On Kaczmarz method with oblique projection for solving large overdetermined linear systems *

Weiguo Li\textsuperscript{a}, Qifeng Wang\textsuperscript{a}, Wendi Bao\textsuperscript{a}, Li Liu\textsuperscript{a}

\textsuperscript{a}College of Science, China University of Petroleum, Qingdao 266580, P .R. China

Abstract

In this paper, an extension of Kaczmarz method, the Kaczmarz method with oblique projection (KO), is introduced and analyzed. Using this method, a number of iteration steps to solve the over-determined systems of linear equations are significantly reduced, and the the computing time is much saved, especially for those problems that contain some linear equations with near-linear correlation. Simultaneously, a randomized version–randomized Kaczmarz method with oblique projection (RKO) is established. The convergence proofs of these two methods are given and numerical experiments show the effectiveness of the two methods for uniformly distributed random data. Especially when the system has correlated rows, the improvement of experimental results is very prominent.

Keywords: Large linear system, Oblique projection method, Randomized Kaczmarz method, Correlation

2010 MSC: 65H10, 65F20, 65J20

1. Introduction

Consider solving an overdetermined consistent linear system of equations

\[ Ax = b, \]  

where the matrix \( A \in \mathbb{R}^{m \times n} (m \geq n) \), \( b \in \mathbb{R}^{m} \). Denote the rows of \( A \) by \( a_{1}^{T}, a_{2}^{T}, \ldots, a_{m}^{T} \) and let \( b = (b_{1}, b_{2}, \ldots, b_{m})^{T} \). The Kaczmarz method \cite{1} (or the algebraic reconstruction technique (ART) \cite{2}) is one of the most popular solvers. At each iteration, the Kaczmarz method uses the cyclic rule to choose a row of the matrix and projects the current iteration onto the corresponding hyperplane. The convergence rates for Kaczmarz-type algorithms are given by C. Popa \cite{3}. In 2009, Strohmer and Vershynin \cite{4} proved firstly that the randomized Kaczmarz (RK) method converges with the linear rate when the rows are selected randomly with probability proportional to the square of the Euclidean norm of the rows.

As far back as 1954 \cite{5}, the linear convergence of a greedy projection method, known as Motzkin’s method (or the relaxation method) \cite{6}, was proved. This method was also called Kaczmarz method with the ‘maximal-residual control’ in the numerical linear algebra literatures \cite{7, 8}. A discussion on the full history of this method’s linear

\*This research is supported by National Key Research and Development Program of China (grant number 2019YFC1408400), the Fundamental Research Funds for the Central Universities (grant number 19CX05003A-2) and the Fundamental Research Funds for the Central Universities (grant number 18CX02041A)

\*Corresponding author

Email address: baowendi@sina.com (Wendi Bao)
convergence results and this greedy projection method is given in [7] or [9]. There are at least two greedy selection rules: the maximum residual (MR) rule and the maximum distance (MD) rule, respectively:

\[ i_k = \arg\max_i |\langle a_i, x \rangle - b_i| \quad (MR); \quad i_k = \arg\max_i |\langle a_i, x \rangle - b_i|/\|a_i\|_2 \quad (MD) \] (1.2)

where \( i_k \) is the row index that should be selected at the \( k \)th iteration.

With the MR rule, the proof of linear convergence rate was provided by R. Ansorge in [10]. For further developments of Ansorge’s Maximal residual algorithm, we can refer to [11] and [3]. Recently, a series of effective probability criteria for selecting the working rows from the coefficient matrix are introduced. For example, the greedy randomized Kaczmarz (GRK) method [12, 13], the new randomized Kaczmarz (NRK) method [14] are constructed. These methods converge to the unique least-norm solution of the linear system when it is consistent. Theoretical analysis demonstrates that the convergence rates of the GRK and NRK methods are much faster than that of the randomized Kaczmarz method [12, 13, 14].

In the above methods, each iteration is an orthogonal projection. For coherent over-determined systems, the speed of iterative improvement is very slow. So it is very meaningful to introduce an accelerate projection to improve the convergence rate of this type of equations. In this paper, we introduce and analyze an oblique projection method to improve the convergence rate of the original Kaczmarz method.

Here is a short outline of the paper: in Section 2, we introduce a new method–Kaczmarz method with oblique projection (KO) and present its algorithm and its convergence proof. Then, an randomized version of KO method (RKO) for coherent overdetermined systems and its proof of convergence rate are provided in Section 3. In Section 4, we present a variety of numerical experiments for over-determined systems of linear equations with uniformly distributed random data. The final section is devoted to some remarks and conclusions.

2. Kaczmarz Method with Oblique Projection

2.1. Kaczmarz Method with Orthogonal Projection

We use the following notations. \( \|x\| \) is the Euclid norm of \( x \in \mathbb{R}^n \), \( \|A\| = \max_{\|x\|=1} \|Ax\| \) for \( A \in \mathbb{R}^{m \times n} \); \( A^\dagger \) is the Moore-Penrose inverse of \( A \); \( A^T \) is the transpose of \( A \); \( R(A) \) is the range of the matrix \( A \); \( N(A) \) is the null space of the matrix \( A \); \( P_C(x) \) is the orthogonal projection of \( x \) onto \( C \); \( \sigma_{\min}(A) \) is the smallest nonzero singular value of \( A \); \( \bar{x} \) is a solution of (1.1); \( x^* = A^\dagger b \) is the least-norm solution of (1.1).

The Kaczmarz algorithm with orthogonal projection is described as follows.

(Cyclic) Kaczmarz Algorithm

Initialization: \( x(0) \in \mathbb{R}^n \).

