Filomat 37:14 (2023), 4603–4620
https://doi.org/10.2298/FIL2314603Z

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

Randomized average block iterative methods for solving factorised linear systems

Jing Zhao[a], Xiang Wang[a], Jianhua Zhang[b,*]

[a] *Department of Mathematics, Nanchang University, Nanchang, Jiangxi, 330031, China*
[b] *School of Science, East China University of Technology, Nanchang, Jiangxi, 330013, China*

Abstract. Recently, some randomized iterative methods are proposed to solve large-scale factorised linear systems. In this paper, we present two randomized average block iterative methods which still take advantage of the factored form and need not perform the entire matrix. The new methods are pseudoinverse-free and can be implemented for parallel computation. Furthermore, we analyze their convergence behaviors and obtain the exponential convergence rate. Finally, some numerical examples are carried out to show the effectiveness of our new methods.

1. Introduction

Some science and engineering applications, such as recommender systems in machine learning [1–5], topic modeling of text data and linear regression from statistics [6], require the solution of the large-scale factorised linear system

$$UV\beta = y, \tag{1}$$

where $U \in \mathbb{C}^{m\times k}$, $V \in \mathbb{C}^{k\times n}$, $\beta \in \mathbb{C}^n$ and $y \in \mathbb{C}^m$. If we set $X = UV \in \mathbb{C}^{m\times n}$, then we obtain the full linear system $X\beta = y$. We know that a system is consistent if it has at least one solution (and inconsistent otherwise), and for more theoretical analysis and practical applications, we refer to [7] and the references therein. Recently, instead of solving the full linear system $X\beta = y$, randomized iterative methods for solving the individual subsystems

$$Ux = y \tag{2}$$

and

$$V\beta = x \tag{3}$$

2020 *Mathematics Subject Classification*. 65F10; 65H10
Keywords. Factorised linear systems, Randomized average block Kaczmarz method, Randomized extended average block Kaczmarz method
Received: 24 July 2022; Revised: 12 October 2022; Accepted: 24 January 2023
Communicated by Yimin Wei
Research supported by the National Natural Science Foundation of China under grant No.12061009 and 11961048. Research supported by Jiangxi Provincial Natural Science Foundation under grant No.20202BAB201002.
* Corresponding author: Jianhua Zhang
Email addresses: zhaojingwust@ecut.edu.cn (Jing Zhao), wangxiang49@ncu.edu.cn (Xiang Wang), zhangjh_496@163.com (Jianhua Zhang)

in an alternating fashion have been established.

Recently, the randomized Kaczmarz (RK) method [8] which converges in expectation to the solution of the consistent linear systems has been successfully applied in many practical applications and reignited many researches. The RK method was generalized to solve inconsistent, underdetermined or rank-deficient linear systems [9–11], and some acceleration strategies, such as greedy techniques and matrix sketching [12–16], were studied. The randomized coordinate descent (RCD) method presented by Leventhal and Lewis [17] is another basic randomized iterative method for solving overdetermined linear systems. However, RCD does not converge to the least norm solution of the underdetermined linear systems. To overcome these drawbacks, Zouzias, Freris and Ma et al. proposed the randomized extended Kaczmarz (REK) method [18] and the randomized extended Gauss-Seidel (REGS) method [10], respectively. The convergence properties of RK, RGS, REK and REGS for linear system $X\beta = y$ with full rank is summarized in the following Table 1 [7, 10]. In Table 1, we let β_{uniq}, β_{LN} and β_{LS} denote the optimal unique solution, the least Euclidean norm solution and the ordinary least squares solution, respectively.

Table 1: (Table 1 in [7, 10])Summary of convergence properties of RK, RGS, REK and REGS for linear system $X\beta = y$ with full rank

Method	Overdetermined, consistent: convergence to β_{uniq}?	Overdetermined, inconsistent: convergence to β_{LS}?	Underdetermined: convergence to β_{LN}?
RK	Yes[8]	No[9]	Yes[10]
REK	Yes[18]	Yes[18]	Yes[10]
RGS	Yes[17]	Yes[17]	No[10]
REGS	Yes[10]	Yes[10]	Yes[10]

The projection-based block variants of basic Kaczmarz method, such as the block Kaczmarz (BK) [19] method, the randomized block Kaczmarz (RBK) method [20] and the randomized double BK (RDBK) method [21], have been developed to solve consistent or inconsistent linear systems and numerical results demonstrate that the convergence rate can be significantly accelerated if appropriate blocks of the coefficient matrix are provided. However, these projection-based block methods are difficult to parallelize and are required to compute the Pseudoinverse or solve the least-square problems. In [22], Necoara developed a randomized average block Kaczmarz (RaBK) method and the kth iteration x_k is computed by

$$x_k = x_{k-1} + \alpha_k \left(\sum_{l \in \mathcal{I}_i} \omega_l \frac{b_i - A_{l,:}x_{k-1}}{\|A_{l,:}\|_2^2} (A_{l,:})^T \right),\ k \geq 0,$$

where the weights $\omega_l \in (0, 1]$ such that $\sum_{l \in \mathcal{I}_i} \omega_l = 1$ and the stepsize $\alpha_k > 0$. RaBK is a pseudoinverse-free method and very effective if a good sampling of the rows introduced into well-conditioned blocks. Motivated by RaBK, Du et al. [23] presented a simple randomized extended average block Kaczmarz (REABK) method which works for all types of linear systems and demonstrates remarkable convergence properties in terms of computing time. Other pseudoinverse-free block methods, we refer to [24–26] and the references therein.

In [7, 27], Ma et al. proposed RK-RK and REK-RK to solve the factorised systems with consistent or inconsistent full linear system. Recently, inspired by the effectiveness of RGS, Zhao, Wang and Zhang [28] established RGS-RK which interlaces the RGS iterates to solve subsystem (2) and the RK iterates to solve subsystem (3). Recently, the relaxed GRK-GRK and GRGS-GRK methods [29] based on the greedy randomized Kaczmarz (GRK) method and the greedy randomized Gauss-Seidel (GRGS) method are developed. Du introduce regularized randomized iterative algorithms [30] for factorised linear systems. In this paper, inspired by the works in [23], we also propose pseudoinverse-free block extension of RK-RK (BRK-RK) and REK-RK (BREK-RK), respectively. In addition, we establish their convergence theories and provide some numerical examples to illustrate that BRK-RK and BREK-RK outperform the corresponding RK-RK and REK-RK in terms of computing time, respectively.

Here and throughout the paper, we adopt the same notations introduced in [7]. For example for a matrix $X = (x_{ij}) \in \mathbb{C}^{m\times n}$, $X_{(j)}$, $X^{(i)}$ and $\|X\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|x_{ij}|^2}$ denote its *jth* column, *ith* row and Frobenius norm, respectively. We use $\sigma_{\max}(\cdot)$ and $\sigma_{\min}(\cdot)$ to denote the largest and the smallest nonzero singular value of matrix and $\lambda_{\min}(\cdot)$ to denote the smallest nonzero eigenvalue of matrix. For index set $\mathcal{I} \subseteq [m]$ and $\mathcal{J} \subseteq [k]$, we use $A_{\mathcal{I},:}$ and $A_{:,\mathcal{J}}$ to denote the row submatrix indexed by $\mathcal{I}$ and the column submatrix indexed by $\mathcal{J}$, respectively. In addition, we use $(x_t)_{\mathcal{I}}$ to denote the subvector indexed by $\mathcal{I}$ at tth iteration. Similar to [7], for simplicity, we also refer to the matrix X of a linear system as consistent or inconsistent when the system itself is consistent or inconsistent.

We denote by $\mathbb{E}_{t-1}$ the expected value conditional on the first $t-1$ iterations, that is,

$$\mathbb{E}_{t-1}[\cdot] = \mathbb{E}[\cdot \mid j_1, i_1, \cdots, j_{t-1}, i_{t-1}],$$

where j_l is the *lth* column chosen and i_l is the *lth* row chosen. Then, based on the law of iterated expectations, we obtain

$$\mathbb{E}[\mathbb{E}_{t-1}[\cdot]] = \mathbb{E}[\cdot].$$

For more details, see [23, 28].

The paper is organized as follows. In Section 2, we introduce some necessary preliminaries and review RK-RK and REK-RK, respectively. In Section 3, we propose the randomized average block Kaczmarz (REBK) method and give its convergence theory. In section 4, we present BRK-RK and BREK-RK and study their convergence property. In Section 5, we test some numerical examples. Finally, we give some brief concluding remarks in Section 6.

2. Preliminaries and the REK-RK method

In this paper, we set X be rank deficient and assume that U and V are of full rank. For simplicity in notation, Table 2 in [7] summarizes the optimal solution of (1), (2) and (3). For overdetermined consistent, underdetermined and overdetermined inconsistent linear systems, the optimal solution for (2) and (3) denotes the unique, least norm, or the least squares solution, respectively.

Table 2: (Table 2 in [7])Summary of notation for linear systems discussed and their solutions

Linear system	Optimal solution
$X\beta = y$ (1)	β_*
$Ux = y$ (2)	x_*
$Vb = x$ (3)	b_*

In [7], Ma et al. showed that when U is overdetermined and consistent or X is inconsistent and V is underdetermined, solving subsystems (2) and (3) will converge to the optimal solution of the full system. Other scenarios are fully explained in Table 3 of [7]. In this paper, we only focus on the case in which $k < m, n$.

