On three-term conjugate gradient method for optimization problems with applications on COVID-19 model and robotic motion control

Ibrahim Mohammed Sulaiman, Maulana Malik, Aliyu Muhammed Awwal, Poom Kumam, Mustafa Mamut, and Shadi Al-Ahmad

Abstract
The three-term conjugate gradient (CG) algorithms are among the efficient variants of CG algorithms for solving optimization models. This is due to their simplicity and low memory requirements. On the other hand, the regression model is one of the statistical relationship models whose solution is obtained using one of the least square methods including the CG-like method. In this paper, we present a modification of a three-term conjugate gradient method for unconstrained optimization models and further establish the global convergence under inexact line search. The proposed method was extended to formulate a regression model for the novel coronavirus (COVID-19). The study considers the globally infected cases from January to October 2020 in parameterizing the model. Preliminary results have shown that the proposed method is promising and produces efficient regression model for COVID-19 pandemic. Also, the method was extended to solve a motion control problem involving a two-joint planar robot.

MSC: 90C30; 90C06; 90C56

Keywords: Finite difference; Three-term CG algorithms; Optimization models; Motion control; Line search procedure; Coronavirus (COVID-19); Regression analysis

1 Introduction
Consider the following optimization model:

\[ \min f(x), \quad x \in \mathbb{R}^n, \] (1.1)

where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is a smooth function whose gradient \( \nabla f(x) = g(x) \) is available. Problems of the form (1.1) can be traced to many professional fields of science, astronomy, engineering, economics, and many more (see, for example, [1, 2]). Throughout this paper, we shall abbreviate \( g(x_k) \) and \( f(x_k) \) by \( g_k \) and \( f_k \), respectively. Also, \( \| \cdot \| \) represents the Euclidean norm of vectors.
The nonlinear CG methods play an important role in solving large-scale optimization models due to the modesty of their memory requirements and nice convergence properties. Generally, the iterates of the CG methods are usually determined through the following recursive computational scheme:

\[ x_{k+1} = x_k + s_k, \quad s_k = t_k d_k, \quad k \geq 0, \]  

(1.2)

where \( t_k \) is the step-size computed along the search direction \( d_k \). For the first iteration, \( d_0 \) is always the steepest descent direction, that is, \( d_0 = -g_0 \) [3]. However, subsequent directions are recursively determined by

\[ d_k = -g_k + \beta_k d_{k-1}, \quad k \geq 1, \]  

(1.3)

where the scalar \( \beta_k \) is known as the CG coefficient whose different form determines a different CG methods.

The following line search procedures have been used in the convergence analysis and implementations of the already existing CG methods [4]. The convergence analysis often requires the line search to satisfy the exact line search, the Wolfe or strong Wolfe (SWP) line search. The exact line search requires the step-size \( t_k \) to satisfy

\[ f(x_k + t_k d_k) := \min_{t \geq 0} f(x_k + td_k). \]  

(1.4)

The standard line search requires computing \( t_k \) such that the cost function is minimized along \( d_k \) satisfying

\[ f(x_k + t_k d_k) \leq f(x_k) + \delta t_k g_k^T d_k, \]  

(1.5)

\[ g(x_k + t_k d_k)^T d_k \geq \sigma g_k^T d_k. \]  

(1.6)

The SWP is to compute \( t_k \) satisfying (1.5) and

\[ g(x_k + t_k d_k)^T d_k \leq -\sigma \| g_k \| d_k, \]  

(1.7)

where \( 0 < \delta < \sigma < 1 \).

Presently, there are several known formulas for different CG parameters (see [4–10]). One of the most efficient algorithms among the well-known formulas is the PRP [7, 8] defined by

\[ \beta_{PRP k} = \frac{g_k^T y_{k-1}}{\| g_{k-1} \|^2}, \]  

(1.8)

where \( y_{k-1} = g_k - g_{k-1} \). From the computational point of view, the PRP algorithm performs better than most CG algorithms, and the convergence result has been established under some line search procedures. However, for a general function, the PRP method fails with regard to the global convergence under the Wolfe line search procedure. This is because the direction of search \( d_k \) is not descent for a general objective function [4]. This problem inspired numerous researchers to study the global convergence of PRP method under inexact line search. Interestingly, considering the general function, Yuan et al. [11]
proved the global convergence of PRP method using a modified Wolfe line search procedure. More practical approaches of the line search have been employed to identify a step-size capable of achieving adequate reduction in the objective function \( f(x) \) at minimal cost.

Recently, Rivaie et al. [12] proposed a variant of PRP method by replacing the term \( \|g_{k-1}\|^2 \) in the denominator of PRP with \( \|d_{k-1}\|^2 \) as follows:

\[
\beta_{k}^{\text{RMIL}} = \frac{g_k^T y_{k-1}}{\|d_{k-1}\|^2}, \tag{1.9}
\]

and showed that the method converges globally under the exact line search. However, Dai [13] pointed out a wrong inequality used in the convergence result of RMIL method and suggested some necessary corrections as follows:

\[
\beta_{k}^{\text{RMIL}+} = \begin{cases} 
\frac{g_k^T y_{k-1}}{\|d_{k-1}\|^2}, & \text{if } 0 \leq g_k^T g_{k-1} \leq \|g_k\|^2, \\
0, & \text{otherwise},
\end{cases} \tag{1.10}
\]

and further established the global convergence under the exact line search. Preliminary results have been presented using the same benchmark test problems with different initial guess to illustrate the efficiency of the modified method. More recently, Yousif [14] modified the work of Dai [13] and showed that \( \text{RMIL}^+ \) converges globally under the strong Wolfe line search. For more reference on the convergence analysis of the CG method, please refer to the following references [15–19].

It is worthy to note that the sufficient descent property

\[
g_k^T d_k \leq \lambda \|g_k\|^2, \quad \lambda > 0, \tag{1.11}
\]

plays a crucial role in the convergence analysis of the CG methods including the RMIL method. In this regard, several variants of the CG methods have been defined to satisfy (1.11) independent of the line search technique used.

One of the efficient variants of the CG methods is the three-term CG method where the search direction \( d_k \) contains three terms. One of the classical three-term methods was proposed by Beale [20], using the coefficient \( \beta_k^{\text{HS}} \) [5]. The author constructed a new direction of search as follows:

\[
d_k = -g_k + \beta_k d_{k-1} + \gamma_k d_t,
\]

where \( d_t \) is the restart direction and

\[
\gamma_k = \begin{cases} 
0, & \text{if } k = t + 1, \\
\frac{g_k^T y_t}{d_t^T g_t}, & \text{if } k > t + 1.
\end{cases}
\]

The performance of this method was improved using an efficient restart strategy developed by McGuire [21]. The first three-term PRP algorithm (TTPRP) was defined by Zhang et al. [22] with the formula given as

\[
d_k = -g_k + \beta_k d_{k-1} + \theta_k y_{k-1},
\]
where βₖ is the PRP method defined in (1.8) and θₖ = \( \frac{\mathbf{g}_k^T \mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^T \mathbf{d}_{k-1}} \). An attractive feature of this method is that the descent condition

\[ \mathbf{g}_k^T \mathbf{d}_k \leq -\|\mathbf{g}_k\|^2, \]

holds independent of any line search, and the global convergence was established under a modified Armijo line search. Based on the structure of TTPRP, Liu et al. [23] extended the coefficient of RMIL (1.9) to define a three-term CG method known as TTRMIL with formula as follows:

\[ d_0 = -\mathbf{g}_0, \quad d_k = -\mathbf{g}_k + \beta_k \mathbf{d}_{k-1} + \theta_k \mathbf{y}_{k-1}, \quad k \geq 1, \quad (1.13) \]

where \( \beta_k \) is defined by (1.9) and \( \theta_k = \frac{\mathbf{g}_k^T \mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^T \mathbf{d}_{k-1}} \).

The global convergence of this method was proved under the standard Wolfe line search. However, the proposed TTRMIL method in (1.13) employed the RMIL method; Dai [13] pointed out some errors in the convergence result and suggested some correction given in [14]. Motivated by this, we propose a modification of TTRMIL in the next section. For more references about the three-term CG method, interested readers may refer to [24–27].

The rest of the paper would be structured as follows. In the next section, a modified TTRMIL method is given with its algorithm. The sufficient descent property and the global convergence of the new modification are studied in Sect. 3. Preliminary results based on some unconstrained optimization problems are presented to illustrate the performance of the method in Sect. 4. The proposed modification was extended to formulate a parameterized model for cases of COVID-19 in Sect. 5. In Sect. 6, the application in motion control is presented. Finally, the concluding remark and some recommendations of the study are presented in Sect. 7.