Iterative step: for \( k = 0, 1, \cdots \) select \( i_{k+1} = k \ (\text{mod} \ m) + 1 \) and compute \( x^{(k+1)} \) as

\[ x^{(k+1)} = x^{(k)} + \frac{b_{i_{k+1}} - \langle a_{i_{k+1}}, x^{(k)} \rangle}{\|a_{i_{k+1}}\|^2} a_{i_{k+1}}. \] (2.1)

The next result came from Theorem 4 in [3] about the convergence rate of this algorithm.
Lemma 2.1 ([3]) Let \( x^{(0)} \in \mathbb{R}^n \) be an arbitrary initial approximation, \( \tilde{x} \) is a solution of (1.1) such that \( P_{N(A)}(\tilde{x}) = P_{N(A)}(x^{(0)}) \), and the sequence \( \{x^{(k)}\} \) is generated with this Kaczmarz algorithm. Then, there exists a constant \( \delta \in [0, 1) \) such that
\[
\|x^{(k)} - \tilde{x}\|_2 \leq \delta^{m_k},
\]
where \( m_k \) and \( q_k \in \{0, 1, \ldots, m-1\} \) are (uniquely) defined by \( k = m \cdot m_k + q_k \).

In fact, the Kaczmarz method has the following convergence properties (see Theorem 1 in [15]).

Lemma 2.2 ([15]) If (1.1) is consistent, then the sequence \( \{x^{(k)}\} \) generated with the Kaczmarz algorithm converges to a solution of (1.1). If, in addition, \( x^{(0)} \in R(A^T) \), then \( \{x^{(k)}\} \) converges to the least-norm solution of (1.1), i.e.,
\[
\lim_{k \to \infty} x^{(k)} = x^*.
\]

2.2. Kaczmarz Method with Oblique Projection

Consider Kaczmarz method with oblique projection. A question is raised: if we have computed \( x^{(k)} \), how to find the next iteration \( x^{(k+1)} \) which is on the intersection of two hyperplanes? Here we give a simple strategy to find a next iteration \( x^{(k+1)} \) that can converge to \( \tilde{x} \) much quickly, where \( \tilde{x} \) is a solution of the linear system (1.1).

Our Kaczmarz method with oblique projection is described as follows (refer to Fig. 2.1):

Assume that \( x^{(k)} \) is the \( k \)th iteration of solving the systems of equations (1.1), and \( x^{(k)} \) is on the hyperplane \( \langle a_{i_k}, x \rangle = b_{i_k} \). Orthogonal projection from point \( x^{(k)} \) to the hyperplane \( \langle a_{i_{k+1}}, x \rangle = b_{i_{k+1}} \), and get the projection point \( \hat{y}^{(k)} \). And then orthogonal projection from point \( \hat{y}^{(k)} \) to the hyperplane \( \langle a_{i_k}, x \rangle = b_{i_k} \), and get the projection point \( \hat{z}^{(k)} \). Let line \( L \) pass through point \( x^{(k)} \) and along direction \( w^{(i_k)} = \hat{z}^{(k)} - x^{(k)} \), then the intersection of \( L \) and the hyperplane \( \langle a_{i_{k+1}}, x \rangle = b_{i_{k+1}} \) is chosen as the next iteration point \( x^{(k+1)} \) (denote \( x^{(k+1)} \) as the oblique projection point of \( x^{(k)} \) along \( w^{(i_k)} \) to the hyperplane \( \langle a_{i_{k+1}}, x \rangle = b_{i_{k+1}} \)). In the following lemma, we will deduce the iterative formula:
\[
x^{(k+1)} = x^{(k)} + t_k w^{(i_k)},
\]
where \( t_k \) is step size.

Figure 2.1: KO, m=2
Lemma 2.3 Let the step size $t_k$, the direction vector $w^{(i_k)}$, $x^{(k)}$ and $x^{(k+1)}$ be the same as the definitions of the iterative scheme (2.3), then in the KO method, we have

\[
t_k = \frac{b_{k+1} - \langle a_{k+1}, x^{(k)} \rangle}{\langle a_{k+1}, w^{(i_k)} \rangle}, \quad w^{(i_k)} = a_{k+1} - \frac{\langle a_{i_k}, a_{k+1} \rangle}{\langle a_{i_k}, a_{i_k} \rangle} a_{i_k}.
\]

Proof 1 According to the above description of Kaczmarz method with oblique projection and Figure 2.1 and 2.2,

\[
y^{(k)} = x^{(k)} + \frac{b_{k+1} - \langle a_{k+1}, x^{(k)} \rangle}{\|a_{k+1}\|^2} a_{k+1},
\]
and

\[
z^{(k)} = y^{(k)} + \frac{b_{k} - \langle a_{i_k}, y^{(k)} \rangle}{\|a_{i_k}\|^2} a_{i_k}.
\]
Hence,

\[
w^{(i_k)} = z^{(k)} - x^{(k)}
= \frac{b_{k+1} - \langle a_{k+1}, x^{(k)} \rangle}{\|a_{k+1}\|^2} a_{k+1} + \frac{b_{k} - \langle a_{i_k}, y^{(k)} \rangle}{\|a_{i_k}\|^2} a_{i_k}
= \lambda (a_{k+1} + \mu a_{i_k}).
\]

For convenience, the factor $\lambda$ can be omitted, which does not affect the change of the unit direction of $w^{(i_k)}$. According to the orthogonality of $w^{(i_k)}$ and $a_{i_k}$, we get $\mu = -\frac{\langle a_{i_k}, a_{k+1} \rangle}{\langle a_{i_k}, a_{i_k} \rangle}$, then

\[
w^{(i_k)} = a_{k+1} - \frac{\langle a_{i_k}, a_{k+1} \rangle}{\langle a_{i_k}, a_{i_k} \rangle} a_{i_k}.
\]
Taking an inner product on $a_{ik+1}$ with both sides of the equation (2.2) and subtracting $b_{ik+1}$, we get

$$0 = \langle a_{ik+1}, x^{(k)} \rangle - b_{ik+1} + t_k \langle a_{ik+1}, w^{(k)} \rangle,$$

so

$$t_k = \frac{b_{ik+1} - \langle a_{ik+1}, x^{(k)} \rangle}{\langle a_{ik+1}, w^{(k)} \rangle}.$$

The reason why the left equation of (2.4) is equal to 0 is that $x^{(k+1)}$ is on the hyperplane $\langle a_{ik+1}, x \rangle = b_{ik+1}$, i.e. $\langle a_{ik+1}, x^{(k+1)} \rangle = b_{ik+1}$.

