based on the structure of (1.1) include the work of Kobayashi *et al.* [7] where they came up with a structured algorithm using conjugate gradient algorithm; they develop their approach using a modified secant equation. Moreover, motivated by the success of this approach, Dehghani and Mahdavi-Amiri [3] coined another conjugate gradient-based algorithm for solving (1.1) where unlike in [7], their proposal is based upon a higher-order approximation of the Hessian. In another paradigm, Mohammed and Santos [10] proposed an algorithm for solving (1.1) through a diagonal approximation of the Hessian, where some structured secant conditions are used in deriving the diagonal approximation of the Hessian matrix. However, their derived approximation of the Hessian matrix in equation (1.3) is composed of approximating the first-term and second-term of (1.3) and also they had to come up with some restrictive strategies to prevent the values of the diagonal matrix from being negative. To mitigate some of these shortcomings, Yahaya *et al.* [17] proposed a variant of a diagonal matrix approximation of the Hessian. Here, the Hessian is composed of the first-term in (1.3) and some approximation of the matrix-vector approximation of the Hessian's second-term. Their formulation is based on diagonal quasi-Newton methods, where a correction matrix with diagonal entries is derived from a *least change secant* condition subject to a modified weak secant condition. The algorithm was shown to perform well numerically. Motivated by the contributions mentioned above, in this article, we develop a new algorithm whose formulation of the search direction is basically obtained as an approximation of the Hessian using a diagonal matrix in (1.3). Unlike in [17], and [19] where the correction matrix with diagonal entries was derived using the least change secant subject to a modified weak secant equation, the entries of the proposed correction matrix with diagonal structure were obtained using the least change secant subject to a modified secant condition. This derivation makes the proposed formulation less complex, and consequently, the algorithm performance is better, as seen from the preliminary experiments discussed in the numerical section. The summarized contributions of this article are as follows:

1. We propose a new structured diagonal matrix update based on the quasi-Newton method for solving (1.1).
2. The proposed algorithm is globally convergent under some standard assumptions and linear convergent rate for strongly convex functions.
3. We numerically demonstrated the efficiency and effectiveness of the coined algorithms on test problems in the literature and some data-fitting problems.

We segmented the paper into the derivation of the algorithm and the steps for its implementation in Section 2. More so, the convergence properties of the proposal while considering some standard assumptions were presented in Section 3. Lastly, in Section 4, we offer the numerical result by solving some benchmark test problems. Wherever $\|\cdot\|$ is used, it represents an Euclidean norm.

2. Development of the proposed algorithm

It can be observed in (1.3) that there is a need for evaluating a 2^{nd} order derivative of the residuals. The computation is intractable, particularly for higher-dimensional problems. As a result, approximating the second term may be helpful so that computing the Hessian is made simpler.

Now at some iterate, say, k, the term, $P(x)$ in expression, (1.3) can be written in the following form:

$$P(x_{k+1}) = \sum_{i=1}^{m} r_i(x_{k+1}) Z_i(x_{k+1}), \tag{2.1}$$

where we can represent $r_i(x_{k+1})$, and $Z_i(x_{k+1})$ as i^{th}-component of $\mathbf{r}(x_{k+1})$, and $\nabla^2 r_i(x_{k+1})$ respectively for, $i = 1, 2, \ldots, m$.

Therefore, we aim to look for an approximate matrix update $B(x_{k+1})$ of the Hessian, $H(x_{k+1})$ such that a secant condition is fulfilled, *i.e.*

$$B(x_{k+1})s_k \approx J(x_{k+1})^T J(x_{k+1})s_k + P(x_{k+1})s_k,$$

Thus, approximating the product, $P(x_{k+1})s_k$ of (2.1) entails approximating the action of the matrix, $Z_i(x_{k+1})$ on the s_k, without directly evaluating $Z_i(x_{k+1})$ throughout the iteration process. To simplify the notations, we denote $P(x_{k+1}) = P_{k+1}$, $r_i(x_{k+1}) = r^i_{k+1}$, and $Z_i(x_{k+1}) = Z^i_{k+1}$. Now, if we let $\bar{g}^i_{k+1}$ to be the gradient of the residual vector, r^i_{k+1}, then applying Taylor's series techniques on the gradient, $\bar{g}^i_{k+1}$ will give

$$\bar{g}^i_k \approx \bar{g}^i_{k+1} - Z^i_{k+1}s_k, \quad for \ \ all \ i = 1, 2, 3, \ldots, m,$$

thus, we can have,

$$Z^i_{k+1}s_k \approx \bar{g}^i_{k+1} - \bar{g}^i_k, \quad for \ \ all \ i = 1, 2, 3, \ldots, m, \tag{2.2}$$

and

$$P_{k+1}s_k = \sum_{i=1}^{m} r^i_{k+1} Z^i_{k+1} s_k. \tag{2.3}$$

Thus, substituting (2.2) into (2.3) and then performing some summation gives

$$P_{k+1}s_k \approx (J_{k+1} - J_k)^T r_{k+1}, \tag{2.4}$$

Therefore, we intend here to get an updated diagonal representation, B_{k+1}, such that

$$B_{k+1}s_k \approx H(x_{k+1})s_k = (J^T_{k+1}J_{k+1})s_k + P_{k+1}s_k,$$

is satisfied, and thus we have as follows:

$$B_{k+1}s_k = (J^T_{k+1}J_{k+1})s_k + P_{k+1}s_k, \tag{2.5}$$

in which, B_{k+1} is a suitable diagonal matrix approximation of the Hessian, H_{k+1} obtained such that the above-modified condition is achieved and $B_{k+1} = B_k + \Delta_k$, where Δ_k is a correction matrix with diagonal entries. This Δ_k is formulated using the *Lemma* as given below:

Lemma 2.1. *Suppose Δ_k and B_k are matrices (diagonal) with the entries δ^i_k, and b^i_k for, $i = 1, 2, \ldots, m$, respectively. Then the entries, δ^i_k, of the correction matrix with diagonal entries, Δ_k, are obtained by solving the following optimization problem*

$$\min_{\Delta_k} \frac{1}{2}\|\Delta_k\|_F, \tag{2.6}$$

$$s.t \ \ (B_k + \Delta_k)s_k = \beta_k, \tag{2.7}$$

is defined as follows

$$\delta^i_k = \frac{(\beta^i_k - b^i_k s^i_k)}{s^i_k}, \qquad i = 1, 2, \ldots, m, \tag{2.8}$$

where

$$\beta_k = (J_{k+1}^T J_{k+1})s_k + (J_{k+1} - J_k)^T r_{k+1}, \tag{2.9}$$

and $\|\cdot\|_F$, *is a Frobenius norm matrix.*

Proof. The formulation given in (2.6) can be re-written in the following way,

$$\min_{\delta} \frac{1}{2}\sum_{i=1}^{m}(\delta_k^i)^2 \tag{2.10}$$

$$\text{s.t} \quad (B_k + \Delta_k)s_k = \beta_k. \tag{2.11}$$

Note that problem (2.6) is convex. The Lagrangian function of (2.10) is given as:

$$L(\delta_k, \gamma_k) = \frac{1}{2}\sum_{i=1}^{m}(\delta_k^i)^2 + \sum_{i=1}^{m}\gamma_k^i\left((b_k^i + \delta_k^i)s_k^i - \beta_k^i\right),$$

where γ_k is a Lagrangian operator. Now, computing $\frac{\partial L}{\partial \delta_k^i}$ and setting it to zero we have

$$\frac{\partial L}{\partial \delta_k^i} = \delta_k^i + \gamma_k^i s_k^i = 0, \quad \textit{for} \quad i = 1, 2, \ldots, m,$$

this leads to,

$$\delta_k^i = -\gamma_k^i s_k^i, \quad \textit{for} \quad i = 1, 2, \ldots, m. \tag{2.12}$$