For the consistent or inconsistent setting, Ma et al. proposed RK-RK and REK-RK [7] for solving the factorised linear systems, respectively. RK-RK and REK-RK are outlined in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 RK-RK

Input: U, V, y, x_0, b_0;
Output: the last b_t.
Set $t = 1$;
While stopping criteria not reached **do**
Choose row $U^{(i)}$ with probability $\frac{\|U^{(i)}\|_2^2}{\|U\|_F^2}$;
Update $x_t = x_{t-1} + \frac{(y^{(i)} - U^{(i)} x_{t-1})}{\|U^{(i)}\|_2^2}(U^{(i)})^*$;
Choose row $V^{(p)}$ with probability $\frac{\|V^{(p)}\|_2^2}{\|V\|_F^2}$;
Update $b_t = b_{t-1} + \frac{(x_t^{(p)} - V^{(p)} b_{t-1})}{\|V^{(p)}\|_2^2}(V^{(p)})^*$;
Update $t = t + 1$;
End

Algorithm 2 REK-RK

Input: U, V, y, z_0, x_0, b_0;
Output: the last b_t.
Set $t = 1$;
While stopping criteria not reached **do**
Choose row $U^{(i)}$ with probability $\frac{\|U^{(i)}\|_2^2}{\|U\|_F^2}$;
Choose column $U_{(j)}$ with probability $\frac{\|U_{(j)}\|_2^2}{\|U\|_F^2}$;
Update $z_t = z_{t-1} - \frac{U_{(j)}^* z_{t-1}}{\|U_{(j)}\|_2^2} U_{(j)}$;
Update $x_t = x_{t-1} + \frac{(y^{(i)} - z_t^{(i)} - U^{(i)} x_{t-1})}{\|U^{(i)}\|_2^2}(U^{(i)})^*$;
Choose row $V^{(p)}$ with probability $\frac{\|V^{(p)}\|_2^2}{\|V\|_F^2}$;
Update $b_t = b_{t-1} + \frac{(x_t^{(p)} - V^{(p)} b_{t-1})}{\|V^{(p)}\|_2^2}(V^{(p)})^*$;
Update $t = t + 1$;
End

For the convergence property of RK-RK and REK-RK, Ma et al. gave the following theorem.

Theorem 2.1. *(Theorem 1 in [27]) Let $X = UV$, where $U \in \mathbb{C}^{m\times k}$ and $V \in \mathbb{C}^{k\times n}$ are of full rank, and the systems $X\beta = y$, $Ux = y$, and $Vb = x$ have the optimal solutions β_*, x_* and b_*, respectively. Set $b_0 = \mathbf{0}$ and assume that $k < m, n$.*
(a) If $X\beta = y$ is consistent, then $b_* = \beta_*$ and RK-RK converges with expected error

$$\mathbb{E}\|b_t - \beta_*\|^2 \le \begin{cases} \alpha_V^t\|b_*\|^2 + (1-\gamma_1)^{-1}\alpha_{\max}^t\dfrac{\|x_*\|^2}{\|V\|_F^2}, & if \quad \alpha_U \neq \alpha_V \\ \alpha_V^t\|b_*\|^2 + t\alpha_{\max}^t\dfrac{\|x_*\|^2}{\|V\|_F^2}, & else \end{cases} \tag{4}$$

where $\alpha_U = 1 - \frac{\sigma^2_{\min}(U)}{\|U\|_F^2}$, $\alpha_V = 1 - \frac{\sigma^2_{\min}(V)}{\|V\|_F^2}$, $\alpha_{\max} = \max\{\alpha_U, \alpha_V\}$ and $\gamma_1 = \min\{\frac{\alpha_U}{\alpha_V}, \frac{\alpha_V}{\alpha_U}\}$.
(b) If $X\beta = y$ is inconsistent, then $b_* = \beta_*$ and REK-RK converges with expected error

$$\mathbb{E}\|b_t - \beta_*\|^2 \le \begin{cases} \alpha_V^t\|b_*\|^2 + (1-\gamma_2)^{-1}\widetilde{\alpha}_{\max}^{t-1}\dfrac{(1+2\kappa_U^2)\|x_*\|^2}{\|V\|_F^2}, & if \quad \sqrt{\alpha_U} \neq \alpha_V \\ \alpha_V^t\|b_*\|^2 + t\widetilde{\alpha}_{\max}^{t-1}\dfrac{(1+2\kappa_U^2)\|x_*\|^2}{\|V\|_F^2}, & else \end{cases} \tag{5}$$

where $\alpha_U = 1 - \frac{\sigma^2_{\min}(U)}{\|U\|_F^2}$, $\alpha_V = 1 - \frac{\sigma^2_{\min}(V)}{\|V\|_F^2}$, $\widetilde{\alpha}_{\max} = \max\{\sqrt{\alpha_U}, \alpha_V\}$, $\gamma_2 = \min\{\frac{\sqrt{\alpha_U}}{\alpha_V}, \frac{\alpha_V}{\sqrt{\alpha_U}}\}$ and $\kappa_U^2 = \frac{\sigma^2_{\max}(U)}{\sigma^2_{\min}(U)}$.

3. The RABK method

In this section, adopting the same techniques introduced in [23], we propose the randomized average block Kaczmarz (RABK) method for solving consistent linear system $Ax = b$, where $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^m$. RABK is outlined in Algorithm 3.

Algorithm 3 The randomized average block Kaczmarz (RABK) method

1: Let $\{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_s\}$ be partitions of $[m]$;
2: Let $\widetilde{\alpha} > 0$ and initialize $x_0 \in \mathbb{R}^n$
3: **For** $k = 1, 2, \ldots,$ **do**
4: Choose index $i \in [s]$ with probability $\frac{\|A_{\mathcal{I}_i,:}\|_F^2}{\|A\|_F^2}$;
5: Update $x_k = x_{k-1} - \frac{\widetilde{\alpha}}{\|A_{\mathcal{I}_i,:}\|_F^2}(A_{\mathcal{I}_i,:})^*(A_{\mathcal{I}_i,:}x_{k-1} - b_{\mathcal{I}_i})$;
6: **Endfor**

We note that RABK can be obtained from the DSBGS method proposed in [26] by setting $\mathcal{J} = 1, 2, \cdots, n$. In addition, if we set $\omega_l = \frac{\|A_{l,:}\|_2^2}{\|A_{\mathcal{I}_i,:}\|_F^2}$, we can also obtain RABK from RaBK presented by Necoara [22]. Similar to Theorem 2.7 in [23], for the convergence property of RABK, we can establish the following theorem.

Theorem 3.1. *Assume that the matrix $A \in \mathbb{C}^{m\times n}$, $b \in \mathbb{C}^m$ and $0 < \widetilde{\alpha} < 2/\widetilde{\beta}^{\mathcal{I}}_{\max}$. Then, the iteration sequence $\{x_k\}_{k=0}^\infty$ generated by RABK starting from any initial guess $x_0 \in$ range(A^*) exists and converges to the unique least-norm solution $x_* = A^\dagger b$ of the consistent linear system $Ax = b$. Moreover, it holds that*

$$\mathbb{E}[\|x_k - x_*\|_2^2] \le \widetilde{\rho}^k\|x_0 - x_*\|_2^2, \tag{6}$$

where $\widetilde{\rho} = 1 - \frac{(2\widetilde{\alpha} - \widetilde{\alpha}^2\widetilde{\beta}^{\mathcal{I}}_{\max})\sigma^2_{\min}(A)}{\|A\|_F^2}$ and $\widetilde{\beta}^{\mathcal{I}}_{\max} = \max\limits_{i\in[s]}\frac{\sigma^2_{\max}(A_{\mathcal{I}_i,:})}{\|A_{\mathcal{I}_i,:}\|_F^2}$

Proof. From Algorithm 3, we have

$$x_k - x_* = x_{k-1} - x_* - \frac{\widetilde{\alpha}}{\|A_{\mathcal{I}_i,:}\|_F^2}(A_{\mathcal{I}_i,:})^*(A_{\mathcal{I}_i,:}x_{k-1} - b_{\mathcal{I}_i}). \tag{7}$$

By direct computations, we obtain

$$\|x_k - x_*\|_2^2 = \|x_{k-1} - x_*\|_2^2 - \frac{2\widetilde{\alpha}}{\|A_{\mathcal{I}_i,:}\|_F^2}\|A_{\mathcal{I}_i,:}(x_{k-1} - x_*)\|_2^2 + \widetilde{\alpha}^2\|(\frac{A_{\mathcal{I}_i,:}}{\|A_{\mathcal{I}_i,:}\|_F^2})^*(\frac{A_{\mathcal{I}_i,:}}{\|A_{\mathcal{I}_i,:}\|_F^2})(x_{k-1} - x_*)\|_2^2. \tag{8}$$

Using Lemma 2.5 in [23], it holds that

$$\begin{aligned}\|x_k - x_*\|_2^2 &\le \|x_{k-1} - x_*\|_2^2 - (2\widetilde{\alpha} - \frac{\widetilde{\alpha}^2\sigma_{\max}^2(A_{\mathcal{I}_i,:})}{\|A_{\mathcal{I}_i,:}\|_F^2})\frac{\|A_{\mathcal{I}_i,:}(x_{k-1} - x_*)\|_2^2}{\|A_{\mathcal{I}_i,:}\|_F^2} \\ &\le \|x_{k-1} - x_*\|_2^2 - (2\widetilde{\alpha} - \widetilde{\alpha}^2\beta_{\max}^{\mathcal{I}})\frac{\|A_{\mathcal{I}_i,:}(x_{k-1} - x_*)\|_2^2}{\|A_{\mathcal{I}_i,:}\|_F^2}. \end{aligned} \tag{9}$$

From (9) and the definition of conditional expectation conditioned on the first $k-1$ iterations, we have

$$\begin{aligned}\mathbb{E}_{k-1}[\|x_k - x_*\|_2^2] &\le \|x_{k-1} - x_*\|_2^2 - (2\widetilde{\alpha} - \widetilde{\alpha}^2\widetilde{\beta}_{\max}^{\mathcal{I}})\frac{\|A(x_{k-1} - x_*)\|_2^2}{\|A\|_F^2} \\ &\le \|x_{k-1} - x_*\|_2^2 - (2\widetilde{\alpha} - \widetilde{\alpha}^2\widetilde{\beta}_{\max}^{\mathcal{I}})\frac{\sigma_{\min}^2(A)\|(x_{k-1} - x_*)\|_2^2}{\|A\|_F^2} \quad (0 < \widetilde{\alpha} < 2/\widetilde{\beta}_{\max}^{\mathcal{I}}) \\ &= \widetilde{\rho}\|x_{k-1} - x_*\|_2^2. \end{aligned} \tag{10}$$