**Lemma 2.4** Assume that $\tilde{x}$ is a solution of the linear system (1.1), $w^{(k)}$ is the direction vector from $x^{(k)}$ to $x^{(k+1)}$ in the KO method. Then $w^{(k)}$ and $x^{(k+1)} - \tilde{x}$ are orthogonal, i.e., $\langle w^{(k)}, x^{(k+1)} - \tilde{x} \rangle = 0$.

**Proof 2** On the basis of the description of the KO algorithm (for the three dimensions, we can see Fig. 2.2), $x^{(k+1)}$ is the oblique projection point of $x^{(k)}$ along $w^{(k)}$ to the hyperplane $\langle a_{i+1}, x \rangle = b_{i+1}$. So the points $x^{(k+1)}$ and $\tilde{x}$ are on the hyperplane $\langle a_{i+1}, x \rangle = b_{i+1}$, then $\langle a_{i+1}, x^{(k+1)} - \tilde{x} \rangle = 0$. In addition, the points $x^{(k+1)}$ and $\tilde{x}$ are also on the hyperplane $\langle a_i, x \rangle = b_i$, therefore, $\langle a_i, x^{(k+1)} - \tilde{x} \rangle = 0$. According to the definition of $w^{(k)}$, $\langle w^{(k)}, x^{(k+1)} - \tilde{x} \rangle = 0$.

With Lemma 2.3, the algorithm is described as in Algorithm 2.1. Without losing generality, we assume that all rows of $A$ are not zero vectors.

**Algorithm 2.1 Kaczmarz Method with Oblique Projection (KO)**

Require: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $x^{(0)} \in \mathbb{R}^n$, $K$, $\epsilon > 0$

1: For $i = 1 : m$, $M(i) = \|a_i\|^2$
2: Compute $x^{(1)} = x^{(0)} + \frac{b_i - \langle a_i, x^{(0)} \rangle}{M(1)} a_1$ and set $i_{k+1} = 1$
3: for $k = 1, 2, \cdots, K$ do
4: Set $i_k = i_{k+1}$ and choose a new $i_{k+1}$: $i_{k+1} = \text{mod}(k, m) + 1$
5: Compute $D_{ik} = \langle a_{ik}, x^{(k)} \rangle$ and $r_{ik+1} = b_{ik+1} - \langle a_{ik+1}, x^{(k)} \rangle$
6: Compute $w^{(k)} = a_{ik+1} - \frac{D_{ik}}{M(I_k)} a_{ik}$ and $h_k = \|w^{(k)}\|^2 = M(i_{k+1}) - \frac{D_{ik}}{M(I_k)} D_{ik}$
7: if $h_k > \epsilon$ then
8: $\alpha_k^{(k)} = \frac{r_{ik+1}}{h_k}$ and $x^{(k+1)} = x^{(k)} + \alpha_k^{(k)} w^{(k)}$
9: end if
10: end for
11: Output $x^{(k+1)}$

Assume the system (1.1) is consistent, then it must be $b_i = \lambda b_{ik+1}$ if the two rows of the coefficient matrix $A$ have relation $a_i = \lambda a_{ik+1}$. In this case, the two hyperplanes $\langle a_i, x \rangle = b_i$ and $\langle a_{ik+1}, x \rangle = b_{ik+1}$ are coincident, and we can eliminate one of them without affecting the solution of the equations. So in the following proof, we always assume
that $0 < \theta_{k+1} \leq \pi/2$, here $\theta_{k+1}$ is the angle between any two hyperplanes $\langle a_{i,k}, x \rangle = b_{i,k}$ and $\langle a_{i,k+1}, x \rangle = b_{i,k+1}$. In the Algorithm 2.1,
\[
  h_{i,k} = \frac{1}{\|a_{i,k}\|^2} \|a_{i,k}\|^2 \|a_{i,k}\|^2 (1 - \cos^2(\theta_{k+1})) = \|a_{i,k+1}\|^2 \sin^2(\theta_{k+1}),
\]
thus $h_{i,k} \geq \varepsilon > 0$ ensures $\theta_{k+1} > 0$ because $h_{i,k} = \|a_{i,k+1}\|^2 \sin^2(\theta_{k+1})$.

