ISSN 1346-5597

NII National Institute of Informatics

NII Technical Report

Kaczmarz-type inner-iteration preconditioned flexible GMRES methods for consistent linear systems

Yi-shu Du, Ken Hayami, Ning Zheng, Keiichi Morikuni,
Jun-Feng Yin

NII-2020-003E
June 2020

URL: http://www.nii.ac.jp/ Tokyo 101-8430, JAPAN +81-3-4212-2000 FAX: +81-3-3556-1916

KACZMARZ-TYPE INNER-ITERATION PRECONDITIONED FLEXIBLE GMRES METHODS FOR CONSISTENT LINEAR SYSTEMS

YI-SHU DU*, KEN HAYAMI†, NING ZHENG‡, KEIICHI MORIKUNI§, AND JUN-FENG YIN¶

Abstract. We propose using greedy and randomized Kaczmarz inner-iterations as preconditioners for the right preconditioned flexible GMRES method to solve consistent linear systems, with a parameter tuning strategy for adjusting the number of inner iterations and the relaxation parameter. We also present theoretical justifications of the right-preconditioned flexible GMRES for solving consistent linear systems. Numerical experiments on overdetermined and underdetermined linear systems show that the proposed method is superior to the GMRES method preconditioned by NE-SOR inner iterations in terms of total CPU time.

Key words. Kaczmarz method, randomized algorithm, linear system, overdetermined system, underdetermined system, least squares problem, iterative method, inner-outer iteration, preconditioner, GMRES, flexible GMRES

AMS subject classifications. 65F08, 65F10, 65F50, 15A06

1. Introduction.

Consider solving consistent linear systems

$$Ax = b, \quad b \in \mathcal{R}(A), \tag{1}$$

where $A \in \mathbb{R}^{m\times n}$ is not necessarily of full rank and $\mathcal{R}(A)$ is the range space of A. In particular, consider the minimum Euclidean-norm solution

$$\min_{x\in\mathbb{R}^n} \|x\|_2 \quad \text{s.t.} \quad Ax = b, \quad b \in \mathcal{R}(A). \tag{2}$$

The problem (2) is equivalent to the normal equations of the second kind

$$AA^{\mathsf{T}}u = b, \quad x = A^{\mathsf{T}}u, \quad b \in \mathcal{R}(A), \tag{3}$$

where $(\cdot)^{\mathsf{T}}$ denotes the transpose.

Direct methods for solving problem (2) or (3) are generally expensive when the coefficient matrix is large and sparse. A well-established iterative method for solving (2) is the (preconditioned) CGNE method [15, 35], which is mathematically equivalent to the (preconditioned) Conjugate Gradient (CG) method [23] applied to (3). Another method is the (preconditioned) MRNE method [29, 16], which applies the (preconditioned) MINRES method [30] to (3). Note that iterative methods may be slow to converge for ill-conditioned problems since the condition number of AA^{T} is the square of that of A, and preconditioning becomes necessary. In [22], Hayami, Yin and Ito proposed a right-preconditioned generalized minimal residual (GMRES) method called the AB-GMRES method by applying GMRES to $\min_{u\in\mathbb{R}^m} \|b - ABu\|_2$, where $B \in \mathbb{R}^{n\times m}$.

In order to accelerate the convergence of iterative methods and save the storage requirement, inner iterations can be applied as a preconditioner inside the Krylov subspace methods instead of applying preconditioning matrices explicitly. Such techniques are often called inner-outer iteration methods [34]. Morikuni and Hayami [28, 29] proposed a class of inner-iteration Krylov subspace methods by applying stationary inner iterations as implicit preconditioners, and showed their efficiency particularly for ill-conditioned and rank-deficient problems. (See also [16].)

In AB-GMRES, common choices for stationary inner iterations are the normal error Gauss-Seidel (NE-GS) and normal error successive overrelaxation (NE-SOR) methods [9, 35], which are also commonly referred to as the Kaczmarz [25] or row action methods [1, 3, 11, 12, 21, 36]. Since it was proposed in

*School of Mathematical Sciences, Tongji University, N.O. 1239, Siping Road, Shanghai, 200092, China, and LIP, École Normale Supérieure de Lyon, INRIA, 46 Allée d'Italie, Lyon, 69364, France (duyishu@tongji.edu.cn).

†National Institute of Informatics, and The Graduate University for Advanced Studies (SOKENDAI), 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 100-0003, Japan (hayami@nii.ac.jp).

‡Research Center for Statistical Machine Learning, The Institute of Statistical Mathematics, 10-3 Midori-cho, Tachikawa Tokyo 190-8562, Japan (nzheng@ism.ac.jp).

§Faculty of Engineering, Information and Systems, University of Tsukuba, 1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan (morikuni@cs.tsukuba.ac.jp).

¶School of Mathematical Sciences, Tongji University, N.O. 1239, Siping Road, Shanghai, 200092, China (yinjf@tongji.edu.cn).

Funding: This work was funded by JSPS KAKENHI Grant (No. 15K04768 and No. 20K14356), the National Natural Science Foundation of China (No. 11971354) and the China Scholarship Council (No. 201906260146).

the 1930s, the Kaczmarz method has gained great theoretical development and plentiful practical applications [10, 13, 18, 19, 31, 39]. Research on the Kaczmarz method was reignited in 2006 and 2009 when Strohmer and Vershymin [37, 38] proposed the randomized Kaczmarz method with expected exponential rate of convergence. In [5], Bai and Wu constructed a greedy randomized Kaczmarz method by proposing a more effective probability criterion. In [32], Popa summarized convergence rates for Kaczmarz-type methods, including greedy Kaczmarz [1], randomized Kaczmarz methods [38] and so on. For more literature on Kaczmarz-type methods, we refer the reader to [4, 6, 7, 8]. Numerical results show that these randomized or greedy Kaczmarz-type methods accelerate the original Kaczmarz method and reduce the required number of iterations and CPU time effectively. Inspired by this randomized framework, we replace the NE-SOR method by the greedy and randomized Kaczmarz methods in AB-GMRES preconditioned by stationary inner iterations.

A motivation for developing such a greedy/randomized inner-iteration preconditioning arises in applications where the operation on a row of a matrix is relatively expensive, such as in basis pursuit problems [14, 16]. We mention related previous work [2, 26, 33] on randomized preconditioners for least squares problems.

When the randomized or greedy Kaczmarz method is used as the inner iteration, the rows of A are selected randomly or greedily in each iteration and thus the preconditioner is not fixed during the outer iteration. Therefore, we use the flexible GMRES method [34] as the outer iteration and propose a new algorithm called the flexible AB-GMRES method with Kaczmarz-type inner iterations. Theoretically, an optimality property of minimizing residual norm can be given under the framework of flexible GMRES. We also propose a parameter tuning procedure for adjusting the number of inner iterations and the relaxation parameter for the new method. Numerical results show that flexible AB-GMRES preconditioned by Kaczmarz-type methods outperform the AB-GMRES method preconditioned by NE-SOR iterations [28, 29] in terms of total CPU time.

The organization of this paper is as follows. In section 2, we review the AB-GMRES method. In section 3, we present the flexible AB-GMRES method for consistent linear systems, and give an optimality property of the proposed method. In section 4, we propose a parameter tuning procedure for the new method and present numerical experiment results. In section 5, we conclude the paper.

