Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 8(8): 18512–18527.
DOI: 10.3934/math.2023941
Received: 22 December 2022
Revised: 08 May 2023
Accepted: 15 May 2023
Published: 31 May 2023

Research article

A partially block randomized extended Kaczmarz method for solving large overdetermined inconsistent linear systems

Feng Yin[1,3,*], **Bu-Yue Zhang**[1,3] **and Guang-Xin Huang**[2,3]

[1] College of Mathematical and Physics, Chengdu University of Technology, Chengdu 610059, China
[2] College of Computer Science and Cyber Security, Chengdu University of Technology, Chengdu 610059, China
[3] Sichuan Geomathematics Key Laboratory, Chengdu University of Technology, Chengdu 610059, China

* **Correspondence:** Email: fyinsuse@163.com.

Abstract: This paper presents a partial block randomized extended Kaczmarz (PBREK) method for solving large overdetermined inconsistent linear system of equations $Ax = b$. The convergence theorem of the PBREK method is derived. Several examples are given to illustrate the effectiveness of the proposed PBREK method compared with the prevuious PREK method and the randomized extended Kaczmarz (REK) method.

Keywords: partial block randomized extended Kaczmarz; overdetermined inconsistent systems; matrix paving; Kaczmarz method
Mathematics Subject Classification: 65F10, 65F20

1. Introduction

In this paper, we discuss a randomized method for solving large linear inconsistent system of equations of the form

$$Ax = b, \tag{1.1}$$

where the matrix $A \in \mathbb{R}^{m\times n}$ with $m > n$ and the vector $b \in \mathbb{R}^m$ is known, while the vector $x \in \mathbb{R}^n$ is unknown.

Many problems in science and engineering fields such as computed tomography and image reconstruction need to solve (1.1). The algebraic reconstruction technique is one of important method to solve large-scale linear systems of Eq (1.1).

The classical Kaczmarz method in [7] is an iterative projection method for solving the system of Eq (1.1). The main idea of this method is to carry out a series of orthogonal projection iterations continuously. Each iteration projects the current point onto the hyperplane formed by the selected row. Let x_0 be the initial vector and the $(k+1)$th iteration vector be given by

$$x^{k+1} := x^k + \frac{b_i - < A^{(i)}, x^k >}{\left\|A^{(i)}\right\|_2^2} A^{(i)}, \tag{1.2}$$

where $A^{(i)}$ represents the i-th row of the coefficient matrix A. We assume that throughout this paper that A has no zero rows, and b_i represent the i-th entry of the vector b. $\|\cdot\|_2$ denotes the Euclidean norm and $< \cdot, \cdot >$ is the Euclidean inner product. The exact solution of the system of linear equations is gradually approached after many iterations. With the order of each row of the coefficient matrix A, the current approximate vector is orthogonally projected onto the hyperplane of the solution, i.e., $\{x| < A^{(i)}, x >= b_i\}$ and the resulting vector is put into the next iteration.

There are several extended Kaczmarz methods having been published in recent years. In 2009, Strohmer and Vershynin [13] proposed a randomized version of the Kaczmarz method to solve consistent and overdetermined linear systems. The authors specified a rule for selecting the working rows of the coefficient matrix A, and showed a convergence theorem of the randomized Kaczmarz (RK) method, which improves the RK method with exponential convergence. The RK method can be represented as follows:

$$x^{k+1} := x^k + \frac{b_{r(i)} - < A_{r(i)}, x_k >}{\left\|A_{r(i)}\right\|_2^2} A_{r(i)}, \tag{1.3}$$

where $r(i)$ denotes the working row index of the coefficient matrix A, which is randomly selected from the set $\{1, 2, ..., m\}$. The working rows are selected with the probability $Pr(row = i_k) = \frac{\left\|A_{r(i)}\right\|_2^2}{\|A\|_F^2}$. The RK method randomly select the rows of matrix A according the row selection probability, which is the ratio of the Euclidean norm of each row of A and the Frobenius norm of A. In this way, the RK method will give priority to the row vector with large norm in the process of selecting the working row of matrix A, and does not need to use the whole matrix A, which greatly saves the calculation cost. Based on these advantages, Liu and Wright [8] proposed an accelerated randomized Kaczmarz (ARK) algorithm. Wu and Xiang [14] proposed a projected randomized Kaczmarz (PRK) method. The Kaczmarz algorithm is used to solve Sylvester matrix equations in [6]. However, when A is a quantity matrix, the norm of some row vectors is the same, and the probability of selecting each row is the same. In this case, the RK method is similar to the classical Kaczmarz method. Let x^* be the least-squares solution of the overdetermined inconsistent system of linear equation (1.1), then $x^* = A^\dagger b$, where $A^\dagger$ denotes the Moore-Penrose pseudoinverse of A. Consider the system of linear equation

$$Ax^* = b - r, \tag{1.4}$$

where $r \in \mathbb{R}^m$ is a nonzero vector in the null space of A, i.e., $A^T r = 0$. Apparently, the RK method does not converge to x^* when it is applied to solve the linear system $Ax = b$. We represent b by $b = b_{R(A)} + b_{R(A)^\perp}$, where $b_{R(A)}$ is the projection of b on the column space of coefficient matrix A, i.e., $R(A) := \{Ax | x \in \mathbb{R}^n\}$. Zouzias and Freris [15] proposed the randomized extended Kaczmarz method and is effective for obtaining the least squares solution of the system (1.1). In 2016, Petra and Popa [12] further studied the theory of REK method. Subsequently, Dumitrescu [2] proposed a randomized coordinate descent method, which combines the coordinate descent method with standard Kacmarz method and solves the least square problem of an overdetermined linear equations. Elfving [5] and Eggermont et al. [4] presented block iterative methods to solve (1.1). In 2015, Needell and Zhao [10] further proposed a randomized double block Kaczmarz (RDBK) method for solving the system (1.1), which is based on the block Kaczmarz method [9] and the randomized extended Kaczmarz method. The row block and column block are selected according to uniform distribution, which improve the convergence speed of the RDBK algorithm. The disadvantage of the RDBK method is that it has to calculate the pseudoinverse, which increases the calculation time, and the calculation is very difficult when the size of A is large enough. Necoara [11] proposed the randomized block Kaczmarz (RBK) method in 2019. The RBK algorithm partitions the rows of A and introduces weight and relaxation parameters. In 2020, Du and Si [3] presented a randomized extended average block Kaczmarz (REABK) method for solving (1.1). The REABK method avoids the computation of the pseudoinverse of A. In 2019, Bai and Wu [1] improved the REK method [15] on the column selection strategy and proposed a partially randomized extended Kaczmarz (PREK) method for overdetermined inconsistent systems. In the k-th step of the PREK method, the working columns are selected according to the order of the columns of A, i.e., $j_k = (k \ mod \ n) + 1$, rather than the probability, which is the ratio of the square of the Euclidean norm of the jth column of A to the square of the Frobenius norm of A. This rule of column section ensures that every column is selected and the frequency distribution is uniform, thus avoids always selecting those columns with the largest column norm in the REK method and ignoring those columns with smaller norm. Furthermore, the cost of calculation is reduced and the solution converges faster than the PREK method.