Now, performing some pre-multiplication operation on (2.12) by s_k^i, we have

$$s_k^i \delta_k^i = -\gamma_k^i (s_k^i)^2, \quad \textit{for} \quad i = 1, 2, \ldots, m. \tag{2.13}$$

We re-write (2.7) entries-wise into,

$$s_k^i \delta_k^i = \beta_k^i - s_k^i b_k^i, \quad \textit{for} \quad i = 1, 2, \ldots, m, \tag{2.14}$$

where δ_k^i and b_k^i are diagonal entries of the diagonal matrices, Δ_k and B_k respectively. Hence, evaluating for γ_k^i using the above expressions, (2.13) and (2.14) yields

$$-\gamma_k^i = \frac{(\beta_k^i - s_k^i b_k^i)}{(s_k^i)^2}, \quad \textit{for} \quad i = 1, 2, \ldots, m. \tag{2.15}$$

Therefore, substituting equation (2.15) into equation (2.12), gives the correction matrix with diagonal entries as

$$\delta_k^i = \frac{(\beta_k^i - s_k^i b_k^i)}{s_k^i}, \quad \textit{for} \quad i = 1, 2, \ldots, m. \tag{2.16}$$

□

Thus, we can define the search direction, d_{k+1}, of the proposed algorithm as follows:

$$d_0 = -B_0^{-1} g_0, \quad \text{for} \quad k = 0 \qquad \text{and} \qquad d_{k+1} = -B_{k+1}^{-1} g_{k+1}, \quad \text{for} \quad k = 1, 2, 3, \ldots, \tag{2.17}$$

in which $B_0 = \text{diag}(b_0^i)$, $b_0^i = 1$ for $i = 1, 2, 3, \ldots, m$ and the values of the update, B_{k+1}, is:

$$b_{k+1}^i = b_k^i + \delta_k^i, \quad \textit{for} \quad i = 1, 2, \ldots, m \quad \textit{and} \quad k = 0, 1, 2, \ldots \tag{2.18}$$

where δ_k^i is given by (2.16).

2.1. A simple safeguarding approach

In an attempt to make sure that the diagonal matrix, B_k remains positive definite for all k, we safeguard its entries, b^i_{k+1} for $i = 1, 2, \dots, m$. This is purposely done to prevent the entries from becoming too large, negative or zero. So, suppose that $\epsilon > 0$ and $\eta > 0$, where $\epsilon << \eta << +\infty$. Thus, the diagonal entries with the safeguard defined in (2.18) is re-stated as follows:

$$\bar{b}^i_{k+1} = \min\left\{\max\{b^i_k + \delta^i_k,\ \epsilon\},\ \eta\right\},\quad for\ \ i = 1, 2, \dots, m\ \ and\ \ k = 0, 1, 2, \dots \tag{2.19}$$

with δ^i_k is given by (2.16) for each i. So, $\bar{B}_{k+1} = diag(\bar{b}^i_{k+1})$ for $i = 1, 2, \dots, m$.
Next, we formally state the algorithm of the new proposed iterative scheme.

Algorithm 2: Structured **D**iagonal based on **M**odified **S**ecant **C**ondition (SDMSC)

Input: $x_0 \in \mathbb{R}^n$, $0 < \epsilon << \eta << +\infty$, $\sigma \in (0, 1)$, $tol > 0$, and set $k := 0$, $R_0 = f(x_0)$, $B_0 = I$,
$\vartheta_k \in [\vartheta_{\min}, \vartheta_{\max}]$, $0 \leq \vartheta_{\min} \leq \vartheta_{\max} \leq 1$, $\sigma \in (0, 1)$.
Compute r_0, f_0 and g_0;
Compute

$$d_0 = -B_0^{-1} g_0 \tag{2.20}$$

while $\|g_k\| > tol$ **do**
  Evaluate, α_k using Algorithm 1
  Then, update $x_{k+1} = x_k + \alpha_k d_k$;
  Compute the, $\bar{b}^i_{k+1}$, for $i = 1, 2, \dots, m$ using (2.19) of the diagonal matrix, $\bar{B}_{k+1}$.
  Update r_{k+1}, f_{k+1} and g_{k+1};
  Update $d_{k+1} = -\bar{B}^{-1}_{k+1} g_{k+1}$.
  Set $k := k + 1$.
end

Remark 2.2. It is paramount to note that we have implemented (in MATLAB) the above algorithm in such a way that for each problem considered, the structured gradient g_k and the Jacobian terms in β_k are computed directly as a matrix-vector product without explicitly forming or storing a matrix throughout the iteration process.

Remark 2.3. In the implementation of Algorithm 2 named SDMSC, as you will see in Section 4, we have two variants, namely SDMSC1 and SDMSC2. These are obtained based on the ϑ_k for $k = 0, 1, 2, \dots$ used. If $\vartheta = 0$, $\forall k$, then the non-monotone line search is reduced to Armijo line search, and we named Algorithm 2 as SDMSC2, otherwise it is called SDMSC1.

3. Analysing the convergence properties

For analysing the convergence of the above, *SDMSC*, algorithm, the assumptions stated below are quite helpful:

Assumption 3.1. *The function f is twice continuously differentiable on a bounded set, $\psi = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ and f is bounded below, where $x_0 \in \mathbb{R}^n$ is an initial point.*

Assumption 3.2. *Assume that the Jacobian and the residual are Lipschitz continuous in some neighbourhood N of ψ with Lipschitz constants, $l_1 > 0$ and $l_2 > 0$, i.e. $\|J(x) - J(y)\| \leq l_1\|x - y\|$, $\|J(x)\| < c_1$, and $\|r(x) - r(y)\| \leq l_2\|x - y\|$, $\|r(x)\| < c_2$, $\forall x, y \in \psi$, where c_1, and c_2 are positive constants.*

Assumption 3.3. *The gradient of (1.1), denoted by $\nabla f(x) = J(x)^T R(x)$, exhibits uniform continuity within an open convex set, confining the level set ψ.*

We can easily deduce from the Assumption 3.2 that there exist some positive constants, $l_3,$ andc_3, the following inequalities hold

$$\|g(x) - g(y)\| \leq l_3\|x - y\|, \quad \|g(x)\| \leq c_3, \tag{3.1}$$

the inequality in equation (3.1) is true because

$$\begin{aligned}\|g(x) - g(y)\| &= \|J(x)^{\mathrm{T}}(r(x) - r(y)) + (J(x) - J(y))^{\mathrm{T}} r(x)\| \\ &\leq \|J(x)\|\|(r(x) - r(y))\| + \|(J(x) - J(y))\|\|r(x)\| \\ &\leq (c_1 l_2 + l_1 c_2)\|x - y\|,\end{aligned}$$

by taking $l_3 = (c_1 l_2 + l_1 c_2)$, the required inequality (3.1) is achieved.

