Networks and Heterogeneous Media

https://www.aimspress.com/journal/nhm

NHM, 19(4): 1448–1469.
DOI: 10.3934/nhm.2024062
Received: 05 September 2024
Revised: 27 November 2024
Accepted: 11 December 2024
Published: 18 December 2024

Research article

On randomized multiple row-action methods for linear feasibility problems

Hui Song, Wendi Bao*, **Lili Xing and Weiguo Li**

College of Science, China University of Petroleum, Qingdao 266580, China

* **Correspondence:** Email: baowendi@sina.com, baowd@upc.edu.cn.

Abstract: In this paper, for solving linear feasibility problems we propose two randomized methods: a multiple row-action method (RMR) based on partial rows of residual vectors and its generalized method (GRMR) with history information in updating the current update. By introducing a linear combination of the information from the previous and subsequent iterative steps with the relaxation parameter ξ, the GRMR method unifies various RMR-type algorithms. A thorough convergence analysis for the proposed methods is provided. The theoretical results show the theoretical convergence rate of the GRMR method with $0 \le \xi \le 1$ is always worse or equal compared to that of the RMR method. Therefore, a global linear rate for the GRMR method is explored for $-1 \le \xi \le 0$. Finally, numerical experiments on both randomly generated and real-world data show our algorithms outperform the original methods in terms of computing time and iteration counts. In particular, when the appropriate parameters are selected, the GRMR method is the competitive row-action method for solving linear feasibility problems.

Keywords: linear feasibility; Kaczmarz method; multiple row-action method; acceleration; convergence

1. Introduction

Consider solving the large-scale linear inequalities

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

where $A \in \mathbb{R}^{m\times n}$ is an m-by-n ($m > n$) real coefficient matrix, $b \in \mathbb{R}^m$ is an m-dimensional right-hand side, and $x \in \mathbb{R}^n$ is an unknown n-dimensional vector. We confine the scope of this work to the regime of $m \gg n$, where iterative methods are typically employed. We denote the feasible region of Eq (1.1) by $S = \{x \in \mathbb{R}^n | Ax \le b\}$. Through this paper, we assume that the coefficient matrix A has no zero rows and $S \neq \emptyset$.

The Kaczmarz method [1], or the algebraic reconstruction technique (ART) [2], is one of the most popular solvers to solve linear systems of equations. Originally proposed by Polish mathematician

Stefan Kaczmarz in 1937, it has found a wide range of applications in many fields such as image reconstruction, medical scanners, computerized tomography, and digital signal processing. At each iteration, the Kaczmarz method uses a cyclic rule to select a row of the matrix and projects the current iteration onto the corresponding hyperplane. Let $A_{i,:}$ stand for the i-th row of the coefficient matrix A and $b = (b_1, b_2, \cdots, b_m)^T$, hence, for the algebraic reconstruction technique (Kaczmarz)

$$x^k = x^{k-1} + \frac{b_i - \langle (A_{i,:})^T, x^{k-1} \rangle}{\|A_{i,:}\|_2^2}(A_{i,:})^T, \quad k = 1, 2, \cdots, \tag{1.2}$$

where $i = (k \bmod m)+1$, $\langle \cdot, \cdot \rangle$ is the Euclidean inner product, and $\| \cdot \|_2$ is the corresponding norm in $\mathbb{R}^n$. A lot of applications have demonstrated that random row selection in the coefficient matrix A can significantly enhance its convergence rate compared to sequential selection. Strohmer and Vershyn [3] proposed the randomized Kaczmarz (RK) method by selecting the working row with a probability proportional to its Euclidean norm and proved its expected linear convergence rate in 2009. Then, Bai and Wu in [4] discussed an improved estimate of the convergence rate of the randomized Kaczmarz method. Subsequently, based on the RK method, many iterative methods have been proposed to further improve its convergence rate and computational efficiency. For instance, many variants of the RK method with different selection rules for working rows or various generalizations of the RK method are explored (see [5–8] for more details). Especially, the study on the block Kaczmarz algorithm is deepening. The block Kaczmarz method was first proposed by Elfving [9]. Unlike the Kaczmarz single-projection method, the block Kaczmarz method is equivalent to solving multiple equations at each iteration. Needell and Tropp [10] proposed the random block Kaczmarz (RBK) method. For the randomized (block) Kaczmarz method, it is necessary to traverse all rows of the coefficient matrix. There is a large amount of data in the coefficient matrix, which leads to a huge amount of computation and storage. To avoid computing the pseudoinverses, Gower and Richtárik [11] proposed the Gaussian Kaczmarz (GK) method. The Gaussian Kaczmarz (GK) method, defined by

$$x^{k+1} = x^k - \frac{\eta^T(Ax^k - b)}{\|A^T\eta\|_2^2}A^T\eta, \tag{1.3}$$

can be regarded as another kind of block Kaczmarz method that writes directly the increment in the form of a linear combination of all columns of A^T at each iteration, where η is a Gaussian vector with mean $0 \in \mathbb{R}^m$ and the covariance matrix $I \in \mathbb{R}^{m\times m}$, i.e., $\eta \sim N(0, I)$. Here I denotes the identity matrix. In Eq (1.3), all columns of A^T are used. The expected linear convergence rate was analyzed in [11] in the case that A is of full column rank. Recently, Chen and Huang [12] proposed a fast deterministic block Kaczmarz (FDBK) method, in which a set U_k is first computed according to the greedy index selection strategy [13] and then the vector η_k is constructed by

$$\eta_k = \sum_{i\in U_k}(b_i - A_{i,:}x^k)\mu_i. \tag{1.4}$$

Its relaxed version was given by [14]. In these methods [11–14], the calculations on the pseudoinverses are not needed, while, to compute U_k, one has to scan the residual vector from scratch during each iteration. In fact, Eq (1.3) is viewed as a special case of the following prototype projection iteration (for more details, see Section 5.1.2 in [15]),

$$x^{k+1} = x^k + V(W^TAV)^+W^T(b - Ax^k), \tag{1.5}$$

for $k = 0, 1, 2, \cdots$, where V and W are two parameter matrices, which is proposed by Saad based on Petrov-Galerkin (PG) conditions. It is easy to see that with different V and W, one can obtain different popular iterations as special cases, including the multiple row-action iterate scheme. For example, let $W = \eta$ and $V = A^T W$ with $\eta \in \mathbb{R}^m$ being any non-zero vector, the iteration step (1.5) becomes the form (1.3).

For solving the linear feasibility problem (1.1), Leventhal and Lewis [16] extended the randomized Kaczmarz method. At each iteration k, if the inequality is already satisfied for the selected row i, then set $x^{k+1} = x^k$. If the inequality is not satisfied, the previous iterate only projects onto the solution hyperplane $\{x| \langle A_{i,:}, x\rangle = b_i\}$. The update rule for this algorithm is thus

$$x^k = x^{k-1} - \frac{(\langle (A_{i,:})^T, x^{k-1}\rangle - b_i)^+}{\|A_{i,:}\|_2^2}(A_{i,:})^T, \quad k = 1, 2, \cdots . \tag{1.6}$$

One can see that x^{k+1} in Eq (1.6) is indeed the projection of x^k onto the set $\{x| \langle A_{i,:}, x\rangle \le b_i\}$. Leventhal and Lewis [16] (Theorem 4.3) proved that a randomized projection (RP) method converges to a feasible solution linearly in expectation. Recently, by combining the ideas of Kaczmarz and Motzkin methods [17, 18], Loera et al. proposed the sampling Kaczmarz-Motzkin (SKM) method for solving the linear feasibility problem (1.1) in [19]. Later, Morshed et al. [20] developed a generalized framework, namely the generalized sampling Kaczmarz-Motzkin (GSKM) method, that extends the SKM algorithm and proves the existence of a family of SKM-type methods. In addition, they also proposed a Nesterov-type acceleration scheme in the SKM method called probably accelerated sampling Kaczmarz-Motzkin (PASKM), which provides a bridge between Nesterov-type acceleration of machine learning and sampling Kaczmarz methods for solving linear feasibility problems.

In this paper, inspired by [13, 20], we develop a randomized multiple row-action (RMR) method for the linear feasibility problem (1.1). Partial rows indexed by U_k are instead of that of [13] to reduce the calculation on the residual vector in the RMR method, Moreover, by using history information in updating the current update, we establish a generalized version of randomized multiple row-action (GRMR) method. This general framework will provide an ideal platform for researchers to experiment with a wide range of iterative projection methods and design efficient algorithms for solving optimization problems in areas such as artificial intelligence, machine learning, etc. We emphasize that our algorithms are pseudoinverse-free and therefore different from projection-based block algorithms. We prove the linear convergence of our algorithms in the mean-square sense. Numerical results are presented to illustrate the efficiency of our algorithms.

The rest of the paper is organized as follows. In Section 2, the notations and preliminaries are provided. In Section 3, we introduce the RMR method and analyze its convergence properties. Experimental results on both randomly generated and real-world data are reported and discussed in Section 4. Finally, we present some conclusions in Section 5.