2 TTRMIL+ method and its algorithm

Motivated by the comments made by Dai [13] on the convergence of RMIL method, as discussed in the preceding section, we propose a modified TTRMIL, named TTRMIL+, by replacing \( \beta_k \) in (1.13) with the \( \beta_k \) given in (1.10) as follows:

\[ d_k = \begin{cases} 
-\mathbf{g}_k, & k = 0, \\
-\mathbf{g}_k + \beta_k \mathbf{d}_{k-1} + \theta_k \mathbf{y}_{k-1}, & k \geq 1,
\end{cases} \quad (2.1) \]

where

\[ \theta_k = \frac{\mathbf{g}_k^T \mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^T \mathbf{d}_{k-1}}. \quad (2.2) \]

From (1.13) and (2.2), it is obvious that the difference between these two methods is the CG parameter \( \beta_k \) employed by each method in defining their search directions \( \mathbf{d}_k \). This is a little change that has a great impact in the convergence analysis of RMIL+. It is interesting to note that the TTRMIL+ reduces to the classical RMIL+ method under the exact minimization condition. The following algorithm describes the proposed TTRMIL+.
Algorithm 1 The modified TTRMIL+ algorithm.
Stage 0. Given \( x_0 \in \mathbb{R}^n, d_0 = -g_0 = -\nabla f_0 \), set \( k := 0 \).
Stage 1. Check if \( \|g_k\| \leq \epsilon \), then stop.
Stage 2. Compute \( t_k \) using (1.5) and (1.6).
Stage 3. Update the new point based on (1.2). If \( \|g_k\| \leq \epsilon \), terminate the process.
Stage 4. Compute \( \beta_k \) by (1.10) and update \( d_k \) by (2.1).
Stage 5. Go to Stage 2 with \( k := k + 1 \).

The following assumptions are very important and usually required in the convergence analysis of most CG algorithms.

Assumption 2.1
(A1) The level set \( \Omega_1 = \{ x \in \mathbb{R}^n | f(x) \leq f(x_0) \} \) is bounded, where \( x_0 \) is an arbitrary initial point.
(A2) In some neighborhood \( N \) of \( \Omega_1 \), \( f \) is smooth and \( g(x) \) is Lipschitz continuous on an open convex set \( N \) that contains \( \Omega_1 \) such that there exists \( L > 0 \) (constant) satisfying
\[
\|g(x) - g(y)\| \leq L \|x - y\|, \quad \forall x, y \in N. \tag{2.3}
\]

From Assumption 2.1 and \([16,28]\), it implies that there exist positive constants \( \gamma \) and \( b \) such that
\[
\|g(x_k)\| \leq \gamma, \quad \forall x_k \in \Omega, \tag{2.4}
\]
\[
\|x - y\| \leq b, \quad \forall x, y \in \Omega. \tag{2.5}
\]

But the function \( f(x) \) decreases as \( k \to +\infty \), hence, from Assumption 2.1, the sequence \( \{x_k\} \) generated by Algorithm 1 is said to be contained in a bounded region. This implies that the sequence \( \{x_k\} \) is bounded.

The convergence analysis of the new method would be studied in the next section.

3 Convergence analysis
In this section, we establish the sufficient descent condition and global convergence properties of the proposed TTRMIL+ method.

The following theorem indicates that the search direction of TTRMIL+ method satisfies the sufficient descent condition.

Theorem 3.1 Suppose that the sequence \( \{x_k\} \) is generated by Algorithm 1. The search direction \( d_k \) defined by (2.1) with \( \beta_k = \beta_{RMIL+}^k \) (1.10) satisfies the sufficient descent condition (1.12).

Proof We will prove by induction. For \( k = 0 \) and from (2.1), we have \( g_0^T d_0 = -\|g_0\|^2 \), so that the sufficient descent condition (1.12) is satisfied. Suppose that (1.12) is true for \( k - 1 \), that is, \( g_{k-1}^T d_{k-1} = -\|g_{k-1}\|^2 \). According to the value of \( \beta_{RMIL+}^k \) (1.10), we have two cases.
• Case 1: $\beta_k^{RMIL+} = 0$. Since (1.6), (2.1), (2.2), and $g_k^T g_{k-1} > \|g_k\|^2$ hold, we have

$$g_k^T d_k = -g_k^T g_{k-1} - \frac{g_k^T d_{k-1}}{\|d_{k-1}\|^2} g_k^T y_{k-1}$$

$$\leq -\|g_k\|^2 + \sigma \frac{\|g_{k-1}\|^2}{\|d_{k-1}\|^2} g_k^T y_{k-1}$$

$$= -\|g_k\|^2 + \sigma \frac{\|g_{k-1}\|^2}{\|d_{k-1}\|^2} (\|g_k\|^2 - g_k^T g_{k-1})$$

$$\leq -\|g_k\|^2.$$

• Case 2: $\beta_k^{RMIL+} = \frac{g_k^T y_{k-1}}{\|d_{k-1}\|^2}$. From (2.1) and (2.2), we get

$$g_k^T d_k = -\|g_k\|^2 + \frac{g_k^T y_{k-1}}{\|d_{k-1}\|^2} g_k^T d_{k-1} - \frac{g_k^T d_{k-1}}{\|d_{k-1}\|^2} g_k^T y_{k-1} = -\|g_k\|^2.$$

Hence, the search direction $d_k$ defined by the TTRMIL+ method always satisfies the sufficient descent condition (1.12).

□

Remark 3.2 Since the proposed method satisfies the sufficient descent condition (1.12), then, for all $k \geq 0$, we have

$$\|d_k\| \geq \|g_k\|.$$ (3.1)

Now, we will establish the global convergence of the TTRMIL+ method by first providing the following lemma to show that the standard Wolfe line search gives a lower bound for the step-size $t_k$ as follows.

Lemma 3.3 Suppose that the sequence $\{x_k\}$ is generated by Algorithm 1, where $d_k$ is a descent direction and Assumption 2.1 holds. If $t_k$ is calculated by standard Wolfe line search (1.5) and (1.6), then we have

$$t_k \geq \frac{(1 - \sigma) \|g_k\|^2}{L \|d_k\|^2}. (3.2)$$

Proof From the standard Wolfe condition (1.6) and by subtracting $g_k^T d_k$ in the both sides, and using Lipschitz continuity (2.3), we get

$$(\sigma - 1) g_k^T d_k \leq (g_{k+1} - g_k)^T d_k$$

$$\leq \|g_{k+1} - g_k\| \|d_k\|$$

$$\leq L \|x_{k+1} - x_k\| \|d_k\|$$

$$= L t_k \|d_k\|^2.$$

Since $d_k$ is a descent direction and also $\sigma < 1$, that implies (3.2) is true.

□

The following lemma is the Zoutendijk condition [29], which plays an important role in the analysis of the global convergence properties for CG method.
Lemma 3.4 Let Assumption 2.1 hold and \( d_k \) be generated by (1.10), (2.1), and (2.2), where \( t_k \) is calculated by the standard Wolfe line search (1.5) and (1.6). Then
\[
\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \tag{3.3}
\]

Proof From the standard Wolfe condition (1.5) and (3.2), we have
\[
f(x_k) - f(x_k + t_k d_k) \geq -\delta t_k g_k^T d_k \geq \delta \frac{(1 - \sigma)(g_k^T d_k)^2}{L\|d_k\|^2}.
\]
Hence, from Assumption (2.1), we get the Zoutendijk condition (3.3) and hence the proof. □

We present a global convergence results of the proposed TTRMIL+ CG method using the standard Wolfe line search.

Theorem 3.5 Suppose that the sequence \( \{x_k\} \) is generated by Algorithm 1, we have
\[
\lim_{k \to \infty} \inf \|g_k\| = 0. \tag{3.4}
\]

Proof Suppose by contradiction that (3.4) is not true. Then \( \forall k \geq 0 \), we can find a positive constant \( c \) so that
\[
\|g_k\| \geq c. \tag{3.5}
\]
Here, we have two cases.

• Case 1: If \( \beta^\text{RMIL+}_k = 0 \), then based on the Cauchy–Schwarz inequality and from (2.1), (2.2), (2.3), (2.4), (2.5), (3.1), and (3.5), we get
\[
\|d_k\| = \| - g_k + \theta_k y_{k-1} \|
= \left\| - g_k - \frac{g_k^T d_{k-1}}{d_{k-1}^T d_{k-1}} y_{k-1} \right\|
\leq \|g_k\| + \frac{\|g_k\| \|d_{k-1}\| \|y_{k-1}\|}{\|d_{k-1}\|^2}
\leq \gamma \frac{\|g_k\| L \|x_k - x_{k-1}\|}{\|d_{k-1}\|}
\leq \gamma \frac{\|g_k\| L b}{\|d_{k-1}\|}
\leq \gamma \frac{\|g_k\| L b}{c} \triangleq \nu. \tag{3.6}
\]
Furthermore, by using (1.12), (3.5), and (3.6), we obtain
\[
\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \geq \sum_{k=0}^{\infty} \frac{\|g_k\|^4}{\|d_k\|^2} \geq \sum_{k=0}^{\infty} \frac{c^4}{\nu^2} = +\infty.
\]
This is a contradiction with (3.3). Hence, (3.4) holds.