Lemma 3.4. *Let Assumptions 3.1, and 3.2 hold, then there exist some positive constants M_1 and $\bar{M}$ such that,* $\forall k > 0$,

$$\|\beta_k\| \leq \bar{M}\|s_k\|. \tag{3.2}$$

Proof. Now, using the β_k defined in (2.9), we have

$$\begin{aligned}\|\beta_k\| &= \|J_{k+1}^T J_{k+1} s_k + (J_{k+1} - J_k)^T r_{k+1}\| \\ &\leq \|J_{k+1}^T J_{k+1} s_k\| + \|(J_{k+1} - J_k)^T r_{k+1}\| \quad \text{(using triangular inequality)} \\ &\leq \|J_{k+1}\|^2\|s_k\| + \|J_{k+1} - J_k\|\|r_{k+1}\| \quad \text{(using matrix norm property)} \\ &\leq c_1^2\|s_k\| + l_1\|x_{k+1} - x_k\|\|r_{k+1}\| \\ &\leq c_1^2\|s_k\| + l_1 c_2\|s_k\| \\ &= (c_1^2 + l_1 c_2)\|s_k\|.\end{aligned}$$

by setting $\bar{M} := c_1^2 + l_1 c_2$. Then, the inequality (3.2) holds. □

Next, we state a lemma which shows that the search direction, d_k, is *sufficiently descent*, and it is upper bounded.

Lemma 3.5. *Suppose $\{d_k\}$, be a sequence of search directions generated by the proposed $SDMSC$, then we can obtain some positive constants, c_4 and c_5 such that these inequalities given below hold*

$$g_k^T d_k \leq -c_4\|g_k\|^2, \tag{3.3}$$

$$\|d_k\| \leq c_5\|g_k\|, \tag{3.4}$$

for all $k \geq 0$.

Proof. Suppose that the diagonal entries of the approximation of the Hessian B_k are defined as stated in equation (2.19). Thus, the following inequality, (2.19), holds

$$\epsilon \leq \min\left\{\max\{b_{k-1}^i + \delta_{k-1}^i,\ \epsilon\},\ \eta\right\} \leq \eta, \tag{3.5}$$

and therefore, by pre-multiplying (2.20) by g_k^T we have,

$$\begin{aligned}g_k^T d_k &= -g_k^T B_k^{-1} g_k \\ &= -\sum_{i=1}^{m}(g_k^i)^2 / \min\left\{\max\{b_{k-1}^i + \delta_{k-1}^i,\ \epsilon\},\ \eta\right\} \\ &\leq -\frac{1}{\eta}\sum_{i=1}^{m}(g_k^i)^2 \\ &= -c_4\|g_k\|^2,\end{aligned}$$

with $c_4 = \frac{1}{\eta}$, and thus, (3.3) holds.
Moreover, using the symmetric property of the diagonal matrix, B_k, we have

$$\begin{aligned}\|d_k\|^2 &= g_k^T B_k^{-2} g_k \\ &= \sum_{i=1}^{m} \left(g_k^i / \min\left\{\max\{b_{k-1}^i + \delta_{k-1}^i,\ \epsilon\},\ \eta\right\}\right)^2 \\ &\leq \frac{1}{\epsilon^2} \sum_{i=1}^{m} (g_k^i)^2 \\ &= c_5 \|g_k\|^2,\end{aligned}$$

with $c_5 = \frac{1}{\epsilon^2}$ and hence the proof of the Lemma. □

Utilizing the findings from Lemmas 3.4 and 3.5, in conjunction with the uniform continuity of ∇f, leads to the conclusion presented in Proposition 1 [10]. This proposition demonstrates the existence of suitably small positive step sizes that ensure the fulfilment of the Armijo rule (1.5) within a finite number of steps. Consequently, we establish that *SDMSC* is well-defined, as supported by Proposition 1, whose detailed proof can be found in [10]. From the Remark 1.2, the following Lemma 3.6 also follows whose proof can also be found in [18].

Lemma 3.6. *Let the proposed Algorithm, SDMSC generates the sequence* $\{x_k\}$ *and suppose,* R_k *and* φ_k *are defined by* (1.5) *and* (1.7)*, respectively, then for all* k *the inequality* $f(x_k) \leq R_k \leq \varphi_k$ *holds.*

Lemma 3.7. *Let the proposed Algorithm, SDMSC generate the sequences* $\{x_k\}$ *and* $\{d_k\}$ *such that the relation* (3.3) *holds, then* $f(x_k) \leq f(x_0)$*, for each* $k \geq 0$.

Proof. Suppose R_{k+1} and W_{k+1} be as defined in (1.6) together with (1.5) and (3.3) gives

$$\begin{aligned}R_{k+1} &= \frac{\vartheta_k W_k R_k + f(x_{k+1})}{W_{k+1}}, \\ &= \frac{(W_{k+1} - 1)R_k + f(x_{k+1})}{W_{k+1}}, \\ &\leq \frac{W_{k+1} R_k + \alpha_k \sigma g_k^T d_k}{W_{k+1}}, \\ &\leq \frac{W_{k+1} R_k - \alpha_k \sigma c_4 \|g_k\|^2}{W_{k+1}}, \\ &= R_k - \frac{\alpha_k \sigma c_4 \|g_k\|^2}{W_{k+1}} \\ &\leq R_k,\end{aligned}$$

where the last inequality holds by dropping the negative term in the preceding line. This means the sequence $\{R_{k+1}\}$ is decreasing and since $f_k \leq R_k$, (see, Lem. 3.6), we have

$$f(x_{k+1}) \leq R_{k+1} \leq R_k \leq R_{k-1} \leq \ldots \leq R_0 = f(x_0),$$

and thus, the proof is complete. □

Lemma 3.8. *Suppose the Armijo-condition* (1.5) *of the non-monotone line search is employed, and* ∇f *satisfies the Lipschitz conditions in Assumption 3.2 with Lipschitz constant* p_3*. Then*

$$\alpha_k \geq \min\left\{\frac{\omega}{\delta},\ \left(\frac{2(\delta - 1)}{p_3}\right) \frac{g_k^T d_k}{\|d_k\|^2}\right\}. \tag{3.6}$$

Proof. Suppose $\delta\alpha_k \geq \omega$, then we have $\alpha_k \geq \omega/\delta$. Thus, the proof is achieved. On the other hand, suppose $\delta\alpha_k < \omega$, then j_k is the largest integer such that $\alpha_k = \bar{\alpha}_k \delta^{j_k}$ satisfies the inequality (1.5). Now, we have

$$\begin{aligned} f(x_k + \alpha_k d_k) &> R_k + \delta\alpha g_k^T d_k \\ &\geq f(x_k) + \delta\alpha g_k^T d_k, \quad (\text{ since } \; f(x_k) \leq R_k), \end{aligned} \tag{3.7}$$

since ∇f is Lipschitz continuous,

$$\begin{aligned} f(x_k + \alpha d_k) - f(x_k) &= \alpha g_k^T d_k + \int_0^\alpha [\nabla f(x_k + u d_k) - \nabla f(x_k)] d_k \; du \\ &\leq \alpha g_k^T d_k + \int_0^\alpha u p_3 \|d_k\|^2 du \\ &= \alpha g_k^T d_k + \frac{1}{2} p_3 \alpha^2 \|d_k\|^2. \end{aligned}$$

Thus, combining the above inequality with (3.7) indicate that,

$$\alpha(\delta - 1) g_k^T d_k \leq \frac{1}{2} p_3 \alpha^2 \|d_k\|^2. \tag{3.8}$$

Hence, we can get the required result (3.6) from (3.8) and $\alpha_k \geq \omega/\delta$. □

We now state and prove (for completeness) a Theorem, which is motivated from [21] that shows the global convergence property of $SDMSC$ Algorithm.

