Journal of Applied Mathematics and Physics, 2021, 9, 3032-3044
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

A Relaxed Greedy Block Kaczmarz Method for Solving Large Consistent Linear Systems

Yimou Liao[1], Feng Yin[1,2*], Guangxin Huang[3]

[1]College of Mathematics and Statistics, Sichuan University of Science & Engineering, Zigong, China
[2]Sichuan Province University Key Laboratory of Bridge Non-destruction Detecting and Engineering Computing, Zigong, China
[3]College of Mathematics and Physics, Chengdu University of Technology, Chengdu, China
Email: *fyinsuse@163.com, liaoyimou596@163.com, huangx@cdut.edu.cn

How to cite this paper: Liao, Y.M., Yin, F. and Huang, G.X. (2021) A Relaxed Greedy Block Kaczmarz Method for Solving Large Consistent Linear Systems. *Journal of Applied Mathematics and Physics*, **9**, 3032-3044.
https://doi.org/10.4236/jamp.2021.912196

Received: October 29, 2021
Accepted: December 12, 2021
Published: December 15, 2021

Abstract

Many problems in science and engineering require solving large consistent linear systems. This paper presents a relaxed greedy block Kaczmarz method (RGBK) and an accelerated greedy block Kaczmarz method (AGBK) for solving large-size consistent linear systems. The RGBK algorithm extends the greedy block Kaczmarz algorithm (GBK) presented by Niu and Zheng in [1] by introducing a relaxation parameter to the iteration formulation of GBK, and the AGBK algorithm uses different iterative update rules to minimize the running time. The convergence of the RGBK is proved and a method to determine an optimal parameter is provided. Several examples are presented to show the effectiveness of the proposed methods for overdetermined and underdetermined consistent linear systems with dense and sparse coefficient matrix.

Keywords

Linear Consistent Systems, Convergence Properties, Relaxed Greedy Block Kaczmarz

1. Introduction

We are concerned with the solution of the large consistent linear system

$$Ax = b, \tag{1}$$

where $A \in \mathbb{R}^{m\times n}$, and $b \in \mathbb{R}^m$. The Kaczmarz method in [2] is possible one of the most popular, simple while efficient algorithms for solving (1). It was revised to be applied to image reconstruction in [3], which is called algebraic reconstruction technique, and has a large range of fields of applications such as image reconstruction in computerized tomography [4] [5] [6] and parallel computing [7].

There are many extended Kaczmarz algorithms [8]-[16] developed to solve (1). Strohmer and Vershynin in [17] introduced a randomized Kaczmarz method (RK) for consistent overdetermined systems (1). The RK method has a convergence bound with the expected exponential convergence, which was called linear convergence. Zhang [18] proposed an improved greedy Kaczmarz (GK) method for solving (1). Bai and Wu in [19] presented a greedy randomized Kaczmarz algorithm (GRK) for (1) when the system is consistent. In each step of iteration, GRK is based on a probability criterion trying to grasp larger entries of the residual vector. Bai and Wu [20] further developed a relaxed GRK method for large sparse Equations (1). Due to its fast convergence, the block method [21] [22] [23] [24] has also been extensively developed in linear or nonlinear optimization problems. Recently, Liu and Zheng in [1] presented a greedy block Kaczmarz algorithm (GBK) with the iteration

$$x_{k+1} = x_k + \boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\left(b_{\mathcal{J}_k} - \boldsymbol{A}_{\mathcal{J}_k} x_k\right), \tag{2}$$

where the index of the selected row

$$\mathcal{J}_k = \left\{ i_k \,|\, \left|b_{i_k} - \boldsymbol{A}^{(i_k)} x_k\right|^2 \geq \varepsilon_k \left\|\boldsymbol{A}^{(i_k)}\right\|^2 \right\} \tag{3}$$

with the parameter

$$\varepsilon_k = \eta \max_{1\leq i\leq m}\left\{ \frac{\left|b_i - \boldsymbol{A}^{(i)} x_k\right|^2}{\left\|\boldsymbol{A}^{(i)}\right\|_2^2} \right\}. \tag{4}$$

$\boldsymbol{A}_{\mathcal{J}_k}$ stands for the submatrix $\boldsymbol{A}(\mathcal{J}_k,:)$ of $\boldsymbol{A}$, b_i represents the ith element of the vector b, $\boldsymbol{A}^{(i)}$ denotes the ith *row* of $\boldsymbol{A}$ and $\boldsymbol{A}^{\dagger}$ denotes the Moore-Penrose pseudoinverse of $\boldsymbol{A}$.

In this paper, based on the GBK method in [1], we develop a new relaxed greedy block Kaczmarz algorithm (RGBK) for (1). RGBK extends the GBK algorithm by introducing a relaxed parameter. The convergence of the algorithm is provided and the optimal relaxed parameter is discussed. The rest of this paper is organized as follows. In Section 2, a new relaxed greedy block Kaczmarz algorithm is presented. The convergence is provided and the optimal relaxed parameter is determined. Several types of examples are shown in Section 3, including the overdetermined or underdetermined systems with dense and sparse coefficient matrices. Some conclusions are drawn in Section 4.

At the end of this section, we introduce some mathematical symbols that will be used. For a matrix $\boldsymbol{Q} \in \mathbb{R}^{m\times n}$, $\boldsymbol{Q}^{(i)}$ denotes the ith row vector of $\boldsymbol{Q}$. $\boldsymbol{Q}_{\mathcal{J}_k}$ stands for the submatrix $\boldsymbol{Q}(\mathcal{J}_k,:)$ of $\boldsymbol{Q}$, where $\mathcal{J}_k$ is an index set composed of positive integers not exceeding m, and m is the number of rows of $\boldsymbol{Q}$. Let $\delta_{\min}(\boldsymbol{Q})$ and $\delta_{\max}(\boldsymbol{Q})$ be the maximum and minimum positive singular values of $\boldsymbol{Q}$, respectively. $\|\boldsymbol{Q}\|_2^2$ and $\|\boldsymbol{Q}\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}\left|q_{ij}\right|^2$ are the spectral norm and Frobenius norm, respectively. For any vector $p \in \mathbb{R}^m$, p_i represents the ith component of p.

