Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

UNIVERSAL WISER PUBLISHER

Research Article

On Randomized Average Block Gauss-Seidel Method and Its Greedy Version for Inconsistent Factorized Linear System

Peng Zhang[1], **Yi Wu**[2], **Bing Wei**[1*]

[1] School of Cyberspace Security, Hainan University, Haikou, 570228, China
[2] School of International Communication and Arts, Hainan University, Haikou, 570228, China
E-mail: weibing@hainanu.edu.cn

Received: 14 July 2025; **Revised:** 12 September 2025; **Accepted:** 20 October 2025

Abstract: For solving inconsistent factorized linear systems, the Randomized Gauss Seidel-Randomized Kaczmarz (RGS-RK) method is very effective. In order to improve its convergence, we first develop the randomized average block Gauss-Seidel method. Then, a simple block version of the RGS-RK method is proposed. By combining the sketching technique with greedy strategy, we present a Greedy Block version of the RGS-RK (GBRGS-RK) method. In addition, the convergence rates of the presented methods are analyzed. Several numerical examples from Phillips's problem, The University of California, Irvine machine learning datasets to random matrices are given to demonstrate that the proposed algorithms perform better in computing cost and iteration. In particular, the GBRGS-RK method secures excellent computational efficiency, which renders it extremely appropriate for large-scale inconsistent factorized linear system.

Keywords: inconsistent factorized linear system, Randomized average block Gauss-Seidel, pseudoinverse-free block iterative algorithm, Randomized Average Block Kaczmarz (RABK), randomized extended average block Kaczmarz

MSC: 15A06, 65F10, 65F20

1. Introduction

Consider the iterative solution of large-scale inconsistent factorized linear system

$$UV\tilde{\beta} = b, \tag{1}$$

with $U \in \mathbb{R}^{m\times k}$, $V \in \mathbb{R}^{k\times n}$, where $b \in \mathbb{R}^m$ is a real m dimensional vector. Let $A = UV$, then we derive the full linear system, represented by $A\tilde{\beta} = b$. This system arises naturally in a wide range of practical applications, such as "User-Item" interaction matrix from machine learning [1], data analysis [2] and systems infrastructure constraints [3] and the references therein. As the coefficient matrix A from the above applications often reaches the million-level scale, calculating the matrix product of U and V to form A directly results in memory overflow, and traditional iterative approaches exhibit insufficient

Copyright ©2026 Bing Wei, et al.
DOI: https://doi.org/10.37256/cm.7220267834
This is an open-access article distributed under a CC BY license
(Creative Commons Attribution 4.0 International License)
https://creativecommons.org/licenses/by/4.0/

convergence speed to meet real-time needs. Stochastic iterative methods, owing to their simplicity, are employed for resolving the individual subsystems

$$Uy = b, \tag{2}$$

and

$$Vx = y, \tag{3}$$

in an alternative manner.

Ma et al. [4] analyze the variations exhibited by (1). If the full system is consistent that two subsystems are consistent, Ma et al. [4] use the Randomized Kaczmarz (RK) method [5] to solve each subsystem and propose the Randomized Kaczmarz-Randomized Kaczmarz (RK-RK) method.. In this paper, we keep a watchful eye on the inconsistent case that (2) is inconsistent and (3) is consistent. The Randomized Extended Kaczmarz-Randomized Kaczmarz (REK-RK) method is proposed to resolve this case, which interlaces the REK method [6] for solving first subsystem and the RK method for second subsystem. Dumitrescu [7] has showed that randomized Gauss-Seidel (GRS) method exhibits superiority compared to the REK method. Then, Zhao et al. [8] introduce the RGS-RK method. Drawing inspiration from greedy random algorithms [9, 10] the relaxed greedy extension of the RGS-RK (Greedy Randomized Gauss Seidel-Randomized: GRGS-GRK) has been introduced in [11]. In the view of optimization, Du [12] establish two regularized randomized iterative methods for factorized linear system. Very recently, combining the random average tech, Zhao et al. [13] give block versions of the RK-RK, REK-RK methods, for short as Block RK-RK method (BRK-RK) and Block REK-RK method (BREK-RK), which can be executed concurrently.

The core contribution of our paper lies in, inspired by the works [8, 13], developing the Randomized Average Block Gauss-Seidel (RABGS) method, which eliminates the need to compute the pseudoinverse of submatrix. Then, we make full use of the RABGS method to find the solution to the equation (2) followed by the Randomized Average Block Kaczmarz (RABK) method [13] to solve (3) and introduce the pseudoinverse-free block version of the RGS-RK (Block RGS-RK method: BRGS-RK) method for solving inconsistent factorized linear systems. To steer clear of performing row and column partitioning, we combine the greedy selection strategy with sketch tech to obtain a greedy version of the BRGS-RK method and abbreviate it as Greedy Block Randomized Gauss Seidel-Randomized Kaczmarz (GBRGS-RK). Besides, we illustrate the linear convergence in expectation toward the least squares solution of inconsistent factorized linear systems for both intertwined methods. Numerical experiments demonstrate that the proposed methods requires less computing time and iteration than that of the RGS-RK, GRGS-GRK, BREK-RK methods.

The organization of the paper is as follows. In Section 2, we introduce some notation and revisit the RGS-RK, GRGS-GRK and BREK-RK methods. The GBRGS-RK and BRGS-RK methods and convergence rates are proposed in Section 3. Section 4 contains experimental results demonstrating the efficiency of the presented methods. Finally, in Section 5, we end this paper with some conclusions.

2. Notation and preliminaries

Throughout this paper, we adopt the notation introduced by Ma et al. [4]. For any matrix $G = (g_{ij}) \in \mathbb{R}^{m\times n}$, $\| G \|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} | g_{ij} |^2}$, $\sigma_{\min}(G)$, $\sigma_{\max}(G)$ indicate the Frobenius norm, the smallest nonzero singular value, the largest singular value, respectively. Let I denote the identity matrix with appropriate dimension, $[m] = \{1, 2, \cdots, m\}$ and $[n] = \{1, 2, \cdots, n\}$. For index set $\tilde{I} \subseteq [m]$, $J \subseteq [k]$ and $C \subseteq [n]$, $G_{\tilde{I},:}$ and $G_{:,J}$ or $G_{:,C}$ indicate the row submatrix indexed by $\tilde{I}$ and the column submatrix indexed by J or C, respectively. Besides, for a vector x, we use $x^{\tilde{I}}$ to denote the subvector indexed

by $\tilde{I}$. Let $\mathbb{E}_t$ denote the expected value conditional on the first t iterations i.e. $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \mid j_0,\ i_0,\ j_1,\ i_1 \cdots,\ j_{t-1},\ i_{t-1}]$, where j_t and i_t $(t = 0,\ 1,\ \cdots,\ t-1)$ are the column index and row index chosen at the t-th iteration. Then, from the law of iterated expectation, we have $\mathbb{E}[\mathbb{E}_t[\cdot]] = \mathbb{E}[\cdot]$. We utilize the row partition and column partition approach outlined in [16]. To simplify the notation, Table 1 in [4, 8] summarises the optimal solution of (1), (2) and (3).

Table 1. Summary of notation for linear systems discussed and their solutions

Linear system	Optimal solution
$A\tilde{\beta} = b$	$\tilde{\beta}_\star$
$Uy = b$	$y_\star$
$Vx = y$	$x_\star$

We end this section with revisiting the RGS-RK method [8] in Algorithm 1, the BREK-RK method [13] in Algorithm 2 and the GRGS-GRK method, seen as a greedy version of the RGS-RK method [11], seen as a greedy version of the RGS-RK method, in Algorithm 3.

Algorithm 1 The RGS-RK method

Input: Let U, V, x_0 and y_0.

Output: x_t.

1. For $t = 1,\ 2,\ \ldots$, do.
2. Choose column $U_{:,\ j}$ with probability $\dfrac{\| U_{:,\ j} \|_2^2}{\|U\|_F^2}$.
3. Set

$$y_{t+1} = y_t + \frac{U_{:,\ j}^T (b - U y_t)}{\|U_{:,\ j}\|_2^2} e_j.$$

4. Choose row $V_{i,:}$ with probability $\dfrac{\| V_{i,\ :} \|_2^2}{\|V\|_F^2}$.
5. Set

$$x_{t+1} = x_t + \frac{y_{t+1}^i - V_{i,\ :} x_t}{\|V_{i,\ :}\|_2^2} V_{i,\ :}^T.$$

6. End For.

Algorithm 2 The BREK-RK method

Input: Let $\{\tilde{I}_1,\ \ldots,\ \tilde{I}_{\tilde{s}}\}$, $\{J_1,\ \ldots,\ J_s\}$ be partitions of $[m]$, $[k]$, respectively. U, V, z_0, x_0 and y_0.

Output: x_t.