2. Preliminaries and notations

Throughout the paper, for any random variables ξ and ζ, we use $\mathbb{E}[\xi]$ and $\mathbb{E}[\xi|\zeta]$ to denote the expectation of ξ and the conditional expectation of ξ given ζ. For an integer $m \ge 1$, let $[m] := \{1, \cdots, m\}$. For any real matrix A, we use a_i, a_j, $a_{i,j}$, A^T, $A^\dagger$, $\|A\|_2$, $\|A\|_F$ and $Range(A)$ to denote the i-th row, the j-th column, the (i, j)-th entry, the transpose, the *Moore-Penrose* pseudoinverse, the spectral norm, the

Frobenius norm, and the column space of A, respectively. The nonzero singular values of a matrix A are $\sigma_{max}(A) = \sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_r(A) := \sigma_{min}(A) > 0$, where r is the rank of A, and we use $\sigma_{max}(A)$ and $\sigma_{min}(A)$ to denote the biggest and smallest nonzero singular values of A. We see that $\|A\|_2 = \sigma_1(A)$ and $\|A\|_F = \sqrt{\sum_{i=1}^{r} \sigma_i^2(A)}$. Matrix A with m rows and n columns belongs to $\mathbb{R}^{m\times n}$, the corresponding compact singular value decomposition of $A \in \mathbb{R}^{m\times n}$ is denoted as $A = UDV^T$, where U and V are unitary matrices with appropriate size and D is the nonsingular and diagonal matrix with singular value on the diagonal. Let $P_S(x)$ be the projection of x onto the nonempty closed convex set S: that is, $P_S(x)$ is the vector y that is the optimal solution to $min_{z\in S} = \|x - z\|_2$. Additionally, define the distance from x to a set S by $d(x, S) = min_{z\in S}\|x - z\| = \|x - P_S(x)\|$ as denoted in [21]. For any $c \in R$, $u \in \mathbb{R}^n$, we define $c^+ = max\{0, c\}$, $u^+ = ((u_1)^+, \cdots, (u_n)^+)^T$. For an index set τ , we use $A(\tau, :)$, $A(:, \tau)$, $v(\tau)$ and $|\tau|$ to denote the row and column submatrix of A indexed by τ , the subvector of v with component indices listed in τ and the cardinality of the set τ, respectively. We use I_n to denote the n-order identity matrix and I for short if its size is without confusion. We use $e_i = I(:, i)$ to represent the ith column of the identity matrix.

Lemma 1 (Hoffman [22])**.** *Let $x \in \mathbb{R}^n$ and S be the feasible region of the linear feasibility problem* (1.1). *There exists a constant $L > 0$ such that the following inequality holds:*

$$\|x - P_S(x)\|_2^2 \leq L^2\|(Ax - b)^+\|_2^2. \tag{2.1}$$

Lemma 1 is a famous result of Hoffmann's work on systems of linear inequalities. The constant L is called the Hoffman constant. For a consistent system of equations (i.e., there exists a unique x^* such that $Ax = b$), L can be expressed in terms of the smallest singular value of matrix A, i.e.,

$$L^2 = \frac{1}{\|A^{-1}\|^2} = \frac{1}{\sigma_{min}^2(A)}.$$

Lemma 2 ([20])**.** *For any $x \in \mathbb{R}^n$ and $\bar{x} \in S$, the following identity holds:*

$$d(x, S)^2 = \|x - P_S(x)\|_2^2 \leq \|x - \bar{x}\|_2^2. \tag{2.2}$$

In the paper, for any $\phi_1, \phi_2 \geq 0$, the following parameters are defined:

$$\begin{aligned} \phi &= \frac{-\phi_1 + \sqrt{\phi_1^2 + 4\phi_2}}{2}, \ \rho = \phi + \phi_1, \\ R_1 &= \frac{1+\phi}{\phi+\rho}, \ \ R_2 = \frac{1-\rho}{\phi+\rho}, \ \ R_3 = \frac{\phi_2+\rho}{\phi+\rho}, \ \ R_4 = \frac{\phi-\phi_2}{\phi+\rho}. \end{aligned} \tag{2.3}$$

Lemma 3 ([20])**.** *Let $\{G^k\}$ be a non-negative real sequence satisfying the following relation:*

$$G_{k+1} \leq \phi_1 G_k + \phi_2 G_{k-1}, \ \ \forall k \geq 1, \ \ G_1 = G_0 \geq 0,$$

if $\phi_1, \phi_2 \geq 0$ and $\phi_1 + \phi_2 < 1$, then the following bounds hold:

1. Let ϕ be the largest root of $\phi^2 + \phi_1\phi - \phi_2 = 0$, then

$$G_{k+1} \leq (1 + \phi)(\phi + \phi_1)^k G_0, \ \ \forall k \geq 1.$$

2. *Define* $\rho = \phi + \phi_1$, *then we have the following:*

$$\mathbb{E}\begin{bmatrix} G_{k+1} \\ G_k \end{bmatrix} \le \begin{cases} \begin{bmatrix} R_1\rho^{k+1} + R_2\phi^{k+1} \\ R_1\rho^k - R_2\phi^k \end{bmatrix} G_0, & k \quad even; \\ \begin{bmatrix} R_3\rho^k - R_4\phi^k \\ R_3\rho^{k-1} + R_4\phi^{k-1} \end{bmatrix} G_0, & k \quad odd, \end{cases}$$

where $0 \le \phi < 1$ *and* $0 < \rho < 1$.

Lemma 4 ([20]). *Let the real sequences* $H_k \ge 0$ *and* $F_k \ge 0$ *satisfy the following recurrence relation:*

$$\begin{bmatrix} H_{k+1} \\ F_{k+1} \end{bmatrix} \le \begin{bmatrix} \Pi_1 & \Pi_2 \\ \Pi_3 & \Pi_4 \end{bmatrix} \begin{bmatrix} H_k \\ F_k \end{bmatrix}, \tag{2.4}$$

where $\Pi_1, \Pi_2, \Pi_3, \Pi_4 \ge 0$ *such that the following relations*

$$\Pi_1\Pi_4 - \Pi_2\Pi_3 \ge 0, \quad \Pi_1 + \Pi_4 < 1 + min\{1, \Pi_1\Pi_4 - \Pi_2\Pi_3\} \tag{2.5}$$

hold. Then the sequences $\{H_k\}$ *and* $\{F_k\}$ *converge and the following result holds:*

$$\begin{bmatrix} H_{k+1} \\ F_{k+1} \end{bmatrix} \le \begin{bmatrix} \Pi_1 & \Pi_2 \\ \Pi_3 & \Pi_4 \end{bmatrix}^k \begin{bmatrix} H_1 \\ F_1 \end{bmatrix} = \begin{bmatrix} \Gamma_3\left(\Gamma_1\rho_2^k - \Gamma_2\rho_1^k\right) & \Gamma_1\Gamma_2\Gamma_3\left(\rho_1^k - \rho_2^k\right) \\ \Gamma_3\left(\rho_2^k - \rho_1^k\right) & \Gamma_3\left(\Gamma_1\rho_1^k - \Gamma_2\rho_2^k\right) \end{bmatrix} \begin{bmatrix} H_1 \\ F_1 \end{bmatrix},$$

where

$$\begin{aligned} \Gamma_1 &= \frac{\Pi_1 - \Pi_4 + \sqrt{(\Pi_1 - \Pi_4)^2 + 4\Pi_2\Pi_3}}{2\Pi_3}, \\ \Gamma_2 &= \frac{\Pi_1 - \Pi_4 - \sqrt{(\Pi_1 - \Pi_4)^2 + 4\Pi_2\Pi_3}}{2\Pi_3}, \quad \Gamma_3 = \frac{\Pi_3}{\sqrt{(\Pi_1 - \Pi_4)^2 + 4\Pi_2\Pi_3}}, \\ \rho_1 &= \frac{1}{2}\left[\Pi_1 - \Pi_4 - \sqrt{(\Pi_1 - \Pi_4)^2 + 4\Pi_2\Pi_3}\right], \\ \rho_2 &= \frac{1}{2}\left[\Pi_1 - \Pi_4 + \sqrt{(\Pi_1 - \Pi_4)^2 + 4\Pi_2\Pi_3}\right], \end{aligned} \tag{2.6}$$

and $\Pi_1, \Pi_3 \ge 0$ *and* $0 \le \rho_1 \le \rho_2 < 1$.

3. The randomized multiple row-action methods and convergence analysis

In this paper, for solving the linear feasibility problem (1.1), combined with the iteration schemes (1.3), (1.6), and the parameter $W = \eta_k$ in Eq (1.4), we propose a randomized multiple row-action (RMR) method and its variant (GRMR), respectively, described in Algorithms 1 and 2.

Before we delve into the main theorems, we give an important lemma as follows:

Lemma 5. *Assume that* $\eta_k = \sum_{i\in U_{i_k}}(A_{i,:}x - b_i)^+ e_i$ *with partition* $\{U_{i_k}\}_{i_k=1}^s$, $\beta_{max}^U := max_{j\in[s]}\{\sigma_{max}^2(A_{U_j,:})/\|A_{U_j,:}\|_F^2\}$, *and* $\beta_{min}^U := min_{j\in[s]}\{\sigma_{min}^2(A_{U_j,:})/\|A_{U_j,:}\|_F^2\}$. *Then, for any* $x \in \mathbb{R}^n$ *there exists the following relation:*

$$\mu_1\|x - P_S(x)\|_2^2 \le \mathbb{E}_k\left[\frac{|\eta_k^T(Ax - b)^+|^2}{\|A^T\eta_k\|_2^2}\right] \le \mu_2\|x - P_S(x)\|_2^2, \tag{3.1}$$

Algorithm 1: The RMR method for $Ax \leq b$

Input partition $\{U_i\}_{i=1}^{s}$, initial vector $x^0 \in \mathbb{R}^n$, $0 < \alpha < 2$, and maximum iteration number l.
for k=1, 2, $\cdots$, $l-1$
Pick $i_k \in [s]$ with probability $\frac{\|A_{U_{i_k},:}\|_F^2}{\|A\|_F^2}$;
Compute $x^{k+1} = x^k - \alpha \frac{\eta_k^T (Ax^k - b)^+}{\|A^T \eta_k\|_2^2} A^T \eta_k$, where $\eta_k = \sum_{i \in U_{i_k}} (A_{i,:}x^k - b_i)^+ e_i$;
End for
Output x^l.

where $0 < \mu_1 = \frac{1}{\beta_{max}^U \|A\|_F^2 L^2} \leq \mu_2 = min\{1, \frac{\sigma_{max}^2(A)}{\beta_{min}^U \|A\|_F^2}\} \leq 1.$