This paper present a partially block randomized extended Kaczmarz method. In the PBREK method, the rows of the coefficient matrix A are randomly divided into several blocks and stored in the working area. In each iteration, the submatrix is selected to enter the iteration according to the probability that is directly proportional to the Frobenius norm of the submatrix. Our block method avoids pseudoinverse and avoids the repeated selection of row vectors with the same probability. The specific algorithm is shown in Algorithm 2. The convergence is considered and several numerical examples are given to show effectiveness of the proposed method for the overdetermined inconsistent linear systems.

The rest of the paper is organized as follows. Section 2.1 introduces some symbols and notations, and Section 2.2 summarizes the PREK method. We present the PBREK method and prove its convergence in Section 3. Section 4 shows several numerical examples to illustrate the proposed method and Section 5 draws some conclusions.

2. Preliminaries and notations

2.1. Notations

Let $A_{(j)}$ be the j-th column of the coefficient matrix $A \in \mathbb{R}^{m\times n}$ and $A^{(i)}$ be the i-th row of A. i_k represents the ith row of matrix A in the kth iteration. $A^\dagger$ denotes the Moore–Penrose pseudoinverse of A. $R(A)$ represents the column space of A, i.e., $R(A) := \{Ax|x \in \mathbb{R}^n\}$. $R(A)^\perp$ is the orthogonal complement of $R(A)$. $b_{R(A)}$ is the projection vector of b onto the column space of matrix A. $b^\perp_{R(A)}$ denotes the orthogonal complement of $b_{R(A)}$, i.e., $b = b_{R(A)} + b^\perp_{R(A)}$. Let $\kappa^2(A) := \dfrac{\sigma^2_{max}}{\sigma^2_{min}}$ denote the condition number. We use $||A||_F := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$ and $||A||_2 := max_{x\neq 0}\frac{||Ax||_2}{||x||_2}$ to denote the Frobenius norm and Euclidean norm of matrix A. Let $\lfloor m \rfloor$ denote for the largest integer that does not exceed m and τ be the number of rows of the submatrix. In addition, we also use $\mathbb{E}_k$ to denote the expected value conditional on the first k iterations, i.e.,

$$\mathbb{E}_k[\cdot] = \mathbb{E}[\cdot|i_0, j_0, i_1, j_1, ..., i_{k-1}, j_{k-1}] \tag{2.1}$$

and

$$\mathbb{E}_k^i[\cdot] = \mathbb{E}[\cdot|i_0, j_0, i_1, j_1, ..., i_{k-1}, j_{k-1}, j_k]. \tag{2.2}$$

It holds that $\mathbb{E}[\mathbb{E}_k[\cdot]] = \mathbb{E}[\cdot]$ and $\mathbb{E}_k[\cdot] = \mathbb{E}_k^i[\mathbb{E}_k^j[\cdot]]$.

2.2. *The PREK method*

Algorithm 1 summarizes the PREK algorithm in [1]. The PREK algorithm consists of two parts. The first part is a randomized version of Kaczmarz method, which selects the row vectors according to the ratio of the norm, and the second part is an orthogonal projection method, which selects the column vectors in order.

Algorithm 1: The PREK Algorithm

```
1 Input: $A, b, l, x_0 = 0, z_0 = b$.
2 Output: $x_l$
3 for $k = 0, 1, 2, ..., l-1$ do
4     Select $i_k \in \{1, 2, ..., m\}$ with probability $Pr(row = i_k) = \frac{||A^{(i_k)}||_2^2}{||A||_F^2}$.
5     Set $x_{k+1} = x_k + \frac{(b^{(i_k)} - z_k^{(i_k)} - A^{(i_k)}x^k)}{||A^{(i_k)}||_2^2}(A^{(i_k)})^T$.
6     Select $j_k = (k \ mod \ n) + 1$.
7     Set $z_{k+1} = z_k - \frac{A_{(j_k)}^T z_k}{||A_{(j_k)}||_2^2} A_{(j_k)}$.
8 end
```

3. The partially block randomized extended Kaczmarz algorithm

3.1. *The PBREK method*

In this section, we present the PBREK algorithm. For the PREK method in [1], it may repeatedly select rows with a high probability or the same probability. We introduce the block idea to improve the first part of the PREK algorithm.

In the first part of the PBREK method, the coefficient matrix A is divided into several submatrices. We assume that the submatrix of A has τ rows. If the partition size can partition the number of rows of matrix A exactly, there are $\frac{m}{\tau}$ submatrices in total. If the partition size does not accurately divide the number of rows in matrix A, the remaining row vectors are added to the previous submatrix. For example, let $\{I_1, I_2 ... I_s\}$ be a row partition. When m is divisible by s, then the partitioning is easy.

Otherwise let c be the remainder of m divided by s, then we first arrange the number of τ in each of the s partitions, and put the remaining c rows in the first s partitions respectively. Thus, $|I_j|$ is τ or $\tau + 1$, where $|I_j|$ denotes the number of elements in set I_j.

After partitioning the coefficient matrix A, we can select the index of block. Step 4 selects row index of each block according to the probability, which is equal to the ratio of the square of the Frobenius norm of the submatrix to the square of the Frobenius norm of matrix A. We set $z_0 = b$, so $b_{R(A)} \approx 0$. In step 5, we update x_k with the iterative method of the RK method, and update vector z. The iteration terminates when it satisfies the iteration stop citerion $RSE = \frac{\|x_k - x^*\|_2^2}{\|x^*\|_2^2} < 10^{-6}$. In the second part of orthogonal projection method, the column selection strategy of the PREK method is still adopted, i.e., $j_k = (k \ mod \ n) + 1$. The vector z_k is updated by sequential iteration according to the original column vectors of matrix A. As the number of iterations increases, $b - z_k$ gets closer and closer to $b_{R(A)}$. Thus the linear system $Ax \approx b_{R(A)}$, and the RK method can be used to solve this new linear system. Algorithm 2 summarizes the PBREK algorithm.