- **Case 2:** If $\beta_k^{\text{RMIL}+} = \beta_k^{\text{RMIL}}$, then based on the Cauchy–Schwarz inequality and from (1.9), (2.1), (2.2), (2.3), (2.4), (2.5), and (3.1), we obtain

$$
\| d_k \| = \left\| -g_k + \beta_k^{\text{RMIL}} d_{k-1} + \theta_k y_{k-1} \right\|
\leq \| g_k \| + \left\| \frac{g_k^T y_{k-1}}{\|y_{k-1}\|^2} d_{k-1} \right\| + \left\| -\frac{g_k^T d_{k-1}}{d_{k-1}^T d_{k-1}} y_{k-1} \right\|
\leq \| g_k \| + \frac{\| g_k \| \| g_k - g_{k-1} \| \| d_{k-1} \|}{\| d_{k-1} \|^2} + \frac{\| g_k \| \| d_{k-1} \| \| g_k - g_{k-1} \|}{\| d_{k-1} \|^2}
\leq \| g_k \| + 2\frac{\| g_k \| \| g_k - g_{k-1} \|}{\| g_{k-1} \|}
\leq \gamma + \frac{2\gamma L b}{\epsilon} \equiv \zeta.
$$

By using the same argument as in Case 1, we obtain (3.4) and the proof is complete. $\square$

### 4 Numerical experiments

In this part, we report the numerical experiments to demonstrate the efficiency of the TTRMIL+ method in comparison with the RMIL [12], RMIL+ [13], PRP [7, 8], and TTRMIL [23] methods. For comparing the computational performance, we consider some test problems from Andrei [30], and Jamil and Yang [31]. Most of initial points are also considered by Andrei [30] and implemented using dimensions starting from 2 to 50,000. The test problems and their initial points are presented in Table 1. The codes were written in Matlab R2019a and run using a personal laptop with specification Intel Core i7 processor, 16 GB RAM, 64 bit Windows 10 Pro operating system. All algorithms are terminated when $\| g_k \| \leq 10^{-6}$, and for objective comparison, all the methods are executed under the standard Wolfe line search (1.5) and (1.6) with parameter $\delta = 10^{-4}$, $\sigma = 0.8$ for the TTRMIL method, and $\delta = 0.01$, $\sigma = 0.1$ for the RMIL, RMIL+, PRP, and TTRMIL+ methods. The metrics used for comparison include the number of iterations (NOI), the number of function evaluations (NOF), and the central of processing unit (CPU) time.

All numerical results of the RMIL, RMIL+, and PRP methods are listed in Table 2 and those of the TTRMIL and TTRMIL+ methods in Table 3. A method is said to have failed if the NOI is more than 10,000 and the terminating criteria stated above have not been satisfied. The failure is symbolized with ‘F’. We also use the performance profile tool of Dolan and Moré [32] to show the performance profile curve of RMIL, RMIL+, PRP, TTRMIL, and TTRMIL+. The performance profile figures on NOI, NOF, and CPU are presented in Figs. 1, 2, and 3, respectively.

Let $P$ be the set of test problems with $n_p$ being the number of test problem. $S$ is the set of methods and $n_s$ is the number of methods. For each method $s \in S$ and problem $p \in P$, let $j_{p,s}$ denote either NOI, NOF, or CPU time required to solve problem $p$ by method $s$. Then the performance profile is defined as follows:

$$
\rho_s(\tau) = \frac{1}{n_p} \text{size}(p \in P : \log_2 r_{p,s} \leq \tau),
$$
Table 1 List of test problems, dimensions, and initial points

| Number | Problems                          | Dimensions | Initial points       |
|--------|-----------------------------------|------------|----------------------|
| 1      | Extended White & Holst            | 1000       | (−1.2, 1, ..., −1.2, 1) |
| 2      | Extended White & Holst            | 1000       | (10, ..., 10)        |
| 3      | Extended White & Holst            | 10,000     | (−1.2, 1, ..., −1.2, 1) |
| 4      | Extended White & Holst            | 10,000     | (5, ..., 5)          |
| 5      | Extended Rosenbrock               | 1000       | (−1.2, 1, ..., −1.2, 1) |
| 6      | Extended Rosenbrock               | 1000       | (10, ..., 10)        |
| 7      | Extended Rosenbrock               | 10,000     | (−1.2, 1, ..., −1.2, 1) |
| 8      | Extended Rosenbrock               | 10,000     | (5, ..., 5)          |
| 9      | Extended Freudenstein & Roth      | 10,000     | (−5, ..., −5)        |
| 10     | Extended Freudenstein & Roth      | 50,000     | (−5, ..., −5)        |
| 11     | Extended Beale                    | 1000       | (1.08, ..., 1.08)    |
| 12     | Extended Beale                    | 1000       | (0.5, ..., 0.5)      |
| 13     | Extended Beale                    | 10,000     | (−1, ..., −1)        |
| 14     | Extended Beale                    | 10,000     | (0.5, ..., 0.5)      |
| 15     | Raydan 1                          | 10         | (1, ..., 1)          |
| 16     | Raydan 1                          | 10         | (−10, ..., −10)      |
| 17     | Raydan 1                          | 100        | (−1, ..., −1)        |
| 18     | Raydan 1                          | 100        | (−10, ..., −10)      |
| 19     | Extended tridiagonal 1            | 500        | (2, ..., 2)          |
| 20     | Extended tridiagonal 1            | 500        | (10, ..., 10)        |
| 21     | Extended tridiagonal 1            | 1000       | (1, ..., 1)          |
| 22     | Extended tridiagonal 1            | 1000       | (−10, ..., −10)      |
| 23     | Diagonal 4                        | 500        | (1, ..., 1)          |
| 24     | Diagonal 4                        | 500        | (−20, ..., −20)      |
| 25     | Diagonal 4                        | 1000       | (1, ..., 1)          |
| 26     | Diagonal 4                        | 1000       | (−30, ..., −30)      |
| 27     | Extended Himmelblau              | 1000       | (1, ..., 1)          |
| 28     | Extended Himmelblau              | 1000       | (20, ..., 20)        |
| 29     | Extended Himmelblau              | 10,000     | (−1, ..., −1)        |
| 30     | Extended Himmelblau              | 10,000     | (50, ..., 50)        |
| 31     | FLETCHCR                         | 10         | (0, ..., 0)          |
| 32     | FLETCHCR                         | 10         | (10, ..., 10)        |
| 33     | Extended Powel                   | 100        | (3, −1, 0, 1, ...)   |
| 34     | Extended Powel                   | 100        | (5, ..., 5)          |
| 35     | NONSCOMP                          | 2          | (3, 3)               |
| 36     | NONSCOMP                          | 2          | (10, 10)             |
| 37     | Extended DENSCINB                | 10         | (1, ..., 1)          |
| 38     | Extended DENSCINB                | 10         | (10, ..., 10)        |
| 39     | Extended DENSCINB                | 100        | (10, ..., 10)        |
| 40     | Extended DENSCINB                | 100        | (−50, ..., −50)      |
| 41     | Extended penalty                 | 10         | (1, 2, ..., 10)      |
| 42     | Extended penalty                 | 10         | (−10, ..., −10)      |
| 43     | Extended penalty                 | 100        | (1, ..., 1)          |
| 44     | Extended penalty                 | 100        | (−2, ..., −2)        |
| 45     | Hager                            | 10         | (1, ..., 1)          |
| 46     | Hager                            | 10         | (−10, ..., −10)      |
| 47     | Extended Maratos                 | 10         | (1.1, 0, ..., 1.1, 1, 1,) |
| 48     | Extended Maratos                 | 10         | (−1, ..., −1)        |
| 49     | Six hump camel                   | 2          | (−1, 2)              |
| 50     | Six hump camel                   | 2          | (−5, 10)             |
| 51     | Three hump camel                 | 2          | (−1, 2)              |
| 52     | Three hump camel                 | 2          | (2, −1)              |
| 53     | Booth                            | 2          | (5, 5)               |
| 54     | Booth                            | 2          | (10, 10)             |
| 55     | Trecanni                         | 2          | (−1, 0, 5)           |
| 56     | Trecanni                         | 2          | (−5, 10)             |
| 57     | Zettl                            | 2          | (−1, 2)              |
| 58     | Zettl                            | 2          | (10, 10)             |
| 59     | Shallow                          | 1000       | (0, ..., 0)          |
Table 1 (Continued)