2. AB-GMRES method. In this section, the inner-iteration preconditioned AB-GMRES method is briefly introduced. Consider solving equation (2) using AB-GMRES. AB-GMRES corresponds to GMRES applied to $\min_{u\in\mathbb{R}^m} \|b - ABu\|_2$ with $x = Bu$ and works in an m-dimensional space [22]. In order to achieve fast convergence of AB-GMRES and to avoid storing the preconditioner B, stationary inner iterations in combination with AB-GMRES were proposed in [28, 29]. This algorithm can be described as follows.

Algorithm 1 AB-GMRES method preconditioned by inner iterations [29]

Let x_0 be the initial approximate solution and $r_0 = b - Ax_0$.
$\beta = \|r_0\|_2$, $v_1 = r_0/\beta$
for $k = 1, 2, \ldots$ until convergence **do**
  Apply ℓ iterations of a stationary iterative method to $AA^{\mathsf{T}}y = v_k$, $z = A^{\mathsf{T}}y$ to obtain $z_k = B^{(\ell)}v_k$.
  $w_k = Az_k$
  for $i = 1, 2, \ldots, k$, **do**
    $h_{i,k} = {w_k}^{\mathsf{T}} v_i$, $w_k = w_k - h_{i,k}v_i$
  end for
  $h_{k+1,k} = \|w_k\|_2$, $v_{k+1} = w_k/h_{k+1,k}$
end for
$y_k = \arg\min_{y\in\mathbb{R}^k} \left\|\beta e_1 - \bar{H}_k y\right\|_2$, $u_k = [v_1, v_2, \ldots, v_k]\, y_k$, where $\bar{H}_k = \{h_{ij}\}_{1\le i\le k+1; 1\le j\le k}$
Apply ℓ iterations of a stationary iterative method to $AA^{\mathsf{T}}y = u_k$, $z = A^{\mathsf{T}}y$ to obtain $z_k = B^{(\ell)}u_k$.
$x_k = x_0 + z_k$

In the AB-GMRES preconditioned by inner iterations, one common choice for stationary inner iterations is the NE-SOR method, which is mathematically equivalent to the SOR method applied to the normal equations of the second kind [9, 35]. More specifically, if we use α_i^{T} to represent the ith row of the matrix A, and v_i to represent the ith entry of the vector v, then the NE-SOR method for $AA^{\mathsf{T}}y = v, z = A^{\mathsf{T}}y$ can be described as follows.

Algorithm 2 NE-SOR method [35], Kaczmarz method with relaxation

Let $z^{(0)}$ be the initial approximate solution and $\omega \in \mathbb{R}$ be the relaxation parameter.
for $p = 0, 1, 2, \ldots, \ell - 1$ **do**
  for $i = 1, 2, \ldots, m$ **do**
    $z^{(p)} = z^{(p)} + \omega \frac{v_i - \alpha_i^{\mathsf{T}} z^{(p)}}{\|\alpha_i\|_2^2} \alpha_i$
  end for
  $z^{(p+1)} = z^{(p)}$
end for

The Kaczmarz method [25] is equivalent to Algorithm 2 with $\omega = 1$. In fact, the iteration scheme of NE-GS (NE-SOR) is exactly the same as that of the Kaczmarz method (relaxed Kaczmarz method) [25]. The relaxed Kaczmarz (NE-SOR) method is one of the most efficient row action methods. The method cycles through the rows of the linear system and form each iterate by orthogonally projecting the current point onto the hyperplane $\alpha_i^{\mathsf{T}} z^{(p+1)} = v_i$ formed by the active row, and all the m equations in the linear system are swept through consecutively in m iterations.

The convergence theorem of AB-GMRES preconditioned by NE-SOR is precisely restated below.

THEOREM 1 *[29, Theorem 5.6]. AB-GMRES preconditioned by NE-SOR inner iterations with* $0 < \omega < 2$, *determines the minimum-norm solution of* $Ax = b$ *without breakdown for all* $b \in \mathcal{R}(A)$ *and for all* $x_0 \in \mathcal{R}(A^{\mathsf{T}})$.

2.1. Flexible AB-GMRES method. We adopt the flexible preconditioning framework proposed in FGMRES [34] to AB-GMRES, and consider using Kaczmarz-type inner iterations in it.

2.1.1. Outer-iteration algorithm. It is well known that the preconditioning matrix needs to be fixed in the preconditioned GMRES method for all the outer iterations. In fact, in order to keep the preconditioner $B^{(\ell)}$ in Algorithm 1 fixed, the number of inner iterations in AB-GMRES should not be changed for each outer iteration. However, if we were to adopt the randomized or greedy algorithm as the preconditioner in AB-GMRES, the preconditioning matrix for each outer iteration may change even though the number of inner iterations for each outer iteration is fixed. In [34], Saad presented a variant of the GMRES algorithm called flexible GMRES (FGMRES) for solving square linear systems $Ax = b$, which allows the preconditioning matrix to change for each outer iteration. This algorithm can be described as follows.

Algorithm 3 Flexible GMRES (FGMRES) [34]

Let x_0 be the initial approximate solution and $r_0 = b - Ax_0$.
$\beta = \|r_0\|_2$, $v_1 = r_0/\beta$
for $k = 1, 2, \ldots$ until convergence **do**
  $z_k = B^{(k)} v_k$, $w_k = A z_k$
  for $i = 1, 2, \ldots, k$, **do**
    $h_{i,k} = (w_k, v_i)$, $w_k = w_k - h_{i,k} v_i$
  end for
  $h_{k+1,k} = \|w_k\|_2$, $v_{k+1} = w_k / h_{k+1,k}$
  Define $Z_k = [z_1, z_2, \ldots, z_k]$.
end for
$y_k = \arg\min_{y \in \mathbb{R}^k} \left\| \beta e_1 - \bar{H}_k y \right\|_2$, where $\bar{H}_k = \{h_{ij}\}_{1 \le i \le k+1; 1 \le j \le k}$
$x_k = x_0 + Z_k y_k$

2.1.2. Randomized inner-iteration algorithms. Instead of using the rows of the coefficient matrix A in the consecutive order, Ansorge [1] proposed selecting the i_kth row corresponding to the residual component with maximum absolute value. We will call this method the greedy Kaczmarz method, or GK method for short. See also [36] for earlier work. Another approach is to choose the i_kth row randomly [38], which is called the randomized Kaczmarz (RK) method. Finally, Bai and Wu [5] combined these two ideas to propose a more effective probability criterion, which aims to diminish entries of the residual vector with relatively large absolute value at each iteration. We will call the corresponding algorithm the greedy randomized

Kaczmarz (GRK) method. These algorithms are called Kaczmarz-type methods. It can be proved that GK, RK and GRK methods converge to the minimum-norm solution whether the system is overdetermined or underdetermined and the coefficient matrix has full rank or is rank deficient [32]. The algorithms of GK, RK and GRK are given below.