Then, we obtain

$$\mathbb{E}[\|x_k - x_*\|_2^2] \le \widetilde{\rho}\mathbb{E}[\|x_{k-1} - x_*\|_2^2] \le \widetilde{\rho}^k\|x_0 - x_*\|_2^2. \square$$

□

4. The BRK-RK and BREK-RK method

Algorithm 4 The BRK-RK method

Let $\{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_s\}$ and $\{\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_v\}$ be partitions of $[m]$ and $[k]$, respectively;
Input: U, V, y, z_0, x_0, b_0;
Output: the last b_t.
Set $t = 1$ and $\alpha > 0$;
While stopping criteria not reached **do**
Choose index $i \in [s]$ with probability $\frac{\|U_{\mathcal{I}_i,:}\|_F^2}{\|U\|_F^2}$;
Update $x_t = x_{t-1} + \frac{\alpha}{\|U_{\mathcal{I}_i,:}\|_F^2}(U_{\mathcal{I}_i,:})^*(y_{\mathcal{I}_i} - U_{\mathcal{I}_i,:}x_{t-1})$;
Choose index $j \in [v]$ with probability $\frac{\|V_{\mathcal{J}_j,:}\|_F^2}{\|V\|_F^2}$;
Update $b_t = b_{t-1} + \frac{\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}(V_{\mathcal{J}_j,:})^*((x_t)_{\mathcal{J}_j} - V_{\mathcal{J}_j,:}b_{t-1})$;
Update $t = t + 1$;
End

Inspired by the works in [23], we propose simple average block RK-RK (BRK-RK) method and block REK-RK (BREK-RK) method which interlace the RABK or REABK iterates to solve subsystem (2) and the

Algorithm 5 The BREK-RK method

Let $\{\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_s\}$ and $\{\mathcal{J}_1, \mathcal{J}_2, \dots, \mathcal{J}_v\}$ be partitions of $[m]$ and $[k]$, respectively;
Input: U, V, y, z_0, x_0, b_0;
Output: the last b_t.
Set $t = 1$ and $\alpha > 0$;
While stopping criteria not reached **do**
Choose index $i \in [s]$ with probability $\frac{\|U_{\mathcal{I}_i,:}\|_F^2}{\|U\|_F^2}$;
Choose index $j \in [v]$ with probability $\frac{\|U_{:,\mathcal{J}_j}\|_F^2}{\|U\|_F^2}$;
Update $z_t = z_{t-1} - \frac{\alpha}{\|U_{:,\mathcal{J}_j}\|_F^2}(U_{:,\mathcal{J}_j})(U_{:,\mathcal{J}_j})^* z_{t-1}$;
Update $x_t = x_{t-1} + \frac{\alpha}{\|U_{\mathcal{I}_i,:}\|_F^2}(U_{\mathcal{I}_i,:})^*(y_{\mathcal{I}_i} - (z_t)_{\mathcal{I}_i} - U_{\mathcal{I}_i,:}x_{t-1})$;
Choose column $j \in [v]$ with probability $\frac{\|V_{\mathcal{J}_j,:}\|_F^2}{\|V\|_F^2}$;
Update $b_t = b_{t-1} + \frac{\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}(V_{\mathcal{J}_j,:})^*((x_t)_{\mathcal{J}_j} - V_{\mathcal{J}_j,:}b_{t-1})$;
Update $t = t + 1$;
End

RABK iterates to solve subsystem (3). We note that BRK-RK and BREK-RK are pseudoinverse-free block randomized iterative methods and can be implemented for parallel computation. The BRK-RK method and the BREK-RK method are outlined in Algorithm 4 and Algorithm 5, respectively.

Next, we give our convergence results of BRK-RK and BREK-RK.

Theorem 4.1. *Let $X = UV$, where $U \in \mathbb{C}^{m\times k}$ and $V \in \mathbb{C}^{k\times n}$ are of full rank, and the full system $X\beta = y$, the subsystem $Ux = y$ and the subsystem $Vb = x$ have optimal solutions β_*, x_* and b_*, respectively. Set x_0 and b_0 be two zero vectors and $k < m, n$.*
(a) *Assume that $0 < \alpha < 2/\max(\beta^{\mathcal{I}}_{\max}, \beta^{\mathcal{J}}_{\max})$. If $X\beta = y$ is consistent, then we have $b_* = \beta_*$ and BRK-RK converges with the average error*

$$\begin{aligned}\mathbb{E}\|b_t - \beta_*\|_2^2 &\le (1 + \frac{1}{\varepsilon})\frac{\alpha^2\beta^{\mathcal{J}}_{\max}}{\|V\|_F^2}\|x_*\|_2^2\sum_{l=0}^{t-1}\rho^{t-l}(1+\varepsilon)^l\eta^l + (1+\varepsilon)^l\eta^l\|b_*\|_2^2\\ &\le (1+\varepsilon)^t\widehat{\rho}^t\left(\|b_*\|_2^2 + \frac{(1+\varepsilon)\alpha^2\beta^{\mathcal{J}}_{\max}\|x_*\|_2^2}{\varepsilon^2\|V\|_F^2}\right),\end{aligned} \tag{11}$$

where $\rho = 1 - \frac{(2\alpha-\alpha^2\beta^{\mathcal{I}}_{\max})\sigma^2_{\min}(U)}{\|U\|_F^2}$, $\eta = 1 - \frac{(2\alpha-\alpha^2\beta^{\mathcal{J}}_{\max})\sigma^2_{\min}(V)}{\|V\|_F^2}$, $\beta^{\mathcal{I}}_{\max} = \max\limits_{i\in[s]}\frac{\sigma^2_{\max}(U_{\mathcal{I}_i,:})}{\|U_{\mathcal{I}_i,:}\|_F^2}$, $\beta^{\mathcal{J}}_{\max} = \max\limits_{j\in[v]}\frac{\sigma^2_{\max}(V_{\mathcal{J}_j,:})}{\|V_{\mathcal{J}_j,:}\|_F^2}$ and $\widehat{\rho} =$ $\max\{\rho, \eta\}$.
(b) *Assume that $0 < \alpha < 2/\max(\widetilde{\beta}^{\mathcal{I}}_{\max}, \widetilde{\beta}^{\mathcal{J}}_{\max}, \beta^{\mathcal{J}}_{\max})$. If $X\beta = y$ is inconsistent, then we have $b_* = \beta_*$ and BREK-RK converges with the average error*

$$\mathbb{E}\|b_t - \beta_*\|^2 \le (1+\varepsilon)^t\widehat{\rho}^t\left(\|b_*\|_2^2 + \frac{(1+\varepsilon)\alpha^2\beta^{\mathcal{J}}_{\max}\|x_*\|_2^2}{\varepsilon^2\|V\|_F^2}(1 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}^{\mathcal{I}}_{\max}\sigma^2_{\max}(U)}{\varepsilon^2\|U\|_F^2})\right) \tag{12}$$

where $\widetilde{\rho} = 1 - \frac{(2\alpha-\alpha^2\widetilde{\beta}^{\mathcal{J}}_{\max})\sigma^2_{\min}(U)}{\|U\|_F^2}$, $\widetilde{\eta} = 1 - \frac{(2\alpha-\alpha^2\widetilde{\beta}^{\mathcal{I}}_{\max})\sigma^2_{\min}(U)}{\|U\|_F^2}$, $\widetilde{\beta}^{\mathcal{I}}_{\max} = \max\limits_{i\in[s]}\frac{\sigma^2_{\max}(U_{\mathcal{I}_i,:})}{\|U_{\mathcal{I}_i,:}\|_F^2}$, $\widetilde{\beta}^{\mathcal{J}}_{\max} = \max\limits_{j\in[v]}\frac{\sigma^2_{\max}(U_{\mathcal{J}_j,:})}{\|U_{\mathcal{J}_j,:}\|_F^2}$, $\widetilde{\widehat{\rho}} = \max\{\widetilde{\rho}, \widetilde{\eta}\}$ and $\widehat{\rho} = \max\{(1+\varepsilon)\widetilde{\widehat{\rho}}, \eta\}$.

Proof. We define $\widetilde{b}_t$ by

$$\widetilde{b}_t = b_{t-1} + \frac{\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}(V_{\mathcal{J}_j,:})^*((x_*)_{\mathcal{J}_j} - V_{\mathcal{J}_j,:}b_{t-1}), \tag{13}$$

which is the one-step RABK update for the exact linear system $Vb = x_*$ from b_{t-1}. From (13), we have

$$\widetilde{b_t} - b_* = b_{t-1} - b_* - \frac{\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}(V_{\mathcal{J}_j,:})^*(V_{\mathcal{J}_j,:}b_{t-1} - (x_*)_{\mathcal{J}_j}). \tag{14}$$

From step 9) of BRK-RK or step 11) of BREK-RK, we obtain

$$b_t = b_{t-1} + \frac{\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}(V_{\mathcal{J}_j,:})^*((x_t)_{\mathcal{J}_j} - V_{\mathcal{J}_j,:}b_{t-1}). \tag{15}$$

From (13) and (15), it holds that

$$b_t - \widetilde{b_t} = \frac{\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}(V_{\mathcal{J}_j,:})^*((x_t)_{\mathcal{J}_j} - (x_*)_{\mathcal{J}_j}). \tag{16}$$