| Number | Problems                      | Dimensions | Initial points |
|--------|-------------------------------|------------|----------------|
| 60     | Shallow                       | 1000       | (10,10)        |
| 61     | Shallow                       | 10,000     | (−1,−1)        |
| 62     | Shallow                       | 10,000     | (−10,−10)      |
| 63     | Generalized quartic           | 1000       | (5,5)          |
| 64     | Generalized quartic           | 1000       | (20,20)        |
| 65     | Quadratic QF2                 | 50         | (0.5,0.5)      |
| 66     | Quadratic QF2                 | 50         | (30,30)        |
| 67     | Leon                          | 2          | (2,2)          |
| 68     | Leon                          | 2          | (8,8)          |
| 69     | Generalized tridiagonal 1     | 10         | (2,2)          |
| 70     | Generalized tridiagonal 1     | 10         | (10,10,10)     |
| 71     | Generalized tridiagonal 2     | 4          | (1,1,1,1)      |
| 72     | Generalized tridiagonal 2     | 4          | (10,10,10,10)  |
| 73     | POWER                         | 10         | (1,1)          |
| 74     | POWER                         | 10         | (10,10,10)     |
| 75     | Quadratic QF1                 | 50         | (1,1)          |
| 76     | Quadratic QF1                 | 50         | (10,10,10)     |
| 77     | Quadratic QF1                 | 500        | (1,1)          |
| 78     | Quadratic QF1                 | 500        | (5,5)          |
| 79     | Extended quadratic penalty QP2| 100        | (1,1)          |
| 80     | Extended quadratic penalty QP2| 100        | (10,10,10)     |
| 81     | Extended quadratic penalty QP2| 500        | (10,10,10)     |
| 82     | Extended quadratic penalty QP2| 500        | (20,20)        |
| 83     | Extended quadratic penalty QP1| 4          | (1,1,1,1)      |
| 84     | Extended quadratic penalty QP1| 4          | (10,10,10,10)  |
| 85     | Quartic                       | 4          | (10,10,10,10)  |
| 86     | Quartic                       | 4          | (15,15,15,15)  |
| 87     | Matyas                        | 2          | (1,1)          |
| 88     | Matyas                        | 2          | (20,20)        |
| 89     | Colville                      | 4          | (2,2,2,2)      |
| 90     | Colville                      | 4          | (10,10,10,10)  |
| 91     | Dixon and Price               | 3          | (1,1)          |
| 92     | Dixon and Price               | 3          | (10,10,10)     |
| 93     | Sphere                        | 5000       | (1,1)          |
| 94     | Sphere                        | 5000       | (10,10,10)     |
| 95     | Sum squares                   | 50         | (0,1,...,0,1)  |
| 96     | Sum squares                   | 50         | (10,10,10)     |
| 97     | ENGVAL1                       | 50         | (2,...,2)      |
| 98     | ENGVAL1                       | 100        | (2,...,2)      |
| 99     | ENGVALB                       | 50         | (0,...,0)      |
| 100    | ENGVALB                       | 100        | (0,...,0)      |
| 101    | QUARTICM                      | 5000       | (2,...,2)      |
| 102    | QUARTICM                      | 10,000     | (2,...,2)      |
| 103    | QUARTICM                      | 15,000     | (2,...,2)      |
| 104    | QUARTICM                      | 20,000     | (2,...,2)      |

where \( \tau > 0 \), and \( r_{p,s} \) is the performance ratio that can be obtained by

\[
r_{p,s} = \frac{j_{p,s}}{\min\{j_{p,s}\}}.
\]

Generally, the method with the high performance profile value \( \rho_1(\tau) \) is considered the best method for a given \( \tau \) value. In other words, the method where the curve dominates the very top is the most efficient method compared to the others.

According to Table 2, the RMIL method was able to solve 66% of the problems, RMIL+ 75%, and PRP 71%. Meanwhile, based on Table 3, the TTRMIL method solved 93% of the problems and the proposed TTRMIL+ 94%. In this regard, the TTRMIL+ method
Table 2 Numerical results of the RMIL, RMIL+, and PRP methods using weak Wolfe line search

| Number | RMIL NOI | RMIL+ NOI | PRP NOI | RMIL NOF | RMIL+ NOF | PRP NOF | CPU  |
|--------|----------|-----------|---------|----------|-----------|---------|------|
| 1      | 25       | 160       | 0.075   | 16       | 102       | 0.0588  | 15   |
| 2      | F        | F         | F       | F        | F         | F       | 21   |
| 3      | 25       | 160       | 0.578   | 16       | 102       | 0.3907  | 15   |
| 4      | F        | F         | F       | 38       | 260       | 0.9512  | 22   |
| 5      | F        | F         | F       | 27       | 176       | 0.0488  | 19   |
| 6      | 44       | 227       | 0.0618  | 40       | 243       | 0.0667  | F    |
| 7      | F        | F         | F       | 32       | 192       | 0.3874  | 19   |
| 8      | 24       | 126       | 0.2573  | 40       | 195       | 0.3768  | 20   |
| 9      | F        | F         | F       | 11       | 63        | 0.1356  | 8    |
| 10     | F        | F         | F       | 11       | 63        | 0.4922  | 8    |
| 11     | 41       | 137       | 0.5728  | 52       | 191       | 0.0992  | 15   |
| 12     | 56       | 175       | 0.0987  | F        | F         | F       | 9    |
| 13     | 22       | 83        | 0.3537  | 11       | 48        | 0.2153  | F    |
| 14     | 58       | 182       | 0.7956  | 58       | 182       | 0.7956  | 10   |
| 15     | 24       | 83        | 0.0015  | 27       | 105       | 0.0026  | 22   |
| 16     | 36       | 143       | 0.0022  | 37       | 170       | 0.0062  | 37   |
| 17     | 110      | 333       | 0.0379  | 109      | 505       | 0.0394  | 74   |
| 18     | 140      | 435       | 0.0439  | 180      | 841       | 0.0609  | F    |
| 19     | 12       | 56        | 0.0203  | 6        | 37        | 0.0144  | F    |
| 20     | F        | F         | F       | 5        | 26        | 0.0145  | 5    |
| 21     | 12       | 56        | 0.0412  | 7        | 40        | 0.0276  | F    |
| 22     | 8        | 41        | 0.0379  | 9        | 55        | 0.0425  | 13   |
| 23     | F        | F         | F       | F        | F         | F       | F    |
| 24     | F        | F         | F       | F        | F         | F       | F    |
| 25     | F        | F         | F       | F        | F         | F       | F    |
| 26     | F        | F         | F       | F        | F         | F       | F    |
| 27     | 12       | 43        | 0.0205  | 11       | 44        | 0.0215  | 8    |
| 28     | 10       | 48        | 0.0196  | 7        | 34        | 0.0165  | 6    |
| 29     | 9        | 39        | 0.0942  | 9        | 42        | 0.0952  | 8    |
| 30     | F        | F         | F       | 11       | 50        | 0.137   | 8    |
| 31     | 72       | 289       | 0.0036  | 72       | 311       | 0.0084  | 56   |
| 32     | 138      | 712       | 0.0198  | 111      | 548       | 0.0183  | 71   |
| 33     | F        | F         | F       | 70       | 863       | 0.0716  | 3337 |
| 34     | F        | F         | F       | 39       | 225       | 0.0443  | 2312 |
| 35     | 8        | 34        | 4.81E-04| 54       | 193       | 0.0183  | 15   |
| 36     | F        | F         | F       | 17       | 94        | 0.2085  | F    |
| 37     | 7        | 22        | 4.32E-04| 6        | 22        | 0.000845| 5    |
| 38     | 8        | 33        | 5.90E-04| 8        | 37        | 0.0022  | 8    |
| 39     | 8        | 33        | 0.0038  | 8        | 37        | 0.0093  | 8    |
| 40     | 11       | 52        | 0.0172  | 9        | 43        | 0.0181  | 7    |
| 41     | F        | F         | F       | 27       | 112       | 0.0038  | 31   |
| 42     | F        | F         | F       | 26       | 103       | 0.0021  | 9    |
| 43     | 26       | 123       | 0.0081  | 19       | 87        | 0.0056  | 12   |
| 44     | F        | F         | F       | 19       | 89        | 0.0124  | 13   |
| 45     | F        | F         | F       | 19       | 89        | 0.0124  | 13   |
| 46     | F        | F         | F       | 19       | 89        | 0.0124  | 13   |
| 47     | F        | F         | F       | 19       | 89        | 0.0124  | 13   |
| 48     | 40       | 191       | 0.0126  | 31       | 195       | 0.0134  | 25   |
| 49     | 9        | 39        | 0.0005699| 8        | 36        | 0.0053  | 6    |
| 50     | 10       | 59        | 0.0081  | 11       | 66        | 0.0026  | F    |
| 51     | 15       | 363       | 0.0034  | F        | F         | F       | F    |
| 52     | 11       | 226       | 0.0075  | 15       | 400       | 0.0108  | F    |
| 53     | 2        | 6         | 0.0001505| 2        | 6         | 2.58E-04| 2    |
| 54     | 2        | 6         | 0.0105  | 2        | 6         | 2.84E-04| 2    |
| 55     | 1        | 3         | 0.0002193| 1        | 3         | 0.0013  | 1    |
| 56     | F        | F         | F       | 5        | 23        | 0.007   | 5    |
| 57     | 18       | 66        | 0.0024  | 16       | 69        | 0.0028  | 10   |
| 58     | 12       | 46        | 0.0075  | F        | F         | F       | 12   |
| 59     | 14       | 46        | 0.0209  | 11       | 39        | 0.0154  | F    |
Table 2 (Continued)