Algorithm 2: The PBREK Algorithm

```
1 Input: Let $\{I_1, I_2, ..., I_s\}$ be partitions of [m], $z_0 \in \mathbb{R}^m$ and $x_0 \in \mathbb{R}^n$, $A$, $b$, $l$, $x_0 = 0$ and $z_0 = b$.
2 Output: $x_l$
3 for $k = 0, 1, 2, ..., l - 1$ do
4     Set $i \in [s]$ with probability $Pr(I_i \in [s]) = \frac{\|A_{(I_i,:)}\|_F^2}{\|A\|_F^2}$.
5     Set $x_{k+1} = x_k + \frac{A^T_{(I_i,:)}(b_{I_i} - z_k^{I_i} - A_{(I_i,:)}x_k)}{\|A_{(I_i,:)}\|_F^2}$.
6     Select $j_k = (k \ mod \ n) + 1$.
7     Set $z_{k+1} = z_k - \frac{A^T_{(j_k)} z_k}{\|A_{(j_k)}\|_2^2} A_{(j_k)}$.
8 end
```

3.2. Convergence of PBREK algorithm

In this subsection we discuss the convergence of Algorithm 2. First we need the following results.

Lemma 3.1. *Let $\{z_k\}$ be the same as that in Algorithm 2, then we have*

$$\left\|z_k - b_{R(A)^\perp}\right\|_2^2 \le \left\|b_{R(A)}\right\|_2^2. \tag{3.1}$$

Proof. The proof of Lemma 3.1 is similar to that of Theorem 3.1 in [1] and is omitted.

It is not difficult to have the results as follows.

Lemma 3.2. *Let $A \in \mathbb{R}^{m \times n}$ be any nonzero real matrix, then for every $x \in \mathbb{R}^n$, it holds that*

$$\|Ax\|_2^2 \le \sigma_{max}^2(A) \|x\|_2^2. \tag{3.2}$$

Based on Lemmas 3.1 and 3.2, we can establish the following convergence theory for the PBREK method.

Theorem 3.1. *Suppose the system of linear equation* (1.1) *is the overdetermined inconsistent and $A \in \mathbb{R}^{m\times n}$ $(m > n)$ is a full rank matrix. Let $x^* = A^\dagger b$ be the least square solution of* (1.1) *and x_k be the iteration sequence produced by Algorithm 2, then for any constant $\rho > 0$, it holds that*

$$E\|x_k - x^*\|_2^2 \le \left[(\eta + \rho\eta)^k + (1 + \frac{1}{\rho})\frac{\beta^I_{max}\sigma^2_{max}(A)}{\|A\|_F^2}\sum_{l=0}^{k-1}(1+\rho)^l\eta^l\right]\|x^*\|_2^2, \tag{3.3}$$

where $\eta = 1 - \frac{(2-\beta^I_{max})}{\|A\|_F^2}\sigma^2_{min}(A)$, $\eta > 0$ and $\beta^I_{max} = \max_{i\in[s]}\frac{\sigma^2_{max}(A_{(I_i,:)})}{\|A_{(I_i,:)}\|_F^2}$, $\beta^I_{max} \le 1$.

Proof. Let

$$X^k = x_{k-1} - \frac{A^T_{(I_i,:)}}{\|A_{(I_i,:)}\|_F^2}(A_{(I_i,:)})(x_{k-1} - x^*),$$

then,

$$\begin{aligned}
\left\|x_k - X^k\right\|_2^2 &= \frac{1}{\left\|A_{(I_i,:)}\right\|_F^4}\left\|(A^T_{(I_i,:)}(b_{I_i} - A_{(I_i,:)}x^* - z^{I_i}_{k-1})\right\|_2^2 \\
&\le \frac{1}{\left\|A_{(I_i,:)}\right\|_F^4}\left\|A^T_{(I_i,:)}\right\|_2^2\left\|b_{I_i} - A_{(I_i,:)}x^* - z^{I_i}_{k-1}\right\|_2^2 \\
&\le \frac{1}{\left\|A_{(I_i,:)}\right\|_F^2}\frac{\sigma^2_{max}(A_{(I_i,:)})}{\left\|A_{(I_i,:)}\right\|_F^2}\left\|b_{I_i} - A_{I_i}x^* - z^{I_i}_{k-1}\right\|_2^2 \\
&\le \frac{\beta^I_{max}}{\left\|A_{(I_i,:)}\right\|_F^2}\left\|b_{I_i} - A_{I_i}x^* - z^{I_i}_{k-1}\right\|_2^2.
\end{aligned}$$

Thus,

$$\begin{aligned}
E_{k-1}\left\|x_k - X^k\right\|_2^2 &= E_{k-1}\left[E^i_{k-1}\left\|x_k - X^k\right\|_2^2\right] \\
&\le E_{k-1}\left[E^i_{k-1}\left[\frac{\beta^I_{max}}{\left\|A_{(I_i,:)}\right\|_F^2}\left\|b_{I_i} - A_{I_i}x^* - z^{I_i}_{k-1}\right\|_2^2\right]\right] \\
&= E_{k-1}\left[\frac{\beta^I_{max}}{\|A\|_F^2}\|b - Ax^* - z_{k-1}\|_2^2\right].
\end{aligned}$$

Furthermore,

$$\begin{aligned}
E\left\|x_k - X^k\right\|_2^2 &= E\left[E_{k-1}\left\|x_k - X^k\right\|_2^2\right] \\
&\le E\left[E_{k-1}\left[\frac{\beta^I_{max}}{\|A\|_F^2}\|b - Ax^* - z_{k-1}\|_2^2\right]\right] \\
&= \frac{\beta^I_{max}}{\|A\|_F^2}E\|b - Ax^* - z_{k-1}\|_2^2 \\
&= \frac{\beta^I_{max}}{\|A\|_F^2}E\left\|b_{R(A)^\perp} - z_{k-1}\right\|_2^2.
\end{aligned} \tag{3.4}$$

We note that the fact that $\left\|b_{R(A)}\right\|_2^2 \leq \lambda_{max}(A^TA)\,\|x^*\|_2^2$. According to Lemma 3.1, we have

$$\left\|z_k - b_{R(A)^\perp}\right\|_2^2 \leq \left\|b_{R(A)}\right\|_2^2 \leq \lambda_{max}(A^TA)\,\|x^*\|_2^2. \tag{3.5}$$

Substituting (3.5) into (3.4) results in

$$E\left\|x_k - X^k\right\|_2^2 \leq \frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\,\|x^*\|_2^2. \tag{3.6}$$

From Algorithm 2, we have that

$$x_k - x^* = x_{k-1} - x^* - \frac{1}{\left\|A_{(I_i,:)}\right\|_F^2}A_{(I_i,:)}^T(A_{(I_i,:)}x_{k-1} - b_{I_i} + z_{k-1}^{I_i}), \tag{3.7}$$

where $x_0 \in R(A^T)$, $A^\dagger b \in R(A^T)$, $A_{(I_i,:)}^T(A_{(I_i,:)}x_{k-1} - b_{I_i} + z_{k-1}^{I_i}) \in R(A^T)$.