Before giving the proof of the convergence of the KO algorithm, we first restate the KO algorithm as the following process. For $x^{(0)} \in \mathbb{R}^n$ as an initial approximation we define $x^{(0,0)}, x^{(0,1)}, \ldots, x^{(0,m)} \in \mathbb{R}^n$ by
\[
  \begin{align*}
    x^{(0,0)} &= x^{(0)}, \\
    x^{(0,1)} &= x^{(0)} + \frac{b_1 - \langle a_1, x^{(0)} \rangle}{\|a_1\|^2} a_1, \\
    x^{(0,2)} &= x^{(0,1)} + \frac{b_2 - \langle a_2, x^{(0,1)} \rangle}{\|w^{(1)}\|^2} w^{(1)}, \\
    &\quad \vdots \\
    x^{(0,m-1)} &= x^{(0,m-2)} + \frac{b_{m-1} - \langle a_{m-1}, x^{(0,m-2)} \rangle}{\|w^{(m-1)}\|^2} w^{(m-1)}, \\
    x^{(0,m)} &= x^{(0,m-1)} + \frac{b_m - \langle a_m, x^{(0,m-1)} \rangle}{\|w^{(m)}\|^2} w^{(m)},
  \end{align*}
\]
where
\[
  w^{(i)} = a_{i+1} - \frac{\langle a_{i+1}, a_i \rangle}{\langle a_i, a_i \rangle} a_i, \quad i = 1, \ldots, m - 1, \quad w^{(m)} = a_1 - \frac{\langle a_1, a_m \rangle}{\langle a_1, a_1 \rangle} a_m. \tag{2.6}
\]
For convenience, we denote $a_{m+1} \equiv a_1, b_{m+1} \equiv b_1$. Then, for an arbitrary $p \geq 0$ and a given approximation $x^{(p,m)} \in \mathbb{R}^n$ we construct the new ones $x^{(p+1,0)}, x^{(p+1,2)}, \ldots, x^{(p+1,m)} \in \mathbb{R}^n$ by
\[
  \begin{align*}
    \text{for } i = 1 : m \\
    x^{(p+1,i)} &= x^{(p+1,i-1)} + \frac{b_{i+1} - \langle a_{i+1}, x^{(p+1,i-1)} \rangle}{\|w^{(i)}\|^2} w^{(i)}, \tag{2.7}
  \end{align*}
\]
with the notational convention
\[
  x^{(p+1,0)} = x^{(p,m)}. \tag{2.8}
\]
Obviously, $x^{(k+1)} = x^{(p,i)}$, if $k = p \cdot m + i$, $0 \leq i < m$. The convergence of the KO method is provided as follows.

**Theorem 2.1** Let $x^{(0)} \in \mathbb{R}^n$ be an arbitrary initial approximation, $\bar{x}$ is a solution of (1.1) such that $P_{N(A)}(\bar{x}) = P_{N(A)}(x^{(0)})$, and the sequence $\{x^{(k)}\}_{k=1}^\infty$ is generated with the KO algorithm. Then,
\[
  \lim_{k \to \infty} x^{(k)} = \bar{x}. \tag{2.9}
\]
In addition, if $x^{(0)} \in R(A^T)$, then $\{x^{(k)}\}$ converges to the least-norm solution of (1.1), i.e.,
\[
  \lim_{k \to \infty} x^{(k)} = x^\ast.
\]

**Proof 3** According to (2.5)-(2.7) we obtain the sequence of approximations (from top to bottom and left to right, and
We define the numbers
\[ y^{(p,i)} = b_{i+1} - (a_{i+1}, x^{(p,i-1)}) , \quad i = 1, 2, \cdots, m, \ \forall p \geq 0. \] (2.11)

By using (2.5)-(2.11), we can obtain
\[ x^{(p,i)} = x^{(p,i-1)} + \frac{y^{(p,i)}}{\|w(i)\|^2} w(i), \quad p \geq 0, \ i = 1, 2, \cdots, m. \] (2.12)

With Lemma 2.4 we have
\[ \langle x^{(p,i-1)} - x^{(p,i)}, x^{(p,i)} - \tilde{x} \rangle = 0. \] (2.13)

Therefore, from \( x^{(p,i-1)} - \tilde{x} = x^{(p,i-1)} - x^{(p,i)} + x^{(p,i)} - \tilde{x} \), it is easy to see
\[ \|x^{(p,i-1)} - \tilde{x}\|^2 = \|x^{(p,i-1)} - x^{(p,i)}\|^2 + \|x^{(p,i)} - \tilde{x}\|^2. \]

From (2.12), we get
\[ \|x^{(p,i-1)} - \tilde{x}\|^2 = \|x^{(p,i)} - \tilde{x}\|^2 + \left| \frac{y^{(p,i)}}{\|w(i)\|^2} \right|^2. \] (2.14)

Obviously, the sequence \( \{\|x^{(p,i)} - \tilde{x}\|\}_{p=0}^{m-1} \) is a monotonically decreasing sequence with lower bounds. There exists a \( \alpha \geq 0 \) such that
\[ \lim_{p \to \infty} \|x^{(p,i)} - \tilde{x}\| = \alpha \geq 0, \ \forall i = 0, 1, \cdots, m - 1. \] (2.15)

Thus, from (2.14) and because \( i \) was arbitrary we get
\[ \lim_{p \to \infty} y^{(p,i)} = 0, \ \forall i = 0, 1, \cdots, m - 1. \] (2.16)

Because the sequence \( \{\|x^{(p,i)} - \tilde{x}\|\}_{p=0}^{m-1} \) is bounded, we obtain
\[ \|x^{(p,i)}\| \leq \|\tilde{x}\| + \|x^{(p,i)} - \tilde{x}\| \leq \|\tilde{x}\| + \|x^{(0,1)} - \tilde{x}\|, \ \forall p \geq 0. \] (2.17)

According to the convention (2.17) we get that the sequence \( \{x^{(p,0)}\}_{p=0}^{m} \) is bounded, thus there exists a convergent subsequence \( \{x^{(j,0)}\}_{j=1}^{\infty} \), let’s denote it as
\[ \lim_{j \to \infty} x^{(j,0)} = \hat{x}. \] (2.18)

But, from (2.12) we get
\[ x^{(p,j,1)} = x^{(p,j,0)} - \frac{y^{(p,j,1)}}{\|w(2)\|^2} w(2), \ \forall j > 0. \] (2.19)
thus, by taking the limit following \( j \) and using (2.16), (2.18)

\[
\lim_{j \to \infty} \hat{x}^{(p_j, 1)} = \hat{x}.
\]  

(2.20)

With the same way we obtain

\[
\lim_{j \to \infty} \hat{x}^{(p_j, i)} = \hat{x}, \; \forall \; i = 0, 1, \cdots, m - 1.
\]  

(2.21)

Then, from (2.21) we get for any \( i = 1, \cdots, m \)

\[
\lim_{j \to \infty} \langle x^{(p_j, i-1)}, a_{i+1} \rangle - b_{i+1} = \langle \hat{x}, a_{i+1} \rangle - b_{i+1},
\]  