Proof. From the definition of η and $A_{U_{i_k},:}x - b_{U_{i_k}} \leq A_{U_{i_k},:}(x - P_S(x))$, there results in

$$\begin{aligned}
\frac{|\eta_k^T (Ax-b)^+|^2}{\|A^T \eta_k\|_2^2} &= \frac{|(\sum_{i \in U_{i_k}} (A_{i,:}x - b_i)^+ e_i)^T (Ax-b)^+|^2}{\|A^T \eta_k\|_2^2} \\
&= \frac{|(I_{:,U_{i_k}} (A_{U_{i_k},:}x - b_{U_{i_k}})^+)^T (Ax-b)^+|^2}{\|A^T \eta_k\|_2^2} \\
&= \frac{\|((A_{U_{i_k},:}x - b_{U_{i_k}})^+)^T (A_{U_{i_k},:}x - b_{U_{i_k}})^+\|_2^2}{\|A^T \eta_k\|_2^2} = \frac{\|((A_{U_{i_k},:}x - b_{U_{i_k}})^+)^T (A_{U_{i_k},:}x - b_{U_{i_k}})\|_2^2}{\|A^T \eta_k\|_2^2} \\
&\leq \frac{\|((A_{U_{i_k},:}x - b_{U_{i_k}})^+)^T A_{U_{i_k},:}(x - P_S(x))\|_2^2}{\|A^T \eta_k\|_2^2} = \frac{\|(A^T I_{:,U_{i_k}} (A_{U_{i_k},:}x - b_{U_{i_k}})^+)^T (x - P_S(x))\|_2^2}{\|A^T \eta_k\|_2^2} \\
&\leq \frac{\|A^T \eta_k\|_2^2 \|(x - P_S(x))\|_2^2}{\|A^T \eta_k\|_2^2} \leq \|x - P_S(x)\|_2^2.
\end{aligned} \tag{3.2}$$

Since $(A_{U_{i_k},:}x - b_{U_{i_k}})^+ \in R(A)$, the inequality comes from $\|A_{U_{i_k},:}^T (A_{U_{i_k},:}x - b_{U_{i_k}})^+\|_2^2 \geq \sigma_{min}^2(A_{U_{i_k},:}) \|(A_{U_{i_k},:}x - b_{U_{i_k}})^+\|_2^2$. Thus, we have the following relation,

$$\begin{aligned}
\frac{|\eta^T (Ax-b)^+|^2}{\|A^T \eta\|_2^2} &= \frac{|(\sum_{i \in U_{i_k}} (A_{i,:}x - b_i)^+ e_i)^T (Ax-b)^+|^2}{\|A_{U_i,:}^T (A_{U_{i_k},:}x - b_{U_{i_k}})^+\|_2^2} \\
&= \frac{\|(A_{U_{i_k},:}x - b_{U_i})^+\|_2^4}{\|A_{U_{i_k},:}^T (A_{U_{i_k},:}x - b_{U_{i_k}})^+\|_2^2} \leq \frac{\|(A_{U_{i_k},:}x - b_{U_{i_k}})^+\|_2^2}{\sigma_{min}^2(A_{U_{i_k},:})}.
\end{aligned} \tag{3.3}$$

Now, taking the expectation conditional on the first k iterations on both sides of Eq (3.3) , we have

$$\begin{aligned}
\mathbb{E}_k\left[\frac{|\eta_k^T(Ax-b)^+|^2}{\|A^T\eta_k\|_2^2}\right] &= \mathbb{E}_k\left[\|A_{U_{i_k},:}\|_F^2\frac{|\eta_k^T(Ax-b)^+|^2}{\|A^T\eta_k\|_2^2}\frac{1}{\|A_{U_{i_k},:}\|_F^2}\right]\\
&\le \mathbb{E}_k\left[\|A_{U_{i_k},:}\|_F^2\frac{\|(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2}{\sigma_{min}^2(A_{U_{i_k},:})}\frac{1}{\|A_{U_{i_k},:}\|_F^2}\right]\\
&\le \frac{1}{\beta_{min}^U}\mathbb{E}_k\left[\frac{\|(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2}{\|A_{U_{i_k},:}\|_F^2}\right]\\
&= \frac{1}{\beta_{min}^U}\sum_{i_k=1}^{s}\frac{\|A_{U_{i_k},:}\|_F^2}{\|A\|_F^2}\frac{\|(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2}{\|A_{U_{i_k},:}\|_F^2} = \frac{\|(Ax-b)^+\|_2^2}{\beta_{min}^U\|A\|_F^2}\\
&\le \frac{\|(Ax-AP_S(x))^+\|_2^2}{\beta_{min}^U\|A\|_F^2} \le \frac{\|Ax-AP_S(x)\|_2^2}{\beta_{min}^U\|A\|_F^2}\\
&\le \frac{\sigma_{max}^2(A)}{\beta_{min}^U\|A\|_F^2}\|x-P_S(x)\|_2^2,
\end{aligned}\tag{3.4}$$

where the third inequality comes from $AP_S(x) \le b$ and the last equality follows from the fact that $\|Ax\|_2^2 \le \sigma_{max}^2(A)\|x\|_2^2$.

Meanwhile, from Eq (3.2), it is easy to see that the following relation exists,

$$\mathbb{E}_k\left[\frac{|\eta_k^T(Ax-b)^+|^2}{\|A^T\eta_k\|_2^2}\right] \le \|x-P_S(x)\|_2^2. \tag{3.5}$$

Therefore, with the use of Eqs (3.4) and (3.5), there holds that

$$\mathbb{E}_k\left[\frac{|\eta_k^T(Ax-b)^+|^2}{\|A^T\eta_k\|_2^2}\right] \le \mu_2\|x-P_S(x)\|_2^2. \tag{3.6}$$

Similarly, we have

$$\frac{|\eta_k^T(Ax-b)^+|^2}{\|A^T\eta_k\|_2^2} = \frac{|(\sum_{i\in U_{i_k}}(A_{i,:}x-b_i)^+e_i)^T(Ax-b)^+|^2}{\|A_{U_{i_k},:}^T(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2} \ge \frac{\|(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2}{\sigma_{max}^2(A_{U_{i_k},:})}.$$

Then, taking the conditional expectation on the above inequalities, we obtain

$$\begin{aligned}
&\mathbb{E}_k\left[\frac{|\eta_k^T(Ax-b)^+|^2}{\|A^T\eta_k\|_2^2}\right] \ge \mathbb{E}_k\left[\frac{\|(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2}{\sigma_{max}^2(A_{U_{i_k},:})}\right]\\
&= \mathbb{E}_k\left[\|A_{U_{i_k},:}\|_F^2\frac{\|(A_{U_{i_k},:}x-b_{U_{i_k}})^+\|_2^2}{\sigma_{max}^2(A_{U_{i_k},:})}\frac{1}{\|A_{U_{i_k},:}\|_F^2}\right]\\
&\ge \frac{1}{\beta_{max}^U}\sum_{i_k=1}^{s}\frac{\|A_{U_{i_k},:}\|_F^2}{\|A\|_F^2}\frac{\|(A_{U_{i_k},:}x-b_{U_i})^+\|_2^2}{\|A_{U_{i_k},:}\|_F^2}\\
&= \frac{\|(Ax-b)^+\|_2^2}{\beta_{max}^U\|A\|_F^2} \ge \frac{\|x-P_S(x)\|_2^2}{\beta_{max}^U\|A\|_F^2L^2} = \mu_1\|x-P_S(x)\|_2^2,
\end{aligned}\tag{3.7}$$

where the last inequality is obtained by Lemma 1.

Hence, from Eqs (3.6) and (3.7), the conclusion is obtained.

Using the above lemma, we have the following theorem that provides the convergence of the RMR method.

Theorem 1. *Assume that the linear feasibility problem* (1.1) *is consistent, and the stepsize is* $0 < \alpha < 2$. *Then the iteration sequence* $\{x^k\}$ *generated by the RMR method for arbitrary* $x^0 \in \mathbb{R}^n$ *satisfies*

$$\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] \leq (h(\alpha))^k \|x^0 - P_S(x^0)\|_2^2,$$

where $h(\alpha) := (1 - \frac{2\alpha-\alpha^2}{L^2\beta_{max}^U\|A\|_F^2}) < 1$ *with* $\beta_{max}^U := max_{j\in[s]}\{\sigma_{max}^2(A_{U_j,:})/\|A_{U_j,:}\|_F^2\}$.

Proof. From direct calculation results, there results in

$$\begin{aligned}\langle A^T\eta_k, P_S(x^k) - x^k\rangle &= \eta_k^T(AP_S(x^k) - Ax^k) \leq \eta_k^T(b - Ax^k) \\ &\leq ((A_{U_{i_k},:}x - b_{U_{i_k}})^+)^T((b_{U_{i_k}} - A_{U_{i_k},:}x)^+) = \eta_k^T(b - Ax^k)^+.\end{aligned} \tag{3.8}$$

Then, there follows that

$$\begin{aligned}&\|x^{k+1} - P_S(x^{k+1})\|_2^2 \leq \|x^{k+1} - P_S(x^k)\|_2^2 \\ &= \|x^k - P_S(x^k) - \alpha\frac{\eta_k^T(Ax^k - b)^+}{\|A^T\eta_k\|_2^2}A^T\eta_k\|_2^2 \\ &= \|x^k - P_S(x^k)\|_2^2 + \alpha^2\frac{|\eta_k^T(Ax^k - b)^+|^2}{\|A^T\eta_k\|_2^2} - 2\alpha\frac{\eta_k^T(Ax^k - b)^+}{\|A^T\eta_k\|_2^2}\langle A^T\eta_k, x^k - P_S(x^k)\rangle \\ &\overset{(3.8)}{\leq} \|x^k - P_S(x^k)\|_2^2 - (2\alpha - \alpha^2)\frac{|\eta_k^T(Ax^k - b)^+|^2}{\|A^T\eta_k\|_2^2}.\end{aligned} \tag{3.9}$$

Taking the conditional expectation on the first k iterations and using Lemma 5, we have

$$\mathbb{E}_k[\|x^{k+1} - P_S(x^{k+1})\|_2^2] \overset{Lemma\,5}{\leq} \left(1 - \frac{2\alpha - \alpha^2}{L^2\beta_{max}^U\|A\|_F^2}\right)\|x^k - P_S(x^k)\|_2^2. \tag{3.10}$$

By the law of total expectation, there holds that

$$\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] \leq \left(1 - \frac{2\alpha - \alpha^2}{L^2\beta_{max}^U\|A\|_F^2}\right)\mathbb{E}[\|x^k - P_S(x^k)\|_2^2].$$

Finally, unrolling the recurrence gives the desired result, i.e.

$$\begin{aligned}\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] &\leq (1 - \frac{2\alpha - \alpha^2}{L^2\beta_{max}^U\|A\|_F^2})^k\|x^0 - P_S(x^0)\|_2^2 \\ &= (h(\alpha))^k\|x^0 - P_S(x^0)\|_2^2.\end{aligned}$$

Remark 1. *It can be seen from Theorem 1 that the rate of the RMR algorithm is given by* $h(\alpha) = (1 - \frac{2\alpha-\alpha^2}{L^2\beta_{max}^U\|A\|_F^2})$, *and it reaches the minimum value* $h(\alpha) = (1 - \frac{1}{L^2\beta_{max}^U\|A\|_F^2})$ *when* $\alpha = 1$.