1. For $t = 1,\ 2,\ \ldots$, do.
2. Choose $j \in [s]$ with probability $\dfrac{\| U_{:,\ J_j} \|_F^2}{\|U\|_F^2}$.
3. Set

$$z_{t+1} = z_t + \frac{\tilde{\alpha}}{\|U_{:,\ J_j}\|_F^2} U_{:,\ J_j} U_{:,\ J_j}^T z_t.$$

4. Choose $i \in [\tilde{s}]$ with probability $\frac{\| V_{\tilde{I}_i,:} \|_F^2}{\|V\|_F^2}$.
5. Set

$$y_{t+1} = y_t + \frac{\tilde{\alpha}}{\|U_{\tilde{I}_i,:}\|_F^2} U_{\tilde{I}_i,:}^T \left(b^{\tilde{I}_i} - z_{t+1}^{\tilde{I}_i} - U_{\tilde{I}_i,:} y_t \right).$$

6. Choose $j \in [\tilde{s}]$ with probability $\frac{\| V_{J_j,:} \|_F^2}{\|V\|_F^2}$.
7. Set

$$x_{t+1} = x_t + \frac{\tilde{\alpha}}{\|V_{J_j,:}\|_F^2} V_{J_j,:}^T \left(y_{t+1}^{J_j} - V_{J_j,:} x_t \right).$$

8. End For.

Algorithm 3 The GRGS-GRK method
Input: $U, V, x_0, \hat{\omega}_U$, and $\hat{\omega}_V$.
Output: x_t.

1. For $t = 1, 2, \ldots$, do.
2. Compute

$$\varepsilon_t^U = \frac{1}{2} \left(\frac{1}{\|U^T(b - Uy_t)\|_2^2} \max_{j \in [k]} \left\{ \frac{\left| U_{:,j}^T (b - Uy_t) \right|^2}{\|U_{:,j}\|_2^2} \right\} + \frac{1}{\|U\|_F^2} \right).$$

3. Determine the index set

$$\mathscr{V}_t^U = \left\{ j \mid \left| U_{:,j}^T (b - Uy_t) \right|^2 \geq \varepsilon_t^U \left\| U^T (b - Uy_t) \right\|_2^2 \|U_{:,j}\|_2^2 \right\}.$$

4. Choose $j \in \mathscr{V}_t^U$ with probability $P_j = p_j^t$, where $p_j^t = 0$ if $j \notin \mathscr{V}_t^U$, $p_j^t > 0$ if $j \in \mathscr{V}_t^U$ and $\sum_{j \in \mathscr{V}_t^U} p_j^t = 1$.
5. Set

$$y_{t+1} = y_t + \hat{\omega}_U \frac{U_{:,j}^T (b - Uy_t)}{\|U_{:,j}\|_2^2} e_j.$$

6. Compute

$$\varepsilon_t^V = \frac{1}{2} \left(\frac{1}{\|y_{t+1} - Vx_t\|_2^2} \max_{i \in [k]} \left\{ \frac{\left| y_{t+1}^i - V_{i,:} x_t \right|^2}{\|V_{i,:}\|_2^2} \right\} + \frac{1}{\|V\|_F^2} \right).$$

7. Determine the index set

$$\mathscr{V}_t^V = \left\{ i \mid \left| y_{t+1}^i - V_{i,:} x_t \right|^2 \geq \varepsilon_t^V \left\| y_{t+1} - V x_t \right\|_2^2 \left\| V_{i,:} \right\|_2^2 \right\}.$$

8. Choose $i \in \mathscr{V}_t^V$ with probability $P_i = p_i^t$, where $p_i^t = 0$ if $i \notin \mathscr{V}_t^V$, $p_i^t > 0$ if $i \in \mathscr{V}_t^V$ and $\sum_{i \in \mathscr{V}_t^V} p_i^t = 1$
9. Set

$$x_{t+1} = x_t + \hat{\omega}_V \frac{y_{t+1}^i - V_{i,:} x_t}{\left\| V_{i,:} \right\|_2^2}.$$

10. End For.

3. The BRGS-RK method

In this section, the RABGS method outlined in Algorithm 4, is presented to solve inconsistent or consistent linear systems $A\bar{x} = b$ with $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Then, we list the framework of the BRGS-RK method in Algorithm 5. We get the expected convergence rate of the BRGS-RK method beginning with one lemma that gives the convergence bound of Algorithm 4.

Algorithm 4 The RABGS method

Input: Let $\{C_1, \ldots, C_v\}$ be partitions of $[n]$, x_0, A, b, $\beta > 0$.

Output: x_t.

1.For $t = 1, 2, \ldots$, do.

2. Choose index $l \in [v]$ with probability $\mathrm{P}_l = \dfrac{\| A_{:, C_l} \|_F^2}{\|A\|_F^2}$.
3. Set

$$\bar{x}_{t+1} = \bar{x}_t + \beta I_{:, C_l} \frac{A_{:, C_l}^T (b - A\bar{x}_t)}{\left\| A_{:, C_l} \right\|_F^2}.$$

4. End For.

Algorithm 5 The BRGS-RK method

Input: Let $\{J_1, \ldots, J_s\}$ be partitions of $[k]$, x_0, y_0, U, V, b, $\alpha > 0$

Output: x_t.

1.For $t = 1, 2, \ldots$, do.

2. Choose index $j \in [s]$ with probability $\mathrm{P}_j = \dfrac{\| U_{:, J_j} \|_F^2}{\|U\|_F^2}$.
3. Set

$$y_{t+1} = y_t + \alpha I_{:, J_j} \frac{U_{:, J_j}^T (b - U y_t)}{\left\| U_{:, J_j} \right\|_F^2}.$$

4. Choose index $i \in [s]$ with probability $\mathrm{P}_i = \dfrac{\| V_{J_i, :} \|_F^2}{\|V\|_F^2}$.
5. Set

$$x_{t+1} = x_t + \alpha \frac{y^{J_i} - V_{J_i,:} x_t}{\|V_{J_i,:}\|_F^2} V_{J_i,:}^T.$$

6. End For.

Lemma 1 Consider the linear system $A\bar{x} = b$, be consistent or inconsistent, with the full column rank coefficient matrix $A \in \mathbb{R}^{m\times n}$. With any initial guess $x_0 \in \mathbb{R}(A^T)$ and $\beta \in \left(0, \frac{2}{\mu_{\max}}\right)$, the iteration $\{\bar{x}_t\}_{t=0}^{\infty}$, generated by Algorithm 1, converges to the unique least squares solution $\bar{x}_\star = A^\dagger b$. It holds that

$$\mathbb{E} \| \bar{x}_{t+1} - \bar{x}_\star \|_2^2 \leq \left(1 - \left(2\beta - \beta^2 \mu_{\max}\right) \frac{\sigma_{\min}^2(A)}{\|A\|_2^2}\right)^{t+1} \kappa_A^2 \|\bar{x}_0 - \bar{x}_\star\|_2^2,$$

with $\mu_{\max} = \max_{l\in[n]} \left\{ \frac{\sigma_{\max}^2(A_{:,C_l})}{\| A_{:,C_l} \|_F^2} \right\}$ and $\kappa_A^2 = \| A^\dagger \|_2^2 \| A \|_2^2$.

Proof. From the update formula in the RABGS method, we get

$$\begin{aligned}
\mathbb{E}_{t+1} \|A(\bar{x}_{t+1} - \bar{x}_\star)\|_2^2 &= \mathbb{E}_{t+1} \left\| A \left(\bar{x}_t - \bar{x}_\star + \beta I_{:,C_l} \frac{A_{:,C_l}^T (b - A\bar{x}_t)}{\|A_{:,C_l}\|_F^2} \right) \right\|_2^2 \\
&= \|A(\bar{x}_t - \bar{x}_\star)\|_2^2 + \beta^2 \mathbb{E}_{t+1} \frac{\left\| A_{:,C_l} A_{:,C_l}^T A(\bar{x}_t - \bar{x}_\star) \right\|_2^2}{\|A_{:,C_l}\|_F^4} - 2\beta^2 \mathbb{E}_{t+1} \frac{\left\| A_{:,C_l}^T A(\bar{x}_t - \bar{x}_\star) \right\|_2^2}{\|A_{:,C_l}\|_F^2} \\
&\leq \|A(\bar{x}_t - \bar{x}_\star)\|_2^2 + \beta^2 \frac{\sigma_{\max}^2(A_{:,C_l})}{\|A_{:,C_l}\|_F^2} \mathbb{E}_{t+1} \frac{\left\| A_{:,C_l}^T A(\bar{x}_t - \bar{x}_\star) \right\|_2^2}{\|A_{:,C_l}\|_F^2} - 2\beta \mathbb{E}_{t+1} \frac{\left\| A_{:,C_l}^T A(\bar{x}_t - \bar{x}_\star) \right\|_2^2}{\|A_{:,C_l}\|_F^2} \\
&\leq \|A(\bar{x}_t - \bar{x}_\star)\|_2^2 - \left(2\beta - \beta^2 \mu_{\max}\right) \sum_{l\in[n]} \frac{\|A_{:,C_l}\|_F^2}{\|A\|_F^2} \frac{\left\| A_{:,C_l}^T A(\bar{x}_t - \bar{x}_\star) \right\|_2^2}{\|A_{:,C_l}\|_F^2} \\
&\leq \left(1 - \left(2\beta - \beta^2 \mu_{\max}\right) \frac{\sigma_{\min}^2(A)}{\|A\|_F^2}\right) \|A(\bar{x}_t - \bar{x}_\star)\|_2^2. \qquad (4)
\end{aligned}$$