From (14) and by direct computations, we have

$$\begin{aligned}
\|\widetilde{b_t} - b_*\|_2^2 &= \|b_{t-1} - b_*\|_2^2 - \frac{2\alpha}{\|V_{\mathcal{J}_j,:}\|_F^2}\|V_{\mathcal{J}_j,:}(b_{t-1} - b_*)\|_2^2 + \alpha^2\|(\frac{V_{\mathcal{J}_j,:}}{\|V_{\mathcal{J}_j,:}\|_F})^*\frac{V_{\mathcal{J}_j,:}}{\|V_{\mathcal{J}_j,:}\|_F}(b_{t-1} - b_*)\|_2^2 \\
&\le \|b_{t-1} - b_*\|_2^2 - (2\alpha - \frac{\alpha^2\sigma_{\max}^2(V_{\mathcal{J}_j,:})}{\|V_{\mathcal{J}_j,:}\|_F^2})\frac{\|V_{\mathcal{J}_j,:}(b_{t-1} - b_*)\|_2^2}{\|V_{\mathcal{J}_j,:}\|_F^2} \\
&\le \|b_{t-1} - b_*\|_2^2 - \frac{(2\alpha - \alpha^2\beta_{\max}^{\mathcal{J}})\|V_{\mathcal{J}_j,:}(b_{t-1} - b_*)\|_2^2}{\|V_{\mathcal{J}_j,:}\|_F^2}, \quad (\beta_{\max}^{\mathcal{J}} = \max_{j\in[v]}\frac{\sigma_{\max}^2(V_{\mathcal{J}_j,:})}{\|V_{\mathcal{J}_j,:}\|_F^2}).
\end{aligned} \tag{17}$$

From (17) and the definition of conditional expectation conditioned on the first $t-1$ iterations, we have

$$\begin{aligned}
\mathbb{E}_{t-1}[\|\widetilde{b_t} - b_*\|_2^2] &\le \|b_{t-1} - b_*\|_2^2 - \frac{(2\alpha - \alpha^2\beta_{\max}^{\mathcal{J}})\|V(b_{t-1} - b_*)\|_2^2}{\|V\|_F^2} \\
&\le \|b_{t-1} - b_*\|_2^2 - \frac{(2\alpha - \alpha^2\beta_{\max}^{\mathcal{J}})\sigma_{\min}^2(V)\|b_{t-1} - b_*\|_2^2}{\|V\|_F^2} \\
&= \eta\|b_{t-1} - b_*\|_2^2. \quad (0 < \alpha < 2/\beta_{\max}^{\mathcal{J}}, \eta = 1 - \frac{(2\alpha - \alpha^2\beta_{\max}^{\mathcal{J}})\sigma_{\min}^2(V)}{\|V\|_F^2})
\end{aligned} \tag{18}$$

Therefore, from (18) we have

$$\mathbb{E}[\|\widetilde{b_t} - b_*\|_2^2] \le \eta\mathbb{E}[\|b_{t-1} - b_*\|_2^2]. \tag{19}$$

From (16), by direct computations we can get

$$\begin{aligned}
\|b_t - \widetilde{b_t}\|_2^2 &= \frac{\alpha^2}{\|V_{\mathcal{J}_j,:}\|_F^4}\|(V_{\mathcal{J}_j,:})^*((x_t)_{\mathcal{J}_j} - (x_*)_{\mathcal{J}_j})\|_2^2 \\
&\le \frac{\alpha^2}{\|V_{\mathcal{J}_j,:}\|_F^2}\frac{\sigma_{\max}^2(V_{\mathcal{J}_j,:})}{\|V_{\mathcal{J}_j,:}\|_F^2}\|(x_t)_{\mathcal{J}_j} - (x_*)_{\mathcal{J}_j}\|_2^2 \\
&\le \frac{\alpha^2\beta_{\max}^{\mathcal{J}}}{\|V_{\mathcal{J}_j,:}\|_F^2}\|(x_t)_{\mathcal{J}_j} - (x_*)_{\mathcal{J}_j}\|_2^2.
\end{aligned} \tag{20}$$

Similar to (18), we have

$$\begin{aligned}
\mathbb{E}_{t-1}[\|b_t - \widetilde{b_t}\|_2^2] &= \mathbb{E}_{t-1}[[\mathbb{E}_{t-1}^j[\|b_t - \widetilde{b_t}\|_2^2]] \\
&\le \mathbb{E}_{t-1}[[\mathbb{E}_{t-1}^j[\frac{\alpha^2\beta_{\max}^{\mathcal{J}}}{\|V_{\mathcal{J}_j,:}\|_F^2}\|(x_t)_{\mathcal{J}_j} - (x_*)_{\mathcal{J}_j}\|_2^2]] \\
&= \mathbb{E}_{t-1}[\frac{\alpha^2\beta_{\max}^{\mathcal{J}}}{\|V\|_F^2}\|x_t - x_*\|_2^2].
\end{aligned} \tag{21}$$

Then, we also have

$$\mathbb{E}[\|b_t - \widetilde{b_t}\|_2^2] = \mathbb{E}[[\mathbb{E}_{t-1}[\|b_t - \widetilde{b_t}\|_2^2]] \leq \frac{\alpha^2 \beta_{\max}^{\mathcal{J}}}{\|V\|_F^2} \mathbb{E}[\|x_t - x_*\|_2^2]. \tag{22}$$

For any $\varepsilon > 0$, we obtain the following mean inequality

$$\frac{1}{\varepsilon}\|b_t - \widetilde{b_t}\|_2^2 + \varepsilon\|\widetilde{b_t} - b_*\|_2^2 \geq 2\|b_t - \widetilde{b_t}\|_2 \|\widetilde{b_t} - b_*\|_2. \tag{23}$$

From (23), by direct computations we obtain

$$\begin{aligned}\|b_t - b_*\|_2^2 &= \|b_t - \widetilde{b_t} + \widetilde{b_t} - b_*\|_2^2 \\ &\leq (\|b_t - \widetilde{b_t}\|_2 + \|\widetilde{b_t} - b_*\|_2)^2 \\ &\leq \|b_t - \widetilde{b_t}\|_2^2 + \|\widetilde{b_t} - b_*\|_2^2 + 2\|b_t - \widetilde{b_t}\|_2 \|\widetilde{b_t} - b_*\|_2 \\ &\leq (1 + 1/\varepsilon)\|b_t - \widetilde{b_t}\|_2^2 + (1+\varepsilon)\|\widetilde{b_t} - b_*\|_2^2. \end{aligned} \tag{24}$$

Then, we can get

$$\mathbb{E}[\|b_t - b_*\|_2^2] \leq (1 + 1/\varepsilon)\mathbb{E}\|b_t - \widetilde{b_t}\|_2^2 + (1+\varepsilon)\mathbb{E}\|\widetilde{b_t} - b_*\|_2^2. \tag{25}$$

From (19) and (22), it holds that

$$\mathbb{E}[\|b_t - b_*\|_2^2] \leq (1 + 1/\varepsilon)\frac{\alpha^2 \beta_{\max}^{\mathcal{J}}}{\|V\|_F^2}\mathbb{E}\|x_t - x_*\|_2^2 + (1+\varepsilon)\eta\mathbb{E}\|b_{t-1} - b_*\|_2^2. \tag{26}$$

(a) For Algorithm 4, plugging Theorem 3.1 into (26), we immediately get

$$\begin{aligned}\mathbb{E}[\|b_t - b_*\|_2^2] &\leq (1 + 1/\varepsilon)\frac{\alpha^2 \beta_{\max}^{\mathcal{J}}}{\|V\|_F^2}\rho^t\|x_*\|_2^2 + (1+\varepsilon)\eta\mathbb{E}\|b_{t-1} - b_*\|_2^2 \\ &\leq (1 + 1/\varepsilon)\frac{\alpha^2 \beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\|V\|_F^2}[\rho^t + \rho^{t-1}(1+\varepsilon)\eta] + (1+\varepsilon)^2\eta^2\mathbb{E}\|b_{t-2} - b_*\|_2^2 \qquad (27) \\ &\leq \cdots \leq (1 + \frac{1}{\varepsilon})\frac{\alpha^2 \beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\|V\|_F^2}\sum_{l=0}^{t-1}\rho^{t-l}(1+\varepsilon)^l\eta^l + (1+\varepsilon)^t\eta^t\|b_*\|_2^2. \qquad (28)\end{aligned}$$

where $\rho = 1 - \frac{(2\alpha - \alpha^2\beta_{\max}^{\mathcal{I}})\sigma_{\min}^2(U)}{\|U\|_F^2}$. If we set $\widehat{\rho} = \max\{\rho, \eta\}$, then by direct computations we have

$$\begin{aligned}\mathbb{E}[\|b_t - b_*\|_2^2] &\leq (1 + \frac{1}{\varepsilon})\frac{\alpha^2 \beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\|V\|_F^2}\widehat{\rho}^t\sum_{l=0}^{t-1}(1+\varepsilon)^l + (1+\varepsilon)^t\widehat{\rho}^t\|b_*\|_2^2 \\ &\leq (1+\varepsilon)^t\widehat{\rho}^t\left(\|b_*\|_2^2 + \frac{(1+\varepsilon)\alpha^2\beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\varepsilon^2\|V\|_F^2}\right). \end{aligned} \tag{29}$$