Next, to improve the RMR method, we propose a generalized version of the RMR method (GRMR) in which the history information is used. Here, we take two iterates, x^{k-1} and x^k, generated by the successive iteration in the RMR method, and update the next iterate, x^{k+1}, as an affine combination of the previous two updates, i.e., starting with $x^0 = x^1, z^0 = z^1 \in \mathbb{R}^n$,

$$x^{k+1} = (1-\xi)z^k + \xi z^{k-1}, \text{for } k \geq 1,$$

where $z^k = x^k - \alpha \frac{\eta_k^T(Ax^k-b)^+}{\|A^T\eta_k\|_2^2} A^T\eta_k$ is the k-th update of the GRMR algorithm, which is described in Algorithm 2. It is easy to see that when $\xi = 0$, the GRMR algorithm reduces to the original RMR algorithm.

Algorithm 2: The GRMR method for $Ax \leq b$

1: **Input** partition $\{U_i\}_{i=1}^s$, initial vector $x^1 = x^0, z^1 = z^0 \in \mathbb{R}^n$, $0 < \alpha < 2$, $\xi \in Q$ and maximum iteration number l.
2: **for** k=1, 2, $\cdots$, $l-1$
3: Pick $i_k \in [s]$ with probability $\frac{\left\|A_{U_{i_k},:}\right\|_F^2}{\|A\|_F^2}$;
4: Compute $z^k = x^k - \alpha \frac{\eta_k^T(Ax^k-b)^+}{\|A^T\eta_k\|_2^2} A^T\eta_k$, where $\eta_k = \sum_{i\in U_{i_k}}(A_{i,:}x^k - b_i)^+ e_i$;
5: Set $x^{k+1} = (1-\xi)z^k + \xi z^{k-1}$;
6: **End for**
7: **Output** x^l.

Before the convergence analysis of the proposed GRMR method is explored, the following sets are first defined. For any $\xi \in \mathbb{R}$, let us denote the sets Q, Q_1, and Q_2 as

$$\begin{aligned} &Q_1 = \{\xi | 0 \leq \xi \leq 1\}, \quad Q = Q_1 \cup Q_2, \\ &Q_2 = \left\{-1 < \xi \leq 0 \mid (1+\xi)\sqrt{h(\alpha)} - \xi(1+\alpha\sqrt{\mu_2}) < 1\right\}. \end{aligned} \tag{3.11}$$

Theorem 2. *Suppose that the linear feasibility problem* (1.1) *is consistent. For arbitrary* $x^1 = x^0 \in \mathbb{R}^n$, *the sequence of iterates* $\{x^k\}$ *by the GRMR method converges with* $0 < \alpha < 2$ *and* $0 \leq \xi \leq 1$ $(\xi \in Q_1)$. *The following results hold:*

1. *Take* $\rho = \phi_1 + \phi, \phi = \frac{-\phi_1 + \sqrt{\phi_1^2 + 4\phi_2}}{2}$, *then*
$$\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] \leq \rho^k(1+\phi)\|x^0 - P_S(x^0)\|_2^2.$$
2. *Take* R_1, R_2, R_3, *and* R_4 *as in Eq* (2.3), *then*
$$\mathbb{E}\begin{bmatrix} \|x^{k+1} - P_S(x^{k+1})\|_2^2 \\ \|x^k - P_S(x^k)\|_2^2 \end{bmatrix} \leq \begin{cases} \begin{bmatrix} R_1\rho^{k+1} + R_2\phi^{k+1} \\ R_1\rho^k - R_2\phi^k \end{bmatrix} \|x^0 - P_S(x^0)\|_2^2, & k \quad even; \\ \begin{bmatrix} R_3\rho^k - R_4\phi^k \\ R_3\rho^{k-1} + R_4\phi^{k-1} \end{bmatrix} \|x^0 - P_S(x^0)\|_2^2, & k \quad odd, \end{cases}$$
where $0 \leq \phi, \ \phi_1, \ \phi_2 < 1$, $0 < \rho = \phi_1 + \phi < 1$, $h(\alpha) := (1 - \frac{2\alpha - \alpha^2}{L^2\beta_{max}^U\|A\|_F^2}) < 1$ *with* $\beta_{max}^U :=$ $max_{j\in[s]}\{\sigma_{max}^2(A_{U_j,:})/\|A_{U_j,:}\|_F^2\}$.

Proof. Since for any $\xi \in Q_1$, $(1-\xi)P_S(x^k) + \xi P_S(x^{k-1}) \in S$, straightforward calculations, we have

$$
\begin{aligned}
&\|x^{k+1} - P_S(x^{k+1})\|_2^2 \overset{Lemma\ 2}{\leq} \|x^{k+1} - (1-\xi)P_S(x^k) - \xi P_S(x^{k-1})\|_2^2 \\
=&\|(1-\xi)z^k + \xi z^{k-1} - (1-\xi))P_S(x^k) - \xi P_S(x^{k-1})\|_2^2 \\
=&\|(1-\xi)[x^k - P_S(x^k) - \alpha \frac{\eta_k^T (Ax^k - b)^+}{\|A^T \eta_k\|_2^2} A^T \eta_k] \\
&+ \xi[x^{k-1} - P_S(x^{k-1}) - \alpha \frac{\eta_{k-1}^T (Ax^{k-1} - b)^+}{\|A^T \eta_{k-1}\|_2^2} A^T \eta_{k-1}]\|_2^2 \\
\leq&(1-\xi)\|x^k - P_S(x^k) - \alpha \frac{\eta_k^T (Ax^k - b)^+}{\|A^T \eta_k\|_2^2} A^T \eta_k\|_2^2 \\
&+ \xi\|x^{k-1} - P_S(x^{k-1}) - \alpha \frac{\eta_{k-1}^T (Ax^{k-1} - b)^+}{\|A^T \eta_{k-1}\|_2^2} A^T \eta_{k-1}\|_2^2.
\end{aligned}
\tag{3.12}
$$

By taking the conditional expectation on Eq (3.12) with respect to index k, k-1 we have

$$
\begin{aligned}
&\mathbb{E}_{k,k-1}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] \\
&\leq (1-\xi)\mathbb{E}_k[\|x^k - P_S(x^k) - \alpha \frac{\eta_k^T (Ax^k - b)^+}{\|A^T \eta_k\|_2^2} A^T \eta_k\|_2^2] \\
&+ \xi \mathbb{E}_{k-1}[\|x^{k-1} - P_S(x^{k-1}) - \alpha \frac{\eta_{k-1}^T (Ax^{k-1} - b)^+}{\|A^T \eta_{k-1}\|_2^2} A^T \eta_{k-1}\|_2^2].
\end{aligned}
\tag{3.13}
$$

Meanwhile, with the use of Eqs (3.9) and (3.10), there holds that

$$
\begin{aligned}
&\mathbb{E}_k[\|x^k - P_S(x^k) - \alpha \frac{\eta_k^T (Ax^k - b)^+}{\|A^T \eta_k\|_2^2} A^T \eta_k\|_2^2] \\
&\leq \left(1 - \frac{2\alpha - \alpha^2}{L^2 \beta_{max}^U \|A\|_F^2}\right)\|x^k - P_S(x^k)\|_2^2 = h(\alpha)\|x^k - P_S(x^k)\|_2^2,
\end{aligned}
\tag{3.14}
$$

and

$$
\begin{aligned}
&\mathbb{E}_{k-1}[\|x^{k-1} - P_S(x^{k-1}) - \alpha \frac{\eta_{k-1}^T (Ax^{k-1} - b)^+}{\|A^T \eta_{k-1}\|_2^2} A^T \eta_{k-1}\|_2^2 \\
&\leq (1 - \frac{2\alpha - \alpha^2}{L^2 \beta_{max}^U \|A\|_F^2})\|x^{k-1} - P_S(x^{k-1})\|_2^2 = h(\alpha)\|x^{k-1} - P_S(x^{k-1})\|_2^2.
\end{aligned}
\tag{3.15}
$$

Taking the total expectation on Eq (3.13) and combining Eqs (3.14) and (3.15), we can get

$$
\begin{aligned}
&\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] \\
&\leq (1-\xi)h(\alpha)\mathbb{E}[\|x^k - P_S(x^k)\|_2^2] + \xi h(\alpha)\mathbb{E}[\|x^{k-1} - P_S(x^{k-1})\|_2^2] \\
&= \phi_1 \mathbb{E}[\|x^k - P_S(x^k)\|_2^2] + \phi_2 \mathbb{E}[\|x^{k-1} - P_S(x^{k-1})\|_2^2],
\end{aligned}
\tag{3.16}
$$

which satisfies the condition of Lemma 3 with $\phi_1 = (1-\xi)h(\alpha)$ and $\phi_2 = \xi h(\alpha)$. Therefore, using the first part of Lemma 3, we have

$$
\begin{aligned}
\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|_2^2] &\leq (\phi + \phi_1)^k (1+\phi)\|x^0 - P_S(x^0)\|_2^2 \\
&= \rho^k (1+\phi)\|x^0 - P_S(x^0)\|_2^2.
\end{aligned}
$$

Furthermore, using the second part of Lemma 3 and Eq (3.16), we can get the second part of Theorem 2.