Using the low of full expectation gives

$$\mathbb{E}\|A(\bar{x}_{t+1} - \bar{x}_\star)\|_2^2 \leq \left(1 - \left(2\beta - \beta^2 \mu_{\max}\right) \frac{\sigma_{\min}^2(A)}{\|A\|_F^2}\right)^{t+1} \|A(\bar{x}_0 - \bar{x}_\star)\|_2^2. \qquad (5)$$

It follows from (5) that

$$\mathbb{E}\|\bar{x}_{t+1}-\bar{x}_\star\|_2^2 \le \left(1-\left(2\beta-\beta^2\mu_{\max}\right)\frac{\sigma_{\min}^2(A)}{\|A\|_F^2}\right)^{t+1}\|A^\dagger\|_2^2\,\|A(\bar{x}_0-\bar{x}_\star)\|_2^2$$

$$\le \left(1-\left(2\beta-\beta^2\mu_{\max}\right)\frac{\sigma_{\min}^2(A)}{\|A\|_F^2}\right)^{t+1}\kappa_A^2\,\|\bar{x}_0-\bar{x}_\star\|_2^2, \tag{6}$$

last line is based on the fact that $\|A(\bar{x}_0-\bar{x}_\star)\|_2^2 \le \|A\|_2^2\|\bar{x}_0-\bar{x}_\star\|_2^2$. This completes the proof. □

Remark 1 For the RABGS method, if we set $v=n$, then the RABGS method reduces to the relaxed RGS method.

Remark 2 From Algorithm 1, it follows that

$$\begin{aligned} r_{t+1} &= A^T(b-A\bar{x}_{t+1}) \\ &= A^Tb - A^TA\left(\bar{x}_t+\beta I_{:,\,C_l}\frac{A_{:,\,C_l}^T(b-A\bar{x}_t)}{\|A_{:,\,C_l}\|_F^2}\right) \\ &= A^Tb - A^TA\bar{x}_t - \beta A^TAI_{:,\,C_l}\frac{r_t^{C_l}}{\|A_{:,\,C_l}\|_F^2} \\ &= r_t - \beta B_{:,\,C_l}\frac{r_t^{C_l}}{\|A_{:,\,C_l}\|_F^2}, \end{aligned}$$

where $B_{:,\,C_l}$ denotes the columns indexed by C_l of A^TA. If we have $B=A^TA$ at the beginning, then the RABGS method could be implemented via a residual-oriented scheme.

The following theorem may be established for the convergence rate of the BRGS-RK method.

Theorem 1 Consider the inconsistent factorized linear system (1), subsystems (2) and (3) with optimal solutions $\tilde{\beta}_\star$, $y_\star$ and $x_\star$, where $A\in\mathbb{R}^{m\times n}$ is often low rank matrix and $U\in\mathbb{R}^{m\times k}$, $V\in\mathbb{R}^{k\times n}$ are full column rank matrix. Let $\alpha\in\left(0,\ \frac{2}{\max(\gamma_{\max},\ \tilde{\mu}_{\max})}\right)$, the iteration sequence $\{x_t\}_{t=0}^{\infty}$, generated by Algorithm 5, converges to $\tilde{\beta}_\star$. Moreover, we have

$$\begin{aligned} \mathbb{E}\|x_{t+1}-x_\star\|_2^2 &\le \frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\|V\|_F^2}\|y_\star\|_2^2\sum_{\ell=0}^{t+1}\left(\varsigma^{t+1-\ell}(1+\varepsilon)^\ell\xi^\ell\right)+(1+\varepsilon)^{t+1}\xi^{t+1}\|x_\star\|_2^2 \\ &\le (1+\varepsilon)^{t+1}\tilde{\varsigma}^{t+1}\left(\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\alpha^2\delta_{\max}}{\varepsilon^2\|V\|_F^2}\|y_\star\|_2^2\right), \end{aligned} \tag{7}$$

with $\xi=\left(1-\left(2\alpha-\alpha^2\gamma_{\max}\right)\frac{\sigma_{\min}^2(V)}{\|V\|_F^2}\right)$, $\varsigma=\left(1-\left(2\alpha-\alpha^2\tilde{\mu}_{\max}\right)\frac{\sigma_{\min}^2(U)}{\|U\|_F^2}\right)\kappa_U^2$, $\kappa_U^2=\|U^\dagger\|_2^2\|U\|_2^2$, $\tilde{\varsigma}=\max\{\xi,\ \varsigma\}$, $\tilde{\mu}_{\max}=\max_{j\in[s]}\left\{\frac{\sigma_{\max}^2(U_{:,\,J_j})}{\|U_{:,\,J_j}\|_F^2}\right\}$ and $\gamma_{\max}=\max_{i\in[s]}\left\{\frac{\sigma_{\max}^2(V_{J_i,\,:})}{\|V_{J_i,\,:}\|_F^2}\right\}$.

Proof. Applying the RABK method [13] to $Vx_t = y_\star$ gives $\tilde{x}_{t+1} = x_t + \alpha \frac{y_\star - V_{J_i,:}x_t}{\|V_{J_i,:}\|} V_{J_i,:}^T$. By the definition of conditional expectation, we get

$$\begin{aligned}
\mathbb{E}_{t+1}\|\tilde{x}_{t+1} - x_\star\|_2^2 &= \mathbb{E}_{t+1}\left\| x_t - x_\star + \alpha \frac{y_\star - V_{J_i,:}x_t}{\|V_{J_i,:}\|_F^2} V_{J_i,:}^T \right\|_2^2 \\
&= \|x_t - x_\star\|_2^2 + 2\alpha \mathbb{E}_{t+1}\left\langle x_t - x_\star, \frac{V_{J_i,:}(x_\star - x_t)}{\|V_{J_i,:}\|_F^2} V_{J_i,:}^T \right\rangle + \alpha^2 \mathbb{E}_{t+1}\left\| \frac{V_{J_i,:}^T V_{J_i,:}(x_\star - x_t)}{\|V_{J_i,:}\|_F^2} \right\|_2^2 \\
&\le \|x_t - x_\star\|_2^2 - 2\alpha \mathbb{E}_{t+1} \frac{\|V_{J_i,:}(x_t - x_\star)\|_2^2}{\|V_{J_i,:}\|_F^2} + \alpha^2 \frac{\sigma_{\max}^2(V_{J_i,:})}{\|V_{J_i,:}\|_F^2} \mathbb{E}_{t+1} \frac{\|V_{J_i,:}(x_t - x_\star)\|_2^2}{\|V_{J_i,:}\|_F^2} \\
&\le \|x_t - x_\star\|_2^2 - \left(2\alpha - \alpha^2 \gamma_{\max}\right) \sum_{i \in [s]} \frac{\|V_{J_i,:}\|_F^2}{\|V\|_F^2} \frac{\|V_{J_i,:}(x_t - x_\star)\|_2^2}{\|V_{J_i,:}\|_F^2} \\
&= \|x_t - x_\star\|_2^2 - \left(2\alpha - \alpha^2 \gamma_{\max}\right) \frac{\|V(x_t - x_\star)\|_2^2}{\|V\|_F^2} \\
&\le \xi \|x_t - x_\star\|_2^2 .
\end{aligned} \tag{8}$$

Thus, $\mathbb{E}\|\tilde{x}_{t+1} - x_{t+1}\|_2^2 \le \xi \mathbb{E}\|x_t - x_\star\|_2^2$. Along the similar lines as in (8), we have

$$\begin{aligned}
\mathbb{E}_{t+1}\|x_{t+1} - \tilde{x}_{t+1}\|_2^2 &= \mathbb{E}_{t+1}\mathbb{E}_{t+1}^i \|x_{t+1} - \tilde{x}_{t+1}\|_2^2 \\
&\le \mathbb{E}_{t+1}\mathbb{E}_{t+1}^i \frac{\alpha^2 \gamma_{\max}^2(V_{J_i,:})}{\|V_{J_i,:}\|_F^2} \frac{\left\| y_{t+1}^{J_i} - y_\star^{J_i} \right\|_2^2}{\|V_{J_i,:}\|_F^2} \\
&\le \mathbb{E}_{t+1} \alpha^2 \gamma_{\max} \frac{\|y_{t+1} - y_\star\|_2^2}{\|V\|_F^2} .
\end{aligned} \tag{9}$$

Then,

$$\mathbb{E}\|x_{t+1} - \tilde{x}_{t+1}\|_2^2 = \mathbb{E}\mathbb{E}_{t+1}\|x_{t+1} - \tilde{x}_{t+1}\|_2^2 \le \frac{\alpha^2 \gamma_{\max}}{\|V\|_F^2} \mathbb{E}\|y_{t+1} - y_\star\|_2^2 . \tag{10}$$