Theorem 3.9. *Let $f(x)$ as formulated in (1.1) and the Assumptions 3.1 and 3.2 hold and also suppose Lemmas 3.5 and 3.5 hold, then the sequence of iterates $\{x_k\}$ generated using the SDMSC is contained ψ and*

$$\liminf_{k \to \infty} \|g_k\| = 0. \tag{3.9}$$

Moreover, if $\vartheta_{\max} < 1$ then

$$\lim_{k \to \infty} \|g_k\| = 0. \tag{3.10}$$

Proof. To prove this theorem, we need first to show that,

$$f(x_k + \alpha_k d_k) \leq R_k - \Delta \|g_k\|^2, \tag{3.11}$$

where,

$$\Delta = \left(\frac{2\delta(1-\delta)c_4^2}{c_5 p_3} \right) \frac{g_k^T d_k}{\|d_k\|^2}.$$

Now, by using (3.6), and inequalities (3.3), (3.4) from Lemma 3.5, we have

$$\begin{aligned} f(x_k + \alpha_k d_k) &\leq R_k + \delta\alpha_k g_k^T d_k \\ &\leq R_k - \delta c_4 \alpha_k \|g_k\|^2 \\ &\leq R_k + \delta c_4 \left(\frac{2(1-\delta)}{p_3} \right) \left(\frac{g_k^T d_k}{\|d_k\|} \right) \|g_k\|^2 \\ &\leq R_k - \delta c_4^2 \left(\frac{2(1-\delta)}{p_3} \right) \frac{\|g_k\|^4}{\|d_k\|^2} \\ &\leq R_k - \delta c_4^2 \left(\frac{2(1-\delta)}{c_5 p_3} \right) \|g_k\|^2, \end{aligned}$$

this inequality, implies (3.11) with $\Delta = \left(\frac{2\delta(1-\delta)c_4^2}{c_5 p_3}\right)$.
Now, combining the relation in (1.6) and the upper bound in (3.11),

$$\begin{aligned} R_{k+1} &= \frac{\vartheta_k W_k R_k + f(x_{k+1})}{W_{k+1}} \\ &\leq \frac{\vartheta_k W_k R_k + R_k - \Delta\|g_k\|^2}{W_{k+1}} \\ &= \frac{(\vartheta_k W_k + 1)R_k - \Delta\|g_k\|^2}{W_{k+1}} \\ &= R_k - \frac{\Delta\|g_k\|^2}{W_{k+1}}. \end{aligned} \tag{3.12}$$

Recall that f is bounded below and from the result obtained from Lemma 3.6, *i.e.*, $f_k \leq R_k$ for all k, we can conclude that R_k is bounded from below (since f_k is bounded). Thus, it follows from (3.12) that

$$\sum_{k=0}^{\infty} \frac{\|g_k\|^2}{W_{k+1}} < \infty. \tag{3.13}$$

Now, suppose $\|g_k\|$ is to be bounded away from 0, then (3.13) would be violated. To see this assertion, if we assume that, $R_i \leq \varphi_i$, $\forall\ 0 \leq i < k$, from (1.6), with $W_0 = 1$, and $\vartheta_k \in [0,1]$, we have $W_{j+1} = 1 + \sum_{j=0}^{i} \prod_{l=0}^{j} \vartheta_{i-l} \leq i+2$, in a similar manner, we can have $W_{k+1} \leq k+2$.
Therefore, $\liminf_{k\to\infty} \|g_k\| = 0$ holds. Moreover, if strictly $\vartheta_{\max} < 1$, then we will similarly have

$$\begin{aligned} W_{j+1} &= 1 + \sum_{j=0}^{i} \prod_{l=0}^{j} \vartheta_{i-l} \\ &\leq 1 + \sum_{j=0}^{i} \vartheta_{\max}^{j+1} \\ &\leq \sum_{j=0}^{\infty} \vartheta_{\max}^{j} \\ &= \frac{1}{1-\vartheta_{\max}}. \end{aligned} \tag{3.14}$$

This, consequently, means that (3.13) implies (3.10).

□

Now we show the proposed algorithm is linearly convergent under a strong convexity assumption on f.
We can recall that a function, f is strongly convex if there exists a positive scalar, m such that, the following inequality holds

$$f(x) \geq f(y) + \nabla f(y)^T(x-y) + \frac{1}{2m}\|x-y\|^2, \qquad \forall\ x, y \in \mathbb{R}^n. \tag{3.15}$$

Theorem 3.10. *Let f defined in (1.1) be strongly convex such that it has minimizer, say, x^* and the inequalities in Lemmas 3.5 and 3.7 are satisfied. Suppose Assumption 3.2 holds, then with $\vartheta_{\max} < 1$, there exists $\tau \in (0,1)$ such that*

$$f(x_k) - f(x^*) \leq \tau^k (f(x_0) - f(x^*)). \tag{3.16}$$

Proof. Now, using $f(x_{k+1}) \leq R_{k+1} \leq R_k \leq R_{k-1} \leq \ldots \leq R_0 = f(x_0)$, this means that the iterates generated by Algorithm 2 are in the set, ψ, $\forall k$.

By strong convexity assumption on f, it implies that the set, ψ is bounded and the ∇f is Lipschitz continuous on it. Since d_k is bounded (from (3.4)) and the $\|\nabla f\|$ is also bounded on ψ, then $d_{\max} = \sup_k \|d_k\| < \infty$. Now suppose $\bar{\psi}$ represent the collection of $x \in \mathbb{R}^n$ whose distance to ψ is at most $1 \cdot d_{\max}$ (since the step-size, α_k, is upper bounded by 1) and let $\bar{L}$ be a Lipschitz constant for ∇f on the $\bar{\psi}$.

From (3.11) of theorem, 3.9, $f(x_k + \alpha_k d_k) \leq R_k - \Delta \|g_k\|^2$, where, $\Delta = \left(\frac{2\delta(1-\delta)c_4^2}{c_5^2 p_3}\right) \frac{g_k^T d_k}{\|d_k\|^2}$.

Now, using the boundedness of d_k and the upper bound of α_k, we have

$$\begin{aligned} \|x_{k+1} - x_k\| &= \alpha_k \|d_k\|, \\ &\leq 1 \cdot c_5 \|g_k\|, \end{aligned}$$

since ∇f is Lipschitz continuous, we have

$$\begin{aligned} \|g_{k+1} - g_k\| &\leq \bar{\psi} \|x_{k+1} - x_k\|, \\ &\leq \bar{\psi} \cdot 1 \cdot c_5 \|g_k\|. \end{aligned}$$

Thus, we have

$$\begin{aligned} \|g_{k+1}\| &= \|g_{k+1} - g_k + g_k\|, \\ &\leq \|g_{k+1} - g_k\| + \|g_k\|, \\ &\leq \bar{\psi} \|x_{k+1} - x_k\| + \|g_k\|, \\ &\leq \bar{\psi} \cdot 1 \cdot c_5 \|g_k\| + \|g_k\|, \\ &= (1 + \bar{\psi} \cdot 1 \cdot c_5) \|g_k\|. \end{aligned} \tag{3.17}$$

Thus, we aim to show that $R(x_{k+1}) - f(x^*) \leq \tau^k (R(x_k) - f(x^*))$ $\forall k$, in which $\tau = 1 - \Delta c_7 (1 - \vartheta_{\max})$ and $c_7 = \frac{1}{\Delta + \gamma_1 c_6^2}$.