Furthermore, we have

$$\begin{aligned}
\left\|X^k - x^*\right\|_2^2 &= \left\|(x_{k-1} - x^*) - \frac{A_{(I_i,:)}^T}{\left\|A_{(I_i,:)}\right\|_F^2}A_{(I_i,:)}(x_{k-1} - x^*)\right\|_2^2 \\
&= \|x_{k-1} - x^*\|_2^2 + \left\|(\frac{A_{(I_i,:)}}{\left\|A_{(I_i,:)}\right\|_F})^T\frac{A_{(I_i,:)}}{\left\|A_{(I_i,:)}\right\|_F}(x_{k-1} - x^*)\right\|_2^2 \\
&\quad - 2\left\|(x_{k-1} - x^*)^T\frac{A_{(I_i,:)}^T}{\left\|A_{(I_i,:)}\right\|_F^2}A_{(I_i,:)}(x_{k-1} - x^*)\right\| \\
&= \|x_{k-1} - x^*\|_2^2 - \frac{2\left\|A_{(I_i,:)}(x_{k-1} - x^*)\right\|_2^2}{\left\|A_{(I_i,:)}\right\|_F^2} \\
&\quad + \left\|(\frac{A_{(I_i,:)}}{\left\|A_{(I_i,:)}\right\|_F})^T\frac{A_{(I_i,:)}}{\left\|A_{(I_i,:)}\right\|_F}(x_{k-1} - x^*)\right\|_2^2 \\
&\leq \|x_{k-1} - x^*\|_2^2 - (2 - \frac{\sigma_{max}^2A_{(I_i,:)}}{\left\|A_{I_i:}\right\|_F^2})\frac{\left\|A_{(I_i,:)}(x_{k-1} - x^*)\right\|_2^2}{\left\|A_{(I_i,:)}\right\|_F^2} \\
&\leq \|x_{k-1} - x^*\|_2^2 - \frac{(2 - \beta_{max}^I)\left\|A_{(I_i,:)}(x_{k-1} - x^*)\right\|_2^2}{\left\|A_{(I_i,:)}\right\|_F^2}.
\end{aligned}$$

Thus,

$$\begin{aligned}
E_{k-1}\left\|X^k - x^*\right\|_2^2 &\leq \|x_{k-1} - x^*\|_2^2 - \frac{(2 - \beta_{max}^I)}{\|A\|_F^2}\|A(x_{k-1} - x^*)\|_2^2 \\
&\leq \|x_{k-1} - x^*\|_2^2 - \frac{(2 - \beta_{max}^I)}{\|A\|_F^2}\sigma_{min}^2(A)\,\|x_{k-1} - x^*\|_2^2 \\
&= \left[1 - \frac{(2 - \beta_{max}^I)}{\|A\|_F^2}\sigma_{min}^2(A)\right]\|x_{k-1} - x^*\|_2^2 \\
&= \eta\|x_{k-1} - x^*\|_2^2.
\end{aligned}$$

From the above derivation results we know that

$$E_{k-1}\left\|X^k - x^*\right\|_2^2 \leq \eta \|x_{k-1} - x^*\|_2^2, \tag{3.8}$$

where $E_{k-1} \geq 0$ and $\|x_{k-1} - x^*\|_2^2 \geq 0$, thus $\eta \geq 0$. It follows that

$$E\left\|X^k - x^*\right\|_2^2 = E\left[E_{k-1}\left\|X^k - x^*\right\|_2^2\right] \leq \eta E\|x_{k-1} - x^*\|_2^2. \tag{3.9}$$

For any $\rho > 0$, we have

$$\begin{aligned}\|x_k - x^*\|_2^2 &= \left\|(X^k - x^*) + (x_k - X^k)\right\|_2^2 \\ &\leq (\left\|X^k - x^*\right\|_2 + \left\|x_k - X^k\right\|_2))^2 \\ &\leq \left\|x_k - X^k\right\|_2^2 + \left\|X^k - x^*\right\|_2^2 + 2\left\|x_k - X^k\right\|_2\left\|X^k - x^*\right\|_2 \\ &\leq (1 + \frac{1}{\rho})\left\|x_k - X^k\right\|_2^2 + (1+\rho)\left\|X^k - x^*\right\|_2^2.\end{aligned}$$

Thus, we can get

$$E\|x_k - x^*\|_2^2 \leq (1 + \frac{1}{\rho})E\left\|x_k - X^k\right\|_2^2 + (1+\rho)E\left\|X^k - x^*\right\|_2^2. \tag{3.10}$$

Now we do induction on the pth transformation. Assume that

$$E\|x_k - x^*\|_2^2 \leq (1 + \frac{1}{\rho})\frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\|x^*\|_2^2\sum_{l=0}^{p-1}(1+\rho)^l\eta^l + (\eta + \rho\eta)^p E\left[\left\|x_{k-p} - x^*\right\|_2^2\right]. \tag{3.11}$$

When $p = 1$, obviously (3.11) is true,

$$E\|x_k - x^*\|_2^2 \leq (1 + \frac{1}{\rho})\frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\|x^*\|_2^2 + (1+\rho)\eta E\|x_{k-1} - x^*\|_2^2. \tag{3.12}$$

Assume that when $p = k - 1$, (3.11) still holds,

$$E\|x_k - x^*\|_2^2 \leq (1 + \frac{1}{\rho})\frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\|x^*\|_2^2\sum_{l=0}^{k-2}(1+\rho)^l\eta^l + (\eta + \rho\eta)^{k-1}E\|x_1 - x^*\|_2^2. \tag{3.13}$$

Now we prove that $p = k$, (3.11) still holds. From (3.13), we can get

$$\begin{aligned}E\|x_k - x^*\|_2^2 &\leq (1 + \frac{1}{\rho})\frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\|x^*\|_2^2\sum_{l=0}^{k-2}(1+\rho)^l\eta^l \\ &\quad + (\eta + \rho\eta)^{k-1}\left[(1 + \frac{1}{\rho})\frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\|x^*\|_2^2 + (1+\rho)\eta E\|x_0 - x^*\|_2^2\right] \\ &= (1 + \frac{1}{\rho})\frac{\beta_{max}^I}{\|A\|_F^2}\sigma_{max}^2(A)\|x^*\|_2^2\sum_{l=0}^{k-1}(1+\rho)^l\eta^l + (\eta + \rho\eta)^k E\left[\|x_0 - x^*\|_2^2\right]. \end{aligned} \tag{3.14}$$

Obviously, (3.14) satisfies (3.11), therefore, the assumption is true.