| Number | RMIL NOI | RMIL+ NOI | PRP NOI | RMIL NOF | RMIL+ NOF | PRP NOF | RMIL CPU | RMIL+ CPU | PRP CPU |
|--------|----------|-----------|---------|----------|-----------|---------|----------|-----------|---------|
| 60     | 16       | 58        | 0.035   | 14       | 59        | 0.0303  | 13       | 51        | 0.018   |
| 61     | 51       | 155       | 0.323   | F        | F         | F       | F        | F         | F       |
| 62     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 63     | 24       | 301       | 0.0146  | F        | F         | F       | F        | F         | F       |
| 64     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 65     | 78       | 265       | 0.0146  | 78       | 280       | 0.0225  | 70       | 250       | 0.0241  |
| 66     | 78       | 299       | 0.0226  | 77       | 334       | 0.0327  | 58       | 275       | 0.0322  |
| 67     | 35       | 170       | 0.0046  | 31       | 179       | 0.0023  | 17       | 136       | 0.0012  |
| 68     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 69     | 21       | 66        | 0.0019  | 22       | 74        | 0.0058  | 23       | 77        | 0.0057  |
| 70     | 27       | 104       | 0.0155  | 28       | 120       | 0.003   | 27       | 117       | 0.0037  |
| 71     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 72     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 73     | 123      | 369       | 0.0074  | 123      | 369       | 0.0102  | 10       | 30        | 7.66E–04 |
| 74     | 139      | 417       | 0.0139  | 139      | 417       | 0.0123  | 10       | 30        | 8.78E–04 |
| 75     | 69       | 207       | 0.0108  | 69       | 207       | 0.0115  | 38       | 114       | 0.0049  |
| 76     | 78       | 234       | 0.0093  | 78       | 234       | 0.0104  | 40       | 120       | 0.0073  |
| 77     | 447      | 1341      | 0.1754  | 447      | 1341      | 0.1716  | 131      | 393       | 0.0719  |
| 78     | 500      | 1500      | 0.2143  | 500      | 1500      | 0.2046  | 137      | 411       | 0.072   |
| 79     | 37       | 314       | 0.0274  | 34       | 313       | 0.0254  | 22       | 235       | 0.0161  |
| 80     | F        | F         | F       | F        | F         | F       | 11       | 59        | 0.0019  |
| 81     | 60       | 591       | 0.099   | 57       | 620       | 0.1127  | 39       | 493       | 0.087   |
| 82     | 3899     | 12030     | 1.6296  | 14       | 53        | 0.0012  | 6        | 28        | 6.26E–04 |
| 83     | 14       | 48        | 0.000978| 14       | 53        | 0.0012  | 6        | 28        | 6.26E–04 |
| 84     | 20       | 81        | 0.0144  | 15       | 68        | 0.0013  | 9        | 49        | 9.52E–04 |
| 85     | 773      | 2468      | 0.0345  | 802      | 2788      | 0.0517  | 163      | 696       | 0.0138  |
| 86     | 781      | 2558      | 0.0395  | 806      | 2811      | 0.0454  | 113      | 495       | 0.0133  |
| 87     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 88     | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 89     | 773      | 3091      | 0.0278  | 1032     | 4339      | 0.0726  | 148      | 818       | 0.2155  |
| 90     | 897      | 3418      | 0.0425  | 669      | 2819      | 0.0324  | 86       | 372       | 0.0167  |
| 91     | 42       | 149       | 0.0077  | F        | F         | F       | F        | F         | F       |
| 92     | 35       | 141       | 0.0196  | 46       | 194       | 0.0083  | 56       | 266       | 0.0063  |
| 93     | 1        | 3         | 0.0057  | 1        | 3         | 0.0083  | 1        | 3         | 0.00167 |
| 94     | 1        | 3         | 0.0179  | 1        | 3         | 0.0056  | 1        | 3         | 0.0071  |
| 95     | 46       | 138       | 0.0123  | 46       | 138       | 0.0152  | 25       | 75        | 0.0057  |
| 96     | 81       | 243       | 0.5261  | 81       | 243       | 0.2223  | 41       | 123       | 0.0097  |
| 97     | 9        | 112       | 0.0162  | 47       | 817       | 0.0301  | 22       | 409       | 0.0147  |
| 98     | F        | F         | F       | F        | F         | F       | 22       | 416       | 0.0251  |
| 99     | 14       | 46        | 0.0112  | 14       | 63        | 0.2976  | 14       | 78        | 0.2305  |
| 100    | 14       | 60        | 0.0089  | F        | F         | F       | F        | F         | F       |
| 101    | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 102    | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 103    | F        | F         | F       | F        | F         | F       | F        | F         | F       |
| 104    | F        | F         | F       | F        | F         | F       | F        | F         | F       |

is considered a better method when compared to the RMIL, RMIL+, and PRP methods, but competes with the TTRMIL method in terms of NOI, CPU time, and NOF. From the performance profile in Figs. 1–3, we can see that the TTRMIL+ method is efficient and promising with regard to solving unconstrained optimization problems compared to the RMIL, RMIL+, PRP, and TTRMIL methods.

5 Application of TTRMIL+ to parameterized COVID-19 model

Coronavirus disease often called COVID-19 is an acute vector-borne disease that surfaced in 2019. This disease is caused by the newly discovered coronavirus (SARS-CoV-2) and can be transmitted through droplets produced when an infected person exhales, sneezes,
| Number | TTRMIL NOI | NOF | CPU | TTRMIL+ NOI | NOF | CPU |
|--------|-----------|-----|-----|-------------|-----|-----|
| 1      | 93        | 358 | 0.1801 | 23 | 150 | 0.0711 |
| 2      | 9929      | 29,974 | 12.2849 | 84 | 513 | 0.2698 |
| 3      | 88        | 342 | 1.3363 | 30 | 181 | 0.6683 |
| 4      | 4957      | 14,979 | 76.0681 | 45 | 307 | 1.4779 |
| 5      | 78        | 295 | 0.0782 | 35 | 175 | 0.0473 |
| 6      | 120       | 467 | 0.1551 | 54 | 313 | 0.0894 |
| 7      | 108       | 384 | 0.7369 | 19 | 65 | 0.0015 |
| 8      | 50        | 176 | 0.4424 | 59 | 290 | 0.6383 |
| 9      | 24        | 120 | 0.3335 | 16 | 87 | 0.2462 |
| 10     | 24        | 120 | 1.0537 | 27 | 120 | 1.0524 |
| 11     | 38        | 112 | 0.0976 | 20 | 75 | 0.0503 |
| 12     | 34        | 101 | 0.0697 | 46 | 148 | 0.1227 |
| 13     | 59        | 183 | 0.8012 | 24 | 100 | 0.6058 |
| 14     | 37        | 109 | 0.4909 | 48 | 154 | 1.0408 |
| 15     | 70        | 164 | 0.0186 | 19 | 65 | 0.0015 |
| 16     | 122       | 302 | 0.0133 | 39 | 197 | 0.0039 |
| 17     | 109       | 329 | 0.0205 | 110 | 333 | 0.0402 |
| 18     | 179       | 539 | 0.0533 | 173 | 541 | 0.0511 |
| 19     | 369       | 1110 | 0.3641 | 17 | 80 | 0.0435 |
| 20     | 414       | 1216 | 0.4525 | 18 | 83 | 0.0447 |
| 21     | 488       | 1419 | 0.7491 | 17 | 80 | 0.048 |
| 22     | 293       | 935 | 0.4831 | 23 | 104 | 0.0648 |
| 23     | 14        | 39 | 0.0254 | 11 | 30 | 0.0114 |
| 24     | 19        | 53 | 0.0171 | 13 | 36 | 0.0157 |
| 25     | 14        | 39 | 0.0223 | 7 | 19 | 0.012 |
| 26     | 19        | 53 | 0.0234 | 13 | 35 | 0.0186 |
| 27     | 9         | 32 | 0.0359 | 9 | 34 | 0.0144 |
| 28     | 13        | 58 | 0.0297 | 15 | 60 | 0.0254 |
| 29     | 10        | 41 | 0.1842 | 9 | 39 | 0.1533 |
| 30     | 11        | 50 | 0.202 | 13 | 57 | 0.1416 |
| 31     | 73        | 288 | 0.0248 | 72 | 290 | 0.005 |
| 32     | 139       | 699 | 0.018 | 141 | 729 | 0.0115 |
| 33     | F         | F | F | F | F | F |
| 34     | F         | F | F | F | F | F |
| 35     | 31        | 95 | 0.014 | 12 | 44 | 0.0023 |
| 36     | 29        | 102 | 0.0049 | 22 | 100 | 0.0013 |
| 37     | 8         | 24 | 0.0141 | 5 | 16 | 4.05E-04 |
| 38     | 10        | 40 | 0.0018 | 9 | 35 | 6.82E-04 |
| 39     | 11        | 43 | 0.0039 | 9 | 35 | 0.0032 |
| 40     | 13        | 73 | 0.0053 | 9 | 47 | 0.0037 |
| 41     | 27        | 95 | 0.0177 | 23 | 85 | 0.014 |
| 42     | 22        | 83 | 2.70E-03 | 16 | 66 | 0.0043 |
| 43     | F         | F | F | F | F | F |
| 44     | 20        | 107 | 0.3942 | 10 | 46 | 0.0066 |
| 45     | 23        | 60 | 0.0095 | 24 | 63 | 0.0014 |
| 46     | 34        | 89 | 0.0057 | 32 | 86 | 0.0019 |
| 47     | 42        | 206 | 0.0132 | 47 | 261 | 0.0057 |
| 48     | 51        | 213 | 0.005 | 50 | 275 | 0.0073 |
| 49     | 7         | 26 | 0.0071 | 6 | 25 | 3.28E-04 |
| 50     | 8         | 41 | 5.62E-04 | 9 | 44 | 4.61E-04 |
| 51     | 27        | 85 | 0.0073 | 19 | 275 | 0.0078 |
| 52     | F         | F | F | F | F | F |
| 53     | 2         | 6 | 1.98E-02 | 2 | 6 | 4.81E-04 |
| 54     | 2         | 6 | 2.60E-03 | 2 | 6 | 1.71E-04 |
| 55     | 1         | 3 | 6.50E-03 | 1 | 3 | 1.87E-04 |
| 56     | 13        | 48 | 5.90E-03 | 10 | 37 | 9.83E-04 |
| 57     | 21        | 59 | 0.0095 | 17 | 61 | 0.0026 |
| 58     | 25        | 80 | 9.51E-04 | 13 | 50 | 7.63E-04 |
| 59     | 27        | 71 | 0.371 | 13 | 39 | 0.012 |
Table 3 (Continued)