Algorithm 4 Greedy Kaczmarz (GK) method [1]

1: Let $z^{(0)}$ be the initial approximate solution.
2: **for** $p = 0, 1, 2, \ldots, \ell - 1$ **do**
3: Select $i_p \in \{1, 2, \ldots m\}$ according to $i_p = \arg\max_i |v_i - \alpha_i^{\mathsf{T}} z^{(p)}|$.
4: $z^{(p+1)} = z^{(p)} + \omega \frac{v_{i_p} - \alpha_{i_p}^{\mathsf{T}} z^{(p)}}{\|\alpha_{i_p}\|_2^2} \alpha_{i_p}$
5: **end for**

Algorithm 5 Randomized Kaczmarz (RK) method [38]

1: Let $z^{(0)}$ be the initial approximate solution.
2: **for** $p = 0, 1, 2, \ldots, \ell - 1$ **do**
3: Select $i_p \in \{1, 2, \ldots m\}$ with probability $\Pr(\text{row} = i_p) = \frac{\|\alpha_{i_p}\|_2^2}{\|A\|_{\mathrm{F}}^2}$.
4: $z^{(p+1)} = z^{(p)} + \omega \frac{v_{i_p} - \alpha_{i_p}^{\mathsf{T}} z^{(p)}}{\|\alpha_{i_p}\|_2^2} \alpha_{i_p}$
5: **end for**

Algorithm 6 Greedy Randomized Kaczmarz (GRK) method [5]

1: Let $z^{(0)}$ be the initial approximate solution.
2: **for** $p = 0, 1, 2, \ldots, \ell - 1$ **do**
3: $\varepsilon_p = \frac{1}{2} \left(\frac{1}{\|v - Az^{(p)}\|_2^2} \max_{1 \le i_p \le m} \left\{ \frac{\left| v_{i_p} - \alpha_{i_p}^{\mathsf{T}} z^{(p)} \right|^2}{\|\alpha_{i_p}\|_2^2} \right\} + \frac{1}{\|A\|_{\mathrm{F}}^2} \right)$
4: Determine the index set of positive integers

$$U_p = \left\{ i_p : |v_{i_p} - \alpha_{i_p}^{\mathsf{T}} z^{(p)}|^2 \ge \varepsilon_p \|v - Az^{(p)}\|_2^2 \|\alpha_{i_p}\|_2^2 \right\}.$$

5: Compute the ith entry $\tilde{s}_i^{(p)}$ of the vector $\tilde{s}^{(p)}$ according to

$$\tilde{s}_i^{(p)} = \begin{cases} v_i - \alpha_i^{\mathsf{T}} z^{(p)}, & \text{if} \quad i \in U_p, \\ 0, & \text{otherwise.} \end{cases}$$

6: Select $i_p \in U_p$ with probability $\Pr(\text{row} = i_p) = \frac{\left| \tilde{s}_{i_p}^{(p)} \right|^2}{\|\tilde{s}^{(p)}\|_2^2}$.
7: $z^{(p+1)} = z^{(p)} + \omega \frac{v_{i_p} - \alpha_{i_p}^{\mathsf{T}} z^{(p)}}{\|\alpha_{i_p}\|_2^2} \alpha_{i_p}$
8: **end for**

In Algorithms 5 and 6, $\Pr(\text{row} = i)$ represents the probability of selecting the ith row of the matrix A as the working row of this iteration.

We remark that $s = v - Az$ needs to be calculated at each inner iteration for GK and GRK methods. This additional computational work cannot be ignored. On the other hand, we may update the residual

vector s recursively as follows [5]:

$$\begin{aligned} s^{(p+1)} &= v - Az^{(p+1)} \\ &= v - A\left(z^{(p)} + \frac{v_{i_p} - \alpha_{i_p}^{\mathsf{T}} z^{(p)}}{\left\|\alpha_{i_p}\right\|_2^2}\alpha_{i_p}\right) \\ &= v - Az^{(p)} - \frac{s_{i_p}^{(p)}}{\left\|\alpha_{i_p}\right\|_2^2}A\alpha_{i_p} \\ &= s^{(p)} - \frac{s_{i_p}^{(p)}}{\left\|\alpha_{i_p}\right\|_2^2}C_{(i_p)}. \end{aligned} \tag{4}$$

Here, $C_{(i_p)}$ is the i_pth column of $C = AA^{\mathsf{T}}$. Hence, if the matrix product AA^{T} is computed and stored once in the beginning, the computational work can be reduced, assuming that the total number of inner iterations is more than the number of rows of A. (See Appendix.) This condition was satisfied in all our numerical experiments.

3. Flexible AB-GMRES preconditioned by Kaczmarz-type methods as inner iterations. In FGMRES, we can change the preconditioner for each outer iteration. Hence, consider using ℓ_k iterations of a Kaczmarz-type method as the preconditioner for each outer iteration of the flexible AB-GMRES (F-AB-GMRES). We denote the preconditioning matrix given by the ℓ_k inner iterations by $B^{(\ell_k)}$. The algorithm is given as follows.

Algorithm 7 Flexible AB-GMRES preconditioned by Kaczmarz-type methods

1: Let x_0 be the initial approximate solution and $r_0 = b - Ax_0$.
2: $\beta = \|r_0\|_2$, $v_1 = r_0/\beta$
3: **for** $k = 1, 2, \ldots$ until convergence **do**
4: Apply ℓ_k iterations of a Kaczmarz-type method to $Az = v_k$ to obtain $z_k = B^{(\ell_k)}v_k$, where ℓ_k is the smaller of the maximum number of inner iterations $\ell_{\max}$ and the smallest ℓ such that

$$\left\|v_k - AB^{(\ell)}v_k\right\|_2 \leq \max\left\{0.9^k, 10^{-1}\right\}\|v_k\|_2.$$

5: $w_k = Az_k$
6: **for** $i = 1, 2, \ldots, k$, **do**
7: $h_{i,k} = {w_k}^{\mathsf{T}} v_i$, $w_k = w_k - h_{i,k}v_i$
8: **end for**
9: $h_{k+1,k} = \|w_k\|_2$, $v_{k+1} = w_k/h_{k+1,k}$
10: **end for**
11: $y_k = \arg\min_{y\in\mathbb{R}^k}\left\|\beta e_1 - \bar{H}_k y\right\|_2$, $u_k = [z_1, z_2, \ldots, z_k]\, y_k$, where $\bar{H}_k = \{h_{ij}\}_{1\leq i\leq k+1; 1\leq j\leq k}$
12: $x_k = x_0 + u_k$

Here, $\bar{H}_k = \{h_{i,j}\} \in \mathbb{R}^{(k+1)\times k}$. Since the number of inner iterations in each outer iteration does not have to be fixed, we proposed a new inner iterations stopping criterion that varies with the outer iteration to accelerate the convergence, which is given in line 4 of Algorithm 7. An optimality property similar to GMRES is given as in [34, Proposition 2.1] for FGMRES.

THEOREM 2 *The approximate solution x_k obtained at the kth iteration of F-AB-GMRES minimizes the residual norm $\|b - Ax_k\|_2$ over $x_0 + \mathrm{Span}\{z_1, z_2, \ldots, z_k\}$.*

Proof It is easy to show that

$$\begin{aligned} AZ_k &= V_{k+1}\bar{H}_k \\ &= V_k H_k + h_{k+1,k}v_{k+1}e_k^{\mathsf{T}}. \end{aligned} \tag{5}$$

Here, H_k denotes the $k \times k$ matrix obtained from $\bar{H}_k$ by deleting its last row, V_k denotes the $n \times k$ matrix with column vectors v_1, v_2, ..., v_k, and e_k denotes the last column of the identify matrix $\mathrm{I_k}$ of size k.