2. The Relaxed Greedy Block Kaczmarz Algorithm

Replacing the left side x_{k+1} in (2) with the combination of x_k and x_{k+1} in (2) by introducing a relaxed parameter $\lambda \in (0,2)$, we have

$$x_{k+1} = (1-\lambda)x_k + \lambda\left(x_k + \boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\left(b_{\mathcal{J}_k} - \boldsymbol{A}_{\mathcal{J}_k}x_k\right)\right). \tag{5}$$

Thus

$$x_{k+1} = x_k + \lambda\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\left(b_{\mathcal{J}_k} - \boldsymbol{A}_{\mathcal{J}_k}x_k\right). \tag{6}$$

The method presented by (6) is called a relaxed greedy block Kaczmarz algorithm, which is abbreviated by RGBK. **Algorithm 1** summarizes the RGBK algorithm.

We remark that the iteration formulation (6) reduces to (2) when $\lambda = 1$, thus the GBK method in [1] is a special case of **Algorithm 1**.

The results below give the convergence of **Algorithm 1**.

Theorem 1. *Assume the linear system* (1) *is consistent. The iterative sequence* $\{x_k\}$ *generated by* **Algorithm 1** *converges to the minimum norm solution* $x^* = \boldsymbol{A}^{\dagger}b$ *of* (1). *Moreover, it holds that*

$$\left\|x_{k+1} - x^*\right\|_2^2 \le \left[1 - \phi_k(\lambda,\eta)\frac{\delta_{\min}^2(\boldsymbol{A})}{\|\boldsymbol{A}\|_F^2}\right]\cdot\left\|x_k - x^*\right\|_2^2, \quad k = 0,1,\cdots, \tag{7}$$

where $\phi_k(\lambda,\eta) = \eta\lambda(2-\lambda)\dfrac{\left\|\boldsymbol{A}_{\mathcal{J}_k}\right\|_F^2}{\sigma_{\max}^2\left(\boldsymbol{A}_{\mathcal{J}_k}\right)}$, $\|\boldsymbol{A}\|_F$ *denotes the Frobenius norm of* $\boldsymbol{A}$, $\delta_{\max}(\boldsymbol{A})$ *and* $\delta_{\min}(\boldsymbol{A})$ *are the maximum and minimum positive singular values of* $\boldsymbol{A}$, *respectively.*

Proof. Let $e_k = x_k - x^*$, where x^* satisfies $\boldsymbol{A}_{\mathcal{J}_k}x^* = b_{\mathcal{J}_k}$, then we have from (6) that

$$\begin{aligned}\left\|e_{k+1}\right\|_2^2 &= \left\|e_k + \lambda\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\left(b_{\mathcal{J}_k} - \boldsymbol{A}_{\mathcal{J}_k}x_k\right)\right\|_2^2 = \left\|\left(\boldsymbol{I} - \lambda\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\boldsymbol{A}_{\mathcal{J}_k}\right)e_k\right\|_2^2 \\ &= \left\|e_k\right\|_2^2 - \lambda(2-\lambda)\left\|\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\boldsymbol{A}_{\mathcal{J}_k}e_k\right\|_2^2.\end{aligned} \tag{8}$$

According to the definition of $\mathcal{J}_k$ at Step 3 in **Algorithm 1** and the fact that $\delta_{\min}^2\left(\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\right) = \delta_{\max}^{-2}\left(\boldsymbol{A}_{\mathcal{J}_k}\right)$, we have that

Algorithm 1. A relaxed greedy block Kaczmarz algorithm (RGBK).

Input: $\boldsymbol{A}, b, x_0$ parameter $\eta \in (0,1]$ and $\lambda \in (0,2)$.

Output: the approximation solution x_k of (1)

for $k = 0,1,\cdots$ do until termination criterion is satisfied; do

Compute ε_k by (4);

Determine the control index set $\mathcal{J}_k$ by (3);

Update x_{k+1} by (6).

end for

$$\begin{aligned}\left\|\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\boldsymbol{A}_{\mathcal{J}_k}e_k\right\|_2^2 &\geq \delta_{\min}^2\left(\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\right)\left\|\boldsymbol{A}_{\mathcal{J}_k}e_k\right\|_2^2 = \delta_{\min}^2\left(\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\right)\sum_{i_k\in\mathcal{J}_k}\left|\boldsymbol{A}^{(i_k)}e_k\right|^2 \\ &\geq \delta_{\min}^2\left(\boldsymbol{A}_{\mathcal{J}_k}^{\dagger}\right)\sum_{i_k\in\mathcal{J}_k}\varepsilon_k\left\|\boldsymbol{A}^{(i_k)}\right\|^2 = \frac{\left\|\boldsymbol{A}_{\mathcal{J}_k}\right\|_F^2}{\delta_{\max}^2\left(\boldsymbol{A}_{\mathcal{J}_k}\right)}\varepsilon_k.\end{aligned} \tag{9}$$