We can easily see that the above definition satisfied (3.16) since, from Lemma 3.7, it has been shown that $R_0 = f(x_0)$ and $f(x_k) \leq R_k$.

Now, suppose $c_7(R_k - f(x^*)) \leq \|g_k\|^2 < c_7(R_k - f(x^*))$, we would show (3.16) in two stages as follows:

Stage 1: Starting with $c_7(R_k - f(x^*)) \leq \|g_k\|^2$. Now, from the update equations in (1.6), we have

$$\begin{aligned} R_{k+1} - f(x^*) &= \frac{\vartheta_k W_k (R_k - f(x^*)) + (f(x_{k+1}) - f(x^*))}{\vartheta_k W_k + 1} \\ &\leq \frac{\vartheta_k W_k (R_k - f(x^*)) + (R_k - \Delta \|g_k\|^2 - f(x^*))}{\vartheta_k W_k + 1} \quad \text{using equation (3.11)} \\ &= R_k - f(x^*) - \frac{\Delta \|g_k\|^2}{W_{k+1}} \\ &\leq R_k - f(x^*) - \Delta(1 - \vartheta_{\max}) \|g_k\|^2 \quad \text{using relation (3.14)} \\ &\leq R_k - f(x^*) - c_7 (R_k - f(x^*)) \Delta (1 - \vartheta_{\max}) \quad \text{since } c_7(R_k - f(x^*)) \leq \|g_k\|^2 \\ &= [1 - c_7 \Delta (1 - \vartheta_{\max})](R_k - f(x^*)). \end{aligned}$$

Hence, from the last equality, the expression (3.11) holds.

Stage 2: Now, with $\|g_k\|^2 < c_7(R_k - f(x^*))$, using convexity property of f *i.e.*, $f(x) - f(x^*) \leq \gamma_1 \|\nabla f\|$, where $\gamma_1 > 0$ (see, [14]) and (3.17) yields

$$\begin{aligned} f(x_{k+1}) - f(x_k) &\leq \gamma_1 \|g_{k+1}\|^2 \\ &\leq \gamma_1 c_6^2 \|g_k\|^2 \\ &\leq \gamma_1 c_6^2 c_7 (R_k - f(x^*)). \end{aligned} \tag{3.18}$$

Now plugging-in (3.18) into $R_{k+1} - f(x^*) = \frac{\vartheta_k W_k (R_k - f(x^*)) + (f(x_{k+1}) - f(x^*))}{\vartheta_k W_k + 1}$, gives

$$\begin{aligned} R_{k+1} - f(x^*) &= \frac{\vartheta_k W_k (R_k - f(x^*)) + (f(x_{k+1}) - f(x^*))}{\vartheta_k W_k + 1} \\ &\leq \frac{\vartheta_k W_k (R_k - f(x^*)) + \gamma_1 c_6^2 c_7 (R_k - f(x^*))}{\vartheta_k W_k + 1} \\ &= \frac{[\vartheta_k W_k + \gamma_1 c_6^2 c_7](R_k - f(x^*))}{W_{k+1}} \quad \text{using (1.6)}. \end{aligned} \tag{3.19}$$

Re-writing the relation, $c_7 = \frac{1}{\Delta + \gamma_1 c_6^2}$ as $\gamma_1 c_6^2 c_7 = 1 - \Delta c_7$ and then substituting it into (3.19), yields

$$\begin{aligned} R_{k+1} - f(x^*) &= \frac{[\vartheta_k W_k + \gamma_1 c_6^2 c_7](R_k - f(x^*))}{Q_{k+1}} \\ &= \left(1 - \frac{\Delta c_7}{W_{k+1}}\right)(R_k - f(x^*)) \\ &\leq [1 - \Delta c_7 (1 - \vartheta_{\max})](R_k - f(x^*)) \quad \text{using relation (3.14)}. \end{aligned} \tag{3.20}$$

Thus, the proof of $R(x_{k+1}) - f(x^*) \leq \tau^k (R(x_k) - f(x^*))\ \forall k$, is achieved and hence (3.16) is obtained.

□

4. Numerical experiment and discussion

In this section, we describe the proposed algorithms' performance numerically in comparison with other proposals from the literature. Next, we report and discuss the algorithms' performance on some benchmark test problems, and portray the result obtained from one of the proposed algorithms when solving some fitting data task.

We compare the proposed SDMSC algorithms with GSDA in [19] and SDHAM in [10]. The numerical experiments were conducted on the literature's large-scale and small-scale benchmark test problems. the references where these problems were taken are cited respectively for each problem we have considered, and each of the large-scale problems consists of the following dimensions namely: 3000, 6000, 9000, 12000, 15000. The algorithms were implemented in a MATLAB package installed in a personal computer PC with the following specifications: 1.60 GHz speed, Intel Core i5-8265U, and 8 GB - RAM. In the implementation of these algorithms, we initialize the following set of parameters:

1. SDMSC Algorithms: $\eta = 1E+30$ $\epsilon = 1E-4$, $\delta = 1E-3$, $tol = 1E-5$, $\vartheta_{\min} = 0.1$, $\vartheta_{\max} = 0.85$, $k_{\max} = 1000$, and ϑ_k is taken as
   - SDMSC1: $\vartheta_k = 0.85\ \forall k$ - a non-monotone line-search.
   - SDMSC2: $\vartheta_k = 0\ \forall k$ - Armijo line-search with backtracking strategy.
2. GSDA Algorithm: the parameters used are the same as reported in [19].
3. SDHAM Algorithm: similarly, the parameters were taken from [10].

Moreover, the stopping criterion given by $\|g_k\| \leq tol = 1E-5$ is the same for all the algorithms, and each problem under consideration is minimized from the standard starting point as reported in Table 1. During the execution of these algorithms on the set of problems, the following information that consists of a number of iteration ($\#niter$), number of function evaluations ($\#nfvals$), number of gradient evaluations ($\#nmvp$), and CPU were recorded, these serve as metrics of comparison between algorithms. The recorded data is reported in tables, which can be viewed through the following link `https://github.com/MAHMOUDPD/SDMSC_project` and

TABLE 1. Checklist of benchmark test functions, and with initial points, x_0. respectively.