(2.22)

and from (2.11) and (2.16)

\[
\lim_{j \to \infty} \langle x^{(p_j, i-1)}, a_{i+1} \rangle - b_{i+1} = 0, \; i = 1, \cdots, m.
\]  

(2.23)

Thus, from (2.22) – (2.23) it results in

\[
\langle \hat{x}, a_{i+1} \rangle - b_{i+1} = 0, \; \forall \; i = 1, \cdots, m,
\]  

(2.24)

that is

\[
A\hat{x} = b.
\]  

(2.25)

With the use of the iterative relations

\[
x^{(0,1)} = x^{(0,0)} + \frac{b_1 - \langle a_1, x^{(0,0)} \rangle}{\|w^{(1)}\|^2} w^{(1)},
\]

and

\[
x^{(p, i)} = x^{(p, i-1)} + \frac{r^{(p, i)}}{\|w^{(i)}\|^2} w^{(i)},
\]

\( w^{(i)} \) and \( w^{(1)} \) are defined in (2.6). It is easy to deduce that \( P_{N(A)}(x^{(k)}) = P_{N(A)}(x^{(0)}), \) and so

\[
P_{N(A)}(\hat{x}) = P_{N(A)}(x^{(0)}).
\]  

(2.26)

From the hypothesis of the theorem, we know that

\[
A\hat{x} = b, \; P_{N(A)}(\hat{x}) = P_{N(A)}(x^{(0)}).
\]  

(2.27)

By (2.25) – (2.27), we get

\[
\lim_{j \to \infty} x^{(p_j, i)} - \hat{x} = \lim_{j \to \infty} x^{(p_j, i)} - \hat{x} = 0, \; \forall \; i = 0, 1, \cdots, m - 1.
\]

If we set \( k_j = p_j \cdot m + i, \) then \( \lim_{j \to \infty} ||x^{(k_j)} - \hat{x}|| = 0 \). Based on monotonicity, \( \lim_{k \to \infty} ||x^{(k)} - \hat{x}|| = 0 \), so the sequence \( \{x^{(k)}\} \) is convergent to \( \hat{x} \).

In addition, if \( x^{(0)} \in R(A^T) \), then \( P_{N(A)}(x^{(0)}) = 0 \) and so \( \{x^{(k)}\} \) converges to the least-norm solution of (1.1), i.e.,

\[
\lim_{k \to \infty} x^{(k)} = x^*.
\]  

8
Remark 1. For the Kaczmarz method, it holds \( \|x^{(k+1)} - \tilde{x}\|^2 = \|x^{(k)} - \tilde{x}\|^2 - \frac{r_{ik+1}^2}{\|a_{ik+1}\|^2 \sin^2(\theta_{ik+1})} \), and the KO method holds \( \|x^{(k+1)} - \tilde{x}\|^2 = \|x^{(k)} - \tilde{x}\|^2 - \frac{r_{ik+1}^2}{\|a_{ik+1}\|^2} \). So the KO method is faster than the Kaczmarz method if \( 0 < \theta_{ik+1} < \pi/2 \).

Remark 2. When the coefficient matrix \( A \) is a matrix with orthogonal rows, the KO algorithm degenerates to the Kaczmarz algorithm (right now, \( y^{(k)} = x^{(k)} = x^{(k+1)} \)). Generally, we use the KO method in two ways: one is online mode, and the other is preprocessing mode.

1. Online mode. Each iteration only uses all the information of two adjacent equations, and there is no preprocessing information. Considering that the information of one equation is shared by two adjacent iterations, the KO method takes about \( 10n + 2 \) flops per iteration step. In this case, the Kaczmarz method algorithm needs \( 6n - 1 \) floating-point operations per step.

2. Preprocessing mode. Because the norm of row vector of matrix \( A \), the inner product of two adjacent row vectors, the direction \( w_{ik}^{(k)} \) and its norm are fixed, these can be calculated in advance. After preprocessing (assuming that the above quantities have been calculated), the amount of floating-point number operation of the KO method in each step is \( 4n + 1 \) (only \( r_{ik+1}^{(k)} \) and \( \alpha_{ik}^{(k)} \) need to be calculated). In this case, the workload of the KO method per iterative step is the same as that of the Kaczmarz method. But the total cost of pretreatment for the KO method is about \( 6mn + m \), while that for the Kaczmarz method is \( 2mn - 1 \).

Remark 3. Although the workload of the KO method in each step is more than or equals to that of the Kaczmarz method, compared with the Kaczmarz method, the iteration steps of the KO method are significantly reduced, especially for those problems which contain some linear equations with near linear correlation. See Example 2.1 and numerical experiments in Section 4.

Example 2.1 Consider the following systems of linear equations with two equations

\[
\begin{align*}
\begin{cases}
7x_1 - 8x_2 &= -1, \\
8x_1 - 7x_2 &= 1
\end{cases}
\end{align*}
\]  

(2.28)

and

\[
\begin{align*}
\begin{cases}
7x_1 + 8x_2 &= 15, \\
140x_1 + 159x_2 &= 299
\end{cases}
\end{align*}
\]  

(2.29)

The two equations in system (2.29) are close to correlation. So if the Kaczmarz method is used, 817 steps are needed for the system (2.33) and 940,627 steps are needed for the system (2.34) to reach the error requirement \( \|x^{(k)} - x^*\| \leq \frac{1}{2} \times 10^{-6} \); but with the use of the KO method, both systems need only one step to get the exact solutions.