From (8) and (10) and the proof of [13, Theorem 4.1], it follows that

$$\begin{aligned}\mathbb{E}\|x_{t+1}-x_\star\|_2^2 &= \mathbb{E}\|x_{t+1}-\tilde{x}_{t+1}+\tilde{x}_{t+1}-x_\star\|_2^2\\ &\le (1+1/\varepsilon)\mathbb{E}\|x_{t+1}-\tilde{x}_{t+1}\|_2^2+(1+\varepsilon)\mathbb{E}\|\tilde{x}_{t+1}-x_\star\|_2^2\\ &\le \frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\|V\|_F^2}\mathbb{E}\,\|y_{t+1}-y_\star\|_2^2+(1+\varepsilon)\xi\mathbb{E}\|x_t-x_\star\|_2^2.\end{aligned} \tag{11}$$

Substituting Lemma 1 into (11), we obtain

$$\begin{aligned}\mathbb{E}\|x_{t+1}-x_\star\|_2^2 &\le \frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\|V\|_F^2}\varsigma^{t+1}\,\|y_\star\|_2^2+(1+\varepsilon)\xi\mathbb{E}\|x_t-x_\star\|_2^2\\ &\le \frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\|V\|_F^2}\,\|y_\star\|_2^2\left(\varsigma^{t+1}+\varsigma^t(1+\varepsilon)\xi\right)+(1+\varepsilon)^2\xi^2\mathbb{E}\|x_{t-1}-x_\star\|_2^2\\ &\le \ldots\\ &\le \frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\|V\|_F^2}\,\|y_\star\|_2^2\sum_{\ell=0}^{t+1}\left(\varsigma^{t+1-\ell}(1+\varepsilon)^\ell\xi^\ell\right)+(1+\varepsilon)^{t+1}\xi^{t+1}\|x_\star\|_2^2\\ &\le \frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\|V\|_F^2}\,\|y_\star\|_2^2\,\tilde{\varsigma}^{t+1}\sum_{\ell=0}^{t+1}(1+\varepsilon)^\ell+(1+\varepsilon)^{t+1}\xi^{t+1}\|x_\star\|_2^2\\ &\le (1+\varepsilon)^{t+1}\,\tilde{\varsigma}^{t+1}\left(\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\alpha^2\gamma_{\max}}{\varepsilon^2\|V\|_F^2}\,\|y_\star\|_2^2\right).\end{aligned} \tag{12}$$

We complete the proof.

Implementing a column paving for the RABGS method and the BRGS-RK method introduces a substantial overhead in terms of time complexity. To address this problem, we propose the greedy version of the BRGS-RK method, interlacing the greedy randomized row method [14] to (3) and the greedy randomized column method [14] to (4), listed in Algorithm 3. For the convergence rate of the GBRGS-RK method, we can establish the following theorem.

Theorem 2 Consider the inconsistent factorized linear system (1), subsystems (2) and (3) with optimal solutions $\tilde{\beta}_\star$, $y_\star$ and $x_\star$, where $A\in\mathbb{R}^{m\times n}$ is often low rank matrix and $U\in\mathbb{R}^{m\times k}$, $V\in\mathbb{R}^{k\times n}$ are full column rank matrix. The iteration sequence $\{x_t\}_{t=0}^{\infty}$, generated by Algorithm 6, converges to $\tilde{\beta}_\star$. Moreover, we have

$$\begin{aligned}\|x_{t+1}-x_\star\|_2^2 &\le (1+\varepsilon)^{t+1}\vartheta^{t+1}\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\,\|y_\star\|_2^2}{\sigma_{\min}^2(V_{\mathcal{V}_t^V,:})}\sum_{\ell=0}^{t}\rho^{t+1-\ell}+(1+\varepsilon)^t\rho^t\\ &\le (1+\varepsilon)^{t+1}\hat{\vartheta}^{t+1}\left(\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\,\|y_\star\|_2^2}{\varepsilon^2\sigma_{\min}^2(V_{\mathcal{V}_t^V,:})}\right),\end{aligned}$$

where

$$\hat{\vartheta}^{t+1} = \max\{\rho,\ \vartheta\},$$

$$\vartheta = \left(1 - \frac{\sigma^2_{\min}(A)}{\sigma^2_{\max}(V_{\mathscr{V}_t^V,\ :})}\right),$$

$$\rho = \left(1 - \tilde{\gamma}_t \frac{\sum_{j\in\mathscr{V}_t^U} \|U_{:,\ j}\|_2^2}{\|U\|_F^2} \frac{\sigma^2_{\max}(U)}{\sigma^2_{\max}(U_{:,\ \mathscr{V}_t^V})}\right),$$

$$\tilde{\gamma}_t = \frac{1}{2}\left(1 + \frac{\|U\|_F^2}{\|U\|_F^2 - \sum_{j\in\Omega_t} \|U_{:,\ j}\|_2^2}\right),$$

and $\Omega_t = \{j | b^i - U_{:,\ j} y_t = 0,\ j \in [k]\}$.

Algorithm 6 The GBRGS-RK method

Input: x_0, y_0, U, V, b.

Output: x_t.

1. For $t = 1,\ 2,\ ...,$ do.
2. Execute Steps 2-3 of Algorithm 3.
3. Update

$$y_{t+1} = y_t + \frac{\eta_t^T U^T (b - U y_t)}{\|U\eta_t\|_2^2} \eta_t,$$

with $\eta_t = \sum_{j\in\mathscr{V}_t^U} U_{:,\ j}^T (b - U y_t) \upsilon_j$.

4. Execute Steps 6-7 of Algorithm 3.
5. Update

$$x_{t+1} = x_t + \frac{\tilde{\eta}_t^T (y_{t+1} - V x_t)}{\|V^T \tilde{\eta}_t\|_2^2} V^T \tilde{\eta}_t,$$

with $\tilde{\eta}_t = \sum_{i\in\mathscr{V}_t^V} (y_{t+1}^i - V_{i,\ :} x_t) \tilde{\upsilon}_i$.

6. End For.

Proof. Set $\tilde{x}_{t+1} = x_t + \frac{\tilde{\eta}_t^T (y_* - V x_t)}{\|\ V^T \tilde{\eta}_t\ \|^2} V^T \tilde{\eta}_t$ with $y_* = V x_*$, we immediately have

$$\|\tilde{x}_{t+1} - x_*\|_2^2 = \|x_t - x_*\|_2^2 + \left\| \frac{\tilde{\eta}_t^T (y_* - V x_t)}{\|\ V^T \tilde{\eta}_t\ \|^2} V^T \tilde{\eta}_t \right\|_2^2 - 2\left\langle x_t - x_*, \frac{\tilde{\eta}_t^T V (x_t - x_*)}{\|\ V^T \tilde{\eta}_t\ \|^2} V^T \tilde{\eta}_t \right\rangle$$

$$= \|x_t - x_*\|_2^2 - \frac{\left\|\tilde{\eta}_t^T V (x_t - x_*)\right\|_2^2}{\|V^T \tilde{\eta}_t\|_2^2}$$

$$
\begin{aligned}
&\leq \|x_t - x_*\|_2^2 - \frac{\|\tilde{\eta}_t\|_2^2 \|V(x_t - x_*)\|_2^2}{\|V^T \tilde{\eta}_t\|_2^2} \\
&\leq \|x_t - x_*\|_2^2 - \frac{\|V(x_t - x_*)\|_2^2}{\sigma_{\max}^2(V_{\mathscr{Y}_t^V,:})} \\
&\leq \vartheta \|x_t - x_*\|_2^2,
\end{aligned} \tag{13}
$$

the second inequality is derived from

$$
\frac{1}{\|V^T \tilde{\eta}_t\|_2^2} = \frac{1}{\left\| V_{\mathscr{Y}_t^V,:}^T (y_{t+1}^{\mathscr{Y}_t^V} - V_{\mathscr{Y}_t^V,:} x_t) \right\|_2^2} \geq \frac{1}{\sigma_{\max}^2(V_{\mathscr{Y}_t^V,:} x_t)} \frac{1}{\|\tilde{\eta}_t\|_2^2}.
$$

For the bound of $\|x_{t+1} - \tilde{x}_{t+1}\|_2^2$, we get

$$
\begin{aligned}
\|x_{t+1} - \tilde{x}_{t+1}\|_2^2 &= \left\| \frac{\tilde{\eta}_t^T (y_{t+1} - y_*)}{\|V^T \tilde{\eta}_t\|^2} V^T \tilde{\eta}_t \right\|_2^2 \\
&= \frac{\left\| \tilde{\eta}_t^T (y_{t+1} - y_*) \right\|_2^2}{\|V^T \tilde{\eta}_t\|_2^2} \\
&\leq \frac{\left\| \tilde{\eta}_t^T \right\|_2^2 \|y_{t+1} - y_*\|_2^2}{\|V^T \tilde{\eta}_t\|_2^2} \\
&\leq \frac{\|y_{t+1} - y_*\|_2^2}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})},
\end{aligned} \tag{14}
$$

the last line follows by

$$
\frac{1}{\|V^T \tilde{\eta}_t\|_2^2} = \frac{1}{\left\| V_{\mathscr{Y}_t^V,:}^T (y_{t+1}^{\mathscr{Y}_t^V} - V_{\mathscr{Y}_t^V,:} x_t) \right\|_2^2} \leq \frac{1}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:} x_t)} \frac{1}{\|\tilde{\eta}_t\|_2^2}.
$$