Remark 2. *When* $0 \le \xi \le 1$*, we obtain a global linear rate for the GRMR method:*

$$\rho = \phi + \phi_1 = \frac{(1-\xi)h(\alpha) + \sqrt{(1-\xi)^2 h^2(\alpha) + 4\xi h(\alpha)}}{2}.$$

Since $0 < h(\alpha) < 1$ *and* $0 \le \xi \le 1$*, we can obtain that* ρ *is greater than or equal to* $h(\alpha)$*. Therefore, the theoretical convergence rate of the GRMR method with* $0 \le \xi \le 1$ *is always worse or equal compared to that of the RMR method.*

In the next theorem, we will investigate a global linear rate for the GRMR method with $\xi \in Q_2$.

Theorem 3. *Assume that the linear feasibility problem* (1.1) *is consistent. Let* $\{x^k\}$ *and* $\{z^k\}$ *be the sequence of random iterates generated by GRMR with* $0 < \alpha < 2$ *and* $\xi \in Q_2$*. Define*

$$\Pi_1 = \sqrt{h(\alpha)},\ \Pi_2 = |\xi|,\ \Pi_3 = \alpha\sqrt{\mu_2 h(\alpha)},\ \Pi_4 = |\xi|(1 + \alpha\sqrt{\mu_2}), \tag{3.17}$$

and $\Gamma_1, \Gamma_2, \Gamma_3, \rho_1, \rho_2$ *as in Eq* (2.6) *with the parameter choice of Eq* (3.17)*. Then, the following results hold,*

$$\mathbb{E}\begin{bmatrix}\left\|x^{k+1} - P_S(x^{k+1})\right\| \\ \left\|z^{k+1} - z^k\right\|\end{bmatrix} \le \begin{bmatrix}\Gamma_1\Gamma_3\rho_2^k - \Gamma_2\Gamma_3\rho_1^k \\ \Gamma_3\rho_2^k - \Gamma_3\rho_1^k\end{bmatrix} \|x^0 - P_S(x^0)\|,$$

where $\Gamma_1, \Gamma_3 \ge 0$ *and* $0 \le \rho_1 \le \rho_2 < 1$.

Proof. From Step 5 of the GRMR method and Eq (3.14), there follows that

$$\begin{aligned}
&\mathbb{E}_{k+1,k}[\|x^{k+1} - P_S(x^{k+1})\|] \le \mathbb{E}_k[\|x^{k+1} - P_S(x^k)\|] \\
&\overset{Step5}{=} \mathbb{E}_k[\|z^k - P_S(x^k) - \xi(z^k - z^{k-1})\|] \\
&\le \mathbb{E}_k[\|z^k - P_S(x^k)\|] + |\xi|\mathbb{E}_k[\|z^k - z^{k-1}\|] \\
&\le \{\mathbb{E}_k[\|z^k - P_S(x^k)\|^2]\}^{\frac{1}{2}} + |\xi|\|z^k - z^{k-1}\| \\
&\overset{(3.14)}{\le} \sqrt{h(\alpha)}\|x^k - P_S(x^k)\| + |\xi|\|z^k - z^{k-1}\|.
\end{aligned} \tag{3.18}$$

Taking the total expectation on Eq (3.18), we have

$$\mathbb{E}[\|x^{k+1} - P_S(x^{k+1})\|] \le \sqrt{h(\alpha)}\mathbb{E}[\|x^k - P_S(x^k)\|] + |\xi|\mathbb{E}[\|z^k - z^{k-1}\|]. \tag{3.19}$$

Then using the update formula for z^{k+1} in Step 5 of the GRMR method and Lemma 5, we have

$$\begin{aligned}
&\mathbb{E}_{k+1,k}[\|z^{k+1} - z^k\|] = \mathbb{E}_{k+1,k}[\|x^{k+1} - \alpha\frac{\eta_{k+1}^T(Ax^{k+1} - b)^+}{\|A^T\eta_{k+1}\|_2^2}A^T\eta_{k+1} - z^k\|] \\
&\overset{Step5}{=} \mathbb{E}_{k+1,k}[\| - \xi(z^k - z^{k-1}) - \alpha\frac{\eta_{k+1}^T(Ax^{k+1} - b)^+}{\|A^T\eta_{k+1}\|_2^2}A^T\eta_{k+1}\|] \\
&\le |\xi|\,\|z^k - z^{k-1}\| + \alpha\mathbb{E}_{k,k+1}\left[\left\|\frac{\eta_{k+1}^T(Ax^{k+1} - b)^+}{\|A^T\eta_{k+1}\|_2^2}A^T\eta_{k+1}\right\|\right] \\
&\overset{Lemma\ 5}{\le} |\xi|\,\|z^k - z^{k-1}\| + \alpha\sqrt{\mu_2}\mathbb{E}_k[\|x^{k+1} - P_S(x^{k+1})\|].
\end{aligned} \tag{3.20}$$

Taking the total expectation on Eq (3.20) and using Eq (3.19), we have

$$\begin{aligned}\mathbb{E}[\|z^{k+1}-z^k\|] &\overset{(3.20)}{\leq} |\xi|\,\mathbb{E}[\|z^k-z^{k-1}\|]+\alpha\sqrt{\mu_2}\mathbb{E}[\|x^{k+1}-P_S(x^{k+1})\|] \\ &\overset{(3.19)}{\leq} |\xi|\,(1+\alpha\sqrt{\mu_2})\mathbb{E}[\|z^k-z^{k-1}\|]+\alpha\sqrt{\mu_2 h(\alpha)}\mathbb{E}[\|x^k-P_S(x^k)\|].\end{aligned} \tag{3.21}$$

Combining Eqs (3.19) and (3.21), we can deduce the following inequality:

$$\mathbb{E}\begin{bmatrix}\|x^{k+1}-P_S(x^{k+1})\| \\ \|z^{k+1}-z^k\|\end{bmatrix} \leq \begin{bmatrix}\sqrt{h(\alpha)} & |\xi| \\ \alpha\sqrt{\mu_2 h(\alpha)} & |\xi|(1+\alpha\sqrt{\mu_2})\end{bmatrix}\mathbb{E}\begin{bmatrix}\|x^k-P_S(x^k)\| \\ \|z^k-z^{k-1}\|\end{bmatrix}. \tag{3.22}$$

From the definition, we check that $\Pi_1,\Pi_2,\Pi_3,\Pi_4\geq 0$. Since $\xi\in Q_2$, we have

$$\Pi_2\Pi_3-\Pi_1\Pi_4=|\xi|\alpha\sqrt{\mu_2 h(\alpha)}-|\xi|\sqrt{h(\alpha)}-|\xi|\alpha\sqrt{\mu_2 h(\alpha)}=-|\xi|\sqrt{h(\alpha)}\leq 0. \tag{3.23}$$

We have

$$\Pi_1+\Pi_4-\Pi_1\Pi_4+\Pi_2\Pi_3=\sqrt{h(\alpha)}+|\xi|(1+\alpha\sqrt{\mu_2})-|\xi|\sqrt{h(\alpha)}\overset{(3.11)}{<}1. \tag{3.24}$$

From the above formula (3.24), there holds that $\Pi_1+\Pi_4<1+|\xi|\sqrt{h(\alpha)}=1+min\{1,|\xi|\sqrt{h(\alpha)}\}=1+min\{1,\Pi_1\Pi_4-\Pi_2\Pi_3\}$. With the use of Eq (3.23), there results in $\Pi_2\Pi_3-\Pi_1\Pi_4\leq 0$, which is precisely the condition provided in Eq (2.5).

Let the sequences $F^k=\mathbb{E}[\|z^k-z^{k-1}\|]$ and $H^k=\mathbb{E}[\|x^k-P_S(x^k)\|]$. Then, by using Lemma 4, we have

$$\begin{bmatrix}H^{k+1} \\ F^{k+1}\end{bmatrix} \leq \begin{bmatrix}\Gamma_3(\Gamma_1\rho_2^k-\Gamma_2\rho_1^k) & \Gamma_1\Gamma_2\Gamma_3(\rho_1^k-\rho_2^k) \\ \Gamma_3(\rho_2^k-\rho_1^k) & \Gamma_3(\Gamma_1\rho_1^k-\Gamma_2\rho_2^k)\end{bmatrix}\begin{bmatrix}H^1 \\ F^1\end{bmatrix}. \tag{3.25}$$

where $\Gamma_1,\Gamma_2,\Gamma_3,\rho_1,\rho_2$ can be derived from Eq (2.6) using the parameter choice of Eq (3.17).

Since $x^1=x^0$ and $z^1=z^0$, there follows that $F_1=\mathbb{E}[\|z^1-z^0\|]=0$ and $H_1=\mathbb{E}[\|x^1-P_S(x^1)\|]=H_0$. Thus, the formula (3.25) become into the following form:

$$\begin{bmatrix}H^{k+1} \\ F^{k+1}\end{bmatrix}=\begin{bmatrix}\|x^{k+1}-P_S(x^{k+1})\| \\ \|z^{k+1}-z^k\|\end{bmatrix}\leq\begin{bmatrix}\Gamma_1\Gamma_3\rho_2^k-\Gamma_2\Gamma_3\rho_1^k \\ \Gamma_3\rho_2^k-\Gamma_3\rho_1^k\end{bmatrix}\|x^0-P_S(x^0)\|. \tag{3.26}$$

From Lemma 4, we have $\Gamma_1,\Gamma_3\geq 0$ and $0\leq\rho_1\leq\rho_2<1$, which proves the conclusion.

4. Numerical experiments

In this section, we discuss the numerical experiments performed to show the computational efficiency of the proposed algorithms (Algorithms 1 and 2). As mentioned before, we limit our focus on the over-determined systems regime (i.e., $m\gg n$), where iterative methods are competitive in general. We present some numerical examples, both synthetic and real-world data, to demonstrate the convergence of the RMR and GRMR methods.

We suppose that the subset $\{U_i\}_{i=1}^s$ is computed by

$$\{U_i\}=\begin{cases}\{(i-1)\tau+1,(i-1)\tau+2,\cdots,i\tau\}, & i\in[s-1], \\ \{(s-1)\tau+1,(s-1)\tau+2,\cdots,m\}, & i\in s,\end{cases}$$

where $\tau = 20$ is the size of U_i. All experiments are carried out using MATLAB (version R2021b) on a laptop with a 2.50-GHz intel Core i9-12900H processor, 16 GB memory, and Windows 11 operating system.