Consider the residual vector for an arbitrary vector $x = x_0 + Z_k y$ in the affine space $x_0 + \mathrm{Span}\{Z_k\}$, where $Z_k = [z_1, z_2, \ldots, z_k]$. The optimality property is based on the relations

$$\begin{aligned} b - Ax &= b - A(x_0 + Z_k y) \\ &= r_0 - AZ_k y \\ &= \beta v_1 - V_{k+1}\bar{H}_k y \\ &= V_{k+1}\left(\beta e_1 - \bar{H}_k y\right), \end{aligned} \tag{6}$$

where e_1 is the first column of $\mathrm{I_{k+1}}$.

If $J_k(y)$ denotes the function

$$J_k(y) = \|b - A(x_0 + Z_k y)\|_2, \tag{7}$$

observe that by (6) and the fact that the column vectors of V_{k+1} are orthogonal,

$$J_k(y) = \left\|\beta e_1 - \bar{H}_k y\right\|_2. \tag{8}$$

Since the algorithm minimizes this norm over all vectors in $\mathbb{R}^k$ to yield y_k, it is clear that the approximate solution $x_k = x_0 + Z_k y_k$ has the smallest residual norm in $x_0 + \mathrm{Span}\{Z_k\}$. ∎

Next, we consider the possibility of breakdown in F-AB-GMRES preconditioned by Kaczmarz-type methods as inner iterations. A breakdown occurs when the vector v_{k+1} cannot be computed in line 9 of Algorithm 7 because $h_{k+1,k} = 0$. For AB-GMRES with B satisfying the convergence conditions [29, Theorem 5.2], [22, Corollary 3.8], this is not a problem because when this happens, the approximate solution x_k satisfies $Ax_k = b$. The situation for F-AB-GMRES is slightly different, as in [34, Proposition 2.2] for FGMRES.

THEOREM 3 *Assume that $\beta = \|r_0\|_2 \neq 0$ and that $k-1$ steps of F-AB-GMRES have been successfully performed, i.e., that $h_{i+1,i} \neq 0$ for $i < k$. In addition, assume that the matrix H_k is nonsingular. Then x_k is a solution of $Ax = b$ if and only if $h_{k+1,k} = 0$.*

Proof If $h_{k+1,k} = 0$, then $AZ_k = V_k H_k$, and we have

$$J_k(y) = \|\beta v_1 - AZ_k y\|_2 = \|\beta v_1 - V_k H_k y\|_2 = \|\beta e_1 - H_k y\|_2,$$

where e_1 is the first column of $\mathrm{I_k}$.

If H_k is nonsingular, then the above function is minimized for $y = y_k = H_k^{-1}(\beta e_1)$ and the corresponding minimum norm is zero, that is to say, x_k is a solution of $Ax = b$.

On the other hand, if x_k satisfies $Ax_k = b$, then from (5) and (6), we have

$$0 = b - Ax_k = V_k(\beta e_1 - H_k y_k) - w_k e_k^{\mathsf{T}} y_k. \tag{9}$$

We show that $h_{k+1,k} = 0$ by contradiction. Suppose that $h_{k+1,k} \neq 0$ holds. If the last component of y_k is equal to zero $e_k^{\mathsf{T}} y_k = 0$, then a simple backward substitution for the system $H_k y_k = \beta e_1$, starting from the last equation, will show that all components of y_k are zero. This would imply that $\beta = 0$, which contradicts the assumption. Hence, $h_{k+1,k} = 0$ holds. ∎

The only difference of this result from that of AB-GMRES is that the additional assumption that H_k is nonsingular must be made since it is no longer implied by the algorithm. In fact, the following holds

THEOREM 4 *Assume* $\mathrm{rank}A = n$, $\mathrm{rank}Z_k = k$ *and* $h_{k+1,k} = 0$. *Then, H_k is nonsingular.*

Proof Let $c_1 Az_1 + \cdots + c_k Az_k = A(c_1 z_1 + \cdots + c_k z_k) = 0$. Then, $\mathrm{rank}A = n$ implies $c_1 z_1 + \cdots + c_k z_k = 0$, and $\mathrm{rank}Z_k = k$ implies $c_1 = \cdots = c_k = 0$. Hence, $\mathrm{rank}(AZ_k) = k$. Since, $h_{k+1,k} = 0$, $AZ_k = V_k H_k$. Thus, $k = \mathrm{rank}(AZ_k) = \mathrm{rank}(V_k H_k) \leq \min(\mathrm{rank}V_k, \mathrm{rank}H_k) = \min(k, \mathrm{rank}H_k)$. Hence, $\mathrm{rank}H_k = k$, and H_k is nonsingular. ∎

See also [27, Theorem 3] for convergence conditions of FGMRES preconditioned by multistep matrix splitting iterations, which ensure the nonsingularity of H_k.

The additional cost of the flexible variant over AB-GMRES is only the extra memory required to save the set of vectors $\{z_j\}_{j=1,2,\ldots,m}$. However, the added advantage of flexibility may be worth this extra cost.

4. Numerical experiments. We compare the proposed F-AB-GMRES preconditioned by Kaczmarz-type methods as inner iterations with AB-GMRES preconditioned by NE-SOR inner iterations [28, 29] in terms of the central processing unit (CPU) time by numerical experiments on underdetermined and overdetermined problems.

Table 1 gives the number of rows m, the number of columns n, the density of the nonzero elements, the rank and the condition number $\kappa(A)$ on the overdetermined test matrices. The matrices RANDLi, $i = 1, 2, \ldots, 6$ were randomly generated using the MATLAB function `sprandn`, as in [22, 28, 29]. The Maragal_j, $j = 3, 4, 5$ are rank-deficient matrices from [17]. These matrices were transposed to form underdetermined problems. Table 1 shows the effective size of the matrices after removing all zero rows. (If the matrix A has a zero row, then the Kaczmarz-type methods can not work.) The rank and condition number were computed using the MATLAB functions `sprank` [20] and `svd`.

In our implementations, a solution vector $x_\star \in \mathbb{R}^n$ is randomly generated by using the MATLAB function `randn`, and the right-hand side $b \in \mathbb{R}^m$ is taken to be $Ax_\star$. All computations were started from the initial vector $x_0 = 0$, and the stopping criterion is the relative residual

$$\frac{\|r_k\|_2}{\|b\|_2} \leq 10^{-6},$$

where $r_k = b - Ax_k$ is the kth residual vector, or the maximum number of outer iterations 2000. No restarts were used for GMRES.

All experiments were carried out using MATLAB (version R2017b) on a personal computer with 2.80 GHz CPU (Intel(R) Core(TM) i7-7700HQ Q9550), 16 GB memory, and Windows 10 Home version 1909.

4.1. Automatic parameter tuning for Kaczmarz-type methods. The proposed method requires two preconditioning parameters: the maximum number of inner iterations $\ell_{\max}$ in line 4 of Algorithm 7 and the relaxation parameter ω used for the Kaczmarz-type inner iterations. Since the CPU time for the proposed method varies with the values of these parameters, it is desirable to determine the values automatically for any problem. Inspired by the idea in [28, 29], we perform the following procedure given as Algorithm 8 using Kaczmarz-type methods alone for $Az = b$ before starting the outer iterations to determine the values of these parameters $\ell_{\max}$ and ω_{opt}.