Prblms	Names	Initial point, x_0
Large Scale		
P1	Penalty function [8]	$(3, 3, \dots, 3)^T$
P2	Trignometric function [8]	$(1, 1, \dots, 1)^T$
P3	Discrete boundary-valued [8]	$(\frac{1}{n+1}(\frac{1}{n+1} - 1), \dots, \frac{1}{n+1}(\frac{n}{n+1} - 1))^T$
P4	Linear full rank function [13]	$(1, 1, \dots, 1)^T$
P5	Problem 202 [9]	$(2, 2, \dots, 2)^T$
P6	Problem 206 [9]	$\frac{1}{n} * (1, 1, \dots, 1)^T$
P7	Problem 212 [9]	$\frac{1}{2} * (1, 1, \dots, 1)^T$
P8	Strictly convex function (Raydan 1) [15]	$\frac{1}{n} * (1, 1, \dots, 1)^T$
P9	Strictly convex function (Raydan 2) [15]	$(1, 1, \dots, 1)^T$
P10	Sine function 2 [8]	$(1, 1, \dots, 1)^T$
P11	Exponential function 1 [8]	$\frac{n}{n-1} * (1, 1, \dots, 1)^T$
P12	Logarithm function [8]	$(1, 1, \dots, 1)^T$
P13	Extended Powell singular function [8]	$1.5E - 4 * (1, 1, \dots, 1)^T$
P14	Function 21 [8]	$\frac{6}{5}(-1, -1, \dots, -1)^T$
P15	Extended Rosenbrock function [13]	$repmat([-1; 1], [(n/2), 1])$
P16	Extended Himmelblau function [6]	$(1, 1/n, 1, 1/n, \dots, 1, 1/n)^T$
P17	Function 27 [8]	$(1, 1, \dots, 1)^T$
P18	Triglog function [6]	$(1, 1, \dots, 1)^T$
P19	Zero Jacobian function [8]	$\left(\frac{(100(n-100)}{n}, \dots, \frac{(n-500)(n-1000)}{(60n)^2}\right)^T$
P20	Exponential function [8]	$(1, 1, \dots, 1)^T$
P21	Brown almost linear function [13]	$\frac{1}{n} * (1, 1, \dots, 1)^T$
Small Scale		
P22	Brown function [12]	$(1, 1)^T$
P23	Jennrich and Sampson function [4]	$(0.3, 0.4)^T$
P24	Box three–dimensional function [13]	$(0, 10, 20)^T$
P25	Rank deficient function [13]	$(-1, 1)^T$
P26	Rosenbrock function [13]	$(-1, 1)^T$
P27	Parameterized function [5]	$(10, 10)^T$
P28	Freudenstein and Roth function [13]	$(0.5, -2)^T$
P29	Beale function [13]	$(1, 1)^T$

as indicated in the tables, a problem has F for an algorithm whenever the stopping criterion fails and/or the maximum number of iterations, named $k_{\max}$ is exceeded, and no solution is obtained.

By simply skimming through the tables, one can easily see that the proposed SDMSC algorithm was able to solve all the problems except for some instances in problem P12; in contrast, GSDA reported quite some failure cases (although most of which are small-scale problems), as can be seen in problems P9, P14, P15, P22, P24, P26, P29, P28 and similarly, SDHAM algorithm has recorded failures in problems, P20, P21. This obviously underscores the efficiency, robustness, and usefulness of SDMSC algorithms over SDHAM and GSDA. To visualize the results in the tables `https://github.com/MAHMOUDPD/SDMSC_project` and illustrate the performance of these algorithms, we adopted the well-known Dolan-Moré tool.

This performance profile indicates as follows: given a set of optimization problems, say, P and a set of algorithms, say, S. The tool compares a problem $p \in P$ over a particular algorithm, say, $s \in S$. The idea here is if $\tau_{p,s}$ is a result of one of the metrics obtained from solving a given problem using an algorithm, then the

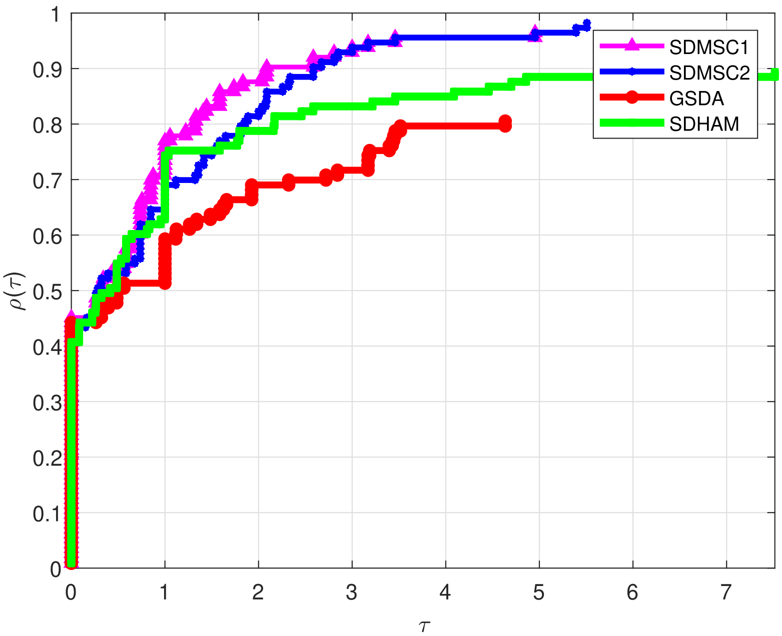

FIGURE 1. Plot based on $\#niter$.

performance ratio is defined as $\tau^*_{p,s} = \frac{\tau_{p,s}}{\min\{\tau_{p,s}:\ s\in S\}}$. It is assumed from this definition that $\tau^*_{p,s}$ is in the interval $[1, f]$, with $f > 1$; however, if $\tau^*_{p,s}$ is twice the largest value in the metric under consideration, then we can conclude that a solver, say $s \in S$, failed to solve a problem, say $p \in P$. A graph is plotted between the probability, $\rho(\tau_{p,s})$, on the $y - axis$ and $\tau^*_{p,s} \in [1, f]$ on the $x - axis$, with log_2 scale. The lowest and best performance ratio recorded by an algorithm is 1, and the algorithm with the highest number of $1'$s is located in the left-most part of the figures. The most efficient algorithm is indicated/represented by the top-most curve in a figure and most likely to have the highest probability as $f \to +\infty$, and consequently, must be preferred.

Furthermore, we discuss and/or portray the behaviour of the result obtained by the proposed $SDMSC1$ algorithm for some nonlinear least-squares data–fitting. The solution obtained is the minimizer of the objective function (1.1), where the residual norm $\|r(x)\|$ is minimized so that we can realize the *"best-fit curve"* of the data points. If the nonlinear mathematical models say, $m(x, t)$ closely fit most of the data points, then the residual vector $r(x^*)$ is small or approximately close to zero. This problem is known as the small residual problem. Otherwise, it is known as a large-scale residual problem. In what follows, we outline the model functions and briefly detailed descriptions.