For $k=1,2,\cdots$, it holds that

$$\left\|b-\boldsymbol{A}x_k\right\|_2^2 = \sum_{i=1}^{m}\frac{\left|b_i-\boldsymbol{A}^{(i)}x_k\right|^2}{\left\|\boldsymbol{A}^{(i)}\right\|_2^2}\left\|\boldsymbol{A}^{(i)}\right\|_2^2 \leq \max_{1\leq i\leq m}\frac{\left|b_i-\boldsymbol{A}^{(i)}x_k\right|^2}{\left\|\boldsymbol{A}^{(i)}\right\|_2^2}\left\|\boldsymbol{A}\right\|_F^2,$$

thus the constant ε_k at step 2 of **Algorithm 1** becomes

$$\begin{aligned}\varepsilon_k &= \eta\max_{1\leq i\leq m}\frac{\left|b_i-\boldsymbol{A}^{(i)}x_k\right|^2}{\left\|\boldsymbol{A}^{(i)}\right\|_2^2} \geq \eta\frac{\left\|b-\boldsymbol{A}x_k\right\|_2^2}{\left\|\boldsymbol{A}\right\|_F^2} \\ &= \eta\frac{\left\|\boldsymbol{A}x^*-\boldsymbol{A}x_k\right\|_2^2}{\left\|\boldsymbol{A}\right\|_F^2} \geq \eta\frac{\delta_{\min}^2\left(\boldsymbol{A}\right)}{\left\|\boldsymbol{A}\right\|_F^2}\left\|e_k\right\|_2^2.\end{aligned} \tag{10}$$

Thus (8) together with (9) and (10) implies (7). This completes the proof. □

We derive easily the results below from Theorem 1.

Corollary 1. *Let* $M(\lambda,\eta)=\max_{0\leq j\leq k-1}\phi_j(\lambda,\eta)$. *Under the conditions of Theorem* 1, *we obtain an upper bound of error*

$$\left\|x_k-x^*\right\|_2^2 \leq \left[1-M(\lambda,\eta)\frac{\delta_{\min}^2\left(\boldsymbol{A}\right)}{\left\|\boldsymbol{A}\right\|_F^2}\right]^k\cdot\left\|x_0-x^*\right\|_2^2. \tag{11}$$

Proof. Using (7) iteratively for $k=1,\cdots$, we have (11) with the definition of $M(\lambda,\eta)$. □

The upper bound of error below is independent of $\mathcal{J}_k$.

Corollary 2. *Under the conditions of Theorem* 1, (7) *becomes*

$$\left\|x_{k+1}-x^*\right\|_2^2 \leq \left[1-\eta\lambda(2-\lambda)\frac{\delta_{\min}^2\left(\boldsymbol{A}\right)}{\left\|\boldsymbol{A}\right\|_F^2}\right]\cdot\left\|x_k-x^*\right\|_2^2. \tag{12}$$

Proof. We notice that

$$\left\|\boldsymbol{A}_{\mathcal{J}_k}\right\|_F^2 = \sum_{i=1}^{|\mathcal{J}_k|}\delta_i^2\left(\boldsymbol{A}_{\mathcal{J}_k}\right) \geq \delta_{\max}^2\left(\boldsymbol{A}_{\mathcal{J}_k}\right),$$

thus

$$\frac{\left\|\boldsymbol{A}_{\mathcal{J}_k}\right\|_F^2}{\delta_{\max}^2\left(\boldsymbol{A}_{\mathcal{J}_k}\right)} \geq 1. \tag{13}$$

Therefore, (12) results from (12) together with (7). □

Remark 1. *The RGBK method reduces to the GBK method in* [1] *when* $\lambda=1$. *Examples in Section* 3 *provide a way to determine the optimal relaxed parameter value of* λ *that minimizes the CPU time or the number of iteration for both overdetermined and underdetermined systems.*

Remark 2. *Taking into account the limitation of computer memory space, we use Gaussian Kaczmarz method defined in* [22] *instead of* (6), *which could avoid the calculation of* $\boldsymbol{A}^{\dagger}$,

$$x_{k+1} = x_k + \lambda \frac{\delta_k^{\mathrm{T}} \left(b - \boldsymbol{A}x_k\right)}{\left\|\boldsymbol{A}^{\mathrm{T}}\delta_k\right\|_2^2} A^{\mathrm{T}}\delta_k,$$

where $\delta_k = \sum_{i \in \mathcal{J}_k} \left(b_i - \boldsymbol{A}^{(i)} x_k\right) e_i$, this method abbreviated as AGBK.

3. Numerical Examples

In this section, we give several examples to show the efficiency of our RGBK and AGBK methods and compare them with GBK in [1]. All experiments are carried out with the MATLAB 2020b on a computer with 3.00 GHz processing unit and 16 GB RAM.

We compute the solution of the consistent system (1) with $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ computed by $b = \boldsymbol{A}x^*$, where $x^* \in \mathbb{R}^n$ denotes the exact solution generated by the MATLAB function *randn*. Denote *the relative solution residuals* $\mathrm{RSE} := \left\|x_k - x^*\right\|_2^2 \Big/ \left\|x^*\right\|_2^2$. We define different acceleration ratios as follows,

$$\text{SU-R} = \frac{\text{CPU of GBK}}{\text{CPU of RGBK}} \quad \text{and} \quad \text{SU-A} = \frac{\text{CPU of GBK}}{\text{CPU of AGBK}}.$$

To avoid calculating the Moore-Penrose inverse $\boldsymbol{A}^{\dagger}$ when implementing the update (6) in **Algorithm 1**, we use the CGLS algorithm in [21] to solve a corresponding least-squares problem. Considering the fairness of the three algorithms, all parameters involved in AGBK are the same as RGBK. We set the initial vector $x_0 = 0$ and the termination criterion satisfying RSE < 10^{-6} for GBK, RGBK and AGBK in all examples.

Example 4.1. We use this example to illustrate how to determine an optimal parameter η in **Algorithm 1** for both overdetermined and underdetermined systems (1). We use different parameter η ranged from 0.05 to 0.9 to compute the number of iteration (IT) and CPU time (CPU) by **Algorithm 1** for the consistent systems (1) with $\boldsymbol{A} \in \mathbb{R}^{2000\times 1000}$ and $\boldsymbol{A} \in \mathbb{R}^{1000\times 2000}$, which are randomly dense matrices generated by the MATLAB function *randn*, respectively, then determine an optimal parameter η that minimizes the IT or CPU.