| Number | TTRMIL | TTRMIL+ |
|--------|--------|---------|
|        | NOI | NOF | CPU | NOI | NOF | CPU |
| 60     | 40  | 124 | 0.1116 | 19  | 78  | 0.0245 |
| 61     | 34  | 98  | 2.2351 | 26  | 74  | 0.1941 |
| 62     | 60  | 195 | 0.4594 | 24  | 82  | 0.1785 |
| 63     | F   | F   | F   | F   | F   | F   |
| 64     | F   | F   | F   | F   | F   | F   |
| 65     | 80  | 261 | 0.2371 | 79  | 268 | 0.0158 |
| 66     | 90  | 361 | 0.019  | 81  | 318 | 0.0018 |
| 67     | 214 | 703 | 0.0263 | 17  | 94  | 0.0012 |
| 68     | 125 | 586 | 0.0222 | 54  | 371 | 0.0042 |
| 69     | 22  | 69  | 0.0138 | 22  | 69  | 0.0013 |
| 70     | 32  | 132 | 0.005  | 27  | 104 | 0.0027 |
| 71     | 16  | 42  | 1.33E–02 | 17  | 46  | 6.91E–04 |
| 72     | 14  | 47  | 7.42E–04 | 23  | 71  | 0.0013 |
| 73     | 123 | 369 | 1.59E–02 | 123 | 369 | 0.0064 |
| 74     | 139 | 417 | 7.80E–03 | 139 | 417 | 0.0083 |
| 75     | 69  | 207 | 0.0162 | 69  | 207 | 0.0084 |
| 76     | 76  | 234 | 0.0182 | 78  | 234 | 0.0105 |
| 77     | 447 | 1341 | 0.2032 | 447 | 1341 | 0.1934 |
| 78     | 500 | 1500 | 0.188  | 500 | 1500 | 0.2052 |
| 79     | 61  | 419 | 0.2168 | 38  | 320 | 0.0215 |
| 80     | 163 | 701 | 0.0447 | 41  | 324 | 0.0225 |
| 81     | 1516 | 5038 | 0.7924 | 61  | 594 | 0.1043 |
| 82     | 84  | 683 | 0.1154 | 82  | 762 | 0.1268 |
| 83     | 10  | 34  | 1.03E–02 | 15  | 51  | 0.0011 |
| 84     | 18  | 71  | 1.10E–03 | 18  | 71  | 0.0013 |
| 85     | 580 | 1890 | 0.8407 | 804 | 2608 | 0.0368 |
| 86     | 740 | 2315 | 0.0271 | 777 | 2471 | 0.0367 |
| 87     | 29  | 145 | 9.60E–03 | 9   | 63  | 6.19E–04 |
| 88     | 37  | 185 | 0.0018 | 11  | 77  | 8.07E–04 |
| 89     | 838 | 3098 | 0.0527 | 683 | 2426 | 0.0262 |
| 90     | 567 | 1924 | 0.017  | 493 | 1847 | 0.023  |
| 91     | 23  | 75  | 1.34E–02 | 20  | 68  | 6.89E–04 |
| 92     | 36  | 154 | 0.0023 | 39  | 153 | 0.0021 |
| 93     | 1   | 3   | 0.0723 | 1   | 3   | 0.0064 |
| 94     | 1   | 3   | 0.008  | 1   | 3   | 0.0058 |
| 95     | 46  | 138 | 0.0085 | 46  | 138 | 0.0071 |
| 96     | 81  | 243 | 0.0185 | 81  | 243 | 0.0072 |
| 97     | 24  | 79  | 0.0159 | 24  | 103 | 0.0048 |
| 98     | 23  | 76  | 0.0107 | F   | F   | F   |
| 99     | 14  | 46  | 0.2267 | F   | F   | F   |
| 100    | F   | F   | F   | 15  | 50  | 0.0131 |
| 101    | 45  | 365 | 0.7906 | 4   | 31  | 0.0886 |
| 102    | 46  | 381 | 1.5578 | 4   | 31  | 0.1567 |
| 103    | 46  | 381 | 2.2696 | 4   | 31  | 0.2059 |
| 104    | 47  | 397 | 2.9721 | 4   | 31  | 0.2747 |

or coughs. Most people infected by the virus will develop mild to moderate symptoms, such as mild fever, cold, difficulty in breathing, and recover without special treatment. Clinically, as of 3:05 pm CEST, 20 October 2020, a total of 40,251,950 confirmed cases of the COVID-19 with 1,116,131 deaths was recorded from 215 territories and countries around the globe since the disease was first reported in Wuhan, China [WHO].

Recently, numerous studies modeled various aspects of the coronavirus outbreak, and application of numerical methods on some COVID-19 models was also studied. In this paper, we consider the global COVID-19 outbreak from January to September, 2020, model the confirmed cases into an unconstrained optimization problem, and finally apply TTRMIL+ to obtain the solution of the parameterized model.
Consider the following function of regression analysis:

\[ y = h(x_1, x_2, \ldots, x_p + \varepsilon), \] (5.1)

where \( x_i, i = 1, 2, \ldots, p, p > 0 \) is the predictor, \( y \) is the response variable, and \( \varepsilon \) is the error. This type of problem often arises in the fields of management, finance, economics, accounting, physics, and many more. The regression analysis is a statistical modeling tool used to estimate the relationships between a dependent variable and one or more independent variables. To derive the linear regression function, we compute \( y \) such that

\[ y = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_p x_p + \varepsilon, \] (5.2)
where the parameters of the regression are defined by $a_0, \ldots, a_p$. The regression analysis estimates the regression parameters $a_0, a_1, \ldots, a_p$ such that the value of the error $\varepsilon$ is minimized. An instance where the linear regression method is the relationship between $y$ and $x$ is approximated with a straight line. However, such a case infrequently occurs, and thus, the nonlinear regression process is often used. In this study, we consider the nonlinear regression approach.

To formulate the approximate function, we consider the data from the global confirmed cases of COVID-19 from January to September, 2020. The detailed description of the process follows from the statistics presented in Table 4 which are taken from the data obtained by the World Health Organization [WHO] [33]. We have data for nine months (Jan–Sept), the data for the months would be denoted by $x$-variable and the confirmed cases corresponding to these months would be denoted by the $y$-variable. For fitting the data, we only consider the data for eight months (Jan–Aug), and reserve the data for September for error analysis.