This completes the proof.

When the number of blocks is the number of rows of A, Theorem 3.1 reduces to the following results.

Corollary 3.1. *For the overdetermined inconsistent system* (1.1)*, when $\tau = 1$, i.e., the number of blocks equals the number of rows of the coefficient matrix A, then the upper bound in* (3.3) *becomes*

$$E\|x_k - x^*\|_2^2 \le \left[\eta^k + \frac{\kappa^2(A)}{1-\eta}\right] \|x^*\|_2^2,$$

where $\eta = 1 - \frac{\sigma_{min}^2(A)}{\|A\|_F^2}$ $(0 \le \eta < 1)$.

Proof. When $s = m$, it holds that $\beta_{max}^I = 1$. According to the assumption in Theorem 3.1, we have that

$$\begin{aligned}\left\|x_k - X^k\right\|_2^2 &= \left\|\frac{A_{i,:}^T}{\left\|A_{i,:}\right\|_2^2}(b_i - z_{k-1}^i - A_{i,:}x^*)\right\|_2^2 \\ &\le \frac{1}{\left\|A_{i,:}\right\|_2^2}\left\|b_{R(A)^\perp}^i - z_{k-1}^i\right\|_2^2.\end{aligned}$$

Therefore, it holds that

$$E\left\|x_k - X^k\right\|_2^2 = E\left[E_{k-1}\left\|x_k - X^k\right\|_2^2\right] \le \frac{\sigma_{max}^2(A)}{\|A\|_F^2}\|x^*\|_2^2,$$

and

$$\begin{aligned}\left\|X^k - x^*\right\|_2^2 &= \left\|(I - \frac{A_{(i,:)}^T A_{(i,:)}}{\left\|A_{(i,:)}\right\|_2^2})(x_{k-1} - x^*)\right\|_2^2 \\ &\le \|x_{k-1} - x^*\|_2^2 - \frac{\left\|A_{(i,:)}(x_{k-1} - x^*)\right\|_2^2}{\|A\|_F^2},\end{aligned}$$

$$\begin{aligned}E\left\|X^k - x^*\right\|_2^2 &= E\left[E_{k-1}\left\|X^k - x^*\right\|_2^2\right] \\ &\le E\left[\|x_{k-1} - x^*\|_2^2 - \frac{\sigma_{min(A)}^2\|x_{k-1} - x^*\|_2^2}{\|A\|_F^2}\right] \\ &= E\left[(1 - \frac{\sigma_{min(A)}^2}{\|A\|_F^2})\|x_{k-1} - x^*\|_2^2\right] \\ &= \eta E\|x_{k-1} - x^*\|_2^2.\end{aligned}$$

According to the equation

$$(X^k - x^*)^T(x_k - X^k) = \left[(I - \frac{A_{(i,:)}^T A_{(i,:)}}{\left\|A_{(i,:)}\right\|_2^2})(x_{k-1} - x^*)\right]^T\left[\frac{A_{i,:}^T}{\left\|A_{i,:}\right\|_2^2}(b_i - z_{k-1}^i - A_{i,:}x^*)\right] = 0,$$

we have

$$\|x_k - x^*\|_2^2 = \left\|x_k - X^k\right\|_2^2 + \left\|X^k - x^*\right\|_2^2.$$

Therefore,

$$\begin{aligned} E\left\|x_k - x^*\right\|_2^2 &= E\left\|x_k - X^k\right\|_2^2 + E\left\|X^k - x^*\right\|_2^2 \\ &\le \frac{\sigma_{max}^2(A)}{\|A\|_F^2}\|x^*\|_2^2 + \eta E\|x_{k-1} - x^*\|_2^2 \\ &\le \sum_{l=0}^{k-1}\eta^l \frac{\sigma_{max}^2(A)}{\|A\|_F^2}\|x^*\|_2^2 + \eta^k E\|x^*\|_2^2 \\ &\le \sum_{l=0}^{\infty}\eta^l \frac{\sigma_{max}^2(A)}{\sigma_{min}^2(A)}\|x^*\|_2^2 + \eta^k\|x^*\|_2^2 \\ &= \left[\eta^k + \frac{\kappa^2(A)}{1-\eta}\right]\|x^*\|_2^2 . \end{aligned}$$

This completes the proof.

From Corollary 3.1 we can see when $\tau = 1$ the algorithm PBREK reduces to the algorithm PREK, and the upper bound of the error in Theorem 3.1 reduces to Theorem 3.1 in [1].

4. Numerical examples

This section shows the application of Algorithm 2 compared with Algorithm 1 and the REK method [15] in solving the overdetermined inconsistent system (1.1) with a dense or sparse coefficient matrix $A \in \mathbb{R}^{m\times n}$. Different sizes of A with $m = 5000, 6000, 7000, 8000, 9000$ and $n = 500$ are considered. The random dense coefficient matrices are generated by MATLAB function randn, and the sparse matrix is collected from the SuiteSparse matrix set. All experiments in this section are repeated 50 times to obtain an average value. All calculations are performed in MATLAB R2020a on a computer with Intel Core i7 and 16GB Ram.

Define the relative error by

$$RSE = \frac{\|x_k - x^*\|_2^2}{\|x^*\|_2^2}, \tag{4.1}$$

where x^* is the least-squares solution of (1.1). When $RSE < 10^{-6}$, the iteration is stopped. We compare the CPU time and the number of iterations (IT) for the PBREK and PREK method. We also list the speed_up defined as follows:

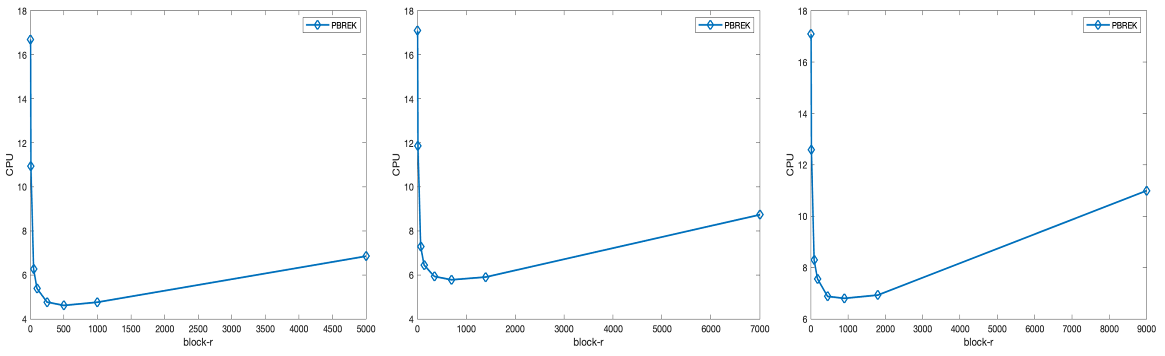
$$speed\_up = \frac{CPU\ of\ PREK}{CPU\ of\ PBREK}. \tag{4.2}$$