Figure 1 shows the plot of CPU versus η. From **Figure 1**, we can choose the optimal parameter $\eta = 0.2$, which minimizes CPU by using **Algorithm 1** for the consistent systems (1) with $\boldsymbol{A} \in \mathbb{R}^{2000\times 1000}$ and $\boldsymbol{A} \in \mathbb{R}^{1000\times 2000}$.

Example 4.2. In this case, we give the same way to determine the optimal relaxation parameters λ of RGBK as shown in **Figure 2**, and strictly abide by the method of controlling variables. We default that GBK and RGBK have the same η.

Figure 2 shows the plot of IT and CPU versus relaxed parameter λ. From **Figure 2(a)** and **Figure 2(b)**, we can choose the optimal relaxed parameter $\lambda_{opt} = 1.2$ and $\lambda_{opt} = 1.3$, which minimizes CPU by using **Algorithm 1** for the consistent systems (1) with $\boldsymbol{A} \in \mathbb{R}^{2000\times 1000}$ and $\boldsymbol{A} \in \mathbb{R}^{1000\times 2000}$.

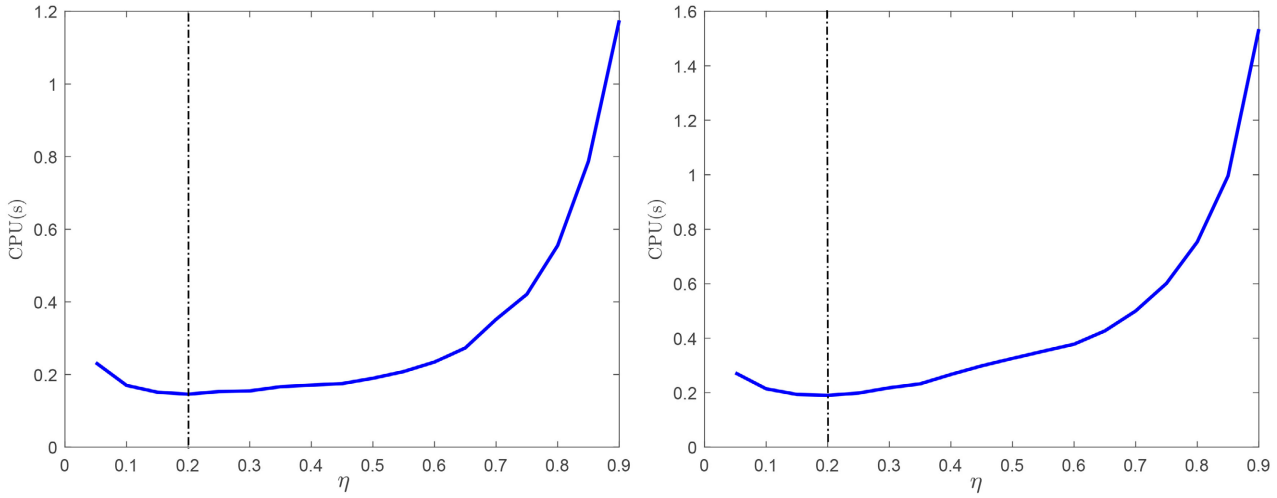

Figure 1. An optimal parameter (η) determined by the minimum of CPU for overdetermined 2000 × 1000 (left) and underdetermined 1000 × 2000 (right) systems in RGBK.