(b) For Algorithm 5, plugging Theorem 2.7 in [23] into (26), we have

$$\mathbb{E}[\|b_t - b_*\|_2^2] \leq (1+\varepsilon)\eta\mathbb{E}\|b_{t-1} - b_*\|_2^2 + (1+1/\varepsilon)\frac{\alpha^2\beta_{\max}^{\mathcal{J}}}{\|V\|_F^2}\Big[(1+\varepsilon)^t\widetilde{\eta}^t\|x_0 - x_*\|_2^2$$

$$+ (1+1/\varepsilon)\frac{\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\|z_0 - y_\perp\|_2^2}{\|U\|_F^2}\sum_{l=0}^{t-1}\widetilde{\rho}^{t-l}(1+\varepsilon)^l\widetilde{\eta}^l\Big](z_0 = y)$$

$$\leq (1+\varepsilon)\eta\mathbb{E}\|b_{t-1} - b_*\|_2^2 + (1+1/\varepsilon)\frac{\alpha^2\beta_{\max}^{\mathcal{J}}}{\|V\|_F^2}(1+\varepsilon)^t\widetilde{\widetilde{\rho}}^t\Big[\|x_*\|_2^2 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\|z_0 - y_\perp\|}{\varepsilon^2\|U\|_F^2}\Big]$$

$$\leq (1+\varepsilon)\eta\mathbb{E}\|b_{t-1} - b_*\|_2^2 + (1+1/\varepsilon)\frac{\alpha^2\beta_{\max}^{\mathcal{J}}}{\|V\|_F^2}(1+\varepsilon)^t\widetilde{\widetilde{\rho}}^t\Big[\|x_*\|_2^2 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\sigma_{\max}^2(U)\|x_*\|_2^2}{\varepsilon^2\|U\|_F^2}\Big]$$

$$\leq (1+\varepsilon)\eta\mathbb{E}\|b_{t-1} - b_*\|_2^2 + (1+1/\varepsilon)\frac{\alpha^2\beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\|V\|_F^2}(1+\varepsilon)^t\widetilde{\widetilde{\rho}}^t\Big[1 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\sigma_{\max}^2(U)}{\varepsilon^2\|U\|_F^2}\Big]$$

$$\leq (1+\varepsilon)^2\eta^2\mathbb{E}\|b_{t-2} - b_*\|_2^2 + (1+1/\varepsilon)\frac{\alpha^2\beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\|V\|_F^2}\Big[((1+\varepsilon)\widetilde{\widetilde{\rho}})^t + ((1+\varepsilon)\widetilde{\widetilde{\rho}})^{t-1}(1+\varepsilon)\eta\Big]$$

$$\Big[1 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\sigma_{\max}^2(U)}{\varepsilon^2\|U\|_F^2}\Big]$$

$$\leq \cdots \leq (1+\frac{1}{\varepsilon})\frac{\alpha^2\beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\|V\|_F^2}\Big[1 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\sigma_{\max}^2(U)}{\varepsilon^2\|U\|_F^2}\Big] \tag{30}$$

$$\sum_{l=0}^{t-1}((1+\varepsilon)\widetilde{\widetilde{\rho}})^{t-l}(1+\varepsilon)^l\eta^l + (1+\varepsilon)^t\eta^t\|b_*\|_2^2.$$

Let $\widehat{\rho} = \max\{\eta, (1+\varepsilon)\widetilde{\widetilde{\rho}}\}$, and then we have

$$\mathbb{E}\|b_t - \beta_*\|^2 \leq (1+\varepsilon)^t\widehat{\rho}^t\left(\|b_*\|_2^2 + \frac{(1+\varepsilon)\alpha^2\beta_{\max}^{\mathcal{J}}\|x_*\|_2^2}{\varepsilon^2\|V\|_F^2}(1 + \frac{(1+\varepsilon)\alpha^2\widetilde{\beta}_{\max}^{\mathcal{I}}\sigma_{\max}^2(U)}{\varepsilon^2\|U\|_F^2})\right). \tag{31}$$

Then, we complete the proof of Theorem 4.1. □ □

Remark 4.2. *For BRK-RK, if $s = m$, $v = k$ and $\alpha = 1$, then we have $\beta_{\max}^{\mathcal{I}} = \beta_{\max}^{\mathcal{J}} = 1$. In addition, we obtain $\eta = 1 - \frac{\sigma_{\min}^2(V)}{\|V\|_F^2} = \alpha_V$ and $\rho = 1 - \frac{\sigma_{\min}^2(U)}{\|U\|_F^2} = \alpha_U$. Similar to the works in [23], we have*

$$\|b_t - b_*\|_2^2 = \|b_t - \widetilde{b}_t\|_2^2 + \|\widetilde{b}_t - b_*\|_2^2. \tag{32}$$

Therefore, by direct computations we have

$$\mathbb{E}[\|b_t - b_*\|_2^2] \leq \alpha_V^t\|b_*\|_2^2 + \frac{\|x_*\|_2^2}{\|V\|_F^2}\sum_{l=0}^{t-1}\alpha_U^{t-l}\alpha_V^h, \tag{33}$$

which is the same as the conclusion proposed in [7, 27].

Remark 4.3. *For BREK-RK, if $s = m$, $v = k$ and $\alpha = 1$, then we have $\widetilde{\beta}_{\max}^{\mathcal{I}} = \widetilde{\beta}_{\max}^{\mathcal{J}} = 1$. In addition, we obtain $\widetilde{\eta} = 1 - \frac{\sigma_{\min}^2(U)}{\|U\|_F^2} = \alpha_U$ and $\widetilde{\rho} = 1 - \frac{\sigma_{\min}^2(U)}{\|U\|_F^2} = \alpha_U$, respectively. Similar to the works in Remark 4.2, we obtain*

$$\mathbb{E}[\|b_t - b_*\|_2^2] \leq \alpha_V\mathbb{E}[\|b_t - b_*\|_2^2] + \frac{1}{\|V\|_F^2}\Big[\alpha_U^t\|x_*\|_2^2 + \frac{\|z_0 - y_\perp\|_2^2}{\|U\|_F^2}\sum_{l=0}^{t-1}\alpha_U^{t-l}\alpha_U^l\Big]. \tag{34}$$

Using the fact that $\|z_0 - y_\perp\|_2^2 \le \sigma_{max}^2(U)\|x_*\|_2^2$, *we obtain*

$$\mathbb{E}[\|b_t - b_*\|_2^2] \le \alpha_V \mathbb{E}[\|b_t - b_*\|_2^2] + \frac{1}{\|V\|_F^2}\Big[\alpha_U^t \|x_*\|_2^2 + \frac{\sigma_{\max}^2(U)\|x_*\|_2^2}{\|U\|_F^2}\sum_{l=0}^{t-1}\alpha_U^t\Big]. \tag{35}$$

By direct computations, we obtain

$$\begin{aligned}
\mathbb{E}[\|b_t - b_*\|_2^2] &\le \alpha_V \mathbb{E}[\|b_t - b_*\|_2^2] + \frac{\|x_*\|_2^2}{\|V\|_F^2}\Big[\alpha_U^t + \frac{t\alpha_U^t \sigma_{\max}^2(U)}{\|U\|_F^2}\Big] \\
&< \alpha_V \mathbb{E}[\|b_t - b_*\|_2^2] + \frac{\|x_*\|_2^2}{\|V\|_F^2}\alpha_U^{\lfloor t/2\rfloor}\Big[1 + 2\lceil t/2\rceil \alpha_U^{t-\lfloor t/2\rfloor}\frac{\sigma_{\max}^2(U)}{\|U\|_F^2}\Big] \\
&< \alpha_V \mathbb{E}[\|b_t - b_*\|_2^2] + \frac{\|x_*\|_2^2}{\|V\|_F^2}\alpha_U^{\lfloor t/2\rfloor}\Big[1 + 2\frac{\sigma_{\max}^2(U)}{\|U\|_F^2}\sum_{l=0}^{\infty}\alpha_U^l\Big] \\
&= \alpha_V \mathbb{E}[\|b_t - b_*\|_2^2] + \frac{\|x_*\|_2^2}{\|V\|_F^2}\alpha_U^{\lfloor t/2\rfloor}\Big[1 + 2\frac{\sigma_{\max}^2(U)}{\|U\|_F^2}\frac{1}{1-\alpha_U}\Big] \\
&= \alpha_V \mathbb{E}[\|b_t - b_*\|_2^2] + \frac{\|x_*\|_2^2}{\|V\|_F^2}\alpha_U^{\lfloor t/2\rfloor}\Big[1 + 2\kappa_U^2\Big],
\end{aligned} \tag{36}$$

where $\kappa_U^2 = \frac{\sigma_{max}^2(U)}{\sigma_{min}^2(U)}$. *Therefore, from (36) and Theorem 1 in [7, 27], we observe that BREK-RK converges faster than REK-RK. Numerical results in Section 5 will assert this conclusion.*

5. Numerical examples

In this section, some numerical examples are tested to compare the effectiveness of RK-RK, REK-RK, BRK-RK and BREK-RK for solving different types of factorised linear systems. We note that IT and CPU denote the medians of the required iterations steps and the elapsed computing time (in seconds) averaged over 50 runs. Note that all experiments are performed in MATLAB (version R2019a) on a computer with an Intel Core i7-7700 processor at 3.60 GHz and 32 GB RAM. In our implementations, solving subsystem (2) and subsystem (3) is started from the initial guess $x_0 = \mathbf{zeros}(k, 1)$ and $b_0 = \mathbf{zeros}(n, 1)$, respectively. We note that the MATLAB function **randn** creates a random matrix with coefficients subject to the standard normal distribution $N(0, 1)$. When X is overdetermined and consistent, we consider different parameter values k and set $y = X\beta$, where β is generated by using the MATLAB function **randn**. On the other hand, when X is overdetermined and inconsistent, we consider different parameter values k, set $y = X\beta + r_0$ and $z_0 = y$, where $r_0 \in \mathbf{null}(X^*)$ (computed in MATLAB using the **null** function) and β is also generated by using the MATLAB function **randn**. We test two types of coefficient matrices. For type I, we set $U = \mathbf{randn}(m, k)$ and $V = \mathbf{randn}(k, n)$. For type II, we consider $U = U_1 \times D$ and $V = V_1^T$, where the matrices D, U_1 and V_1 are generated as follows:

$$[U_1, \sim] = \mathrm{qr}(\mathbf{randn}(m, r), 0), [V_1, \sim] = \mathrm{qr}(\mathbf{randn}(n, r), 0)$$

and

$$D = \mathrm{diag}(1 + (\kappa - 1). * \mathrm{rand}(r, 1)),$$

see [23] for more details. We adopt the same row partition and column partition introduced in [23]. For row partition $\{\mathcal{I}_i\}_{i=1}^s$, we let

$$\mathcal{I}_i = \{(i-1)\tau + 1, (i-1)\tau + 2, \ldots, i\tau\}, i = 1, 2, \ldots, s-1,$$

and

$$\mathcal{I}_s = \{(s-1)\tau + 1, (s-1)\tau + 2, \ldots, m\}, |\mathcal{I}_s| \leq \tau.$$

For column partition $\{\mathcal{J}_j\}_{j=1}^{v}$, we let

$$\mathcal{J}_j = \{(j-1)\tau + 1, (j-1)\tau + 2, \ldots, j\tau\}, j = 1, 2, \ldots, v-1,$$

and

$$\mathcal{J}_v = \{(v-1)\tau + 1, (v-1)\tau + 2, \ldots, k\}, |\mathcal{J}_v| \leq \tau.$$

We set the stopping criterion be **RSE** $= \frac{\|b_k - \beta_*\|_2^2}{\|\beta_*\|_2^2} \leq 10^{-6}$, or the maximum iteration steps be 100000, where **RSE** denotes the relative solution error. The speed-up of BRK-RK against RK-RK and the speed-up1 of BREK-RK against REK-RK are defined by

$$\textbf{speed-up} = \frac{\textbf{CPU of RK-RK}}{\textbf{CPU of BRK-RK}}$$

and

$$\textbf{speed-up1} = \frac{\textbf{CPU of REK-RK}}{\textbf{CPU of BREK-RK}},$$

respectively.