For all examples below, we assume that the residual vector $r = b - Ax$ is the null space of matrix A, i.e., $r \in Null(A^T)$, then $A^TAx = A^Tb + A^Tr = A^Tb$. With this assumption the solution obtained by the PBREK method will be very close to the exact solution, and the relative error will be very close to 0. Define

$$\delta = \frac{\|r\|_2}{\|Ax^*\|_2}. \tag{4.3}$$

We will consider two cases for all examples. One satisfies $\|r\|_2 = 1$ and the other is $\delta = 1$.

We will employ the rule of the partitioning strategy of blocks for the PBREK method, which is similar to that in [9]. We choose the size of row blocks as $\tau = 5, 10, 20$ to compare the quality of all method for both dense and sparse coefficient matrices. Figure 1 shows the CPU-time used to satisfy (4.1) versus τ the PBREK method with different sizes of A with $m = 5000, 7000, 9000$ and $n = 500$.



(a) 5000×500 (b) 7000×500 (c) 9000×500

Figure 1. Plot of CPU versus the number of block sizes τ for the PBREK algorithm with different sizes of coefficient matrix.

Example 4.1 In this example, we consider the overdetermined inconsistent system (1.1) with A being a random matrix. We compare the quality of the PBREK method and compare it with that of the REK, PREK methods with different partitioning strategies. Table 1 lists the IT and CPU of the PBREK algorithm compared with that of the REK and PREK methods for solving (1.1) with different sizes of A when $||r||_2 = 1$. The case when $\delta = 1$ is shown in Table 2. From Tables 1 and 2 we can see that when $||r||_2 = 1$, $\tau = 20$ is the most effective among all cases and when $\delta = 1$, $\tau = 10$ is the most effective. Thus, we take the CPU of PBREK-20 and PBREK-10 to calculate the speed_up in Tables 1 and 2, respectively. The results in Tables 1 and 2 show that the PBREK algorithm needs least CPU time and IT among all methods. Moreover, with the increase of the number of rows, the advantages of PBREK on CPU become more obvious.

Table 1. Example 4.1: Comparison of IT and CPU of the PBREK method compared with those of the REK and PREK algorithms for solving (1.1) with different sizes of A when $||r||_2 = 1$.

Method		5000×500	6000×500	7000×500	8000×500	9000×500
δ		0.0007	0.0006	0.0005	0.0005	0.0005
REK	IT	9025	9166	8585	8857	8497
	CPU	7.485	8.782	9.401	10.66	11.68
PREK	IT	8064	8024	7711	7821	7757
	CPU	6.705	7.809	8.432	9.734	10.65
PBREK-10	IT	5971	5558	5306	5166	5098
	CPU	4.211	5.415	5.381	5.714	6.326
PBREK-5	IT	5797	5649	5362	5273	5244
	CPU	4.212	5.151	5.425	6.028	6.569
PBREK-20	IT	5971	5570	5348	5128	5050
	CPU	4.2	5.276	5.297	5.669	6.303
speed_up		1.5964	1.48	1.5918	1.7171	1.690

Table 2. Example 4.1: Comparison of IT and CPU of the PBREK method compared with those of the REK and PREK algorithms for solving (1.1) with different sizes of A when $\delta = 1$.

Method		5000×500	6000×500	7000×500	8000×500	9000×500
REK	IT	9256	5621	9104	8534	8699
	CPU	7.651	8.259	9.678	10.42	11.53
PREK	IT	7884	7902	7780	7668	7631
	CPU	7.003	7.669	8.616	9.475	10.57
PBREK-10	IT	5938	5621	5214	5078	5076
	CPU	4.353	4.859	5.223	5.787	6.257
PBREK-5	IT	5918	5621	5270	5290	5111
	CPU	4.368	5.194	5.487	6.043	6.567
PBREK-20	IT	5857	5621	5156	5077	4980
	CPU	4.429	4.877	5.346	5.843	6.323
speed_up		1.6088	1.5783	1.65	1.6373	1.69

Figure 2 shows the plot of (a) log10(RSE) versus CPU and (b) log10(RSE) versus IT for the PBREK, PREK and REK methods for A with $m = 9000, n = 500$ when $\|r\|_2 = 1$. The similar case is shown in Figure 3 when $\delta = 1$. We can see from Figures 2 and 3 that REK algorithm needs the most CPU and IT, and the PBREK algorithm is faster than the PREK algorithm. The PBREK algorithm makes no obvious difference between $\tau = 20$ and $\tau = 10$. When $\tau = 1$, the PBREK algorithm is very close to the PREK algorithm. The PBREK algorithm with $\tau = 20$ converges the fastest among all methods.

We compare the PBREK algorithm with the REABK method in [3] and the RDBK method in [10] for solving the overdetermined inconsistent linear systems (1.1). We set $\tau = 20$ for all methods. Different number of rows of matrix A with 1000, 2000, 3000 is considered. We take $\tau = 20$ and $\alpha \approx 17$ for the REABK algorithm. Figure 4 shows plots of log10(RSE) versus CPU for the PBREK-20 algorithm compared with that of the PREK, RDBK and REABK methods with different sizes of A. From Figure 4 we can see the PBREK algorithm outperforms the REABK algorithm and the RDBK algorithm for different sizes of A.