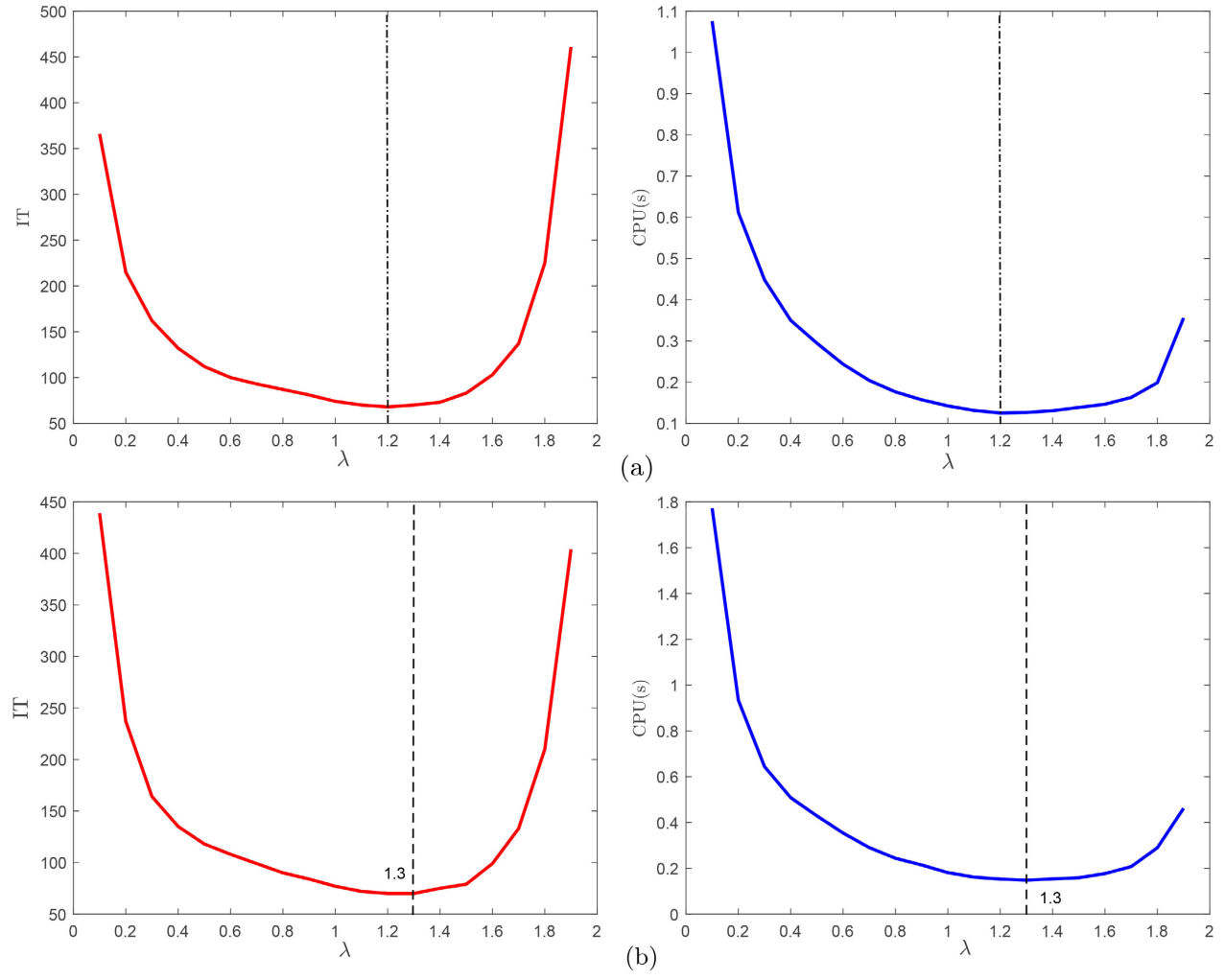

Figure 2. An optimal relaxed parameter (λ) determined by the minimum of IT (left) and CPU (right) for overdetermined and underdetermined systems. (a) IT (left) and CPU (right) versus λ with 2000 × 1000; (b) IT (left) and CPU (right) versus λ with 1000 × 2000.

Example 4.3. We compare the convergence of RGBK and AGBK with the optimal relaxed parameter λ_{opt} determined similarly in Example 4.2 with that of GBK algorithm for overdetermined and underdetermined systems (1) with $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ generated by the MATLAB function *randn*. Table 1 lists IT and CPU(s) of the RGBK and AGBK algorithms with the optimal relaxed parameter λ_{opt} compared with that of GBK algorithm for different overdetermined systems (1) with Gaussian coefficient matrices $\boldsymbol{A}$. The corresponding optimal relaxed parameter computed similarly in Example 4.2 is $\lambda_{opt} = 1.2, 1.2, 1.3, 1.2, 1.2, 1.2, 1.25$, respectively.

Table 2 shows similar results for different underdetermined Gaussian systems (1) with $\boldsymbol{A}$. From Table 1 and Table 2, we can see the advantages of our proposed RGBK and AGBK methods over GBK algorithm.

Example 4.4. We use the RGBK and AGBK methods and compare them with GBK to solve systems (1) with sparse coefficient matrix $\boldsymbol{A}$ from the Florida sparse matrix collection in [25]. Table 3 summarizes the different sparse systems and its *density* and condition number $Cond(\boldsymbol{A})$, where *the density* of $\boldsymbol{A}$ means the ratio of the number of the nonzero elements of $\boldsymbol{A}$ to the total number of the elements of $\boldsymbol{A}$.

Table 4 lists IT and CPU(s) of the RGBK and AGBK algorithms with the optimal

Table 1. Performance of GBK, RGBK, AGBK for overdetermined systems.

method	Index	$m \times n$			
		3000 × 1000	4000 × 1000	5000 × 1000	3000 × 2000
GBK(η)	IT	37 (0.2)	26 (0.2)	24 (0.25)	159 (0.15)
	CPU	0.1064	0.0904	0.0824	1.0852
RGBK(λ)	IT	34 (1.2)	24 (1.2)	23 (1.3)	149 (1.3)
	CPU	0.0884	0.0790	0.0715	0.9881
AGBK	IT	36	25	25	158
	CPU	0.0587	0.0538	0.0635	0.4778
SU-R		**1.2039**	1.1448	1.1542	1.0983
SU-A		1.8146	1.7943	**1.2983**	**2.2713**
		4000 × 2000	5000 × 2000	4000 × 3000	5000 × 3000
GBK(η)	IT	71 (0.15)	43 (0.15)	313 (0.2)	121 (0.15)
	CPU	0.6024	0.4370	3.8447	1.9698
RGBK(λ)	IT	66 (1.2)	42 (1.2)	290 (1.2)	115 (1.25)
	CPU	0.5577	0.4128	3.3780	1.7867
AGBK	IT	69	44	293	120
	CPU	0.2730	0.2149	1.7092	0.8857
SU-R		1.0801	**1.0659**	1.1380	1.1025
SU-A		2.2061	2.0474	2.2491	2.2241

Table 2. Performance of GBK, RGBK, AGBK for underdetermined systems.

name	Index	$m \times n$			
		1000 × 3000	1000 × 4000	1000 × 5000	2000 × 3000
GBK(η)	IT	35 (0.15)	23 (0.1)	20 (0.1)	163 (0.15)
	CPU	0.1488	0.1392	0.1487	1.2579
RGBK(λ)	IT	32 (1.25)	22 (1.2)	19 (1.3)	150 (1.3)
	CPU	0.1231	0.1334	0.1418	1.0974
AGBK	IT	33	22	19	163
	CPU	0.0532	0.0434	0.0457	0.4838
SU-R		1.2087	**1.0431**	1.0487	1.1463
SU-A		2.7982	3.2101	**3.2541**	2.6001
		2000 × 4000	2000 × 5000	3000 × 4000	3000 × 5000
GBK(η)	IT	65 (0.1)	49 (0.15)	299 (0.15)	142 (0.2)
	CPU	0.7464	0.6181	4.3543	2.3230
RGBK(λ)	IT	63 (1.4)	44 (1.25)	280 (1.3)	132 (1.4)
	CPU	0.7112	0.5401	3.9027	1.9067
AGBK	IT	71	45	294	145
	CPU	0.2719	0.2164	1.7181	1.0346
SU-R		1.0496	1.1443	1.1157	**1.2183**
SU-A		2.7454	2.8555	2.5344	**2.2452**