Algorithm 8 Parameter tuning procedure of Kaczmarz-type methods

1: Set $\omega = 1$ and $z^{(0)} = 0$.
2: **for** $\ell = 1, 2, \ldots$ **do**
3: Apply ℓ iterations of a Kaczmarz-type method to $Az = b$ to obtain $z^{(\ell)}$.
4: **if** $\left\|b - Az^{(\ell)}\right\|_2 \leq \eta \|b\|_2$ **then**
5: break
6: **end if**
7: **end for**
8: $\ell_{\max} = \ell$
9: **for** $\omega = 0.1, 0.2, \ldots, 1.9$ **do**
10: Apply $\ell_{\max}$ iterations of a Kaczmarz-type method to $Az = b$.
11: **end for**
12: $\omega_{\text{opt}} = \arg\min_{\omega=0.1,0.2,\ldots,1.9} \left\|r^{(\ell_{\max})}\right\|_2$

We remark that Morikuni and Hayami [28] evaluated the performance of AB-GMRES preconditioned by NE-SOR inner iterations for different values of η and finally chose $\eta = 10^{-1}$. Here, we also choose $\eta = 10^{-1}$. Since the number of inner iterations in F-AB-GMRES does not have to be fixed for each outer iteration, we set $\ell_{\max}$ determined by Algorithm 8 to be the maximum number of inner iterations for each outer iteration.

4.2. Underdetermined problems. We first present numerical experiment results on underdetermined problems ($m < n$). Tables 2 and 3 give the numerical experiment results on full-rank matrices and rank-deficient matrices, respectively. The letter T at the end of the name of a matrix denotes the transposition of the matrix. The first row in each cell in Tables 2 and 3 gives the number of outer iterations outside parentheses and gives the total number of inner iterations (the sum of the number of inner iterations in each

TABLE 1
Information of the matrices.

Name	m	n	Density[%]	Rank	$\kappa(A)$
RANDL1	5000	500	20	500	1.00×10^{1}
RANDL2	5000	500	20	500	1.00×10^{2}
RANDL3	5000	500	20	500	1.00×10^{3}
RANDL4	5000	500	20	500	1.00×10^{4}
RANDL5	5000	500	20	500	1.00×10^{5}
RANDL6	5000	500	20	500	1.00×10^{6}
Maragal_3	1682	858	1.27	613	1.10×10^{3}
Maragal_4	1964	1027	1.32	801	9.33×10^{6}
Maragal_5	4654	3296	0.61	2147	1.19×10^{5}

Name: name of the matrix, m: number of rows of the matrix, n: number of columns of the matrix, Density: density of the nonzero components of the matrix, Rank: maximum number of linearly independent columns of the matrix, $\kappa(A)$: condition number of the matrix $\sigma_{\max}/\sigma_{\min}$, where $\sigma_{\max}$ and $\sigma_{\min}$ are the largest and smallest nonzero singular values of the matrix, respectively.

TABLE 2
Results for full-rank matrices (underdetermined problems)

Outer iteration	Inner iteration	RANDL1T 0.16		RANDL2T 0.15		RANDL3T 0.15	
AB-GMRES	NE-SOR	11	(16500, 1.1)	29	(101500, 1.0)	107	(267500, 1.0)
		1.36	(0.76)	4.38	(1.70)	7.87	(1.21)
F-AB-GMRES	RK	20.5	(34450, 1.2)	47.5	(265240, 1.3)	232.5	(1074371, 0.9)
		4.07	(2.41)	17.21	(5.67)	49.41	(3.99)
	GRK	20	(5371.5, 1.1)	38.5	(29305, 1.3)	122.5	(83258, 1.3)
		1.01	(0.54)	2.82	(1.04)	5.57	(0.76)
	GK	29	(6553, 1.1)	38	(31080, 1.2)	119	(82532, 1.1)
		⋆0.65	(0.29)	⋆1.60	(0.57)	⋆2.91	(0.41)

Outer iteration	Inner iteration	RANDL4T 0.17		RANDL5T 0.16		RANDL6T 0.16	
AB-GMRES	NE-SOR	72	(180000, 0.9)	199	(497500, 0.9)	132	(330000, 1.1)
		5.85	(1.22)	13.67	(1.28)	9.47	(1.22)
F-AB-GMRES	RK	163	(672860, 1.3)	345.5	(1560800, 1.0)	243	(987859, 1.0)
		33.14	(3.68)	72.7	(4.03)	46.69	(3.55)
	GRK	122	(79897, 1.2)	327.5	(233590, 1.1)	157	(116580, 1.2)
		5.29	(0.73)	13.23	(0.76)	7.35	(0.83)
	GK	86	(53544, 1.2)	264	(189135, 1.2)	149	(118184, 1.3)
		⋆2.07	(0.36)	⋆5.78	(0.40)	⋆3.89	(0.45)

First row: Number of (outer) iterations (the total number of inner iterations, ω).
Second row: Total CPU time, which includes parameter tuning time in parentheses, in seconds.

outer iteration) and relaxation parameter in parentheses. Here, the number of inner iterations means ℓm in Algorithm 2, and ℓ_k in line 4 of Algorithm 7. The second row in each cell gives the total CPU time in seconds including the parameter tuning time and the formation of C outside parentheses, and the parameter tuning time in seconds in parentheses. Here, the CPU time means the median of the elapsed CPU time for 10 times of repeated runs for the same b for the corresponding method, respectively. The $\star$ indicates the fastest method regarding the CPU time. In order to improve computing efficiency for the GK and GRK methods, we used the recursive formula (4) to update the residual vector $s^{(p)}$ for each inner iteration. To do so, we need to compute $C = AA^{\mathsf{T}}$ in advance. The CPU time in seconds for computing matrix C is given below the name of the matrix. The total CPU time for F-AB-GMRES preconditioned by GK and GRK inner iterations includes the time for computing C. We also remark that the matrix A^{T} instead of A was used throughout the programs for efficient data access with MATLAB. (The CPU time required to transpose A is negligible.)

Table 2 shows that F-AB-GMRES preconditioned by the GK inner iterations is the fastest among all the methods for the full-rank matrices RANDLi, $i = 1, 2, \ldots, 6$. We remark that the total number of inner iterations of F-AB-GMRES with GRK and GK is also smaller than that of AB-GMRES with NE-SOR. This

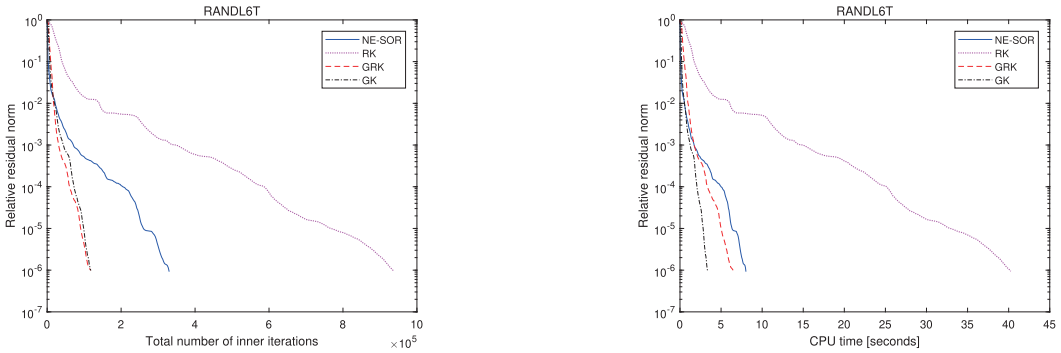

FIG. 1. *Relative residual norm vs. total number of inner iterations (left) and relative residual norm vs. CPU time (right) for RANDL6T.*