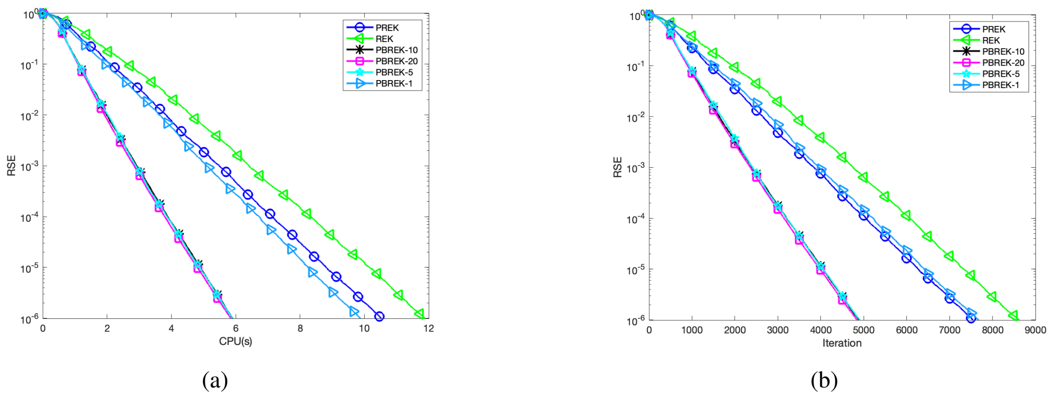
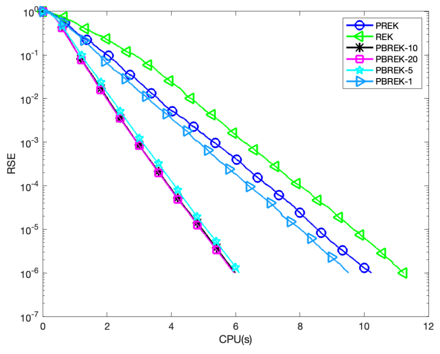
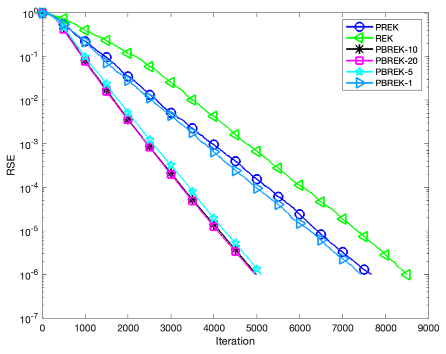


Figure 2. Example 4.1: Plots of (a) log10(RSE) versus CPU and (b) log10(RSE) versus IT for the PBREK, PREK and REK methods for A with $m = 9000, n = 500$ when $\|r\|_2 = 1$.

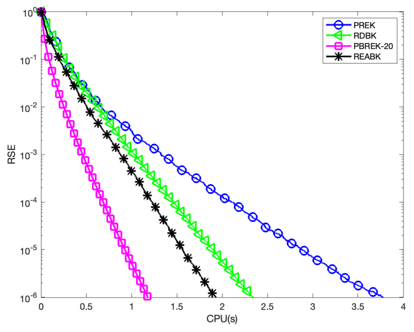


(a)

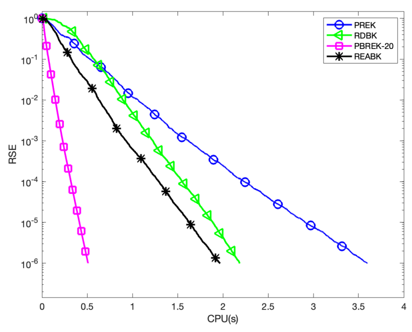


(b)

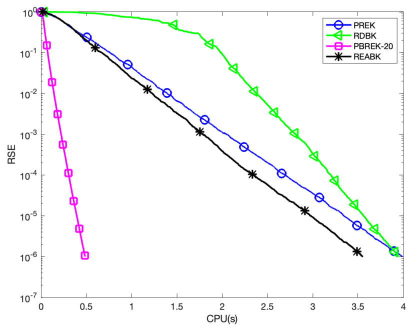
Figure 3. Example 4.1: Plots of (a) log10(RSE) versus CPU and (b) log10(RSE) versus IT for the PBREK, PREK and REK method for A with $m = 9000, n = 500$ when $\delta = 1$.



(a) 1000×500



(b) 2000×500



(c) 3000×500

Figure 4. Example 4.1: Plots of log10(RSE) versus CPU for the PREK, RDBK, REABK, PBREK-20 algorithm with different sizes of A.

Example 4.2 We discuss linear systems (1.1) with a sparse matrix A. Let's consider four sparse matrices, i.e., *ash608*, *ash958*, *divorce* and *ash219*, which comes from the SuiteSparse matrix set. We also discuss two cases when $\|r\|_2 = 1$ and $\delta = 1$, respectively. The relevant properties of four sparse matrices are shown in Table 3, where the density of a matrix is defined as follows:

$$density = \frac{number\ of\ nonzeros\ of\ an\ m-by-n\ matrix}{mn}. \tag{4.4}$$

Table 3. Example 4.2: Properties of sparse matrices.

name	ash608	ash958	divorce	ash219
$m \times n$	608×188	958×292	50×9	219×85
rank	188	292	9	85
density	1.06%	0.68%	50.0%	2.35%

Table 4 lists the IT and CPU of the PBREK algorithm compared with that of the REK and PREK methods for solving (1.1) with different sparse matrice A when $\|r\|_2 = 1$. The case when $\delta = 1$ is shown in Table 5.

Table 4. Example 4.2: Comparison of IT and CPU of the PREK and PBREK algorithms for solving (1.1) with different sizes of A when $\|r\|_2 = 1$.

Method		ash958	ash219	divorce	ash608
REK	IT	8696	2486	3743	5958
	CPU	1.154	0.1633	0.113	0.55
PREK	IT	5827	2284	2974	3604
	CPU	0.7266	0.1433	0.0768	0.2805
PBREK-10	IT	4333	1861	2632	3142
	CPU	0.5006	0.1107	0.0532	0.2203
PBREK-20	IT	4436	1567	2656	3146
	CPU	0.482	0.0714	0.0543	0.2748
PBREK-5	IT	4833	1770	2618	3158
	CPU	0.5097	0.092	0.0526	0.2306
speed_up		1.5074	2.007	1.46	1.2732

Table 5. Example 4.2: Comparison of IT and CPU of the PREK and PBREK algorithms for solving (1.1) with different sizes of A when $\delta = 1$.

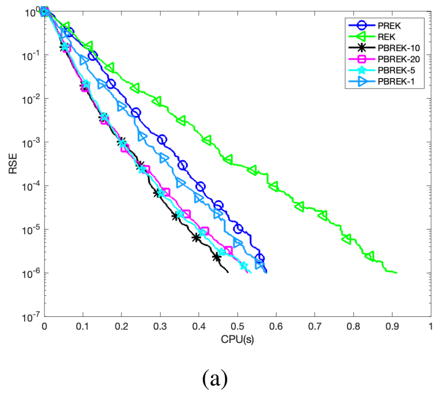
Method		ash958	ash219	divorce	ash608
REK	IT	7721	2167	2546	5056
	CPU	1.09	0.1178	0.0832	0.4931
PREK	IT	5183	1646	2487	3751
	CPU	0.6884	0.0829	0.0677	0.339
PBREK-10	IT	4272	1436	1885	3604
	CPU	0.4697	0.044	0.0404	0.2562
PBREK-20	IT	4598	1643	2100	3445
	CPU	0.5033	0.0516	0.0467	0.2777
PBREK-5	IT	4434	1525	1955	3226
	CPU	0.4911	0.0474	0.0424	0.2696
speed_up		1.4656	1.884	1.6757	1.3231

From Tables 4 and 5 we have the similar results as that in Tables 1 and 2, respectively.