In testing synthetic data and the SuiteSparse Matrix Collection, the stopping criterion is

$$RSE = \|(Ax - b)^+\|_2 \leq 10^{-6},$$

or the maximum iteration steps of 300,000 being reached. Besides, we use the symbol " − " to indicate the case that either the corresponding iteration method can not reach the stopping criterion $RSE \leq 10^{-6}$ within 300,000 iteration steps or the computing time exceeds 1800 seconds.

In testing the sparse Netlib LP data, we set the stopping criterion to be

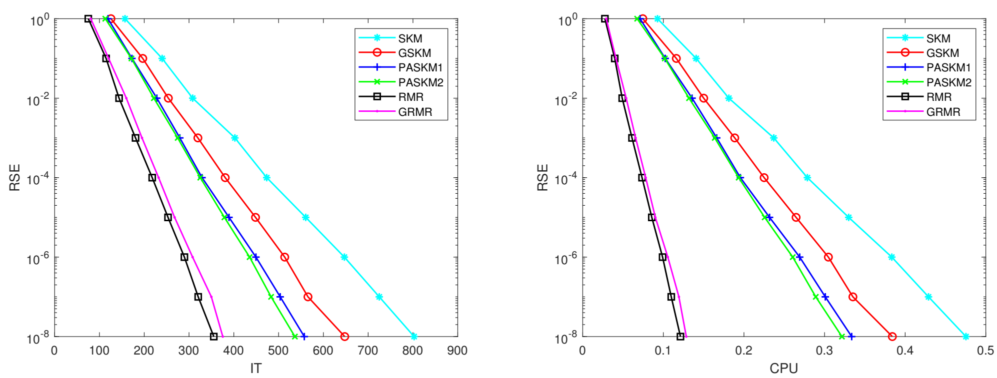
$$\frac{max(Ax^k - b)}{max(Ax^0 - b)} \leq \gamma,$$

where γ is the tolerance gap.

In this section, IT and CPU denote the number of iteration steps and computing times (in seconds), respectively. IT and CPU are the medians of the required iteration steps and the elapsed CPU times for 20 times repeated runs of the corresponding method. The SKM, GSKM, and PASKM algorithms involve the selection of many parameters as well, and we have selected a set of parameters with better performance based on the literature [20]. To ensure that the system (1.1) is consistent, we randomly generate vectors $y^1 \in \mathbb{R}^n$, $y^2 \in \mathbb{R}^n$ and set the right-hand side as $b = 0.5Ay^1 + 0.5Ay^2$. Both y^1 and y^2 are generated randomly by the MATLAB function "**randn**".

4.1. *Experiments on synthetic data*

Example 1. For the coefficient matrix A, we mainly consider two types, namely dense and sparse matrices, respectively. We randomly generate the dense matrix by the MATLAB function "**randn**". The sparse matrix is generated randomly by the MATLAB function "**sprandn**" with a density of $\frac{1}{2logmn}$ for the nonzero elements. We compared RMR and GRMR with SKM, GSKM, PASKM1, and PASKM2 with the initial vector $x^0, z^0 \in \mathbb{R}^n$ (x^0, z^0 generated randomly by the MATLAB function "**randn**") .



(a) IT (b) CPU

Figure 1. The convergence behaviors of RSE versus IT and CPU given by six methods with sparse coefficient matrix $A \in 5000 \times 50, \beta = 100, \delta = 0.5, \xi = 0.2, \alpha = 1$ for Example 1.

From Tables 1–4, we list IT and CPU of SKM, GSKM, PASKM1, PASKM2, RMR, and GRMR methods for the consistent linear feasible problem $Ax \leq b$ in Example 1. We set $\xi = -0.2, \alpha = 1.05$ in tables. The performance of these algorithms was tested on both dense and sparse coefficient matrices in two sets of experiments presented in Tables 1 and 2 with a constant number of rows but an increasing number of columns. Tables 3 and 4 show the performance of the algorithms with different orders of coefficient matrices. The convergence rates of the different methods are observed for consistent systems, as depicted in Figure 1. From Tables 1–4 , it can be observed that the RMR and GRMR methods outperform the SKM, GSKM, PASKM1, and PASKM2 methods for consistent systems. Furthermore, Figure 1 demonstrates that compared with other methods, the RMR and GRMR approaches achieve higher accuracy with fewer iterations (IT) and computational time (CPU).

Table 1. IT and CPU of six methods for $m \times n$ dense matrices A with $m = 5000$ and different n in Example 1.

$m \times n$		5000×100	5000×200	5000×300	5000×400	5000×500	5000×600
SKM	IT	1067	2982	5614	9727	15220	21832
	CPU	0.9855	4.7603	9.1254	17.5679	33.0315	71.7751
GSKM	IT	851	2252	4294	7268	11276	16374
	CPU	0.7875	3.7005	7.3854	13.6847	23.5921	58.1655
PASKM1	IT	750	1819	3340	5707	8850	12717
	CPU	0.6957	3.0986	5.5658	10.3246	16.8315	35.4983
PASKM2	IT	700	1474	2310	3362	4.5721e+03	6.2811e+03
	CPU	0.6484	2.5027	3.7529	5.7056	8.7151	17.7476
RMR	IT	412	868	1297	2132	2871	3505
	CPU	0.0523	0.1780	0.3607	0.7130	1.2153	2.3454
GRMR	IT	**399**	**849**	**1234**	**1777**	**2482**	**3275**
	CPU	**0.0503**	**0.1680**	**0.3227**	**0.6910**	**1.1115**	**2.1774**

Table 2. IT and CPU of six methods for $m \times n$ sparse matrices A with $m = 5000$ and different n in Example 1.

$m \times n$		5000×100	5000×200	5000×300	5000×400	5000×500	5000×600
SKM	IT	1303	3138	5724	10207	14553	21886
	CPU	1.0057	6.6166	8.2686	16.9807	26.5132	43.8631
GSKM	IT	1012	2410	4373	7606	10984	16502
	CPU	0.7751	5.3301	6.7144	12.2361	21.1343	33.7102
PASKM1	IT	877	1993	3478	5967	8693	13133
	CPU	0.6877	4.2751	5.1627	9.5969	14.9724	23.7929
PASKM2	IT	791	1567	2398	3450	4729	6550
	CPU	0.6343	3.5159	3.9884	5.5175	8.1405	12.4132
RMR	IT	459	1034	1563	2419	3129	4684
	CPU	0.0672	0.3124	0.7153	1.1694	1.6621	2.6481
GRMR	IT	**426**	**1018**	**1550**	**2394**	**3054**	**4287**
	CPU	**0.0625**	**0.3106**	**0.7091**	**1.1563**	**1.6199**	**2.4405**

Table 3. IT and CPU of six methods for $m \times n$ dense matrices A with different n and m in Example 1.

$m \times n$		1000×100	2000×200	3000×300	4000×400	5000×500	6000×600
SKM	IT	2847	6058	8876	12224	15171	18685
	CPU	1.3973	7.3557	14.5893	21.0069	32.0768	81.6442
GSKM	IT	2198	4602	6675	9347	11173	13946
	CPU	1.1239	5.6573	11.6139	16.4939	23.1984	63.4526
PASKM1	IT	1.775	3616	5143	7256	9076	10995
	CPU	0.8970	4.4592	9.0139	12.7491	16.9114	48.8534
PASKM2	IT	885	1887	2682	3729	4684	5569
	CPU	0.4631	2.3404	4.6370	6.6436	8.7244	23.4332
RMR	IT	667	1251	1978	2523	3217	3933
	CPU	0.0268	0.0608	0.1731	0.5311	1.2986	1.7975
GRMR	IT	**638**	**1226**	**1860**	**2462**	**3150**	**3740**
	CPU	**0.0255**	**0.0585**	**0.1621**	**0.5126**	**1.2877**	**1.7047**

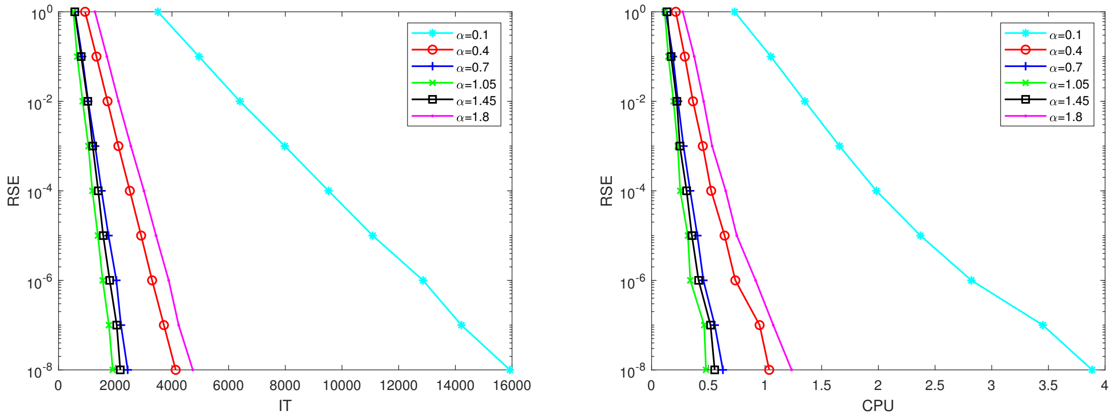
Table 4. IT and CPU of six methods for $m \times n$ sparse matrices A with different n and m in Example 1.