TABLE 3
Results for rank-deficient matrices (underdetermined problems)

Outer iteration	Inner iteration	Maragal_3T 0.01		Maragal_4T 0.02		Maragal_5T 0.21	
AB-GMRES	NE-SOR	57	(195624, 1.2)	51	(209508, 1.1)	144	(1898496, 1.2)
		3.05	(0.79)	3.69	(1.08)	44.7	(5.05)
F-AB-GMRES	RK	305	(1352502, 1.0)	179	(1161812, 1.1)	514	(8856928, 0.9)
		48.59	(3.11)	46.71	(4.79)	730.07	(27.05)
	GRK	104.5	(64322, 1.2)	132	(124250, 1.2)	250.5	(578360, 1.1)
		3.50	(0.62)	7.05	(1.01)	73.57	(6.05)
	GK	121	(73165, 1.0)	99	(84543, 1.1)	228	(555285, 1.1)
		⋆1.32	(0.20)	⋆1.67	(0.31)	⋆19.13	(1.44)

First row: Number of (outer) iterations (the total number of inner iterations, ω).
Second row: Total CPU time, which includes parameter tuning time in parentheses, in seconds.

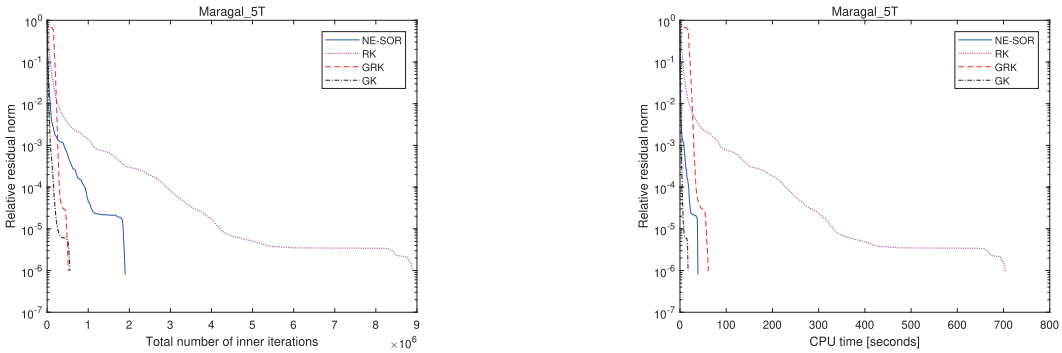

FIG. 2. *Relative residual norm vs. total number of inner iterations (left) and relative residual norm vs. CPU time (right) for Maragal_5T.*

may imply that F-AB-GMRES with GRK and GK has a smaller workload than AB-GMRES with NE-SOR.

In Figure 1, we plot the relative residual norm $\|r_k\|_2/\|b\|_2$ versus total number of inner iterations and CPU time in seconds for the matrix RANDL6T. The figure shows that F-AB-GMRES preconditioned by the GRK inner iterations is best among all the methods regarding the total number of inner iterations, and the GK inner iterations is the fastest among all the methods regarding CPU time. These results are in accordance with Table 2.

Table 3 shows that F-AB-GMRES preconditioned by the GK inner iterations is also the fastest among all the methods for the rank-deficient matrices Maragal_3T, Maragal_4T and Maragal_5T.

In Figure 2, we plot the relative residual norm versus total number of inner iterations and CPU time in seconds for the matrix Maragal_5T. Figure 2 shows that F-AB-GMRES preconditioned by the GRK inner iterations is best among all the methods regarding the total number of inner iterations, and the GK inner iterations is the fastest among all the methods regarding CPU time for the matrix Maragal_5T. These results

TABLE 4
Results for full-rank matrices (overdetermined problems)

Outer iteration	Inner iteration	RANDL1 2.26		RANDL2 2.32		RANDL3 2.24	
AB-GMRES	NE-SOR	2	(10000, 1.0)	2	(10000, 0.9)	2	(10000, 0.7)
		⋆6.10	(3.32)	⋆6.27	(3.43)	6.10	(3.34)
F-AB-GMRES	RK	20	(42352, 0.9)	146	(703920, 0.9)	328.5	(1298700, 0.7)
		16.25	(8.87)	103.41	(13.23)	174.79	(9.59)
	GRK	60	(14281, 1.0)	79.5	(25718, 1.0)	53	(11622, 1.0)
		6.24	(1.26)	8.83	(1.54)	⋆5.81	(1.24)
	GK	220	(71784, 1.0)	383	(178359, 1.0)	469	(140146, 1.0)
		6.81	(0.35)	13.97	(0.95)	12.56	(0.28)

Outer iteration	Inner iteration	RANDL4 2.30		RANDL5 2.35		RANDL6 2.34	
AB-GMRES	NE-SOR	2	(10000, 1.0)	2	(10000, 1.0)	2	(10000, 1.0)
		⋆6.27	(3.44)	6.24	(3.37)	6.26	(3.40)
F-AB-GMRES	RK	278	(1295432, 0.7)	330	(1544889, 0.8)	259.5	(1524497, 1.2)
		176.5	(11.43)	207.13	(11.27)	204.36	(14.46)
	GRK	74.5	(17678, 1.0)	80.5	(21406, 1.0)	53.5	(11422, 1.0)
		7.04	(1.20)	7.9	(1.29)	5.94	(1.27)
	GK	241	(114389, 1.0)	129	(43046, 1.0)	136	(35842, 1.0)
		9.21	(0.32)	⋆5.23	(0.33)	⋆4.81	(0.29)

First row: Number of (outer) iterations (the total number of inner iterations, ω).
Second row: Total CPU time, which includes parameter tuning time in parentheses, in seconds.

TABLE 5
Results for rank-deficient matrices (overdetermined problems)

Outer iteration	Inner iteration	Maragal_3 0.02		Maragal_4 0.02		Maragal_5 0.07	
AB-GMRES	NE-SOR	149	(751854, 1.1)	97	(571524, 1.1)	330	(4607460, 1.1)
		9.35	(1.16)	7.67	(1.33)	78.40	(4.21)
F-AB-GMRES	RK	238	(1303000, 0.9)	165.5	(896606, 1.0)	468	(8781000, 0.9)
		56.60	(4.71)	42.60	(5.08)	906.15	(37.92)
	GRK	262.5	(146302, 1.0)	173.5	(124610, 1.1)	457	(933609, 1.1)
		8.95	(0.71)	8.42	(0.92)	152.79	(6.38)
	GK	253	(156155, 1.0)	200	(161556, 1.1)	520	(1192959, 1.1)
		⋆2.98	(0.24)	⋆3.08	(0.29)	⋆45.46	(1.38)

First row: Number of (outer) iterations (the total number of inner iterations, ω).
Second row: Total CPU time, which includes parameter tuning time in parentheses, in seconds.

are in accordance with Table 3.

4.3. Overdetermined problems. Next, we present numerical experiment results on overdetermined problems ($m > n$). Table 4 and 5 give the numerical experiment results for full-rank matrices and rank-deficient matrices, respectively similarly to Tables 2 and 3.

Table 4 shows that AB-GMRES preconditioned by the NE-SOR inner iterations is the fastest among all the methods for matrices RANDL1, RANDL2 and RANDL4. F-AB-GMRES preconditioned by the GRK inner iterations is the fastest among all the methods for matrices RANDL3, and the GK inner iterations is the fastest among all the methods for matrices RANDL5 and RANDL6.