3. Randomized Kaczmarz Method with Oblique Projection

If the row index \( i_{k+1} \) in Algorithm 2.1 is randomly selected, we get a randomized Kaczmarz method with oblique projection and its convergence as follows. Based on the relationship of the KO and the RKO methods, we can easily prove the expected convergence rate of the RKO method.
Algorithm 3.1 Randomized Kaczmarz Method with Oblique Projection (RKO)

Require: \( A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, x^{(0)} \in \mathbb{R}^n, K, \varepsilon \)

1. For \( i = 1 : m, M(i) = \|a_i\|^2 \)
2. Randomly select \( i_1 \), and compute \( x^{(1)} = x^{(0)} + \frac{b_{i_1} - \langle a_{i_1}, x^{(0)} \rangle}{M(i_1)} a_{i_1} \)
3. Randomly select \( i_2 \neq i_1 \), and compute \( w^{(i_1)} = a_{i_2} - \frac{\langle a_{i_2}, a_{i_1} \rangle}{\|a_{i_1}\|^2} a_{i_1}, x^{(2)} = x^{(1)} + \frac{b_{i_2} - \langle a_{i_2}, x^{(1)} \rangle}{\|w^{(i_1)}\|^2} w^{(i_1)} \)
4. for \( k = 2, 3, \ldots, K \) do
5. Randomly select \( i_{k+1} (i_{k+1} \neq i_k, i_{k-1}) \) (uniformly at random)
6. Compute \( D_{i_k} = \langle a_{i_k}, a_{i_{k+1}} \rangle, r_{i_k} = b_{i_{k+1}} - \langle a_{i_{k+1}}, x^{(k)} \rangle \)
7. Compute \( w^{(i_k)} = a_{i_{k+1}} - \frac{D_{i_k}}{M(i_k)} a_{i_k}, h_k = M(i_{k+1}) - \frac{D_{i_k}}{M(i_k)} D_{i_k} (= \|w^{(i_k)}\|^2) \)
8. if \( h_k > \varepsilon \) then
9. \( \alpha_{i_k} = \frac{r_{i_k}}{h_k} \) and \( x^{(k+1)} = x^{(k)} + \alpha_{i_k} w^{(i_k)} \)
10. end if
11. end for
12. Output \( x^{(k+1)} \)

Lemma 3.1 Let \( x^{(0)} \in R(A^T) \) be an arbitrary initial approximation, \( x^* \) is the least-norm solution of (1.1) \((m \geq 2)\). We select \( i_{k+1} \neq i_k, i_{k-1} \) uniformly at random and compute the next iteration \( x^{(k+1)} = x^{(k)} + r_{i_{k+1}}^{(k)} \frac{w^{(i_k)}}{\|w^{(i_k)}\|^2} \), then we obtain the bound on the following expected conditional on the first \( k \) \((k \geq 2)\) iterations of the RKO method

\[
E_k \frac{r_{i_{k+1}}^{(k)}^2}{\|w^{(i_k)}\|^2} \geq \frac{\sigma_{\min}^2 \|x^{(k)} - x^*\|^2}{(m-2)(\|A\|_F^2 - \sigma_{\min}^2)}
\]

Proof 4 Due to \( x^{(0)} \in R(A^T) \) and \( x^* \) is the least-norm solution of equations (1.1), \( x^{(k)} - x^* \in R(A^T) \).

\[
E_k \frac{r_{i_{k+1}}^{(k)}^2}{\|w^{(i_k)}\|^2} = \frac{1}{m-2} \sum_{s=1, s \neq i_{k-1}}^{m} \frac{|r_{i_k}^{(s)}|^2}{\|w_s\|^2} (r_{i_k} = 0 \text{ and } r_{i_{k-1}} = 0)
\]

\[
\geq \frac{1}{m-2} \sum_{s=1, s \neq i_{k-1}}^{m} \frac{|x^{(s)}|^2}{\|w_s\|^2}
\]

\[
= \frac{1}{m-2} \sum_{s=1}^{m} \frac{|x^{(s)}|^2}{\|w_s\|^2}
\]

\[
= \frac{1}{m-2} \frac{\|b - Ax^{(k)}\|^2}{\|A\|_F^2 - \|w_{i_{k-1}}\|^2} \quad \text{(where } b - Ax^{(k)} = A(x^* - x^{(k)})\})
\]

\[
\geq \frac{\sigma_{\min}^2 \|x^{(k)} - x^*\|^2}{(m-2)(\|A\|_F^2 - \sigma_{\min}^2)}
\]

Since \( x^{(k)} \) is on the intersection of hyperplanes \( \langle a_{i_k}, x \rangle = b_{i_k} \) and \( \langle a_{i_k}, x \rangle = b_{i_{k-1}} \), we have \( r_{i_k} = 0 \) and \( r_{i_{k-1}} = 0 \). Thus the first and second equalities are valid. With \( m \sum_{s=1}^{m} \|w_s\|^2 = \|A\|_F^2 - \sigma_{\min}^2 \), the last equality holds. The first inequality uses the conclusion of \( \frac{\|b_1\|}{\|a_1\|} + \frac{\|b_2\|}{\|a_2\|} \geq \frac{\|b_1\| + \|b_2\|}{\|a_1\| + \|a_2\|} \) (if \( |a_1| > 0, |a_2| > 0 \), and the second one uses the conclusion of \( \|Az\|_2 \geq \sigma_{\min}(A) \|z\|_2 \), if \( z \in R(A^T) \)).
Theorem 3.1 Assume that the system (1.1) is consistent, \( m > 2 \) and \( x^{(0)} \in R(A^T) \). Then the RKO method converges to the least-norm solution of equations (1.1) in expectation and has the following bound

\[
E_k \|x^{(k+1)} - x^*\|^2 \leq \left( 1 - \frac{\sigma^2_{\text{min}}}{(m-2)(\|A\|^2 - \sigma^2_{\text{min}})} \right) \|x^{(k)} - x^*\|^2, \quad k \geq 2.
\]

Proof 5 From lemma 2.4 (see the description of Fig. 2.1 and Fig. 2.2), we know that

\[
\langle x^{(k+1)} - x^*, x^{(k+1)} - x^* \rangle = 0.
\]

Therefore,

\[
\|x^{(k)} - x^*\|^2 = \|x^{(k)} - x^{(k+1)}\|^2 + \|x^{(k+1)} - x^*\|^2.
\]

By

\[
\|x^{(k)} - x^{(k+1)}\|^2 = \|r_k^{(k)}\|_{w_k^{(k)}}^2 = \frac{|x_k^{(k)}|^2}{\|w_k^{(k)}\|^2},
\]

we know that

\[
E_k \|x^{(k+1)} - x^*\|^2 = \|x^{(k)} - x^*\|^2 - E_k \|r_k^{(k)}\|_{w_k^{(k)}}^2 \leq \|x^{(k)} - x^*\|^2 - \|x^{(k+1)} - x^*\|^2 \quad \text{(Lemma 2.3)}
\]

\[
\leq \left( 1 - \frac{\sigma^2_{\text{min}}}{(m-2)(\|A\|^2 - \sigma^2_{\text{min}})} \right) \|x^{(k)} - x^*\|^2.
\]

It yields the desired results.