Table 3. The properties of different sparse matrices.

name	ash-958	Trefethen-700	relat6	flower-5-1	stat96v5
$m \times n$	958 × 292	700 × 700	2340 × 157	211 × 201	2307 × 75,779
Density	0.68%	2.58%	2.20%	1.42%	0.13%
Full rank	Yes	Yes	No	No	No
Cond(A)	3.20	4.71e+3	Inf	2.00e+16	1.2609e+17

Table 4. Performance of GBK, RGBK, AGBK for overdetermined systems.

name		ash958	Trefethen-700	relat6	flower-5-1	stat96v5
GBK(η)	IT	25 (0.25)	468 (0.1)	116 (0.25)	242 (0.25)	55 (0.3)
	CPU	0.0008	0.0219	0.0064	0.0036	0.1006
RGBK(λ)	IT	21 (0.9)	401 (1.2)	115 (0.9)	214 (1.2)	46 (0.8)
	CPU	0.0006	0.0191	0.0062	0.0030	0.0934
AGBK	IT	24	624	111	251	59
	CPU	0.0004	0.0134	0.0035	0.0018	0.0388
SU-R		**1.2246**	1.1417	**1.0360**	1.2123	1.0771
SU-A		2.0521	**1.6324**	1.8407	1.9953	**2.5934**

relaxed parameter λ_{opt} compared with that of GBK algorithm for different sparse systems (1) with coefficient matrices $\boldsymbol{A}$. The corresponding optimal relaxed parameter computed similarly in Example 4.3 is $\lambda_{opt} = 0.9, 1.2, 0.9, 1.2$ and 0.8, respectively. Figure 3 shows the plot of RSE versus IT (left) or RSE versus CPU (right) of Algorithm 1 applied to solve (1) with different sparse coefficient matrix $\boldsymbol{A}$ in Table 4. From Table 4, we can see that the speedup ratio SU-R can reach 1.2246 and SU-A can reach 2.5934, which shows the fast convergence of our proposed algorithm. From Figure 3, RGBK and AGBK converge much faster than GBK does on RSE versus IT (left) and RSE versus CPU (right) for sparse consistent Equation (1) with coefficient matrices $\boldsymbol{A}$ named ash958 and stat96v5 in Table 4.

Example 4.5. This example uses RGBK and AGBK to restore a computer

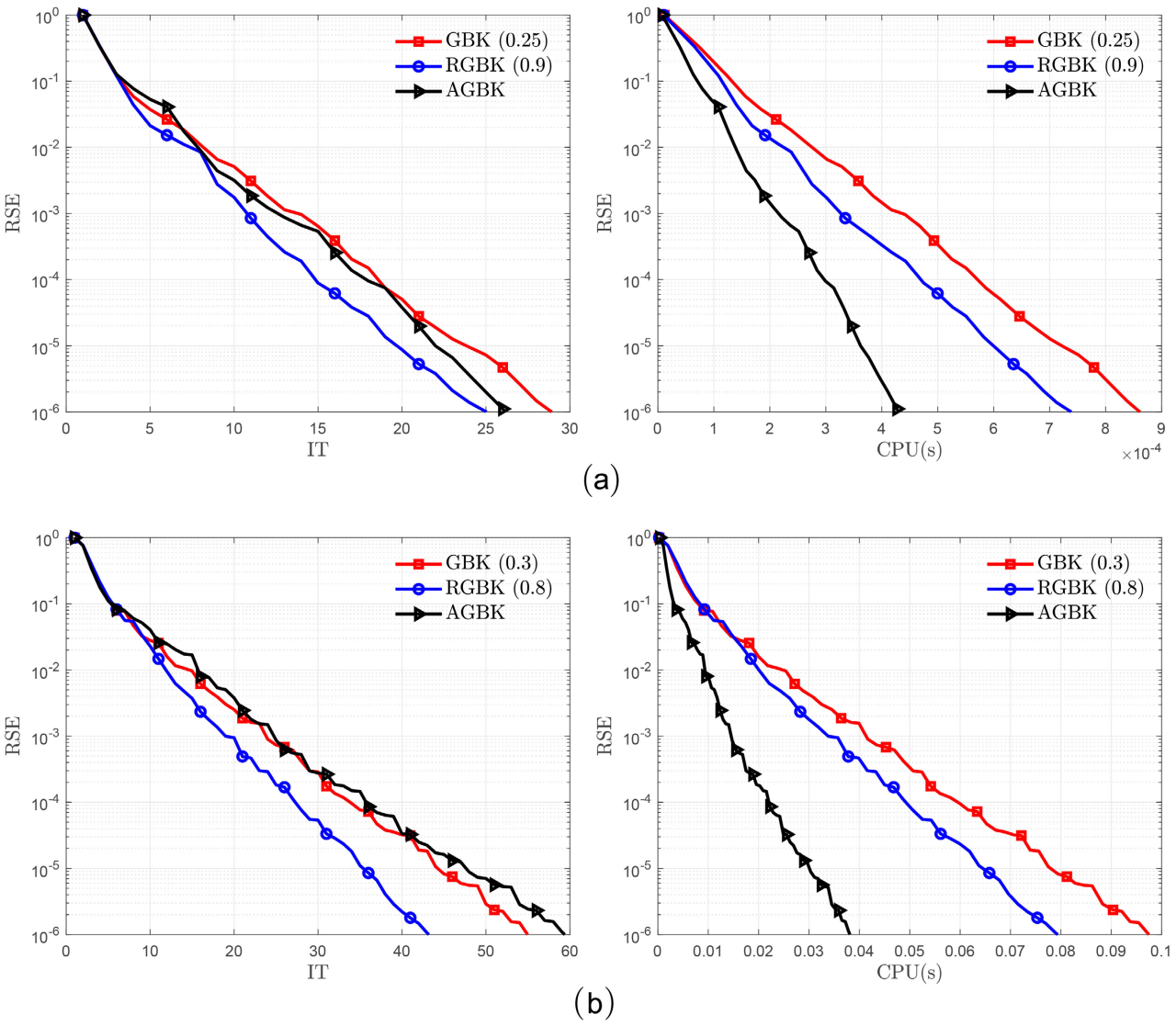

Figure 3. RSE versus IT (left) and RSE versus CPU (right) of RGBK and AGBK compared with GBK to solve consistent Equation (1) with sparse coefficient matrix $\boldsymbol{A}$ named ash958 (a) stat96v5 (b) in Table 3. (a) IT (left) and CPU (right) vesue ash958; (b) IT (left) and CPU (right) vesue stat96v5.