From the above data, we obtain the following approximate function for the nonlinear least square method:

$$f(x) = -26,029.59 + 14,557.39x + 3290.077x^2. \quad (5.3)$$
Function (5.3) is used to approximate the values of $y$ based on values of $x$ from Jan–Aug. Denoting the number of months by $x_j$ and the corresponding confirmed cases by $y_j$, then, we can transform the least squares problem (5.3) into the following unconstrained minimization model:

$$
\min_{x \in \mathbb{R}^n} f(x) = \sum_{j=1}^{n} \left( (u_0 + u_1 x_j + u_2 x_j^2) - y_j \right)^2. \quad (5.4)
$$

The nonlinear quadratic function for the least squares problem is derived using the data utilized from Jan–Aug, 2020, which is further used to formulate the corresponding unconstrained optimization model. Obviously, it can be observed that data $x_j$ and the value of $y_j$ possess some parabolic relations with the regression parameters $u_0, u_1,$ and $u_2$ and the regression function (5.4).

$$
\min_{x \in \mathbb{R}^2} \sum_{j=1}^{n} E_j^2 = \sum_{j=1}^{n} \left( (u_0 + u_1 x_j + u_2 x_j^2) - y_j \right)^2. \quad (5.5)
$$

Using the data from Table 4, we can transform (5.5) to obtain our nonlinear quadratic unconstrained minimization model as follows:

$$
9u_0^2 + 90u_0 u_1 + 570u_0 u_2 - 2,482,956u_0 + 285u_1^2 + 4050u_1 u_2 - 17,172,778u_1 + 15,333u_2^2 - 126,050,318u_2 + 275,210,100,844. \quad (5.6)
$$

The data considered to generate the unconstrained optimization model are data from Jan–August, and the data for Sept is reserved for computing the relative errors of the predicted data. Applying the proposed TTRMIL+ method on model (5.6) under the strong Wolfe line search, we obtain the following results presented in Table 5.

One of the major challenges is computing the values of $u_0, u_1, u_2$ using matrix inverse [34]. To overcome this difficulty, we implement the proposed TTRMIL+ using different initial points. The computation would be terminated if the following conditions hold.

1. The algorithm fails to solve the model.
2. The number of iterations exceeds 1000. This point is denoted as ‘Fail’.

### 5.1 Trend line method

A trend line is a line drawn under pivot lows or over pivot highs to show the prevailing direction of price. In this section, we estimate the data for COVID-19 for a period of nine (9) months using the proposed TTRMIL+ and least squares methods. The trend line is plotted using the Microsoft Excel software based on data from Table 4. The trend line

| Initial points | NOI | CPU time         |
|----------------|-----|------------------|
| (3,3,3)        | 14  | 0.0325913087998213 |
| (5,5,5)        | 13  | 0.04000198696240659 |
| (21,21,21)     | 15  | 0.0406229692033143 |
| (100,100,100)  | 15  | 0.04526743733986786 |

Table 5 Test results for optimization of quadratic model for TTRMIL+
equation appears in a form of nonlinear quadratic equation. Representing the \( y \)-axis by \( y \) and \( x \)-axis by \( x \), we obtain the plot presented in Fig. 4 using the actual data from Table 4. Further, to illustrate the efficiency of the proposed method, we compare the approximation functions of TTRMIL+ method with the functions of trend line and least square methods as follows.

The ideal purpose of regression analysis is estimating the parameters \( a_0, a_1, \ldots, a_p \) such that the error \( \varepsilon \) is minimized. From the results presented in Table 6, it is obvious that the proposed TTRMIL+ CG method has the least relative error compared to the least square and trend line methods which implied that the method is applicable to real-life situations. For other references regarding modeling, analysis, and prediction of COVID-19 cases, one can see [35].

### 6 Application TTRMIL+ in motion control

This section demonstrates the performance of the proposed TTRMIL+ CG method on motion control of a two-joint planar robotic manipulator. As presented in [36], the following model describes a discrete-time kinematics equation of two-joint planar robot manipulator at the position level

\[
\Gamma(\mu_k) = \eta_k, \quad (6.1)
\]

where \( \mu_k \in \mathbb{R}^2 \) and \( \eta_k \in \mathbb{R}^2 \) denote the joint angle vector and the end effector vector position, respectively. The vector-valued function \( \Gamma(\cdot) \) represents the kinematics function which has the following structure:

\[
\Gamma(\mu_k) = \begin{bmatrix}
\tau_1 \cos(\mu_1) + \tau_2 \cos(\mu_1 + \mu_2) \\
\tau_1 \sin(\mu_1) + \tau_2 \sin(\mu_1 + \mu_2)
\end{bmatrix}, \quad (6.2)
\]
with \( \tau_1 \) and \( \tau_2 \) denoting the length of the first and second rod, respectively. In the case of motion control, at each instantaneous computational time interval \([t_k, t_{k+1}] \subseteq [0, T_f]\) with \( T_f \) being the end of task duration, the following nonlinear least squares model is to be minimized:

\[
\min_{\Gamma_k} \frac{1}{2} \| \Gamma_k - \hat{\Gamma}_k \|_2^2, \tag{6.3}
\]

where \( \hat{\Gamma}_k \) denotes the end effector controlled track.

Similar to the approach presented in [37–39], the end effector, used in this experiment, is controlled to track a Lissajous curve given as

\[
\hat{\Gamma}_k = \left[ \begin{array}{c} \frac{3}{2} + \frac{1}{2} \sin\left(\frac{\pi t_k}{5}\right) \\ \sqrt{\frac{3}{2} + \frac{1}{2} \sin\left(\frac{6\pi t_k}{5} + \frac{\pi}{3}\right)} \end{array} \right]. \tag{6.4}
\]

The parameters used in the implementation of the proposed TTRMIL+ CG method are: \( \tau_1 = 1, \tau_2 = 1, \) and \( T_f = 10 \) seconds. The starting point \( \mu_0 = [\mu_1, \mu_2] = [0, \frac{\pi}{3}]^T \) where the task duration \([0, 10]\) is divided into 200 equal parts.

The results of the motion control experiments are depicted in Figs. 5(a)–5(b). The robot trajectories synthesized by the TTRMIL+ are shown in Fig. 5(a), where the end effector trajectory and the desired path are plotted in Fig. 5(b). Finally, the errors recorded on

![Figure 5](image-url)
horizontal and vertical axes by the TTRMIL+ are shown in Figs. 5(c) and 5(d), respectively. Perusing through these figures, it can be seen from Figs. 5(a) and 5(b) that the TTRMIL+ successfully accomplished the task at hand. The error recorded in the course of the task is relatively low as can be seen from Figs. 5(c) and 5(d), which confirms the efficiency of the proposed TTRMIL+.

7 Conclusion
This paper presented a modified conjugate gradient method for unconstrained optimization models. The proposed TTRMIL+ method replaced RMIL in TTRMIL with a new modification known as RMIL+. The sufficient descent condition and the convergence proof of TTRMIL+ are studied under the standard Wolfe line search. Some unconstrained benchmark test problems are considered to illustrate the performance of the proposed method. The result obtained showed that the TTRMIL+ method is efficient and promising. The method was further applied to a parameterized COVID-19 model, and the result obtained showed that TTRMIL+ produced a good regression model and thus can be used in regression analysis. Finally, we applied the method to solve a practical problem of motion control. Future work includes studying the new algorithm on nonlinear least squares problems as discussed in [40]. Furthermore, we shall consider other problems in our future research as presented in the following references [41–44].

Acknowledgements
The authors acknowledge the financial support provided by the Center of Excellence in Theoretical and Computational Science (TaCS-CoE), KMUTT and Center under Computational and Applied Science for Smart Innovation research Cluster (CLASSIC), Faculty of Science, KMUTT. Aliyu Muhammed Awwal would like to thank the Postdoctoral Fellowship from King Mongkut’s University of Technology Thonburi (KMUTT), Thailand.

Funding
This project is funded by National Council of Thailand (NRCT) under Research Grants for Talented Mid-Career Researchers (Contract no. N41A640089).

Abbreviations
CG, conjugate gradient; RMIL, Rivaie, Mustafa, Ismail, and Leong. TTRMIL, Three-term Rivaie, Mustafa, Ismail, and Leong; PRP, Polak–Ribière–Polyak; NOI, Number of iterations; NOF, Number of function evaluations; CPU, CPU time; SARS-CoV-2, Severe acute respiratory syndrome coronavirus 2; COVID-19, Coronavirus disease caused by SARS-CoV-2.

Availability of data and materials
Not applicable.

Declarations
Competing interests
The authors declare that they have no competing interests.

Authors’ contributions
The authors contributed equally to this paper. All authors have read and approved this version of the manuscript.