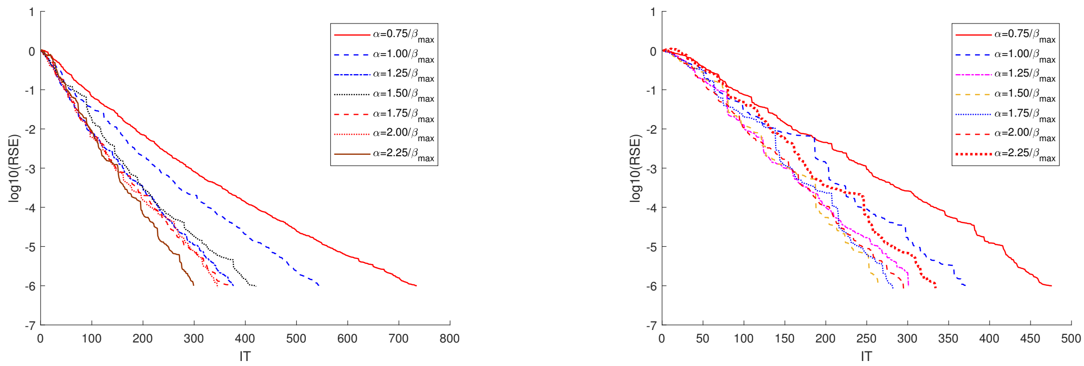

Figure 1: $\log_{10}(RSE)$ versus IT for BRK-RK when $X = \text{randn}(500, 250), k = 100$ (left) and $X = UDV^T, m = 500, n = 250, r = 150, \kappa = 2$ (right)

In Figure 1, we plot the RSE of BRK-RK with a fixed block size ($\tau = 10$) and different stepsizes (α from $0.75/\beta_{\max}$ to $2.25/\beta_{\max}$) for two consistent linear systems with coefficient matrices of Type I ($m = 500, n = 250, k = 100$) and Type II ($m = 500, n = 250, r = 150, \kappa = 2$). In Figure 2, we also plot the RSE of BREK-RK with a fixed block size ($\tau = 10$) and different stepsizes (α from $0.75/\beta_{\max}$ to $2.0/\beta_{\max}$ or $2.25/\beta_{\max}$) for two inconsistent linear systems with coefficient matrices of Type I ($m = 500, n = 250, k = 100$) and Type II ($m = 2000, n = 200, r = 100, \kappa = 2$). From Figure 1 on the left and Figure 2 on the left, we see that the convergence rate of BRK-RK or BREK-RK becomes faster with an increase in stepsize. From Figure 1 on the right and Figure 2 on the right, we see that the convergence rate of BRK-RK or BREK-RK becomes faster with an increase in stepsize and then slows down after reaching the fastest rate.

In Figure 3, we plot the computing time of BRK-RK with different block sizes $\tau = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$ and fixed stepsize $\alpha = 1.75/\beta_{\max}$ for two consistent linear systems with coefficient matrices of Type I ($A = \text{randn}(20000, 1000), k = 300$) and Type II ($A = UDV^T with \quad m = 2000, n = 1000, r = 500, \kappa = 2$). From Figure 3, we see that the CPU value reaches the minimum value at $\tau = 10$. In Tables 3-5, we give the

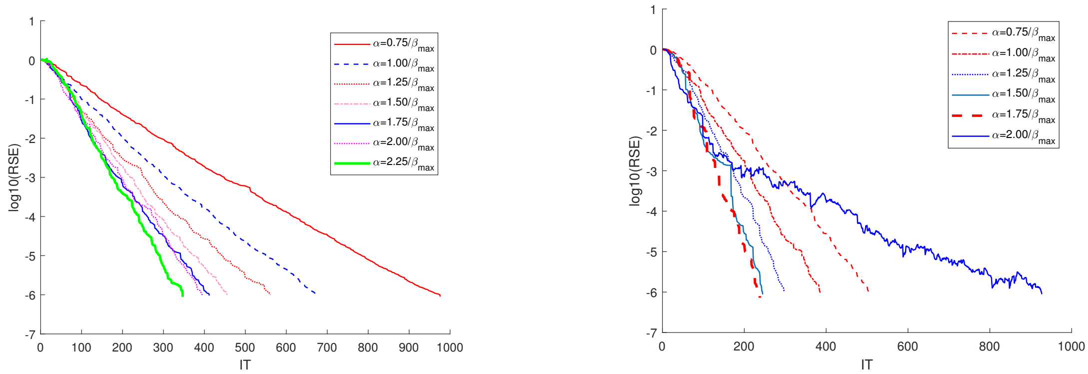

Figure 2: $\log_{10}(RSE)$ versus IT for BREK-RK when $X = \text{randn}(500, 250), k = 100$ (left) and $X = UDV^T, m = 2000, n = 200, r = 100, \kappa = 2$ (right)

Table 3: IT and CPU of RK-RK and BRK-RK for consistent X, where $U = \text{randn}(m, k)$ and $V = \text{randn}(k, n)$ with different k

$m \times n$	Method	k	100	200	300	400	500	600
20000×1000	RK-RK	IT	1712.0	4157.6	7743.3	14274.0	25624.0	52396.0
		CPU	1.387	3.362	6.371	11.920	22.193	44.945
	BRK-RK	IT	177.2	399.9	723.3	1226.0	2147.8	4184.8
		CPU	0.088	0.121	0.178	0.279	0.536	1.039
		speed-up	15.76	27.79	35.79	42.72	41.40	43.26
20000×2000	RK-RK	IT	1598.0	3460.5	5601.0	8225.0	11582.0	16053.0
		CPU	1.349	3.042	5.081	7.673	10.869	15.481
	BRK-RK	IT	164.9	338.9	535.1	785.2	1085.9	1397.7
		CPU	0.168	0.405	0.787	1.367	2.069	2.978
		speed-up	8.03	7.51	6.46	5.61	5.25	5.20
20000×3000	RK-RK	IT	1595.0	3338.6	5239.5	7319.6	9624.9	12460.0
		CPU	2.122	4.827	7.891	11.293	14.845	19.572
	BRK-RK	IT	164.1	332.1	538.7	729.5	948.5	1215.8
		CPU	0.230	0.653	1.239	1.861	2.613	3.478
		speed-up	9.23	7.39	6.37	6.07	5.68	5.63

Table 4: IT and CPU of RK-RK and BRK-RK for consistent X, where $U = \text{randn}(m, k)$ and $V = \text{randn}(k, n)$ with $k = 500$

$m \times n$	RK-RK		BRK-RK		speed-up
	IT	CPU	IT	CPU	
2000×1000	1712.0	1.387	177.2	0.088	15.76
3000×1000	26444.0	29.309	2188.7	0.587	49.93
4000×1000	28351.0	38.434	2340.5	2.200	17.47
5000×1000	25225.0	41.223	2111.5	2.183	18.88
6000×1000	25650.0	48.302	2145.8	2.303	20.93
7000×1000	25251.0	54.644	2172.9	2.445	22.35
8000×1000	26793.0	63.815	2246.6	2.724	23.43
9000×1000	23938.0	60.781	2091.6	2.641	23.01

Table 5: IT and CPU of RK-RK and BRK-RK for consistent X, where $U = U_1 \times D$ and $V = V_1^T$

$m \times n$	Rank	κ	RK-RK		BRK-RK		speed-up
			IT	CPU	IT	CPU	
2000×1000	500	2	9119.0	0.817	974.4	0.263	3.11
		6	35751.0	3.332	2940.9	0.693	4.81
4000×1000	500	2	9058.6	3.948	940.9	0.276	14.30
		6	33823.0	12.757	2961.9	0.713	17.89
6000×1000	500	2	9181.6	4.164	930.3	0.266	15.65
		6	35490.0	17.654	3075.2	0.753	23.44
8000×1000	500	2	9118.0	5.105	944.3	0.245	20.84
		6	33246.0	15.831	2798.9	0.587	26.97
10000×1000	500	2	9251.1	5.449	916.2	0.242	22.52
		6	34539.0	20.207	2975.3	0.706	28.62
12000×1000	500	2	9171.5	5.945	928.3	0.249	23.88
		6	33871.0	21.198	2966.8	0.643	32.99
14000×1000	500	2	9047.8	6.139	912.1	0.245	25.06
		6	31385.0	20.958	2734.0	0.621	33.75
16000×1000	500	2	9057.3	6.543	942.1	0.265	24.69
		6	30715.0	21.879	2708.8	0.624	35.06

numerical results of RK-RK and BRK-RK with two different types of coefficient matrices. Here, we use a fixed block size $\tau = 10$ and stepsize $\alpha = 1.75/\beta_{\max}$. From Tables 3-5, we can conclude several observations. First, RK-RK and BRK-RK are effective to solve the factorised linear systems. Second, BRK-RK outperforms RK-RK in terms of both iteration steps and computing time. Third, for Type I, the minimum of speed-ups is 5.20 and the maximum is 49.93. For type II, the minimum of speed-ups is 3.11 and the maximum is 35.06. In addition, for the fixed n, r and κ, the speed-up is increasing with respect to the increase of m. For the fixed m, r and n, the speed-up is increasing with respect to the increase of κ. In addition, from Figure 4, we observe that BRK-RK outperforms RK-RK in terms of computing time.