tomography (*CT*) image. The MATLAB function $paralleltomo(N,\theta,p)$ from Algebraic Iterative Reconstruction (ART) package in [26], which generates a large sparse matrix $\boldsymbol{A}$ and the exact solution x_* is used, where $N=70$, $\theta=0^\circ:0.7:178^\circ$ and $p=70$, then the size of $\boldsymbol{A}$ is 17,850 × 4900. We compute b by $b=\boldsymbol{A}x_*$, RGBK and AGBK are used to recover x_* from b and compared with the GBK method.

Table 5 reports the IT and CPU(s), and RSE of RGBK and AGBK compared with GBK for overdetermined consistent sparse matrix, where $\mathrm{RSE}\le 10^{-6}$. Figure 4 shows the recovered images by GBK, RGBK and AGBK together with the original image.

It can be seen from Table 5 that RGBK(1.3) and AGBK obtain lower IT and CPU(s) than GBK(0.2) for restoring *CT* images, which show that RGBK and AGBK converge faster than GBK dose if the parameter λ is selected appropriately.

Table 5. GBK, RGBK and AGBK for reconstruction of *CT* image.

Method	IT	CPU(s)	RSE
GBK(0.2)	863	5.6002	9.8576e−07
RGBK(1.3)	753	5.0474	8.3409e−07
AGBK	787	1.9918	9.5513e−07

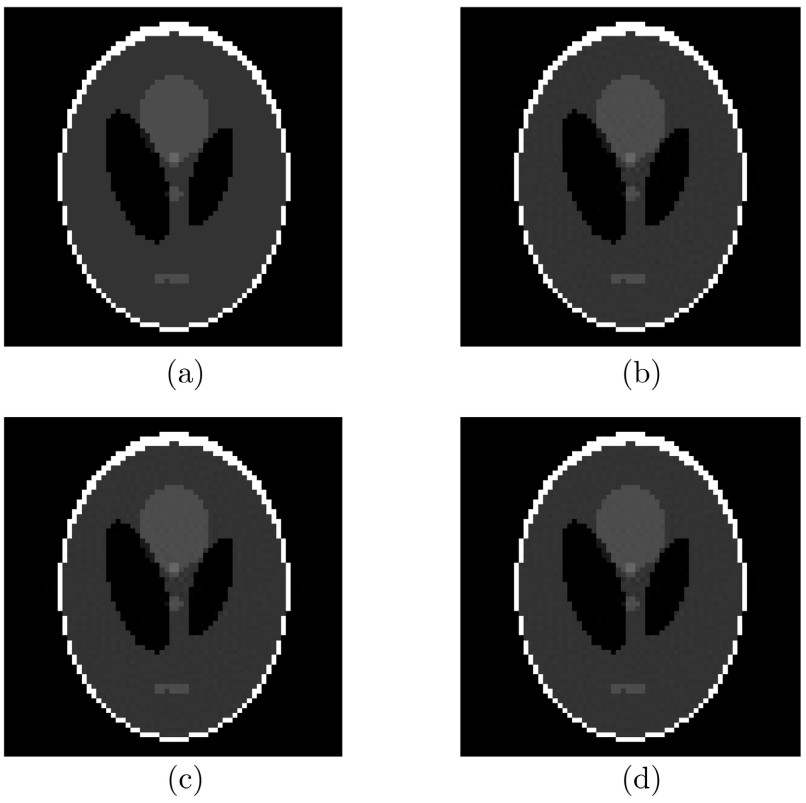

Figure 4. The original "*phantom*" image (a), the recovered images by GBK (b), RGBK(1.3) (c) and AGBK (d).

4. Conclusion

We present a relaxed GBK algorithm abbreviated as RGBK for solving large consistent linear systems. The RGBK method extends the GBK method in [17]. The convergence is provided and a method is provided to determine an optimal relaxed parameter for the RGBK method. In addition, AGBK effectively accelerates the convergence of RGBK in running time. The examples for different cases show the advantage of the proposed RGBK and AGBK methods as long as the optimal relaxation parameter λ_{opt} is determined.

Acknowledgements

The authors thank the reviewers for providing some helpful comments. Research by Y.L. was partially supported by The Innovation Fund of Postgraduate, Sichuan University of Science & Engineering (grant y2021101), Research by F.Y. was partially supported by the Opening Project of Sichuan Province University Key Laboratory of Bridge Non-destruction Detecting and Engineering Computing (grant 2020QZJ03) and SUSE (grant 2019RC09, 2020RC25) and research by G.H. was supported in part by Application Fundamentals Foundation of STD of Sichuan (grant 2020YJ0366).

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Niu, Y.Q. and Zheng, B. (2020) A Greedy Block Kaczmarz Algorithm for Solving Large-Scale Linear Systems. *Applied Mathematics Letters*, **104**, Article ID: 106294. https://doi.org/10.1016/j.aml.2020.106294

[2] Kaczmarz, S. (1937) Angenäherte auflösung von systemen linearer Gleichungen. *Bulletin International de l' Académie Polonaise des Sciences et des Lettres*, **35**, 355-357.