$m \times n$		1000×100	2000×200	3000×300	4000×400	5000×500	6000×600
SKM	IT	3143	5961	8667	11404	14620	17525
	CPU	1.1965	2.0794	14.9667	22.3923	27.1353	47.9605
GSKM	IT	2396	4452	6510	8512	10734	13104
	CPU	0.9519	1.5554	11.9794	16.7697	21.0046	37.0169
PASKM1	IT	1974	3472	5158	6765	8734	10302
	CPU	0.8684	1.2462	9.8658	12.9433	15.4741	28.1906
PASKM2	IT	1119	1959	2870	3694	4700	5546
	CPU	0.4947	0.7069	5.5380	7.1027	8.0391	14.3387
RMR	IT	675	1275	2005	2725	3129	3853
	CPU	0.0506	0.1899	0.4785	1.0729	1.6562	2.4124
GRMR	IT	**668**	**1219**	**1919**	**2575**	**3054**	**3792**
	CPU	**0.0488**	**0.1808**	**0.4536**	**1.0211**	**1.6199**	**2.3684**

In Table 5, we list IT and CPU of the RMR method for $m \times n$ dense matrices A with different α for Example 1. From Table 5, we can see that the RMR algorithm with $\alpha = 1.05$ performs better. As shown in Figure 2, the choice of parameter α affects IT and CPU required by the RMR method to achieve desired accuracy levels. When α is appropriately selected, the IT and CPU needed by the RMR method are significantly reduced.

Table 5. IT and CPU of the RMR method for $m \times n$ dense matrices A with different α in Example 1.

$m \times n$	α	0.10	0.40	0.55	0.80	1.05	1.30	1.55	1.85
5000×100	IT	5808.9	910.0	536.7	436.6	**413.3**	456.8	578.2	1500.0
	CPU	0.5981	0.1007	0.0591	0.0486	**0.0469**	0.0514	0.0650	0.1616
5000×200	IT	13620.0	2093.9	1133.8	851.2	**806.5**	879.4	1161.7	3005.1
	CPU	3.0197	0.5453	0.3023	0.2278	**0.2141**	0.2355	0.3115	0.7872
5000×300	IT	24926.4	3778.3	1962.1	1423.8	**1267.4**	1407.1	1837.5	4595.9
	CPU	13.8342	2.1100	1.1329	0.7885	**0.7346**	0.8189	1.0251	2.5884
5000×400	IT	39569.2	5934.5	2969.8	2013.3	**1726.1**	1865.3	2432.2	6223.0
	CPU	14.6740	2.4406	1.2494	0.8401	**0.7170**	0.7846	1.0214	2.6008
5000×500	IT	61697.4	9158.3	4554.6	2998.8	**2423.2**	2441.9	3094.0	7574.4
	CPU	19.6330	3.0827	1.5505	1.0319	**0.8332**	0.8344	1.0518	2.6854
5000×600	IT	83018.8	12278.2	6063.6	3931.3	**2958.2**	3066.8	3725.5	9557.4
	CPU	48.1672	7.3571	3.6987	2.4025	**1.8190**	1.9194	2.2922	5.8362



(a) RMR IT (b) RMR CPU

Figure 2. The convergence behaviors of RSE versus IT and CPU given by RMR method with dense coefficient matrix $A \in 5000 \times 300$ for Example 1.

Example 2. For given m, n, r and $\kappa > 1$, we construct a matrix A by $A = UDV$, where $U \in \mathbb{R}^{m\times r}$, $D \in \mathbb{R}^{r\times r}$ and $V \in \mathbb{R}^{n\times r}$. These matrices are generated by $[U, \sim] = qr(randn(m, r), 0)$, $[V, \sim] = qr(randn(n, r), 0)$, and $D = diag(1 + (\kappa - 1). * rand(r, 1))$.

This example gives us many flexibilities to adjust the input parameters m, n, r, and κ. We consider two types of rank-deficient cases by setting $m = 30n$, $r = n/2$, and $\kappa = n/10$ in Table 6 and $m = 5000$, $r = n/2$, and $\kappa = n/10$ in Table 7. We compared RMR and GRMR with SKM, GSKM, PASKM1, and PASKM2 with the initial vector $x^0, z^0 \in \mathbb{R}^n$ (x^0, z^0 generated randomly by the MATLAB function "**randn**") .

In Tables 6 and 7, we report the numerical results of the SKM, GSKM, PASKM1, PASKM2, RMR, and GRMR algorithms with $\beta = 100$, $\delta = 0.5$, $\xi = -0.2$, $\alpha = 1$ for two rank-deficient consistent linear systems. We can observe the following phenomena: the relative solution error of GRMR decays faster than those of SKM, GSKM, PASKM1, PASKM2, and RMR when the number of iteration steps and computing time increase.

Table 6. IT and CPU of six methods for $m \times n$ matrices A with $m = 30n$ and different n in Example 2.

$m \times n$		1500×50	3000×100	4500×150	6000×200	7500×250
SKM	IT	1.2447e+03	7.9725e+03	2.1648e+04	5.5140e+04	1.2848e+05
	CPU	0.2798	6.0742	30.2712	72.6378	177.7376
GSKM	IT	9.7291e+02	6.3474e+03	1.7195e+04	4.3673e+04	1.0183e+05
	CPU	0.2223	4.9473	25.7609	59.6574	142.0572
PASKM1	IT	8.0253e+02	5.3334e+03	1.4417e+04	3.6519e+04	8.4791e+04
	CPU	0.1833	4.2592	21.6780	48.1641	118.7923
PASKM2	IT	5.1557e+02	3.4486e+03	9.1972e+03	2.3007e+04	5.2762e+04
	CPU	0.1181	2.8350	13.2988	27.2750	73.9614
RMR	IT	2.3945e+02	1.4493e+03	7.0107e+03	1.9027e+04	3.4009e+04
	CPU	0.0124	0.0844	0.9885	3.4734	9.3129
GRMR	IT	**2.0725e+02**	**1.2183e+03**	**5.7479e+03**	**1.5770e+04**	**2.8827e+04**
	CPU	**0.0109**	**0.0784**	**0.8225**	**2.8958**	**8.2372**

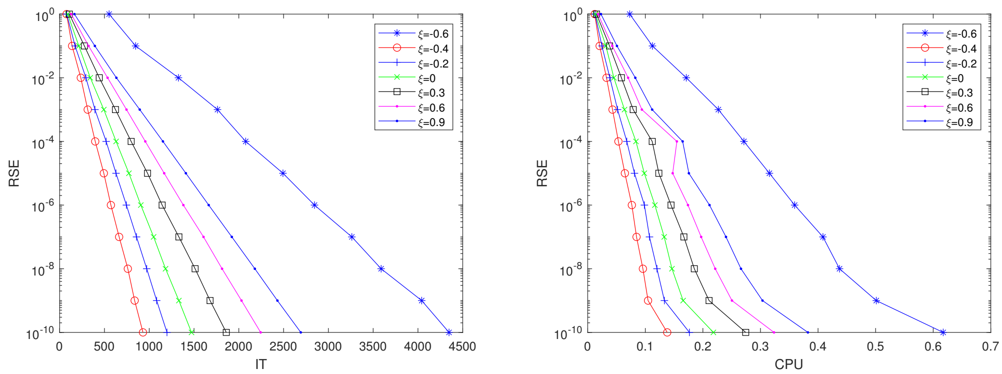
Table 7. IT and CPU of six methods for $m \times n$ matrices A with $m = 5000$ and different n in Example 2.

$m \times n$		5000×50	5000×100	5000×150	5000×200	5000×250
SKM	IT	7.5260e+02	4.9593e+03	2.5683e+04	5.8852e+04	1.06256e+05
	CPU	0.6725	7.4758	29.9510	86.5563	134.1015
GSKM	IT	6.1130e+02	3.9435e+03	2.0453e+04	4.6742e+04	9.5641e+04
	CPU	0.5448	5.9391	21.6146	65.3934	102.7754
PASKM1	IT	5.3000e+02	3.3225e+03	1.7098e+04	3.9103e+04	8.2928e+04
	CPU	0.4740	5.2256	19.0484	49.6410	87.6790
PASKM2	IT	3.6930e+02	2.1724e+03	1.0772e+04	2.4912e+04	5.8286e+04
	CPU	0.3334	3.4637	12.0785	32.7437	61.1034
RMR	IT	3.4125e+02	2.4008e+03	5.8137e+03	1.2747e+04	3.5337e+04
	CPU	0.0405	0.2881	1.0362	5.9427	11.1385
GRMR	IT	**2.905e+02**	**2.0233e+03**	**4.7020e+03**	**1.0390e+04**	**2.8997e+04**
	CPU	**0.0345**	**0.2446**	**0.8953**	**4.9289**	**9.3507**

In Table 8, we list IT and CPU of the GRMR method for $m \times n$ dense matrices A with $m = 5000$, $r = n/2$, $\kappa = n/10$, $\alpha = 0.95$, and different ξ for Example 2. We can find that the choice of $\xi = -0.4$ is the best choice for the GRMR method in Figure 3.

Table 8. IT and CPU of the GRMR method for $m \times n$ dense matrices A with $m = 5000$, $r = n/2$, $\kappa = n/10$, $\alpha = 0.95$ and different ξ in Example 2.

n	ξ	-0.6	-0.4	-0.2	0	0.6	0.9
50	IT	3.0910e+02	**2.1429e+02**	2.7607e+02	3.3015e+02	4.8100e+02	5.9819e+02
	CPU	0.0490	**0.0344**	0.0441	0.0511	0.0747	0.0933
100	IT	1.5200e+03	**1.1347e+03**	1.4166e+03	1.7213e+03	2.5678e+03	3.1229e+03
	CPU	0.9983	**0.9612**	0.9436	0.9793	0.9883	0.9909
150	IT	9.9682e+03	**4.6247e+03**	5.9130e+03	7.2372e+03	1.0993e+04	1.2839e+04
	CPU	2.9349	**1.3518**	1.8325	2.3050	3.4949	4.0744
200	IT	-	**6.4790e+03**	8376e+03	1.0158e+04	1.5754e+04	1.8845e+04
	CPU	-	**2.2413**	2.9027	3.5326	5.8995	7.1385
250	IT	-	**2.4789e+04**	3.1179e+04	3.8087e+04	5.8929e+04	6.9713e+04
	CPU	-	**3.8122**	4.8770	6.5623	9.9870	12.1248
300	IT	-	**2.6105e+04**	3.3267e+04	4.0252e+04	6.2172e+04	7.4024e+04
	CPU	-	**5.6167**	7.1635	8.6136	13.1339	15.7273



(a) IT (b) CPU

Figure 3. The convergence behaviors of RSE versus IT and CPU given by the GRMR method with dense coefficient matrix $A \in 5000 \times 100$, $\alpha = 0.95$, $\beta = 100$, $\delta = 0.5$, and different ξ for Example 2.