Table 5 shows that F-AB-GMRES preconditioned by the GK method is the fastest regarding the CPU time among all the methods for matrices Maragal_3, Maragal_4 and Maragal_5.

In Figure 3, we plot the relative residual norm versus the total number of inner iterations and CPU time in seconds for the matrix Maragal_5. Figure 3 shows that F-AB-GMRES preconditioned by the GRK inner iterations is best among all the methods when comparing the total number of inner iterations, and the GK inner iterations is the fastest among all the methods regarding the CPU time for the matrix Maragal_5. These results are in accordance with Table 5.

We have tried to further speed up the methods based on GK and GRK by computing approximations of AA^T using the method in [24] for over- and underdetermined systems, but so far we have not been successful, and this is left for future research.

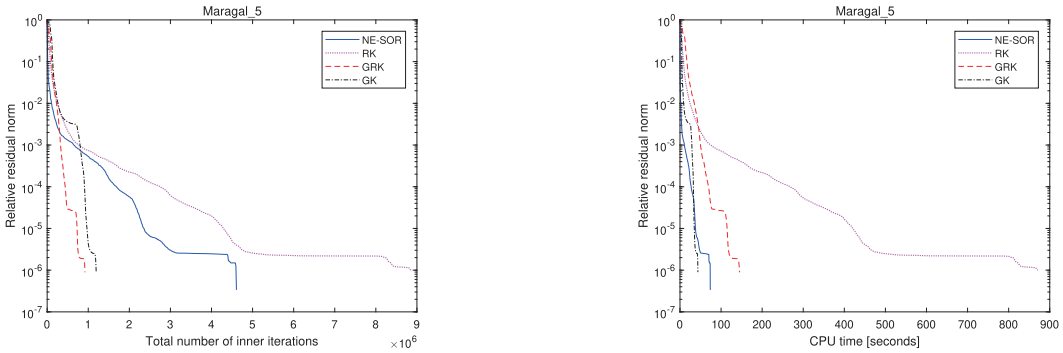

FIG. 3. *Relative residual norm vs. total number of inner iterations (left) and relative residual norm vs. CPU time (right) for Maragal_5.*

5. Conclusion. In this paper, we proposed replacing the NE-SOR method by Kaczmarz-type methods in the previous AB-GMRES method preconditioned by stationary inner iterations for solving consistent systems of linear equations. To do so, we developed a new algorithm called flexible AB-GMRES method preconditioned by Kaczmarz-type methods as inner iterations. An optimality property of minimizing residuals was given for the proposed method. We also proposed a tuning procedure for adjusting the maximum number of inner iterations and value of the relaxation parameter in the method. Numerical experiment results showed that flexible AB-GMRES preconditioned by Kaczmarz-type methods converge faster than the previous method in terms of total CPU time.

Appendix. Comparison of computational work of GK and modified GK inner-iteration preconditioning.

We compare the total computational work (the number of floating-point operations) for the greedy Kaczmarz inner-iteration preconditioning using Algorithm 4 (GK), with GK modified by precomputing and storing $C = AA^{\mathsf{T}}$ once beforehand and updating the residual vector s using equation (4) (MGK).

Let A be an $m \times n$ matrix. Assume that the number of outer iterations of the F-AB-GMRES is k, and that the number of inner Kaczmarz iterations is fixed at ℓ for each outer iteration.

First, consider the case when A is dense. Then, the total work for GK is given by

$$w_{\mathrm{GK}}^{\mathrm{d}} = k\ell(mn + n).$$

The first term corresponds to step 3, and the second corresponds to step 4 of Algorithm 4, respectively. The total work for MGK is

$$w_{\mathrm{MGK}}^{\mathrm{d}} = m^2 n + k\ell(m + n).$$

The first term corresponds to computing $C = AA^{\mathsf{T}}$ once beforehand, the second is for step 4 of Algorithm 4, and the third corresponds to the update in equation (4). Hence,

$$w_{\mathrm{GK}}^{\mathrm{d}} - w_{\mathrm{MGK}}^{\mathrm{d}} = m\left[k\ell(n-1) - mn\right].$$

Therefore,

$$w_{\mathrm{MGK}}^{\mathrm{d}} < w_{\mathrm{GK}}^{\mathrm{d}} \iff k\ell > m\left(1 + \frac{1}{n-1}\right). \tag{10}$$

Next, consider the case when A is sparse and the position of the nonzero elements are random. Let nz be the number of nonzero elements of A. Define $q = \mathrm{nz}/m$ as the average number of nonzero elements per row of A. Thus, the density of A is $d = q/n$. Assume that the computational work to compute $C = (c_{ij}) = AA^{\mathsf{T}}$ is qm^2. Let the density of C be p.

Then, the total work for GK is

$$w_{\mathrm{GK}}^{\mathrm{s}} = k\ell(\mathrm{nz} + q) = k\ell q(m + 1).$$

The first term corresponds to step 3, and the second corresponds to step 4 of Algorithm 4, respectively. The total work for MGK is

$$w^{\mathrm{s}}_{\mathrm{MGK}} = qm^2 + k\ell\,(q + mp)\,.$$

The first term is for computing AA^{T}, the second is for step 4 of Algorithm 4, and the third is for the update in equation (4), respectively. Hence,

$$w^{\mathrm{s}}_{\mathrm{GK}} - w^{\mathrm{s}}_{\mathrm{MGK}} = m\,[k\ell\,(q - p) - mq]\,.$$

Therefore,

$$w^{\mathrm{s}}_{\mathrm{MGK}} < w^{\mathrm{s}}_{\mathrm{GK}} \Longleftrightarrow k\ell > m\left(1 + \frac{p}{q-p}\right). \tag{11}$$

The density p of $C = AA^{\mathsf{T}}$ can be estimated as follows. The probability that $c_{ij} \neq 0$ for $i \neq j$ is $p_{\mathrm{nd}} = 1 - \left(1 - d^2\right)^n$, and the probability that $c_{ii} \neq 0$ is $p_{\mathrm{d}} = 1 - (1-d)^n$. Therefore the probability that $c_{ij} \neq 0$ (or the density of C) is

$$\begin{aligned} p &= \frac{\left(m^2 - m\right) p_{\mathrm{nd}} + m p_{\mathrm{d}}}{m^2} \\ &= 1 - \left(1 + \frac{1}{m}\right)\left(1 - d^2\right)^n + \frac{1}{m}(1-d)^n. \end{aligned} \tag{12}$$

If $d = 1$ (A is dense), (12) implies $p = 1$. Then, also $q = n$, so that (11) agrees with (10). If $d \ll 1$, p can be approximated as

$$p \sim 1 - \left(1 + \frac{1}{m}\right) e^{-nd^2} + \frac{1}{m} e^{-q}.$$

Table 6 gives estimated (using (12)) and actual values of the density of $C = AA^{\mathsf{T}}$ for the matrices used in our experiments.

As for the CPU time, the computation of AA^{T} should perform relatively more efficiently than the flops count suggests, especially for the dense case, due to fast memory access.

Acknowledgement. We would like to thank Professor Zhong-Zhi Bai for stimulating discussions and valuable remarks.