Based on (13) and (14), similar to (11), we give

$$
\begin{aligned}
\|x_{t+1} - x_\star\|_2^2 &= \|x_{t+1} - \tilde{x}_{t+1} + \tilde{x}_{t+1} - x_\star\|_2^2 \\
&\leq (1 + 1/\varepsilon) \|x_{t+1} - \tilde{x}_{t+1}\|_2^2 + (1 + \varepsilon) \|\tilde{x}_{t+1} - x_\star\|_2^2
\end{aligned}
$$

$$\leq \frac{1+1/\varepsilon}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})}\|y_{t+1}-y_\star\|_2^2+(1+\varepsilon)\vartheta\|x_t-x_\star\|_2^2. \tag{15}$$

Plugging [14, Theorem 4] with $\theta_2=\dfrac{1}{2}$ into (15), gives

$$\begin{aligned}
\|x_{t+1}-x_\star\|_2^2 &\leq \frac{(1+1/\varepsilon)\rho^{t+1}}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})}\|y_\star\|_2^2+(1+\varepsilon)\vartheta\|x_t-x_\star\|_2^2\\
&\leq (1+\varepsilon)^2\vartheta^2\|x_t-x_\star\|_2^2+\frac{(1+1/\varepsilon)\|y_\star\|_2^2}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})}\left(\rho^{t+1}+\rho^t(1+\varepsilon)\vartheta\right)\\
&\leq \ldots\\
&\leq (1+\varepsilon)^{t+1}\vartheta^{t+1}\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\|y_\star\|_2^2}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})}\sum_{\ell=0}^{t}\rho^{t+1-\ell}+(1+\varepsilon)^t\vartheta^t\\
&\leq (1+\varepsilon)^{t+1}\hat{\vartheta}^{t+1}\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\|y_\star\|_2^2}{\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})}\hat{\vartheta}^{t+1}\sum_{\ell=0}^{t}(1+\varepsilon)^t\\
&\leq (1+\varepsilon)^{t+1}\hat{\vartheta}^{t+1}\left(\|x_\star\|_2^2+\frac{(1+1/\varepsilon)\|y_\star\|_2^2}{\varepsilon^2\sigma_{\min}^2(V_{\mathscr{Y}_t^V,:})}\right),
\end{aligned} \tag{16}$$

this concludes the proof of the Theorem 2.

Remark 3 In contrast to the randomized block Gauss-Seidel algorithm [15], Algorithm 4 avoids the pseudoinverse of $A_{:,C_l}$. Owing to the predefined block partitions in both the RABGS and BRGS-RK methods, these algorithms are capable of supporting parallel computation, which can notably boost computational efficiency in multi-core CPU or GPU environments. By leveraging a greedy strategy, the GBRGS-RK method dynamically selects components with larger numerical values in the residual vector $U^T(b-Uy_t)$ and $y_{t+1}-Vx_t$ to build submatrix $U_{:,\mathscr{Y}_t^U}$ and $V_{\mathscr{Y}_t^V,:}$ respectively, thereby addressing the time overhead issue caused by partitioning in the Algorithm 5. As the sketching tech, which allows multiple indices to be used in each iteration update, the GBRGS-RK method exhibits a faster convergence speed compared to Algorithm 3.

4. Numerical experiments

We report the numerical results for inconsistent factorized linear systems with the RGS-RK, BREK-RK, GRGS-GRK, BRGS-RK, GBRGS-RK, in this section.We have released the code in our GitHub repository at the following address: https://github.com/Fanyuanhangzhang/GBRGSRK. For the fairness, we set $\alpha=\dfrac{1.75}{\max(\gamma_{\max},\,\tilde{\mu}_{\max})}$ and $\tilde{\alpha}=\dfrac{1.75}{\max(\gamma_{\max},\,\hat{\mu}_{\max},\,\tilde{\mu}_{\max})}$, with $\hat{\mu}_{\max}=\max_{i\in[s]}\left\{\dfrac{\sigma_{\max}^2(U_{\tilde{I}_i,:})}{\|U_{\tilde{I}_i,:}\|_F^2}\right\}$. For the coefficient matrix A, we use the following choices: the random matrices generated by MATLAB function randn, synthetic data, wine quality (winequality-red with $1,599\times 12$ elements, winequality-white with $4,898\times 12$ elements), bike sharing with $17,379\times 16$ elements and the Phillips's problem

with ill-posed coefficient matrix. All experiments presented in this section are performed in MATLAB R2023a on a personal computer with 2.40 GHz central processing unit (Intel(R) Core(TM) i7 CPU), 16.00 GB memory, and Windows operating system (Windows 11).

We compare the above methods in terms of the iteration counts (denoted as IT) and the computing cost (denoted as CPU time), be medians of averaged over 50 runs. All the tests start from an initial vector $x_0 = \text{zeros}\ (k,\ 1)$, $y_0 = \text{zeros}\ (k,\ 1)$, and are terminated once the Relative Solution Error (RES), defined by $\text{RES} = \dfrac{\| x_t - x_\star \|_2^2}{\| x_\star \|_2^2}$, $x_\star = \text{randn}\ (n,\ 1)$, satisfies $\text{RES} < 10^{-6}$ or the number of the iteration steps exceeds 100,000. Besides, the speed-up of BRGS-RK against RGS-RK and BREK-RK and the speed-up of GBRGS-RK against GRGS-GRK are defined as follows:

$$\text{speed} - \text{up1} = \frac{\text{CPU of BRGS-RK}}{\text{CPU of RGS-RK}},$$

$$\text{speed} - \text{up2} = \frac{\text{CPU of BRGS-RK}}{\text{CPU of BREK-RK}},$$

and

$$\text{speed} - \text{up3} = \frac{\text{CPU of GBRGS-RK}}{\text{CPU of GRGS-GRK}}.$$

Example 1 In this example, for the coefficient matrices U and V, we use the following two choices: Type I is generated by synthesis mode and Type II is derived from the MATLAB function randn. For Type I, like to [16], we set $U = U_1 \times D$ and $V = V_1^T$ with $[U_1,\ \sim] = qr(\text{randn}\ (m,\ r),\ 0)$, $[V_1,\ \sim] = qr(\text{randn}\ (n,\ r),\ 0)$ and $D = \text{diag}\ (1 + (\kappa - 1).*\ \text{rand}\ (r,\ 1))$. For Type II, $U = \text{randn}\ (m,\ k)$ and $V = \text{randn}\ (k,\ n)$. We compute the right hand $b = Ax_\star + \theta r_0 / \|r_0\|_2$, where r_0 is in the null space of A^T by using MATLAB function null, with a controlled parameter $\theta = 0.1$.

In Tables 2 and 3, we report the IT and CPU time for the BRGS-RK, RGK-RK and BREK-RK methods for Type I and Type II matrix. We observe that (i) the GBRGS-RK method triumphs over other methods in computing cost and iterations. (ii) For Type I, the speedup1 is at least 0.3152 and the biggest arrives at 0.8090, the minimum of the speedup2 is 0.5859 and the maximum is 1.0472, the speedup3 has a minimum of 0.1200 and peaks at 0.2177. The corresponding value of speedup1, speedup2 and speedup3 for Type II are 0.0602, 0.1026, 0.5981, 0.8891 and 0.1156, 0.2449. In Figures 1 and 2, we plot the curves of $\log_{10}$ (RES) versus the number of IT for type I and type II matrix with different θ, and again conclude (i).

Example 2 In this example, the GBRGS-RK, BRGS-RK, BREK-RK, GRGS-GRK and RGS-RK methods are employed to tackle the real word data sets on wine quality and bike sharing, retrieved from the The University of California, Irvine Machine Learning Repository [17]. The MATLAB function *nnmf* is used to generate nonegative matrices $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{k \times n}$. For the bike sharing data sets, $k = 2$ and the block partitioning contains two elements. For the wine quality data sets, $k = 3$ and the block partitioning encompasses three candidates. We calculate $b = Ax_\star + \theta r_0 / \|r_0\|_2$, where $r_0 \in \text{null}(A^T)$ with $\theta = 0.01$. In Figure 3, we plot the curves of $\log_{10}$ (RES) versus the number of Iteration (IT) for wine quality and bike sharing datasets. From Table 4 and Figure 3, the BGRS-RK method converges faster than the BREK-RK and RGS-RK methods for practical problems in IT and CPU time. The maximum of speedup1 is 0.7376, the minimum of it is 0.5627, and that of speedup2 are 0.4648 and 0.3297. The GBRGS-RK method outperforms better than the GRGS-GRK method with relaxation factors $\hat{\omega}_U = \hat{\omega}_v < 1$. The former less effectively than the latter while $\hat{\omega}_U = \hat{\omega}_v > 1$. From [11], we know that the best relaxation parameter will be got from the experiments, in this point, our Algorithm 6 also makes sense.