Remark 4. From Theorem 3.1, we see that the convergence rate of the RKO method is faster than that of the RK method.

4. Numerical Experiments

In this section, we will present some experiment results of the Kaczmarz (K) method, randomized Kaczmarz (RK) method (with uniform probability), the Kaczmarz method with oblique projection (KO) and randomized Kaczmarz method with oblique projection (RKO) for solving the consistent linear system (1.1) with the coefficient matrix \( A \in R^{m \times n} \) from three sources: Gaussian matrix, some real world matrices and Sprand matrix.

In our implementations, the right vector \( b \in R^n \) is chosen such that the exact solution \( x^* \in R^n \) is a vector with all 1’s. Define the relative solution error (RSE) at the \( k \)th iteration as follows:

\[
\text{RSE} = \frac{\|x^{(k)} - x^*\|^2}{\|x^*\|^2}.
\]

The initial point \( x^{(0)} \in R^n \) is set to be a zero vector, and the iterations are terminated once the relative solution error satisfies \( \text{RSE} < 0.5 \times 10^{-6} \) or the number of iteration steps exceeds 100,000. If the number of iteration steps exceeds 100,000, it is denoted as “...”.

We will compare the numerical performance of these methods in terms of the number of iteration steps (denoted as “IT”) and the computing time in seconds (denoted as “CPU(s)”). Here the CPU(s) and IT mean the arithmetical
averages of the elapsed running times and the required iteration steps with respect to 50 trials repeated runs of the corresponding method.

All experiments are carried out by using MATLAB (version R2017b) on a DESKTOP-8CBRR86 with Intel(R) Core(TM) i7-4790, CPU 3.60GHz, RAM 8GB and Windows 10.

**Example 4.1** Gaussian matrix. The Gaussian matrix is randomly generated by using the MATLAB function rand. Consider the linear system (1.1) with $A = \text{rand}(m, 500)$. The numerical results are reported in Table 1. From the table, we can conclude some observations as follows. First, the KO and RKO methods outperform the K and RK methods in terms of the iteration step. The number of iteration steps of the latter two is approximately twice that of the former two respectively. Second, we see that the KO method has an advantage over the K method in CPU. Third, the RKO method requires almost the same iteration steps as the KO method and consumes much less than the other three methods. Finally, we observe that under the same conditions (the same starting vector, the number of rows $m$ and termination condition) the execution time by the KO and RKO methods are quite less than that by the Kaczmarz method and the RK method, respectively.

**Example 4.2** Real world matrix. The real world sparse matrices are taken from [16], which include well-conditioned matrices and ill-conditioned matrices. The properties of different sparse matrices are shown in the Table 2. We list the numbers of IT and the CPU for the four methods in Table 3. The results show that the KO and RKO methods can always successfully compute an approximate solution to the linear system (1.1), but the K and RK methods fail for the matrices WorldCities and well1033 due to the numbers of the iteration steps exceeding 100,000. For all convergent cases, IT and CPU of the KO method are considerably smaller than those of the K method. In the meantime, the RKO method significantly outperforms the RK method in terms of both IT and CPU, too.

| $m \times n$ | Method | K | KO | RK | RKO |
|--------------|--------|---|----|----|-----|
| 1000 $\times$ 500 | IT | 81858 | 23895 | 67857 | 27482 |
|               | CPU  | 0.4886 | 0.2485 | 5.8256 | 0.7393 |
| 2000 $\times$ 500 | IT | 32535 | 8488 | 24504 | 9657 |
|               | CPU  | 0.2008 | 0.1239 | 1.3120 | 0.3843 |
| 3000 $\times$ 500 | IT | 21224 | 8138 | 23645 | 9830 |
|               | CPU  | 0.1390 | 0.1060 | 1.3236 | 0.3587 |
| 4000 $\times$ 500 | IT | 19360 | 8381 | 21313 | 9940 |
|               | CPU  | 0.1357 | 0.1169 | 1.1556 | 0.3682 |
| 5000 $\times$ 500 | IT | 18889 | 8250 | 20319 | 9653 |
|               | CPU  | 0.1355 | 0.1117 | 1.0759 | 0.3907 |
| 6000 $\times$ 500 | IT | 17529 | 8263 | 20129 | 9054 |
|               | CPU  | 0.1367 | 0.1388 | 1.0678 | 0.3623 |
| 7000 $\times$ 500 | IT | 17406 | 8355 | 18108 | 9779 |
|               | CPU  | 0.1562 | 0.1326 | 0.9087 | 0.4008 |
| 8000 $\times$ 500 | IT | 16959 | 8414 | 17575 | 9686 |
|               | CPU  | 0.1315 | 0.1262 | 0.8727 | 0.4167 |
| 9000 $\times$ 500 | IT | 16658 | 8151 | 18122 | 9579 |
|               | CPU  | 0.1234 | 0.1198 | 0.8863 | 0.4064 |

Table 1: IT and CPU of K, KO, RK and RKO for m-by-n matrices $A$ with $n=500$ and different $m$ when the linear system is consistent.
Example 4.3 Uniformly distributed matrix on \([c, 1]\). Consider the consistent linear system (1.1) with uniformly distributed coefficient matrix \(A \in \mathbb{R}^{m \times m}\) on \([c, 1]\), which is generated from the MATLAB function sprand. We perform several experiments to compare IT and CPU of the four methods. All methods are run with the same fixed initial (zero vector) estimate and the fixed matrix. The numerical results are reported in Table 4.