REFERENCES

[1] R. Ansorge, *Connections between the Cimmino-method and the Kaczmarz-method for the solution of singular and regular systems of equations*, Computing, 33 (1984), pp. 367–375.
[2] H. Avron, P. Maymounkov, and S. Toledo, *Blendenpik: Supercharging LAPACK's least-squares solver*, SIAM J. Sci. Comput., 32 (2010), pp. 1217–1236.
[3] Z.-Z. Bai and X.-G. Liu, *On the Meany inequality with applications to convergence analysis of several row-action iteration methods*, Numer. Math., 124 (2013), pp. 215–236.
[4] Z.-Z. Bai and W.-T. Wu, *On convergence rate of the randomized Kaczmarz method*, Linear Algebra Appl., 553 (2018), pp. 252–269.
[5] Z.-Z. Bai and W.-T. Wu, *On greedy randomized Kaczmarz method for solving large sparse linear systems*, SIAM J. Sci. Comput., 40 (2018), pp. A592–A606.
[6] Z.-Z. Bai and W.-T. Wu, *On relaxed greedy randomized Kaczmarz methods for solving large sparse linear systems*, Appl. Math. Lett., 83 (2018), pp. 21–26.
[7] Z.-Z. Bai and W.-T. Wu, *On greedy randomized coordinate descent methods for solving large linear least-squares problems*, Numer. Linear Algebra Appl., 26 (2019), pp. 1–15.
[8] Z.-Z. Bai and W.-T. Wu, *On partially randomized extended Kaczmarz method for solving large sparse overdetermined inconsistent linear systems*, Linear Algebra Appl., 578 (2019), pp. 225–250.
[9] Å. Björck and T. Elfving, *Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations*, BIT, 19 (1979), pp. 145–163.
[10] C. Byrne, *A unified treatment of some iterative algorithms in signal processing and image reconstruction*, Inverse Problems, 20 (2003), pp. 103–120.

TABLE 6
The estimated and actual densities of the matrix $C = AA^{\mathsf{T}}$.

matrix	m	n	d	p(estimated)	p(actual)
RANDL1	5000	500	0.2	1.00	0.745
RANDL2	5000	500	0.2	1.00	0.724
RANDL3	5000	500	0.2	1.00	0.731
RANDL4	5000	500	0.2	1.00	0.714
RANDL5	5000	500	0.2	1.00	0.719
RANDL6	5000	500	0.2	1.00	0.710
Maragal_3	1682	858	1.27×10^{-2}	0.129	0.160
Maragal_4	1964	1027	1.32×10^{-2}	0.163	0.126
Maragal_5	4654	3296	6.10×10^{-3}	0.115	0.0728
RANDL1T	500	5000	0.2	1.00	0.940
RANDL2T	500	5000	0.2	1.00	0.940
RANDL3T	500	5000	0.2	1.00	0.926
RANDL4T	500	5000	0.2	1.00	0.927
RANDL5T	500	5000	0.2	1.00	0.932
RANDL6T	500	5000	0.2	1.00	0.940
Maragal_3T	858	1682	1.27×10^{-2}	0.237	0.562
Maragal_4T	1027	1964	1.32×10^{-2}	0.289	0.669
Maragal_5T	3296	4654	6.10×10^{-3}	0.159	0.461

m: number of rows of A, n: number of columns of A, d: density of nonzero elements of A, p: density of nonzero elements of $C = AA^{\mathsf{T}}$.

[11] C. L. Byrne, *Applied Iterative Methods*, A K Peters, Wellesley, MA, 2008.
[12] Y. Censor, *Row-action methods for huge and sparse systems and their applications*, SIAM Rev., 23 (1981), pp. 444–466.
[13] Y. Censor, *Parallel application of block-iterative methods in medical imaging and radiation therapy*, Math. Program., 42 (1988), pp. 307–325.
[14] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Rev., 43 (2001), pp. 129–159.
[15] E. J. Craig, *The N-step iteration procedures*, J. Math. Phys., 34 (1955), pp. 64–73.
[16] Y. Cui, K. Morikuni, T. Tsuchiya, and K. Hayami, *Implementation of interior-point methods for LP based on Krylov subspace iterative solvers with inner-iteration preconditioning*, Comput. Optim. Appl., 74 (2019), pp. 143–176.
[17] T. A. Davis and Y. Hu, *The University of Florida sparse matrix collection*, ACM Trans. Math. Softw., 38 (2011), pp. 1–25.
[18] P. P. B. Eggermont, G. T. Herman, and A. Lent, *Iterative algorithms for large partitioned linear systems, with applications to image reconstruction*, Linear Algebra Appl., 40 (1981), pp. 37–67.
[19] J. M. Elble, N. V. Sahinidis, and P. Vouzis, *GPU computing with Kaczmarz's and other iterative algorithms for linear systems*, Parallel Comput., 36 (2010), pp. 215–231.
[20] L. Foster, *San Jose State University Singular Matrix Database*, http://www.math.sjsu.edu/singular/matrices/.
[21] A. Galántai, *Projectors and Projection Methods*, Kluwer Academic Publishers, Norwell, MA, 2004.
[22] K. Hayami, J.-F. Yin, and T. Ito, *GMRES methods for least squares problems*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2400–2430.
[23] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp. 409–436.
[24] J. T. Holodnak and I. C. F. Ipsen, *Randomized approximation of the Gram matrix: Exact computation and probabilistic bounds*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 110–137.
[25] S. Kaczmarz, *Angenäherte Auflösung von Systemen Linearer Gleichungen*, Bull. Int. Acad. Polon. Sci. Lett. A., (1937), pp. 355–357.
[26] X. Meng, M. A. Saunders, and M. W. Mahoney, *LSRN: A parallel iterative solver for strongly over- or underdetermined systems*, SIAM J. Sci. Comput., 36 (2014), pp. C95–C118.
[27] K. Morikuni, *Multistep matrix splitting iteration preconditioning for singular linear systems*, Numer. Algorithms, 75 (2017), pp. 457–475.
[28] K. Morikuni and K. Hayami, *Inner-iteration Krylov subspace methods for least squares problems*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1–22.
[29] K. Morikuni and K. Hayami, *Convergence of inner-iteration GMRES methods for rank-deficient least squares problems*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 225–250.
[30] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.
[31] F. Pasqualetti, R. Carli, and F. Bullo, *Distributed estimation via iterative projections with application to power network monitoring*, Automatica, 48 (2012), pp. 747–758.

[32] C. Popa, *Convergence rates for Kaczmarz-type algorithms*, Numer. Algorithms, 79 (2018), pp. 1–17.
[33] V. Rokhlin and M. Tygert, *A fast randomized algorithm for overdetermined linear least-squares regression*, Proc. Natl. Acad. Sci. USA, 105 (2008), pp. 13212–13217.
[34] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.
[35] Y. Saad, *Iterative Methods for Sparse Linear Systems, Second Ed.*, SIAM, Philadelphia, 2003.
[36] R. Southwell, *Relaxation Methods in Engineering Science*, Oxford University Press, Oxford, 1940.
[37] T. Strohmer and R. Vershynin, *A randomized solver for linear systems with exponential convergence*, in Proceedings of Approximation, Randomization and Combinatorial Optimization, Algorithms and techniques, (2006), pp. 499–507.
[38] T. Strohmer and R. Vershynin, *A randomized Kaczmarz algorithm with exponential convergence*, J. Fourier Anal. Appl., 15 (2009), pp. 262–278.
[39] K. Tanabe, *Projection method for solving a singular system of linear equations and its applications*, Numer. Math., 17 (1971), pp. 203–214.