Example 3 In this example, the GBRGS-RK, BRGS-RK, BREK-RK, GRGS-GRK and RGS-RK methods are applied to the Phillips's problem [18], generated by MATLAB function phillips (n) with $n = 1,024$. The right-hand $b = Ax_\star +$

$\theta \frac{r_0}{\|r_0\|_2^2}$, with $\theta = 0.01$ and $r_0 = \text{sprandn}(1{,}024,\ 1,\ 0.02)$. Here U and V are generated by `nnmf` with $m = n = 1024$ and $k = 8$. The condition numbers of A, U and V are 1.7661×10^{20}, 5.1234 and 3.4983 respectively. We report IT and CPU time in Table 5 and plot the curves of $\log_{10}$ (RES) versus the number of iteration and solutions given by these algorithms in Figure 4. From Table 5 and Figure 4, we see that the performance of the GBRGS-RK method surpasses that of the others and the GBRGS-RK method needs less IT and CPU time.

Example 4 In this example, we generate the random matrices U and V by the MATLAB function randn to compare the GBRGS-RK and BRGS-RK methods with the Random Singular Decomposition (RSVD) [19] and Random QR decomposition (RQR) [20] methods. The speed up of GBRGS-RK against RSVD and RQR defined as

$$\text{speed} - \text{up4} = \frac{\text{CPU of GBRGS-RK}}{\text{CPU of RSVD}},$$

and

$$\text{speed} - \text{up5} = \frac{\text{CPU of GBRGS-RK}}{\text{CPU of RQR}}.$$

The right hand b is computed via the formula $b = Ax_\star + \theta r_0/\|r_0\|_2$, where r_0 is in the null space of A^T by using MATLAB function null, with $\theta = 0.1$ and $A = UV$. From Table 6, we can see that the CPU of the GBRGS-RK method is far lower than that of other methods. These results indicate that the GBRGS-RK method exhibits greater competitiveness compared with other approaches.

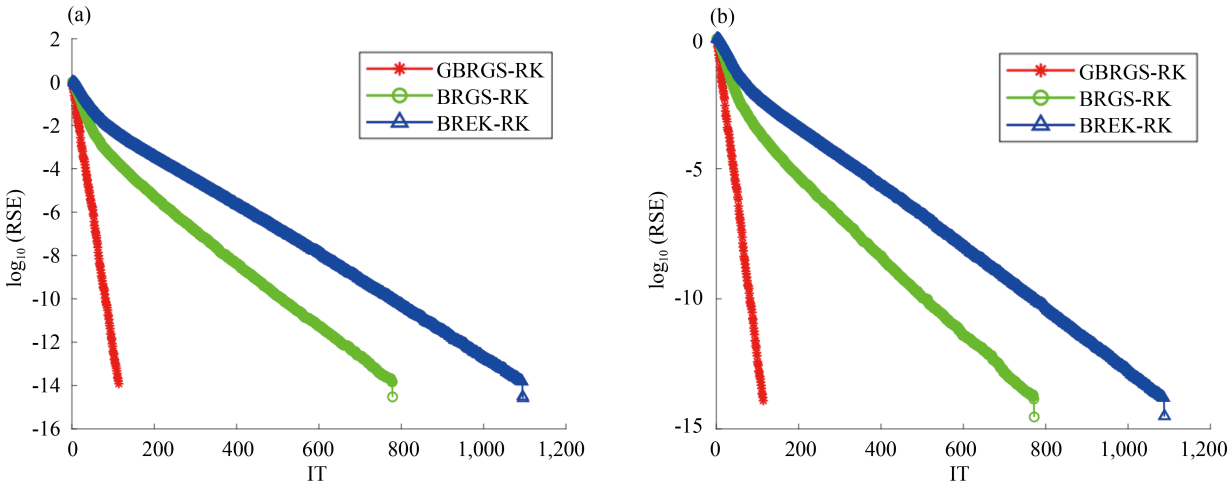

Figure 1. RES versus IT for the GBRGS-RK, BRGS-RK and BREK-RK methods for Type I matrices with $r = 200$, $\kappa = 6$, (a) $\theta = 0.01$; (b) $\theta = 0.1$

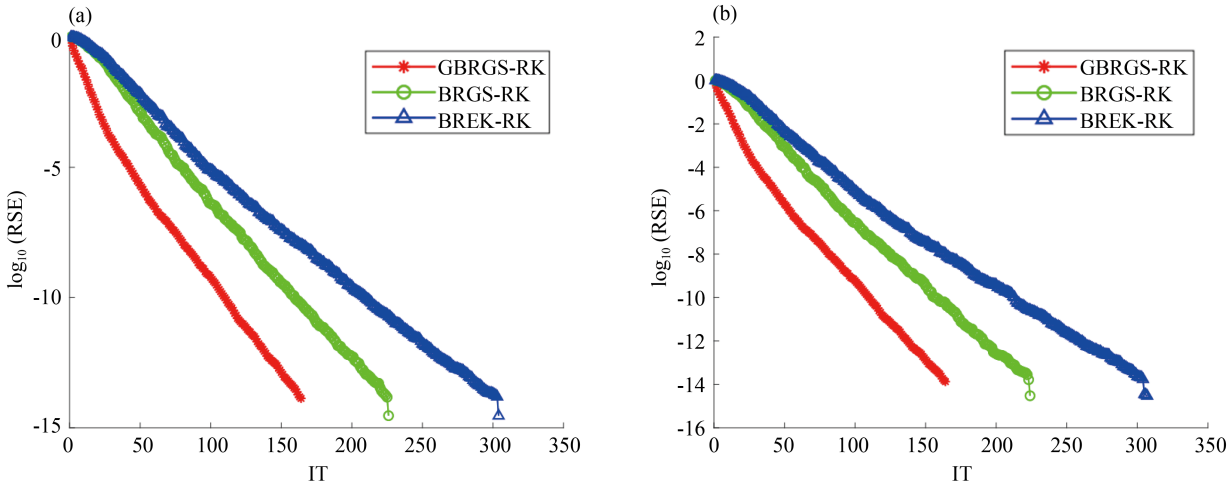

Figure 2. RES versus IT for the GBRGS-RK, BRGS-RK and BREK-RK methods for Type II matrices with $U = \text{randn}(5,000,\ 200)$, $V = \text{randn}(200,\ 500)$ and (a) $\theta = 0.01$; (b) $\theta = 0.1$

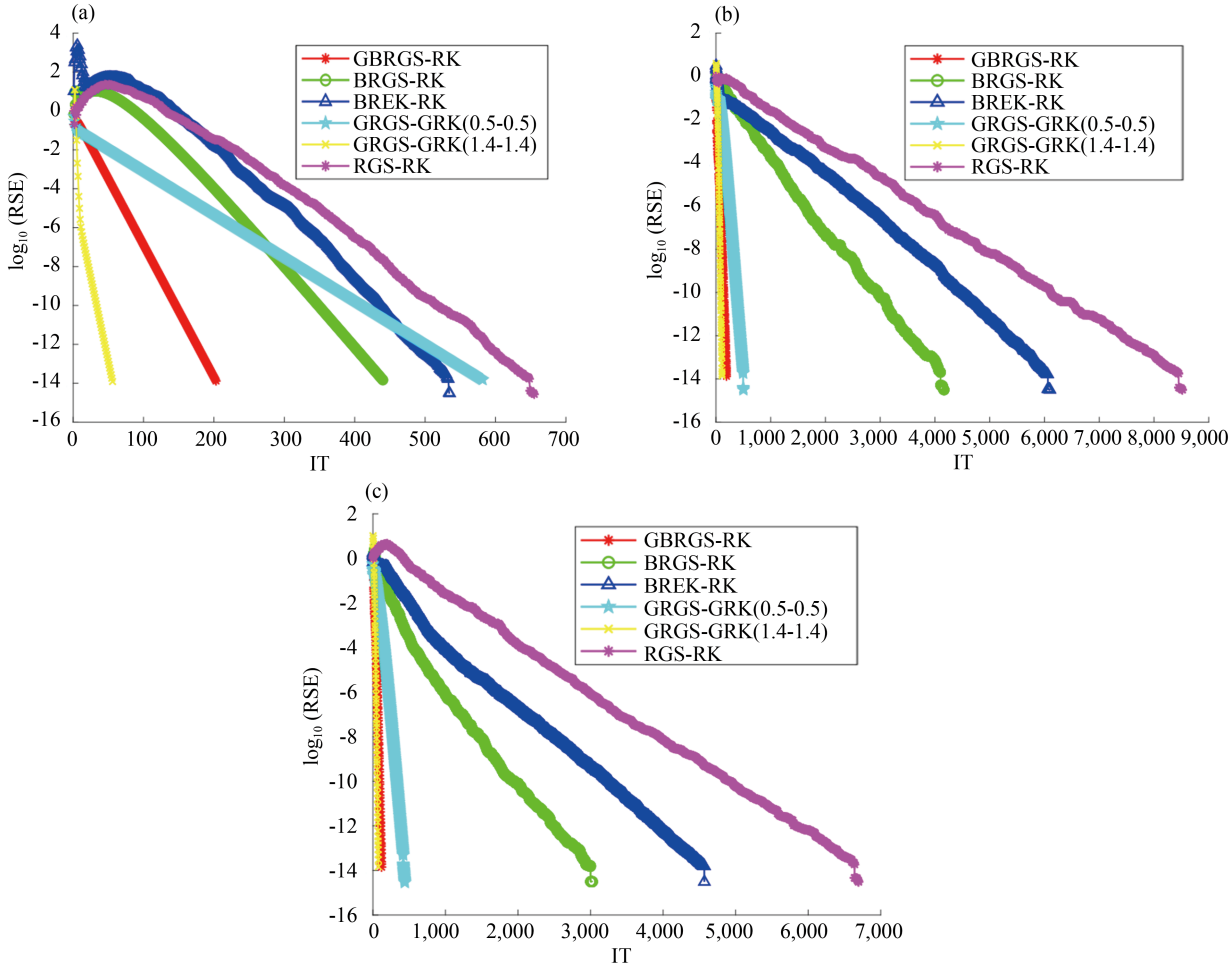

Figure 3. RES versus IT for the GBRGS-RK, BRGS-RK, BREK-RK, GRGS-GRK(1.4, 1.4), GRGS-GRK(0.5, 0.5) and RGS-RK methods for bike datasets (a), red wine (b) and white wine (c)