[3] Gordon, R., Bender, R. and Herman, G. (1970) Algebraic Reconstruction Techniques (ART) for Three-Dimensional Electron Microscopy and X-Ray Photography. *Journal of Theoretical Biology*, **29**, 471-481. https://doi.org/10.1016/0022-5193(70)90109-8

[4] Censor, Y. (1988) Parallel Application of Block-Iterative Methods in Medical Imaging and Radiation Therapy. *Mathematical Programming*, **42**, 307-325. https://doi.org/10.1007/BF01589408

[5] Byrne, C. (2004) A Unified Treatment of Some Iterative Algorithms in Signal Processing and Image Reconstruction. *Inverse Problems*, **20**, 103-120. https://doi.org/10.1088/0266-5611/20/1/006

[6] Herman, G.T. (2009) Fundamentals of Computerized Tomography: Image Reconstruction from Projections. 2nd Edition, Springer, Dordrecht.

[7] Elble, J.M. and Sahinidis, N.V. and Vouzis, P. (2010) GPU Computing with Kaczmarz and Other Iterative Algorithms for Linear Systems. *Parallel Computing*, **36**, 215-231. https://doi.org/10.1016/j.parco.2009.12.003

[8] Eldar, Y.C. and Needell, D. (2011) Acceleration of Randomized Kaczmarz Method

via the Johnson-Lindenstrauss Lemma. *Numerical Algorithms*, **58**, 163-177. https://doi.org/10.1007/s11075-011-9451-z

[9] Needell, D. and Tropp, J.A. (2014) Paved with Good Intentions: Analysis of a Randomized Block Kaczmarz Method. *Linear Algebra and Its Applications*, **441**, 199-221. https://doi.org/10.1016/j.laa.2012.12.022

[10] Zouzias, A. and Freris, N. (2012) Randomized Extended Kaczmarz for Solving Least Squares. *SIAM Journal on Matrix Analysis and Applications*, **34**, 773-793. https://doi.org/10.1137/120889897

[11] Briskman, J. and Needell, D. (2015) Block Kaczmarz Method with Inequalities. *Journal of Mathematical Imaging and Vision*, **52**, 385-396. https://doi.org/10.1007/s10851-014-0539-7

[12] Ma, A., Needell, D. and Ramda, A.S. (2015) Convergence Properties of the Randomized Extended Gauss-Seidel and Kaczmarz Methods. *SIAM Journal on Matrix Analysis and Applications*, **36**, 1590-1604. https://doi.org/10.1137/15M1014425

[13] Needell, D., Zhao, R. and Zouzias A. (2015) Randomized Block Kaczmarz Method with Projection for Solving Least Squares. *Linear Algebra and its Applications*, **484**, 322-343. https://doi.org/10.1016/j.laa.2015.06.027

[14] Bai, Z.Z. and Wu, W.T. (2019) On Partially Randomized Extended Kaczmarz Method for Solving Large Sparse Overdetermined Inconsistent Linear Systems. *Linear Algebra and Its Applications*, **578**, 225-250. https://doi.org/10.1016/j.laa.2019.05.005

[15] Du, K. (2019) Tight Upper Bounds for the Convergence of the Randomized Extended Kaczmarz and Gauss-Seidel Algorithms. *Numerical Linear Algebra with Applications*, **26**, e2233. https://doi.org/10.1002/nla.2233

[16] Richtarik, P. and Takavc, M. (2020) Stochastic Reformulations of Linear Systems Algorithms and Convergence Theory. *SIAM Journal on Matrix Analysis and Applications*, **41**, 487-524. https://doi.org/10.1137/18M1179249

[17] Strohmer, T. and Vershynin, R. (2009) A Randomized Kaczmarz Algorithm with Exponential Convergence. *Journal of Fourier Analysis and Applications*, **15**, 262-278. https://doi.org/10.1007/s00041-008-9030-4

[18] Zhang, J.J. (2018) A New Greedy Kaczmarz Algorithm for the Solution of Very Large Linear Systems. *Applied Mathematics Letters*, **91**, 207-212. https://doi.org/10.1016/j.aml.2018.12.022

[19] Bai, Z.Z. and Wu, W.T. (2018) On Greedy Randomized Kaczmarz Method for Solving Large Sparse Linear Systems. *SIAM Journal on Scientific Computing*, **40**, A592-A606. https://doi.org/10.1137/17M1137747

[20] Bai, Z.Z. and Wu, W.T. (2018) On Relaxed Greedy Randomized Kaczmarz Methods for Solving Large Sparse Linear Systems. *Applied Mathematics Letters*, **83**, 21-26. https://doi.org/10.1016/j.aml.2018.03.008

[21] Björck, A. (1996) Numerical Methods for Least Squares Problems. SIAM, Philadelphia. https://doi.org/10.1137/1.9781611971484

[22] Gower, R.M. and Richtárik, P. (2015) Randomized Iterative Methods for Linear Systems. *SIAM Journal on Matrix Analysis and Applications*, **36**, 1660-1690. https://doi.org/10.1137/15M1025487

[23] Kashkari, B. and Alqarni, S. (2019) Two-Step Hybrid Block Method for Solving Nonlinear Jerk Equations. *Journal of Applied Mathematics and Physics*, **7**, 1897-1910. https://doi.org/10.4236/jamp.2019.78130

[24] Duromola, M. and Momoh, A. (2019) Hybrid Numerical Method with Block Extension for Direct Solution of Third Order Ordinary Differential Equations. *American*

Journal of Computational Mathematics, **9**, 68-80.
https://doi.org/10.4236/ajcm.2019.92006

[25] Davis, T.A. and Hu, Y. (2011) The University of Florida Sparse Matrix Collection. *ACM Transactions on Mathematical Software*, **38**, 1-25.
https://doi.org/10.1145/2049662.2049663

[26] Hansen, R.C. and Jorgensen, J.S (2018) AIR Tools II: Algebraic Iterative Reconstruction Methods. *Numerical Algorithms*, **79**, 107-137.
https://doi.org/10.1007/s11075-017-0430-x