In Table 6, we report the numerical results of RK-RK and BRK-RK for extremely large standard Gaussian matrices U and V. We see that RK-RK and BRK-RK are effective to solve the linear system $UV\beta = y$ without computing the entire matrix. BRK-RK outperforms RK-RK in terms of both iteration steps and computing time, the minimum of speed-ups is 3.95 and the RK-RK method fails to converge with $k = 4000$ and 5000.

Table 6: IT and CPU of RK-RK and BRK-RK for consistent X, where $U = \text{randn}(10^5, k)$ and $V = \text{randn}(k, 10^4)$ with different k

$m \times n$	k	RK-RK		BRK-RK		speed-up
		IT	CPU	IT	CPU	
$10^5 \times 10^4$	1000	17737.0	80.249	1955.7	18.435	4.35
	2000	41612.0	219.823	4547.4	55.631	3.95
	3000	79133.0	446.798	7874.1	106.817	4.18
	4000	–	–	12148.0	187.921	–
	5000	–	–	3944.1	63.856	–

In Figure 5, we plot the computing time of BREK-RK with different block sizes $\tau = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$ and fixed stepsize $\alpha = 1.75/\beta_{max}$ for two inconsistent linear systems with coefficient matrices of Type I

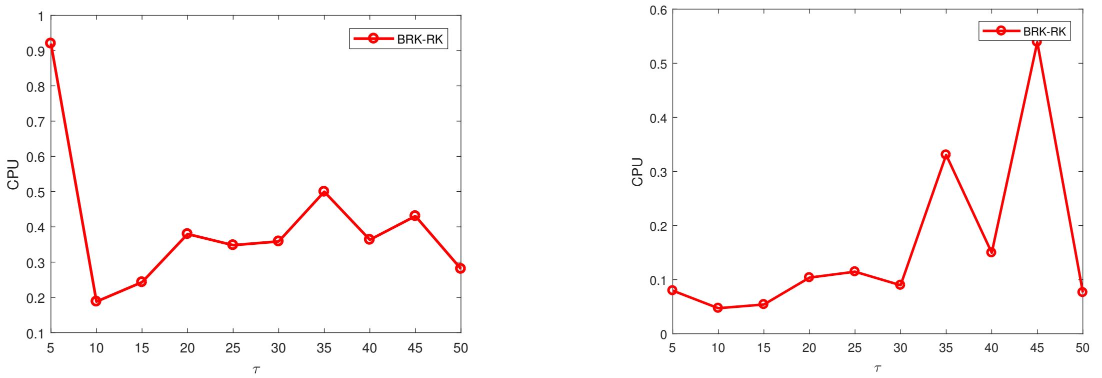

Figure 3: The average CPU of BRK-RK with different block sizes $\tau = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$ and stepsize $\alpha = 1.75/\beta_{max}$ for consistent linear systems. Left: $X = \text{randn}(20000, 1000), k = 300$. Right: $X = UDV^T, m = 2000, n = 1000, r = 500, \kappa = 2$.

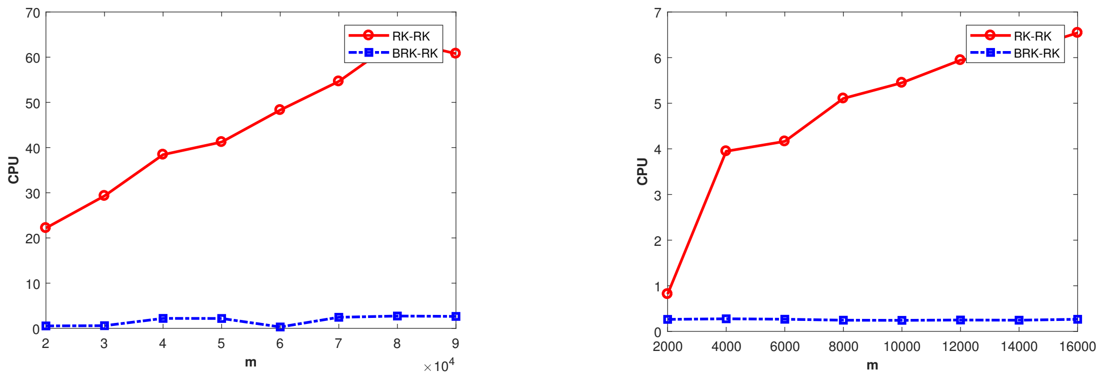

Figure 4: The average CPU of RK-RK, BRK-RK ($\tau = 10$ and stepsize $\alpha = 1.75/\beta_{max}$) for consistent linear systems. Left: $X = \text{randn}(m, 1000), m = 20000, \cdots, 90000, k = 500$. Right: $X = UDV^T, m = 2000, \cdots, 16000, n = 1000, r = 500, \kappa = 2$.

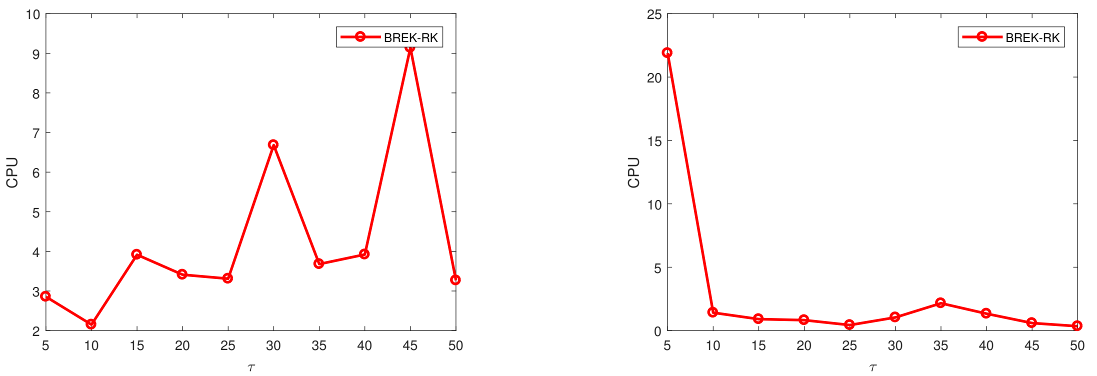

Figure 5: The average CPU of BREK-RK with different block sizes $\tau = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$ and stepsize $\alpha = 1.75/\beta_{max}$ for inconsistent linear systems. Left: $X = randn(10000, 1000), k = 300$. Right: $X = UDV^T, m = 2000, n = 500, r = 250, \kappa = 2$.

Table 7: IT and CPU of REK-RK and BREK-RK for inconsistent X, where U = randn(m, k) and V = randn(k, n) with different k

$m \times n$	Method	k	100	200	300	400	500
10000×1000	REK-RK	IT	1940.0	4557.0	8410.3	15175.0	28734.0
		CPU	2.301	5.266	9.941	18.529	35.035
	BREK-RK	IT	194.0	424.6	761.3	1302.5	2318.8
		CPU	0.501	1.125	2.125	3.821	6.937
		speed-up	4.59	4.68	4.68	4.85	5.05
12000×1000	REK-RK	IT	1904.8	4444.0	8748.2	15539.0	27665.0
		CPU	2.671	5.991	12.026	22.123	38.621
	BREK-RK	IT	191.8	416.7	778.5	1346.4	2254.4
		CPU	0.582	1.207	2.454	4.507	7.559
		speed-up	4.59	4.96	4.90	4.91	5.11
14000×1000	REK-RK	IT	1908.8	4459.8	8258.2	14874.0	26574.0
		CPU	2.721	7.372	14.182	26.013	40.014
	BREK-RK	IT	191.7	421.1	752.6	1314.6	2262.5
		CPU	1.097	2.657	4.715	8.703	14.127
		speed-up	2.48	2.77	3.01	2.99	2.83
16000×1000	REK-RK	IT	1884.8	4581.3	8433.0	15512.0	26565.0
		CPU	3.337	7.652	14.367	27.392	49.337
	BREK-RK	IT	187.8	420.7	758.3	1346.4	2235.5
		CPU	1.211	2.837	5.489	9.499	16.165
		speed-up	2.76	2.70	2.62	2.88	3.05
18000×1000	REK-RK	IT	1866.8	4438.8	8419.8	14636.0	27948.0
		CPU	3.252	6.237	12.500	23.735	53.362
	BREK-RK	IT	199.9	416.7	772.9	1276.0	2307.4
		CPU	1.423	2.634	4.978	8.788	17.729
		speed-up	2.29	2.37	2.51	2.70	3.01

Table 8: IT and CPU of REK-RK and BREK-RK for inconsistent X, where $U = U_1 \times D$ and $V = V_1^T$

$m \times n$	Rank	κ	REK-RK		BREK-RK		speed-up
			IT	CPU	IT	CPU	
4000×1000	100	2	2072.8	1.875	99.4	0.288	6.51
	100	6	6958.9	6.299	505.6	1.471	4.28
6000×1000	100	2	2245.6	2.421	117.1	0.641	3.78
	100	6	7652.7	8.711	590.1	4.106	2.12
8000×1000	100	2	2095.0	2.546	107.4	0.913	2.79
	100	6	9429.3	11.292	671.4	5.267	2.14
10000×1000	100	2	2261.5	3.078	130.4	1.330	2.31
	100	6	7889.1	10.618	578.6	5.372	1.98
12000×1000	100	2	2174.5	3.195	109.9	1.232	2.59
	100	6	9017.4	13.129	641.5	6.781	1.94