1. Gaussian function [13]
   This NLS data-fitting problem has $n = 3$ parameters and $m = 15$ data points or residual functions. The proposed algorithm used to solve this Gaussian model converges to the minimizer, x^*. Then, this solution is used to fit a nonlinear model stated as follows:

$$\mathbf{m}(x^*, t) = x_1^* e^{\frac{1}{2} x_2^* (\theta_t - x_3^*)^2}$$

2. Osborne II function [13]
   The Osborne II function is yet another data-fitting problem with $n = 11$ parameters and $m = 65$ residual functions or data points. These parametric values or solutions, say, x^* obtained using the proposed algorithm

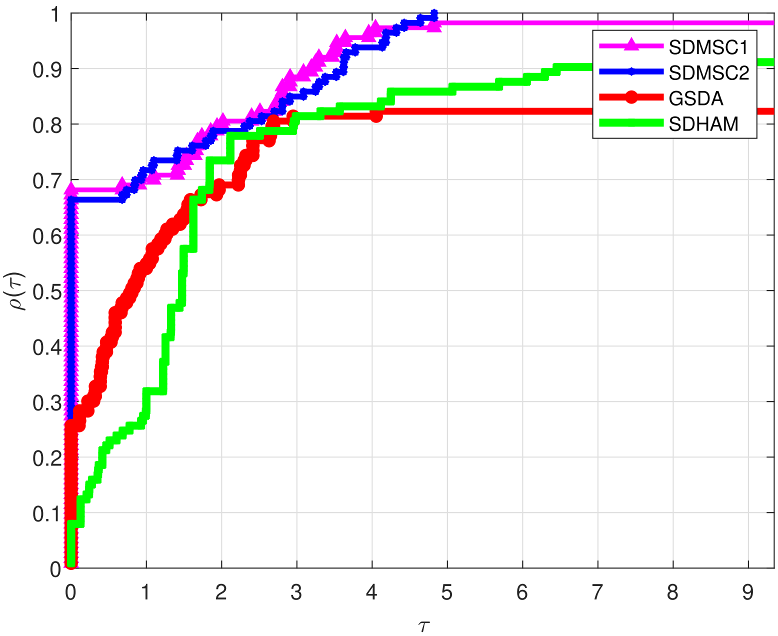

FIGURE 2. Plot based on $\#nfvals$.

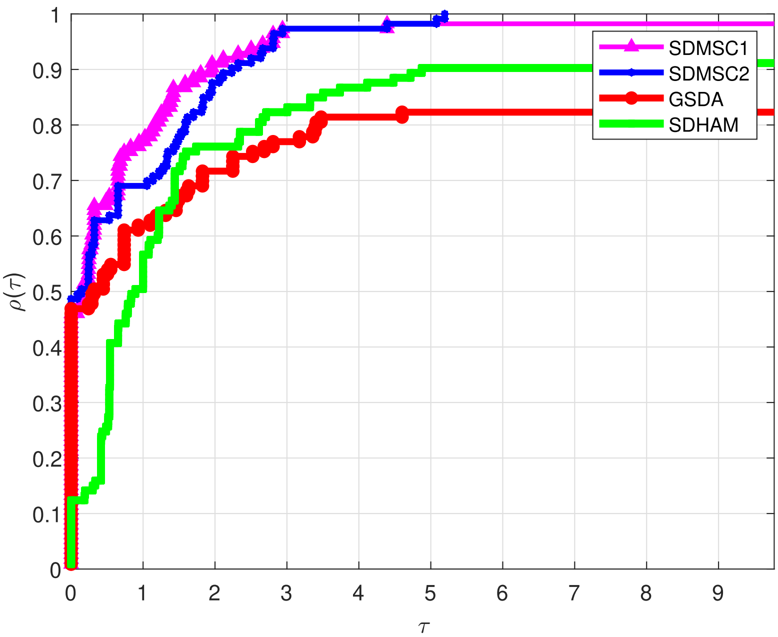

FIGURE 3. Plot based on $\#nmvp$.

and then used to fit the nonlinear model given as follows:

$$\mathbf{m}(x^*, t) = x_1^* e^{-x_5^*(t-1)} + x_2^* e^{-x_6^*(\theta_t - x_9^*)^2} + x_3^* e^{-x_7^*(\theta_t - x_{10}^*)^2} + x_4^* e^{-x_8^*(\theta_t - x_{11}^*)^2}$$

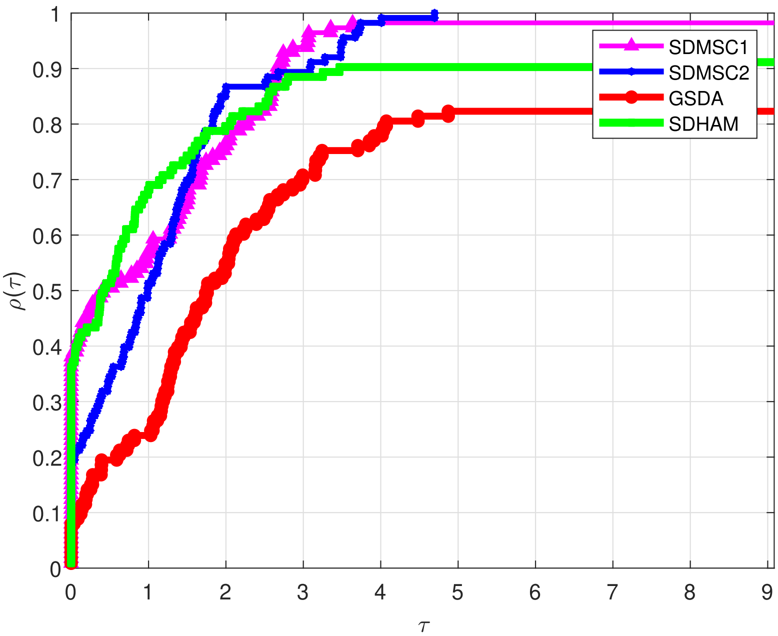

FIGURE 4. Plot based on CPU.

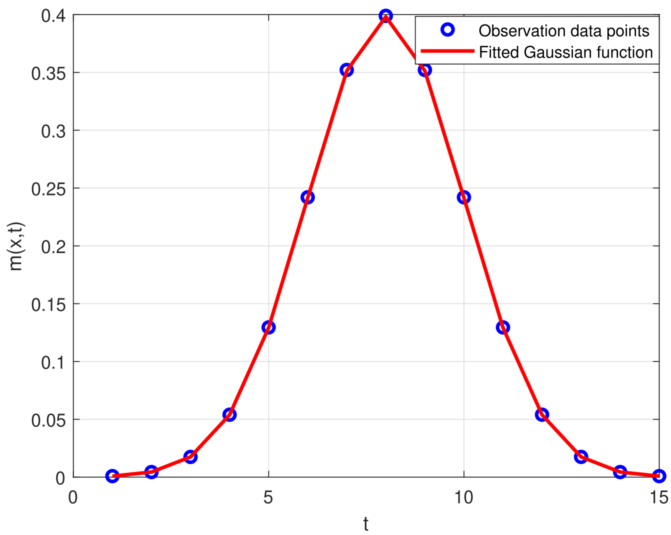

FIGURE 5. A nonlinear fit of a Gaussian function, where the red line represents the Gaussian function with $n = 3$ number of parameters and $m = 15$ data points.

where $\theta_t = \frac{8-t}{2}$ for Gaussian model and $\theta_t = \frac{t-1}{10}$ for the Osborne II model. It can be seen from Figures 5–6 that the least square fit of the Gaussian function and Osborne II function. The fitting models provide a good fit of the data points in all the situations.