Figures 5 and 6 display the plots of RSE versus CPU and RSE versus IT for the PREK, REK, PBREK algorithms on *ash958* with $\delta = 1$ and $\|r\|_2 = 1$, respectively. We take $\tau = 1, 5, 10, 20$ as an example. When $\tau = 1$, the PBREK algorithm is very close to the PREK algorithm. Altogether for both Figures 5 and 6, the PBREK algorithm converges faster than the PREK method, while there are slight difference for different partitioning strategies with $\tau = 5, 10, 20$.

5. Conclusions

In this paper, we present the PBREK algorithm for an overdetermined inconsistent linear system of equations. The convergence is provided. Several examples with random coefficient matrix and sparse coefficient matrices are given to illustrate the efficiency of the proposed PBREK algorithm.



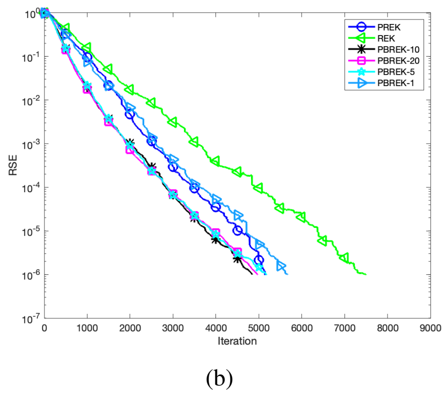
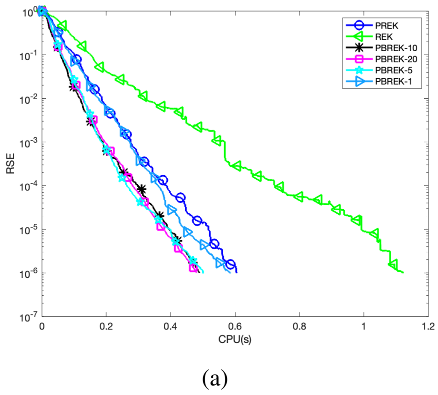


Figure 5. Example 4.2: Plot of (a) RSE versus CPU and (b) RSE versus IT for the PREK, REK, PBREK algorithms with $\delta = 1$ on *ash958*.



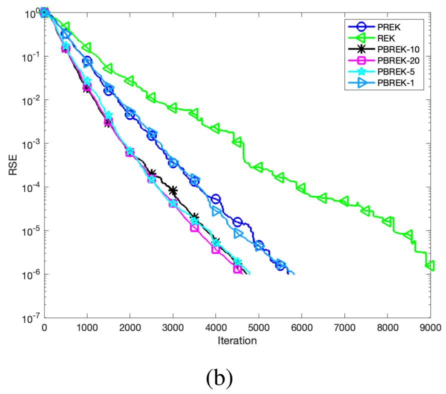


Figure 6. Example 4.2: Plot of (a) RSE versus CPU and (b) RSE versus IT for the PREK, REK, PBREK algorithms with $\|r\|_2 = 1$ on *ash958*.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was supported in part by Sichuan Science and Technology Program (grant 2022ZYD0008).

Conflict of interest

The authors declare no conflicts of interest in this paper.

References

1. Z. Z. Bai, W. T. Wu, On partially randomized extended Kaczmarz method for solving large sparse overdetermined inconsistent linear systems, *Linear Algebra Appl.*, **578** (2019), 225–250. https://doi.org/10.1016/j.laa.2019.05.005
2. B. Dumitrescu, On the relation between the randomized extended Kaczmarz algorithm and coordinate descent, *BIT Numer. Math.*, **55** (2015), 1005–1015. https://doi.org/10.1007/s10543-014-0526-9
3. 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. https://doi.org/10.1137/20M1312629
4. P. P. B. Eggermont, G. T. Herman, A. Lent, Iterative algorithms for large partitioned linear systems, with applications to image reconstruction, *Linear Algebra Appl.*, **40** (1981), 37–67. http://doi.org/10.1016/0024-3795(81)90139-7
5. T. Elfving, Block-iterative methods for consistent and inconsistent linear equations, *Numer. Math.*, **35** (1980), 1–12. http://doi.org/10.1007/BF01396365
6. S. G. Shafiei, M. Hajarian, Developing Kaczmarz method for solving Sylvester matrix equations, *J. Franklin I.*, **359** (2022), 8991–9005. https://doi.org/10.1016/j.jfranklin.2022.09.028
7. S. Kaczmarz, Angenaherte auflosung von systemen linearer glei-chungen, *Bull. Int. Acad. Pol. Sci. Lett. A*, **35** (1937), 335–357
8. J. Liu, S. J. Wright, An accelerated randomized Kaczmarz algorithm, *Math. Comput.,* **85** (2016), 153-178. https://doi.org/10.1090/mcom/2971
9. D. Needell, J. A. Tropp, Paved with good intentions: analysis of a randomized block Kaczmarz method, *Linear Algebra Appl.*, **441** (2014), 199–221. https://doi.org/10.1016/j.laa.2012.12.022
10. D. Needell, R. Zhao, A. Zouzias, Randomized block Kaczmarz method with projection for solving least squares, *Linear Algebra Appl.*, **484** (2015), 322–343. https://doi.org/10.1016/j.laa.2015.06.027
11. I. Necoara, Faster randomized block Kaczmarz algorithms, *SIAM J. Matrix Anal. Appl.*, **40** (2019), 1425–1452. https://doi.org/10.1137/19M1251643
12. S. Petra, C. Popa, Single projection Kaczmarz extended algorithms, *Numer. Algorithms*, **73** (2016), 791–806. https://doi.org/10.1007/s11075-016-0118-7
13. T. Strohmer, R.Vershynin, A randomized Kaczmarz algorithm with exponential convergence, *J. Fourier Anal. Appl.*, **15** (2009), 262–278. https://doi.org/10.1007/s00041-008-9030-4
14. N. C. Wu, H. Xiang, Projected randomized Kaczmarz methods, *J. Comput. Appl. Math.*, **372** (2020), 112672. https://doi.org/10.1016/j.cam.2019.112672
15. A. Zouzias, N. M. Freris, Randomized extended Kaczmarz for solving least squares, *SIAM J. Matrix Anal. Appl.*, **34** (2013), 773–793. https://doi.org/10.1137/120889897