(A = randn(10000, 1000), $k = 300$) and Type II ($A = UDV^T with \quad m = 2000, n = 500, r = 200, \kappa = 2$). From Figure 5, we see that the CPU value reaches the minimum value at $\tau = 10$ for Type I, and the CPU value reaches the minimum value at $\tau = 25$ for Type II. In Tables 7-9, we report the numerical results of REK-RK

Table 9: IT and CPU of REK-RK and BREK-RK for consistent X, where $U = \text{randn}(3000, k)$ and $V = \text{randn}(k, 100)$ with different k

Method	k	200	300	400	500	600	700	800
REK-RK	IT	5284.9	5868.7	8378.5	11542.0	15166.0	19353.0	24588.0
	CPU	2.718	3.088	4.664	6.655	9.110	11.645	15.636
BREK-RK	IT	242.9	266.7	395.5	557.8	769.6	986.5	1193.7
	CPU	0.409	0.494	0.798	1.154	1.769	2.321	3.061
	speed-up	6.65	6.25	5.84	5.77	5.15	5.02	5.10

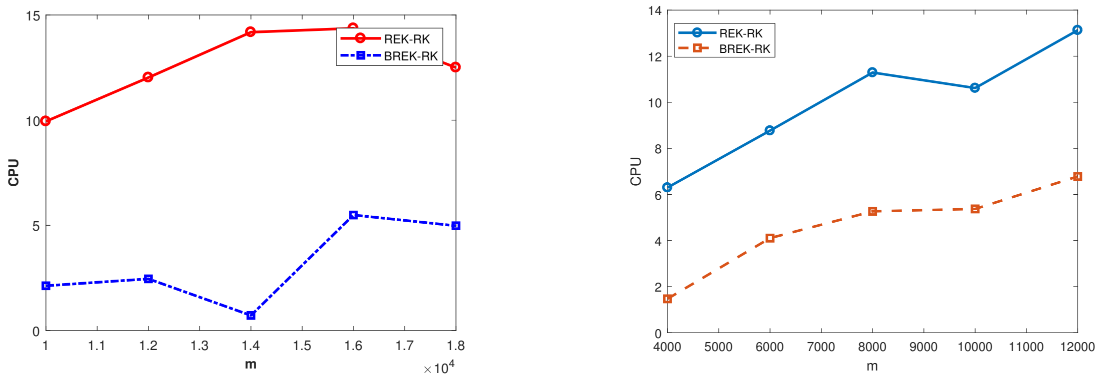

Figure 6: The average CPU of REK-RK, BREK-RK ($\tau = 10, 25$ and stepsize $\alpha = 1.75/\beta_{max}$) for inconsistent linear systems. Left: $X = randn(m, 1000), m = 10000, \cdots, 18000, k = 200$. Right: $X = UDV^T, m = 4000, \cdots, 12000, n = 1000, r = 200, \kappa = 2$.

and BREK-RK with two different types of coefficient matrices. Here, we use a fixed stepsize $\alpha = 1.75/\beta_{max}$. A fixed block size $\tau = 10$ is used in Table 7 and a fixed block size $\tau = 25$ is used in Tables 8-9. From Tables 7-9, we can conclude several observations. First, REK-RK and BREK-RK are effective to solve the factorised linear systems. Second, BREK-RK outperforms REK-RK in terms of both iteration steps and computing time. Third, for Type I, the minimum of speed-ups is 2.29 and the maximum is 6.65, the speed-up is increasing with respect to the increase of k. For Type II, the minimum of speed-ups is 1.94 and the maximum is 6.51. Finally, from Table 9, we also see that BREK-RK converge faster than REK-RK for solving the consistent linear system with $n < k < m$. In addition, from Figure 6, we also observe that BREK-RK outperforms REK-RK for solving two inconsistent linear systems with different coefficient matrices.

6. Conclusion

We have presented two pseudoinverse-free block methods which intertwine RABK or BREK and RABK for solving the factorised linear systems. The convergence theories of two new block iterative methods are also analyzed. Numerical results are provided to confirm the theoretical results and the effectiveness of the new methods. Accelerated variants and extensions will be the future work.

7. Statements and Declarations

The authors declare that they have no conflict of interest.

Acknowledgement(s)

The authors are grateful to the editors and referees for their valuable comments and suggestions which helped to improve the original manuscript of this paper.

References

[1] G. Adomavicius, A. Tuzhilin, *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions*, IEEE Trans. Knowl. Data. Eng. 17 (2005), 734–749.
[2] Y. Koren, B. Bell, C. Volinsky, *Matrix factorization techniques for recommender systems*, Computer. 42 (2009), 30–37.
[3] R. H. Keshavan, A. Montanari, S. Oh, *Matrix completion from noisy entries*, J. Mach. Learn. Res. 11 (2010), 2057–2078.
[4] V. Koltchinskii, K. Lounici, A. B. Tsybakov, *Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion*, Ann. Statist. 39 (2011), 2302–2329.
[5] K. Verbert, N. Manouselis, X. Ochoa, et al., *Context-aware recommender systems for learning: A survey and future challenges*, IEEE Trans. Learn. Technol. 5 (2012), 318–335.
[6] J. Goes, T. Zhang, R. Arora, et al., *Robust stochastic principal component analysis*, Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, PMLR. 33 (2014), 266-274.
[7] A. Ma, D. Needell, A. Ramdas, *Iterative methods for solving factorized linear systems*, SIAM J. Matrix Anal. Appl. 39 (2018), 104–122.
[8] T. Strohmer, R. Vershynin, *A randomized Kaczmarz algorithm with exponential convergence*, J. Fourier Anal. Appl, 15 (2009) 262–278.
[9] D. Needell, *Randomized Kaczmarz solver for nosiy linear systems*, BIT. 50 (2010), 395–403.
[10] A. Ma, D. Needell, A. Ramdas, *Convergence properties of the randomized extended Gauss-Seidel and Kaczmarz methods*, SIAM J. Matrix Anal. Appl. 36 (2015), 1590–1604.
[11] R. M. Gower, P. Richtárik, *Randomized iterative methods for linear systems*, SIAM J. Matrix Anal. Appl. 36 (2015), 1660–1690.
[12] Z.-Z. Bai, W.-T. Wu, *On greedy randomized Kaczmarz method for solving large sparse linear systems*, SIAM J. Sci. Comput. 40 (2018), A592–A606.
[13] Z.-Z. Bai, W.-T. Wu, *On greedy randomized coordinate descent methods for solving large linear least-squares problems*, Numer. Linear Algebra Appl. 26 (2019), e2237:1–15.
[14] Z.-Z. Bai, W.-T. Wu, *On relaxed greedy randomized Kaczmarz methods for solving large sparse linear systems*, Appl. Math. Lett. 83 (2018), 21–26.
[15] K. Du, H. Gao, *A new theoretical estimate for the convergence rate of the maximal weighted residual Kaczmarz algorithm*, Numer. Math. Theor. Meth. Appl. 12 (2019), 627–639.
[16] H.-Y. Li, Y.-J. Zhang, *A novel greedy Kaczmarz method for solving consistent linear systems*, preprint, arXiv:2004.02062, 2020.
[17] D. Leventhal, A. S. Lewis, *Randomized methods for linear constraints: Convergence rates and conditioning*, Math. Oper. Res. 35 (2010), 641–654.
[18] A. Zouzias, N. M. Freris, *Randomized extended Kaczmarz for solving least squares*, SIAM J. Matrix Anal. Appl. 34 (2013), 773–793.
[19] T. Elfving, *Block-iterative methods for consistent and inconsistent linear equations*, Numer. Math. 35 (1980), 1–12.
[20] D. Needell, J. A. Tropp, *Paved with good intentions: analysis of a randomized block Kaczmarz method*, Linear Algebra Appl. 441 (2014), 199–221.
[21] D. Needell, R. Zhao, A. Zouzias, *Randomized blcok Kaczmarz method with projection for solving least squares*, Linear Algebra Appl. 484 (2015), 322–343.
[22] I. Necoara, *Faster randomized block Kaczmarz algorithm*, SIAM J. Matrix Anal. Appl. 40 (2019), 1425–1452.
[23] K. Du, W.-T. Si, X.-H. Sun, *Randomized extended average block Kaczmarz for solving least squares*, SIAM J. Sci. Comput. 42 (2020), A3541–A3559.
[24] J. D. Moorman, T. K. Tu, D. Molitor, D. Needell, *Randomized Kaczmarz with averaging*, BIT. 61 (2021), 337–359.
[25] P. Richtárik, M. Takáč, *Stochastic Reformulations of Linear Systems: Algorithms and Convergence Theory*, SIAM J. Matrix Anal. Appl. 41 (2020), 487–524.
[26] K. Du, X.-H. Sun, *A doubly stochastic block Gauss-Seidel algorithm for solving linear equations*, Appl. Math. Comput. 408 (2021), 126373.
[27] A. Ma A, D. Needell, A. Ramdas, *Iterative methods for solving factorized linear systems*, preprint, arXiv:1701.07453v4, 2019.
[28] J. Zhao, X. Wang, J.-H. Zhang, *A randomised iterative method for solving factorised linear systems*, Linear and Multilinear Algebra. 2022, https://doi.org/10.1080/03081087.2021.2025198.
[29] S-M. Liu, Y. Liu, *On Relaxed Greedy Randomized Iterative Methods for the Solution of Factorized Linear Systems*, Taiwanese J. Math. 26 (2022), 953-979.
[30] K. Du, *Regularized randomized iterative algorithms for factorized linear*, preprint, arXiv:2204.10722v1, 2022.