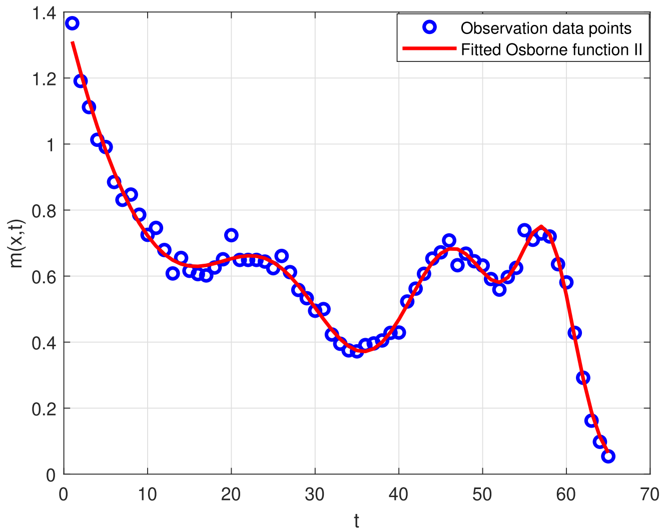

FIGURE 6. A nonlinear fit of Osborne function II where the red line represents the Osborne function with $n = 11$ number of parameters/variables and $m = 65$ data points.

Finally, we can observe from the Figures 5–6 that the proposed algorithm obtained the solution/parameters that best fit the model. This is evident from the *'nearest'* or *'closeness'* of the data points to the model's fitting curves, implying that the residual is almost zero.

5. Conclusion

The paper developed a structured approximation of a diagonal matrix-based approach for solving nonlinear least-squares. We called the algorithm *Structure Diagonal based upon Modified Secant Condition(SDMSC)*. The correction matrix with diagonal entries was derived based upon minimizing the *least change secant* strategy subject to a modified structured secant condition. Also, the algorithm is matrix-free in its implementation and less complicated; this simplicity gives rise to low computational expenses in every iteration process. Moreover, the global convergence of the proposed $SDMSC$ under some standard assumptions and linear rate of convergence for strongly convex functions coupled with a non-monotone line-search approach of Zhang and Hager [21] was proved. Finally, we have shown that the proposed method was comparatively more efficient numerically than other proposed algorithms [10] and [19]. Equally importantly, we successfully applied $SDMSC$ to some data fitting problems.

Acknowledgements

The authors appreciate valuable comments and suggestions by the anonymous reviewers, which have substantially improved the presentation of the paper. More so, the authors acknowledge the financial support provided by the Petchra Pra Jom Klao Scholarship of King Mongkut's University of Technology Thonburi (KMUTT) and Center of Excellence in Theoretical and Computational Science (TaCS-CoE), KMUTT. In addition, the first author was supported by the Petchra Pra Jom Klao Ph.D. Research Scholarship from King Mongkut's University of Technology Thonburi (Contract No. 24/2565). Moreover, this research project is supported by King Mongkut's University of Technology Thonburi (KMUTT), Thailand Science Research and Innovation (TSRI), and National Science, Research and Innovation Fund (NSRF) Fiscal year 2024 Grant number FRB670073/0164.

Data availability statement

The link leads to an Excel file containing the recorded data for the runs of four(4) algorithms: SDMSC1, SDMSC2, GSDA and SDHAM. The data includes four(4) metrics for each algorithm: the number of iterations (#niter), the number of function evaluations(#nfevals), the number of gradient evaluations(#nmvp), and CPU time [20].

References

[1] A.B. Abubakar, P. Kumam, M. Malik and A.H. Ibrahim, A hybrid conjugate gradient based approach for solving unconstrained optimization and motion control problems. *Math. Comput. Simul.* (2021).

[2] A.M. Awwal, P. Kumam, L. Wang, M.M. Yahaya and H. Mohammad, On the barzilai–borwein gradient methods with structured secant equation for nonlinear least squares problems. *Optim. Methods Softw.* (2020) 1–20.

[3] R. Dehghani and N. Mahdavi-Amiri, Scaled nonlinear conjugate gradient methods for nonlinear least squares problems. *Numer. Algorithms* (2018) 1–20.

[4] D.S. Gonçalves and S.A. Santos, Local analysis of a spectral correction for the gauss-newton model applied to quadratic residual problems. *Numer. Algorithms* **73** (2016) 407–431.

[5] J. Huschens, On the use of product structure in secant methods for nonlinear least squares problems. *SIAM J. Optim.* **4** (1994) 108–129.

[6] M. Jamil and X.-S. Yang, A literature survey of benchmark functions for global optimization problems. Preprint: arXiv:1308.4008 (2013).

[7] M. Kobayashi, Y. Narushima and H. Yabe, Nonlinear conjugate gradient methods with structured secant condition for nonlinear least squares problems. *J. Comput. Appl. Math.* **234** (2010) 375–397.

[8] W. La Cruz, C. Venezuela, J.M. Martínez and M. Raydan, Spectral residual method without gradient information for solving large-scale nonlinear systems of equations: Theory and experiments. *P. Optim.* **76** (2004) 79–1008.

[9] L. Lukšan and J. Vlcek, *Test problems for unconstrained optimization.* Academy of Sciences of the Czech Republic, Institute of Computer Science, Technical Report (2003) 897.

[10] H. Mohammad and S.A. Santos, A structured diagonal hessian approximation method with evaluation complexity analysis for nonlinear least squares. *Comput. Appl. Math.* **37** (2018) 6619–6653.

[11] H. Mohammad, M.Y. Waziri and S.A. Santos, A brief survey of methods for solving nonlinear least-squares problems. *Numer. Algebra Control Optim.* **9** (2019) 1–13.

[12] J. Momin and Y. Xin-She, A literature survey of benchmark functions for global optimization problems. *J. Math. Model. Numer. Optim.* **4** (2013) 150–194.

[13] J.J. Moré, B.S. Garbow and K.E. Hillstrom, Testing unconstrained optimization software. *ACM Trans. Math. Softw.* **7** (1981) 17–41.

[14] J. Nocedal and S. Wright. Numerical Optimization. Springer Science & Business Media (2006).

[15] M. Raydan, The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM J. Optim.* **7** (1997) 26–33.

[16] N. Salihu, P. Kumam, A.M. Awwal, I.M. Sulaiman and T. Seangwattana, The global convergence of spectral rmil conjugate gradient method for unconstrained optimization with applications to robotic model and image recovery. *Plos one* **18** (2023) e0281250.

[17] M.M. Yahaya, P. Kumam, A.M. Awwal and S. Aji, A structured quasi–newton algorithm with nonmonotone search strategy for structured nls problems and its application in robotic motion control. *J. Comput. Appl. Math.* (2021) 113582.

[18] M.M. Yahaya, P. Kumam, A.M. Awwal and S. Aji, Alternative structured spectral gradient algorithms for solving nonlinear least-squares problems. *Heliyon* (2021) e07499.

[19] M.M. Yahaya, P. Kumam, A.M. Awwal, P. Chaipunya, S. Aji and S. Salisu, A new generalized quasi-newton algorithm based on structured diagonal hessian approximation for solving nonlinear least-squares problems with application to 3dof planar robot arm manipulator. *IEEE Access* **10** (2022) 10816–10826.

[20] M.M. Yahaya, P. Kumam, P. Chaipunya, A.M. Awwal and L. Wang, Matlab code and Experimental results for "On diagonally structured scheme for nonlinear least squares and data-fitting problems" (2022). https://github.com/MAHMOUDPD/SDMSC_project.

[21] H. Zhang and W.W. Hager, A nonmonotone line search technique and its application to unconstrained optimization. *SIAM J. Optim.* **14** (2004) 1043–1056.