Table 2. IT and CPU of the GBRGS-RK, GRGS-GRK、BRGS-RK、BREK-RK and RGS-RK methods for Type I

$m \times n$			GBRGS-RK		GRGS-GRK		speed-up3	BGRS-RK		BREK-RK		speed-up2	RGS-RK		speed-up1
	r	κ	IT	CPU	IT	CPU		IT	CPU	IT	CPU		IT	CPU	
$4,000 \times 1,500$	100	4	58.0	0.0121	1042.0	0.0847	0.1428	181.5	0.0428	304.0	0.0598	0.7157	1816.0	0.1143	0.3740
	200	6	124.0	0.0407	2144.5	0.3176	0.1282	792.0	0.2075	1116.0	0.2431	0.8539	6861.5	0.6585	0.3152
$6,000 \times 1,500$	100	4	60.0	0.0245	1004.0	0.1126	0.2177	202.5	0.1029	326.0	0.1530	0.6726	1729.5	0.1459	0.7055
	200	6	110.0	0.0768	2088.5	0.4732	0.1624	845.0	0.4582	1169.0	0.5528	0.8289	7039.0	0.9794	0.4678
$8,000 \times 1,500$	100	4	55.0	0.0254	1011.0	0.1367	0.1861	166.0	0.0937	296.0	0.1598	0.5859	1381.0	0.1420	0.6593
	200	6	117.0	0.1154	2103.0	0.7317	0.1577	849.5	0.5693	1212.0	0.6993	0.8142	6652.0	1.0774	0.5284
$10,000 \times 2,500$	100	4	54.0	0.0358	999.0	0.1861	0.1924	145.0	0.1119	283.0	0.1871	0.5982	1317.5	0.1740	0.6432
	200	6	122.0	0.1635	2099.0	1.1800	0.1386	941.5	0.8957	1309.0	0.9133	0.9807	6273.0	1.3009	0.6885
$12,000 \times 3,000$	100	4	59.0	0.0514	1001.0	0.2403	0.2138	211.5	0.1968	345.0	0.2679	0.7345	1639.5	0.2432	0.8090
	200	6	108.0	0.1879	2136.0	1.5653	0.1200	799.5	0.9728	1124.0	0.9290	1.0472	6584.5	1.7415	0.5586

Table 3. IT and CPU of the GBRGS-RK, GRGS-GRK, BRGS-RK, BREK-RK and RGS-RK methods for Type II with different k

$m \times n$	method	k	150	200	250	300
$2,000 \times 500$	GBGRS-RK	IT	88.0	163.0	289.0	543.0
		CPU	0.0088	0.0178	0.0402	0.0791
	BRGS-RK	IT	129.5	237.0	446.5	862.0
		CPU	0.0148	0.0286	0.0598	0.1174
	BREK-RK	IT	174.0	331.0	525.0	1037.5
		CPU	0.0202	0.0391	0.0674	0.1366
	GRGS-GRK	IT	1137.5	1438.0	1799.5	3047.5
		CPU	0.0713	0.1025	0.1725	0.3305
	RGS-RK	IT	3742.5	7167.5	12753.0	2577.4
		CPU	0.1439	0.3150	0.6838	1.5230
		speed-up1	0.1026	0.0908	0.0875	0.0771
		speed-up2	0.7304	0.7312	0.8876	0.8595
		speed-up3	0.1241	0.1736	0.2328	0.2395
$3,000 \times 500$	GBRGS-RK	IT	94.0	158.0	242.0	537.0
		CPU	0.0179	0.0340	0.0620	0.1342
	BRGS-RK	IT	134.0	225.0	418.0	823.5
		CPU	0.0284	0.0508	0.0952	0.1979
	BREK-RK	IT	183.0	316.0	500.5	1053.0
		CPU	0.0375	0.0641	0.1087	0.2226
	GRGS-GRK	IT	1226.0	1566.0	1883.0	2850.5
		CPU	0.1522	0.2287	0.3349	0.5683
	RGS-RK	IT	3815.5	6671.5	11990.0	25990.0
		CPU	0.3152	0.6729	1.3666	3.2897
		speed-up1	0.0900	0.0755	0.0697	0.0602
		speed-up2	0.7570	0.7926	0.8760	0.8891
		speed-up3	0.1179	0.1487	0.1852	0.2362
$4,000 \times 500$	GBRGS-RK	IT	95.0	173.0	268.0	536.0
		CPU	0.0211	0.0451	0.0833	0.1958
	BRGS-RK	IT	129.0	232.0	413.5	849.0
		CPU	0.0327	0.0613	0.1160	0.2494
	BREK-RK	IT	182.5	319.5	531.5	1101.5
		CPU	0.0547	0.0953	0.1641	0.3415
	GRGS-GRK	IT	1246.5	1620.0	1960.0	2942.0
		CPU	0.1827	0.2906	0.4392	0.7994
	RGS-RK	IT	3819.0	6952.5	12338.0	26089.0
		CPU	0.4860	0.9822	1.8093	4.1411
		speed-up1	0.0674	0.0624	0.0641	0.0602
		speed-up2	0.5981	0.6431	0.7072	0.7304
		speed-up3	0.1156	0.1554	0.1896	0.2449

Table 4. IT and CPU of the GBRGS-RK, GRGS-GRK, BRGS-RK, BREK-RK and RGS-RK methods with wine quality and bike sharing datasets

method	name	bike	red wine	white wine
GBRGS-RK	IT	203.0	192.0	119.0
	CPU	0.0210	0.0048	0.0031
GRGS-GRK (0.5, 0.5)	IT	582.0	499.5	428.5
	CPU	0.0446	0.0122	0.0104
	speed-up3	0.4711	0.3973	0.2999
GRGS-GRK (1.4, 1.4)	IT	56.0	114.0	77.0
	CPU	0.0049	0.0032	0.0024
	speed-up3	4.2600	1.5092	1.2793
BRGS-RK	IT	440.0	4134.5	3013.0
	CPU	0.0313	0.0986	0.0725
BREK-RK	IT	534.0	6083.5	4568.5
	CPU	0.0950	0.2121	0.1561
	speed-up2	0.3297	0.4648	0.4646
RGS-RK	IT	651.5	8486.0	6666.5
	CPU	0.0425	0.1636	0.1289
	speed-up1	0.7376	0.6027	0.5627

Table 5. IT and CPU of the GBRGS-RK, GRGS-GRK, BRGS-RK, BREK-RK and RGS-RK methods for the Phillips's problem

method	IT	CPU
GBRGS-RK	45.0	0.0014
GRGS-GRK (0.5, 0.5)	258.5	0.0061
GRGS-GRK (1.4, 1.4)	57.0	0.0015
BRGS-RK	155.5	0.0105
BREK-RK	375.0	0.0229
RGS-RK	508.0	0.0116

Table 6. CPU of the GBRGS-RK, BRGS-RK, RSVD, RQR methods for A with $n = 1,000$ and different m

$m \times n$	$5,000 \times 1,000$	$6,000 \times 1,000$	$7,000 \times 1,000$	$8,000 \times 1,000$	$9,000 \times 1,000$
GBGRS-RK	0.0759	0.1031	0.1341	0.1651	0.1839
BGRS-RK	0.1013	0.1665	0.1940	0.2343	0.2580
RSVD	0.3947	0.4343	0.4998	0.5473	0.6125
RQR	8.6060	8.6237	9.1383	9.3263	10.0631
speedup-4	0.1924	0.2373	0.2682	0.3016	0.3002
speedup-5	0.0088	0.0120	0.0147	0.0177	0.0183