4.2. Experiments on real-world data

We consider the following two types of real-world test data: the SuiteSparse Matrix Collection and the sparse Netlib LP instances.

Example 3. The SuiteSparse Matrix Collection [23]. In Table 9, the coefficient matrix A is chosen from the SuiteSparse Matrix Collection. In testing the SuiteSparse Matrix Collection, we take $\tau = \lceil \|A\|_2^2 \rceil$. We compared RMR and GRMR with SKM, GSKM, PASKM1, and PASKM2 with the initial vector $x^0 = z^0 = 0 \in \mathbb{R}^n$. For details, we list their sizes, densities, condition numbers (i.e., cond(A)), and squared Euclidean norms in Table 10, where the density of a matrix is defined by

$$density = \frac{\text{the number of non-zero elements of an m-by-n matrix}}{mn}.$$

In Table 9, we list the IT and CPU of SKM, GSKM, PASKM1, PASKM2, RMR, and GRMR methods for the linear feasible problem $Ax \leq b$ in Example 3 with $\beta = 100$, $\delta = 0.3$, $\xi = -0.2$, and $\alpha = 0.95$. We observe that the GRMR method outperforms SKM, GSKM, PASKM, and RMR methods in terms of both the iteration counts and the CPU time from Table 9.

Table 9. IT and CPU of six methods for Example 3 with $\beta = 100, \delta = 0.3, \xi = -0.2, \alpha = 0.95$.

name		ash219	ash958	ch7-8-b1	well1033	illc1850	ch6-6-b5
SKM	IT	555	24383	63036	4559	6114	17113
	CPU	0.0237	0.5705	12.3610	1.0811	3.0682	39.7281
GSKM	IT	381	1697	48929	3418	4552	12993
	CPU	0.0169	0.3626	10.4723	0.8127	2.2707	28.0804
PASKM1	IT	258	1142	39865	2.650	3497	10292
	CPU	0.0111	0.2512	8.6279	0.6352	1.7700	46.4743
PASKM2	IT	39	166	20123	1549	1993	6603
	CPU	0.0018	0.0381	4.4998	0.3785	0.9553	13.3613
RMR	IT	113	379	24167	1859	2221	619
	CPU	0.0012	0.0182	0.6021	0.0455	0.1575	0.1430
GRMR	IT	**28**	**109**	**15527**	**551**	**697**	**104**
	CPU	**0.0006**	**0.0112**	**0.3188**	**0.0155**	**0.0592**	**0.0392**

Table 10. The properties of different sparse matrices in Example 3.

name	ash219	ash958	ch7-8-b1	well1033	illc1850	ch6-6-b5
$m \times n$	219×85	958×292	1176×56	1033×320	1850×712	720×4320
density	2.35%	0.68%	3.57%	1.43%	0.66%	0.14%
cond(A)	3.0249	3.2014	3.5819e+15	166.1333	1.404e+03	1
$\|\|A\|\|_2^2$	12.1422	17.9630	49.0000	3.2635	4.5086	6.0000

Example 4. Netlib LP instances. We follow the standard framework used by De Loera et al. [19] and Morshed et al. [20] in their work on linear feasibility problems. The problems are transformed from standard LP problems (i.e., min $c^T x$ subject to $Ax = b, l \leq x \leq u$ with optimum value p^*) to an equivalent linear feasibility formulation (i.e., $Ax \leq b$, where $A = [A^T - A^T I - Ic]^T$ and $b = [b^T - b^T u^T - l^T p^*]^T$). In testing the sparse Netlib LP instances, we take $\tau = 5$ and initial vectors $x^0 = z^0 = 0 \in \mathbb{R}^n$.

Table 11. CPU time comparisons of five methods for different matrices A in Example 4.

name	Dimensions	γ	ξ	α	SKM	GSKM	PASKM2	RMR	GRMR
lp_sc50b	257×78	10^{-2}	-0.1	0.9	0.1746	0.1886	0.3636	0.1059	**0.0874**
lp_adlittle	389×138	10^{-2}	0.01	0.85	0.0222	0.0258	0.1173	0.0149	**0.0076**
lp_recipe	591×204	10^{-4}	-0.2	1.05	2.6012	2.1534	3.1206	2.2878	**1.9061**
degen2	1957×534	10^{-2}	-0.1	1.05	0.0107	0.0184	0.0140	0.0073	**0.0068**

In Table 11, we list the IT and CPU of SKM, GSKM, PASKM2, RMR, and GRMR methods for the linear feasible problem $Ax \leq b$ in Example 4. From Table 11, we know that Algorithm GRMR takes less computing time compared to the other algorithms.

5. Conclusions

In this paper, based on partial rows of the residual vector, the RMR method and its general framework (GRMR) are provided to solve the linear feasibility problems. The GRMR method unifies various RMR-type algorithms and adds the relaxation parameter ξ. The convergence results are proved. Some numerical examples, including synthetic data and real-world applications, demonstrate that the two methods often outperform the original methods. Especially, the GRMR method with $\xi \in Q_2$ takes less computing time. This implies that GRMR is a variant of the competitive row-action type for solving linear feasibility problems. Meanwhile, from the numerical results, it can be seen that the appropriate choice of parameters can lead to more effective methods for different types of problems. In future work, we intend to identify the optimal choices of ξ.

Author contributions

Hui Song: Writing the original draft and deriving the convergence. Wendi Bao: Conceptualization, Methodology. Lili Xing: Visualization, Software. Weiguo Li: Review, Conceptualization.

Use of AI tools declaration

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

Acknowledgments

This article was supported by the National Natural Science Foundation of China [grant number 61931025] and the National Natural Science Foundation of China [grant number 42374156].

Conflict of interest

The authors declare there is no conflict of interest.

References

1. S. Karczmarz, Angenäherte auflösung von systemen linearer gleichungen, *Bull. Int. Acad. Polon. Sci. Lett.*, **35** (1937), 355–357.
2. G. Richard, B. Robert, H. T. Gabor, Algebraic Reconstruction Techniques (ART) for three-dimensional electron microscopy and X-ray photography, *J. Theor. Biol.*, **29** (1970), 471–481. https://doi.org/10.1016/0022-5193(70)90109-8
3. 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
4. Z. Z. Bai, W. T. Wu, On convergence rate of the randomized Kaczmarz method, *Linear Algebra Appl.*, **553** (2018), 252–269. https://doi.org/10.1016/j.laa.2018.05.009

5. Y. C. Eldar, D. Needell, Acceleration of randomized Kaczmarz method via the Johnson-Lindenstrauss lemma, *Numer. Algorithms*, **58** (2011), 163–177. https://doi.org/10.1007/s11075-011-9451-z
6. J. H. Guo, W. G. Li, The randomized Kaczmarz method with a new random selection rule, *Numer. Math. J. Chin. Univ.*, **40** (2018), 65–75.
7. J. J. Zhang, A new greedy Kaczmarz algorithm for the solution of very large linear systems, *Appl. Math. Lett.*, **91** (2019), 207–212. https://doi.org/10.1016/j.aml.2018.12.022
8. Y. Liu, C. Q. Gu, Variant of greedy randomized Kaczmarz for ridge regression, *Appl. Numer. Math.*, **143** (2019), 223–246. https://doi.org/10.1016/j.apnum.2019.04.008
9. T. Elfving, Block-iterative methods for consistent and inconsistent linear equations, *Numer. Math.*, **35** (1980), 1–12. https://doi.org/10.1007/BF01396365
10. 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
11. R. M. Gower, P. Richtárik, Randomized iterative methods for linear systems, *SIAM J. Matrix Anal. Appl.*, **36** (2015), 1660–1690. https://doi.org/10.1137/15M1025487
12. J. Q. Chen, Z. D. Huang, On a fast deterministic block Kaczmarz method for solving large-scale linear systems, *Numer. Algorithms*, **89** (2022), 1007–1029. https://doi.org/10.1007/s11075-021-01143-4
13. 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. https://doi.org/10.1137/17M1137747
14. N. C. Wu, L. X. Cui, Q. Zuo, On the relaxed greedy deterministic row and column iterative methods, *Appl. Math. Comput.*, **432** (2022), 127339. https://doi.org/10.1016/j.amc.2022.127339
15. S. Yousef, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.
16. D. Leventhal, A. S. Lewis, Randomized methods for linear constraints: Convergence rates and conditioning, *Math. Oper. Res.*, **35** (2010), 641–654. https://doi.org/10.1287/moor.1100.0456
17. S. Agmon, The relaxation method for linear inequalities, *Can. J. Math.*, **6** (1954), 382–392. https://doi.org/10.4153/CJM-1954-037-2
18. T. S. Motzkin, I. J. Schoenberg, The relaxation method for linear inequalities, *Can. J. Math.*, **6** (1954), 393–404. https://doi.org/10.4153/CJM-1954-038-x
19. J. A. De Loera, J. Haddock, D. Needell, A sampling Kaczmarz-Motzkin algorithm for linear feasibility, *SIAM J. Sci. Comput.*, **39** (2017), S66–S87. https://doi.org/10.1137/16M1073807
20. M. S. Morshed, M. S. Islam, M. Noor-E-Alam, Sampling Kaczmarz-Motzkin method for linear feasibility problems: Generalization and acceleration, *Math. Program.*, **194** (2022), 719–779. https://doi.org/10.1007/s10107-021-01649-8
21. D. Leventhal, A. S. Lewis, Randomized methods for linear constraints: Convergence rates and conditioning, *Math. Oper. Res.*, **35** (2010), 641–654. https://doi.org/10.1287/moor.1100.0456
22. A. J. Hoffman, On approximate solutions of systems of linear inequalities, *J. Res. Nat. Bur. Stand.*, **49** (1952), 263–265.

23. S. P. Kolodziej, A. Mohsen, B Matthew, D. Jarrett, A. D. Timothy, H. Matthew, et al., The SuiteSparse matrix collection website interface, *J. Open Source Software*, **4** (2019), 1244. https://doi.org/10.21105/joss.01244