Table 4 shows that the KO and RKO methods can always successfully solve the problem (1.1). The K and RK methods cannot obtain a solution when \(c\) is close to 1, because the number of iteration steps exceeds 100,000. Besides, when all four methods converge, the KO and RKO methods are significantly better than the K and RK methods in terms of iteration step and CPU, respectively. It can be seen from Table 5 that the KO method is significantly better than the K method in the number of iteration steps, and the KO method also requires less CPU to achieve convergence. Also, the RKO method is significantly better than the RK method in terms of iterative steps and CPU. In fact, the CPU required by the RK method is about three times that of the RKO method.

| Method | K | KO | RK | RKO |
|--------|---|----|----|-----|
| name   | IT | CPU| IT | CPU | IT | CPU | IT | CPU |
| Stranke94 | 5878 | 0.0707 | 3846 | 0.0446 | 14020 | 0.4748 | 3517 | 0.0633 |
| Trefethen_20 | 201 | 0.0012 | 111 | 0.0008 | 1186 | 0.0157 | 742 | 0.0114 |
| ash608 | 2652 | 0.0219 | 1705 | 0.0186 | 1211 | 0.0194 | 998 | 0.0211 |
| WorldCities | - | - | 20317 | 0.6395 | - | - | 22257 | 0.8953 |
| well1033 | - | - | 66079 | 5.1150 | - | - | 168887 | 28.9278 |

Table 3: IT and CPU of K, KO, RK and RKO for \(m\)-by-\(n\) matrices \(A\) with different \(m\) and \(n\).
### Table 5: $A$ uniformly distributed on $[0, 1]$.

| Method | K | KO | RK | RKO |
|--------|---|----|----|-----|
| $m = 2000$ | 34125 | 0.1982 | 10414 | 0.1247 | 31238.3 | 1.8202 | 11432 | 0.3663 |
| $m = 4000$ | 21480 | 0.1364 | 8412 | 0.1109 | 22301.3 | 1.1980 | 9293.2 | 0.3618 |
| $m = 6000$ | 18425 | 0.1369 | 8485 | 0.1191 | 20010.8 | 1.0364 | 8960.6 | 0.3649 |
| $m = 8000$ | 16966 | 0.1422 | 8114 | 0.1318 | 19509 | 0.9777 | 8913 | 0.3901 |
| $m = 10000$ | 16302 | 0.1267 | 8347 | 0.1232 | 18599 | 0.9458 | 8741 | 0.3746 |

5. Conclusions

Based on the oblique projections to the hyperplanes, we derive a new extension of the Kaczmarz method. The non-random KO method greatly improves the Kaczmarz method. Compared with the RK method, the Randomized version of oblique projection (RKO) can greatly reduce the number of iterations and running time for solving large-scale overdetermined consistent systems of equations $Ax = b$, especially for the uniformly distributed random data $A$ and $b$. Numerical experiments show the effectiveness of the two methods for uniformly distributed random data.

References

[1] S. Kaczmarz, Angenäherte auflösung von systemen linearer gleichungen, Bull. Internat. Acad. Polon.Sci. Lettres A 29 (1937) 335–357.

[2] R. Gordon, R. Bender, G. T. Herman, Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and x-ray photography, J. Theor. Biol. 29 (1970) 471–481.

[3] C. Popa, Convergence rates for Kaczmarz-type algorithms, Numer. Algor. 79 (2018) 1–17.

[4] T. Strohmer, R. Vershynin, A randomized Kaczmarz algorithm with exponential convergence, J. Fourier Anal. Appl. 15 (2009) 262–278.

[5] S. Agmon, The relaxation method for linear inequalities, Canadian J. Math. 6 (1954) 382–392.

[6] T. S. Motzkin, I. J. Schoenberg, The relaxation method for linear inequalities, Canadian J. Math. 6 (1954) 393–404.

[7] Y. Censor, Row-action methods for huge and sparse systems and their applications, SIAM Rev. 23 (1981) 444–466.

[8] J. Nutini, B. Sepehry, A. Virani, I. Laradji, M. Schmidt, H. Koepke, Convergence Rates for Greedy Kaczmarz Algorithms, UAI (2016).
[9] J. De Loera, J. Haddock, D. Needell, A sampling Kaczmarz-Motzkin algorithm for linear feasibility, SIAM J. Sci. Comput. 39 (2017) S66–S87.

[10] R. Ansorge, Connections between the Cimmino-method and the Kaczmarz-method for the solution of singular and regular systems of equations, Computing 33 (1984) 367–375.

[11] S. Petra, C. Popa, Single projection Kaczmarz extended algorithms, Numer. Algor. 73 (2016) 791–806.

[12] Z. Bai, W. Wu, On greedy randomized Kaczmarz method for solving large sparse linear systems, SIAM J. Sci. Comput. 40 (2018) A592–A606.

[13] Z. Bai, W. Wu, On greedy randomized coordinate descent methods for solving large linear least-squares problems, Numer. Linear Algebr. 26(4) (2019) 1–15.

[14] Y. Guan, W. Li, L. Xing, T. Qiao, A note on convergence rate of randomized Kaczmarz method, Calcolo 57(26) (2020) 1–11.

[15] P. Eggermont, G. Herman, A. Lent, Iterative algorithms for large partitioned linear systems, with applications to image reconstruction, Linear Algebra Appl. 40 (1981) 37–67.

[16] T. Davis, Y. Hu, The university of Florida sparse matrix collection, ACM Trans. Math. Softw. 38(1) (2011) 1–25.