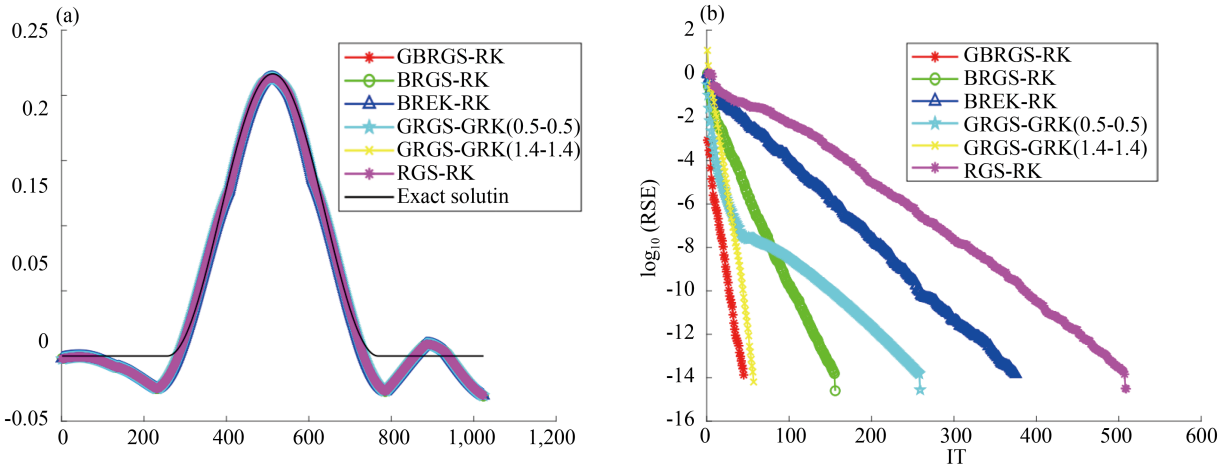

Figure 4. The solutions generated by the GBRGS-RK, GRGS-GRK, BRGS-RK, BREK-RK and RGS-RK methods (a) and RES versus IT for the GBRGS-RK, GRGS-GRK, BRGS-RK, BREK-RK and RGS-RK methods for the Phillips's problem (b)

5. Conclusion

The BRGS-RK method is developed in this work to solve factorized linear systems without assessing the products of U and V. The approach uses the BRAGS and BRAK methods to solve subsystems. In addition, a greedy version of the BRGS-RK method is also considered based on matrix sketch tech. The convergence rates of the proposed algorithms are taken into account in this study. Both the random matrix examples and the UCI dataset examples demonstrate that the GBRGS-RK and BRGS-RK methods have a significantly lower number of iterations than the BREK-RK, GRGS-GRK and RGS-RK methods, while the Phillips's problem example further proves the stability of the GBRGS-RK and BRGS-RK methods in ill-posed systems with a large condition number. Recently, the Kaczmarz-type algorithms [21, 22] and its block variants [23, 24] have been applied to solving nonlinear systems of equations. It's notably worthwhile to apply the methods introduced in this paper to nonlinear models, with the Variable-Coefficient Caudrey-Dodd-Gibbon-Kotera-Sawada Equation [25, 26] serving as a case in point.

Acknowledgement

This work was supported by the National Key Research and Development Program of China under Grant No. 2024YFC3308500, the National Natural Science Foundation of China under Grant No. 62362019, and the Hainan Provincial Natural Science Foundation of China under No. 725QN279.

Conflict of interest

The authors declare no competing financial interest.

References

[1] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. *Computer*. 2009; 42(8): 30-37. Available from: https://doi.org/10.1109/MC.2009.263.

[2] Dhanaraj M, Markopoulos P. Robust stochastic principal component analysis via Barron loss. In: *2022 56th Asilomar Conference on Signals, Systems, and Computers*. CA, USA: Pacific Grove; 2022. p.1286-1290.

[3] Ma A, Flenner A, Needell D, Percus AG. Improving image clustering using sparse text and the wisdom of the crowds. In: *2014 48th Asilomar Conference on Signals, Systems and Computers*. CA, USA: Pacific Grove; 2014. p.1555-1557.
[4] Ma A, Needell D, Ramdas A. Iterative methods for solving factorized linear systems. *SIAM Journal on Matrix Analysis and Applications*. 2018; 39(1): 104-122. Available from: https://doi.org/10.1137/17M1115678.
[5] Strohmer T, Vershynin R. A randomized Kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*. 2009; 15: 262-278. Available from: https://doi.org/10.1007/s00041-008-9030-4.
[6] Zouzias A, Freris N. Randomized extended Kaczmarz for solving least squares. *SIAM Journal on Matrix Analysis and Applications*. 2013; 34(2): 773-793. Available from: https://doi.org/10.1137/120889897.
[7] Dumitrescu B. On the relation between the randomized extended Kaczmarz algorithm and coordinate descent. *BIT Numerical Mathematics*. 2015; 55: 1005-1015. Available from: https://doi.org/10.1007/s10543-014-0526-9.
[8] Zhao J, Wang X, Zhang J. A randomised iterative methods for solving factorised linear systems. *Linear and Multilinear Algebra*. 2023; 71(2): 242-255. Available from: https://doi.org/10.1080/03081087.2021.2025198.
[9] Bai ZZ, Wu WT. On greedy randomized Kaczmarz method for solving large sparse linear systems. *SIAM Journal on Scientific Computing*. 2018; 40(1): A592-A606. Available from: https://doi.org/10.1137/17M1137747.
[10] Bai ZZ, Wu WT. On greedy randomized coordinate descent methods for solving large linear least-squares problems. *Numerical Linear Algebra with Applications*. 2019; 26: e2273. Available from: https://doi.org/10.1002/nla.2237.
[11] Liu SM, Liu Y. On relaxed greedy randomized iterative methods for the solution of factorized linear systems. *Taiwanese Journal of Mathematics*. 2022; 26(5): 953-979. Available from: https://doi.org/10.11650/tjm/220305.
[12] Du K. Regularized randomized iterative algorithms for factorized linear systems. *Applied Mathematics and Computation*. 2024; 466: 128468. Available from: https://doi.org/10.1016/j.amc.2023.128468.
[13] Zhao J, Wang X, Zhang JH. Randomized average block iterative methods for solving factorised linear systems. *Filomat*. 2023; 37(14): 4603-4620. Available from: https://doi.org/10.2298/FIL2314603Z.
[14] Wu NC, Cui LX, Zuo Q. On the relaxed greedy deterministic row and column iterative methods. *Applied Mathematics and Computation*. 2022; 432: 127339. Available from: https://doi.org/10.1016/j.amc.2022.127339.
[15] Wu W. *Paving the Randomized Gauss-Seidel Method*. Available from: https://scholarship.claremont.edu/cgi/viewcontent.cgi?article=2025&context=scripps_theses [Accessed 10 July 2025].
[16] Du K, Si WT, Sun XH. Randomized extended average block Kaczmarz for solving least squares. *SIAM Journal on Scientific Computing*. 2020; 42(2): A3541-A3559. Available from: https://doi.org/10.1137/20M1312629.
[17] UCI Machine Learning Repository. Available from: https://archive.ics.uci.edu. [Accessed 10 July 2025].
[18] Kaveev A, Kropotov G, Tsygankova E, Tzibizov I, Ganichev S, Danilov S, et al. Terahertz polarization conversion with quartz wave plate set. *Applied Optics*. 2013; 52: B60-B69. Available from: https://doi.org/10.1364/AO.52.000B60.
[19] Halko N, Martinsson PG, Tropp JA. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*. 2011; 53(2): 217-288. Available from: https://doi.org/10.1137/090771806.
[20] Balabanov O, Grigori L. Randomized Gram-Schmidt process with application to GMRES. *SIAM Journal on Scientific Computing*. 2022; 44(3): A1450-A1474. Available from: https://doi.org/10.1137/20M138870X.
[21] Wang QF, Li WG, Bao WD, Gao XQ. Nonlinear Kaczmarz algorithms and their convergence. *Journal of Computational and Applied Mathematics*. 2022; 399: 113720. Available from: https://doi.org/10.1016/j.cam.2021.113720.
[22] Liu L, Li WG, Bao WD, Xing LL. Greedy Kaczmarz methods for nonlinear equation. *Journal of Computational and Applied Mathematics*. 2025; 467: 116630. Available from: https://doi.org/10.1016/j.cam.2025.116630.
[23] Ye YX, Yin JF. A residual-based weighted nonlinear Kaczmarz method for solving nonlinear systems of equations. *Computational and Applied Mathematics*. 2024; 43: 276. Available from: https://doi.org/10.1007/s40314-024-02797-1.
[24] Zhang JH, Wang YQ, Zhao J. On pseudoinverse-free block maximum residual nonlinear Kaczmarz method for solving large-scale nonlinear system of equations. *Japan Journal of Industrial and Applied Mathematics*. 2024; 41: 637-657. Available from: https://doi.org/10.1007/s13160-023-00620-8.
[25] Li J, Manafian J, Wardhana A, Othman A, Husein I, Al-Thamir M, et al. N-Lump to the (2 + 1)-dimensional variable-coefficient Caudrey-Dodd-Gibbon-Kotera-Sawada equation. *Complexity*. 2022; 2022: 4383100. Available from: https://doi.org/10.1155/2022/4383100.

[26] Pan YF, Manafian J, Zeynalli SM, Al-Obaidi R, Sivaraman R, Kadi A. N-Lump solutions to a (3 + 1)-dimensional variable-coefficient generalized nonlinear wave equation in a liquid with gas bubbles. *Qualitative Theory of Dynamical Systems*. 2022; 21: 127. Available from: https://doi.org/10.1007/s12346-022-00658-y.