Author details
1Department of Mathematics and Statistics, School of Quantitative Sciences, College of Art and Sciences (CAS), Universiti Utara Malaysia (UUM), 06010, Sintok, Kedah, Malaysia. 2Department of Mathematics, Universitas Indonesia (UI), Depok 16424, Indonesia. 3Department of Mathematics, Faculty of Science, Gombe State University, Gombe, Nigeria. 4Center of Excellence in Theoretical and Computational Science (TaCS-CoE), Faculty of Science, King Mongkut’s University of Technology Thonburi (KMUTT), 126 Pracha Uthit Rd., Bang Mod, Thung Khru, Bangkok, 10140, Thailand. 5KMUTT-Fixed Point Theory and Applications Research Group, Theoretical and Computational Science Center (TaCS), Science Laboratory Building, Faculty of Science, King Mongkut’s University of Technology Thonburi (KMUTT), 126 Pracha-Uthit Road, Bang Mod, Thung Khru, Bangkok, 10140, Thailand. 6Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 40402, Taiwan. 7Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), Terengganu 22200, Malaysia.
Publisher's Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Received: 2 March 2021 Accepted: 19 October 2021 Published online: 04 January 2022

References
1. Xia, Z., Wang, X., Sun, X., Wang, Q.: A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans. Parallel Distrib. Syst. 27(2), 340–352 (2015)
2. Yuan, G., Lu, S., Wei, Z.: A new trust-region method with line search for solving symmetric nonlinear equations. Int. J. Comput. Math. 88(10), 2109–2123 (2011)
3. Sulaiman, I.M., Supian, S., Mamat, M.: New Class of Hybrid Conjugate Gradient Coefficients with Guaranteed Descent and Efficient Line Search. In IOP Conference Series: Materials Science and Engineering, vol. 621, p. 012021. IOP Publishing, Bristol (2019)
4. Hager, W.W., Zhang, H.: A survey of nonlinear conjugate gradient methods. Pac. J. Optim. 21(1), 35–58 (2006)
5. Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand. 49, 409–435 (1952)
6. Fletcher, R., Powell, M.J.D.: A rapidly convergent descent method for minimization. Comput. J. 6(2), 163–168 (1963)
7. Polak, E., Ribiere, G.: Note sur la convergence de méthodes de directions conjuguées. ESAIM: Math. Model. Numer. Anal. 3(1), 35–43 (1969)
8. Poljak, B.T.: The conjugate gradient method in extremal problems. USSR Comput. Math. Math. Phys. 9(4), 94–112 (1969)
9. Liu, Y., Storey, C.: Efficient generalized conjugate gradient algorithms, part I: theory. J. Optim. Theory Appl. 69(1), 129–137 (1991)
10. Dai, Y., Han, J., Liu, G., Sun, D., Yin, H., Yuan, Y.X.: Convergence properties of nonlinear conjugate gradient methods. SIAM J. Optim. 10(2), 345–358 (2000)
11. Yuan, G., Wei, Z., Lu, X.: Global convergence of BFGS and PRP methods under a modified weak Wolfe–Powell line search. Appl. Math. Model. 47, 811–825 (2017)
12. Rivaie, M., Mamat, M., June, L.W., Mohd, I.: A new class of nonlinear conjugate gradient coefficients with global convergence properties. Appl. Math. Comput. 218(22), 11323–11332 (2012)
13. Dai, Z.: Comments on a new class of nonlinear conjugate gradient coefficients with global convergence properties. Appl. Math. Comput. 276, 297–300 (2016)
14. Yousif, O.O.O.: The convergence properties of RMIL+ conjugate gradient method under the strong Wolfe line search. Appl. Math. Comput. 367, 124777 (2020)
15. Al-Baali, M.: Descent property and global convergence of the Fletcher–Reeves method with inexact line search. IMA J. Numer. Anal. 8(1), 121–124 (1988)
16. Gilbert, J.C., Nocedal, J.: Global convergence properties of conjugate gradient methods for optimization. SIAM J. Optim. 2(1), 21–42 (1992)
17. Touati-Ahmed, D., Storey, C.: Efficient hybrid conjugate gradient techniques. J. Optim. Theory Appl. 64(2), 379–397 (1990)
18. Hu, Y.F., Storey, C.: Global convergence result for conjugate gradient methods. J. Optim. Theory Appl. 71(2), 399–405 (1991)
19. Avval, A.M., Sulaiman, I.M., Malik, M., Mamat, M., Kumam, P., Siriththakerngkit, K.: A spectral RMIL+ conjugate gradient method for unconstrained optimization with applications in portfolio selection and motion control. IEEE Access 9, 75398–75414 (2021)
20. Beale, E.M.L.: A deviation of conjugate gradients. In: Numerical Methods for Nonlinear Optimization, pp. 39–43 (1972)
21. McGuire, M.F., Wolfe, P.: Evaluating a restart procedure for conjugate gradients. IBM Thomas J. Watson Research Division (1973)
22. Zhang, L., Zhou, W., Li, D.H.: A descent modified Polak–Ribiere–Polyak conjugate gradient method and its global convergence. IMA J. Numer. Anal. 26(4), 629–640 (2006)
23. Liu, J.K., Feng, Y.M., Zou, L.M.: Some three-term conjugate gradient methods with the inexact line search condition. Calcolo 55(2), 1–16 (2018)
24. Zhang, L., Zhou, W., Li, D.: Some descent three-term conjugate gradient methods and their global convergence. Optim. Methods Softw. 22(4), 697–711 (2007)
25. Andrei, N.: A simple three-term conjugate gradient algorithm for unconstrained optimization. J. Comput. Appl. Math. 241, 19–29 (2013)
26. Al-Bayati, A.Y., Altaw, H.W.: A new three-term non-linear conjugate gradient method for unconstrained optimization. Can. J. Sci. Eng. Math. Can. 1, 108–124 (2010)
27. Dong, X., Liu, H., He, Y., Babaie-Kafaki, S., Ghanbari, R.: A new three-term conjugate gradient method with descent direction for unconstrained optimization. Math. Model. Anal. 21(3), 399–411 (2016)
28. Sun, M., Liu, J.: Three modified Polak–Ribiere–Polyak conjugate gradient methods with sufficient descent property. J. Inequal. Appl. 2013(1), 1 (2015)
29. Zoutendijk, G.: Nonlinear Programming, computational methods. In: Integer and Nonlinear Programming, pp. 37–86 (1970)
30. Andrei, N.: Nonlinear Conjugate Gradient Methods for Unconstrained Optimization. Springer, Berlin (2020)
31. Jamil, M., Yang, X.S.: A literature survey of benchmark functions for global optimisation problems. Int. J. Math. Model. Numer. Optim. 4(2), 150–194 (2013)
32. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. 91(2), 201–213 (2002)
33. World health organization: Report on coronavirus (COVID-19) (2020)
34. Sulaiman, I.M., Mamat, M.: A new conjugate gradient method with descent properties and its application to regression analysis. J. Numer. Anal. Ind. Appl. Math. 14(1–2), 25–39 (2020)
35. Ul Rehman, A., Singh, R., Agarwal, P.: Modeling, analysis and prediction of new variants of COVID-19 and Dengue co-infection on complex network. Chaos Solitons Fractals 2021, 111008 (2021)
36. Zhang, Y., He, L., Hu, C., Guo, J., Li, J., Shi, Y.: General four-step discrete-time zeroing and derivative dynamics applied to time-varying nonlinear optimization. J. Comput. Appl. Math. 347, 314–329 (2019)
37. Awwal, A.M., Kumam, P., Wang, L., Huang, S., Kumam, W.: Inertial-based derivative-free method for system of monotone nonlinear equations and application. IEEE Access 8, 226921–226930 (2020)
38. Yahaya, M.M., Kumam, P., Awwal, A.M., Aji, S.: A structured quasi–Newton algorithm with nonmonotone search strategy for structured NLS problems and its application in robotic motion control. J. Comput. Appl. Math. 395, 113582 (2021)
39. Aji, S., Kumam, P., Awwal, A.M., Yahaya, M.M., Kumam, W.: Two hybrid spectral methods with inertial effect for solving system of nonlinear monotone equations with application in robotics. IEEE Access 9, 30918–30928 (2021)
40. Awwal, A.M., Kumam, P., Mohammad, H.: Iterative algorithm with structured diagonal Hessian approximation for solving nonlinear least squares problems. J. Nonlinear Convex Anal. 22(6), 1173–1188 (2021)
41. Agarwal, P., Ahsan, S., Akbar, M., Nawaz, R., Cesarano, C.: A reliable algorithm for solution of higher dimensional nonlinear (1 + 1) and (2 + 1) dimensional Volterra–Fredholm integral equations. Dolomites Res. Notes Approx. 14(2), 18–25 (2021)
42. Shah, N.A., Agarwal, P., Chung, J.D., El-Zahar, E.R., Hamed, Y.S.: Analysis of optical solitons for nonlinear Schrödinger equation with detuning term by iterative transform method. Symmetry 12(11), 1850 (2020)
43. Saoudi, K., Agarwal, P., Mursaleen, M.: A multiplicity result for a singular problem with subcritical nonlinearities. J. Nonlinear Funct. Anal., 1–18 (2017)
44. Rahmoune, A., Ouchenane, D., Boulaaras, S., Agarwal, P.: Growth of solutions for a coupled nonlinear Klein–Gordon system with strong damping, source, and distributed delay terms. Adv. Differ. Equ. 2020(1), 1 (2020)