Splitting-based randomized iterative methods for solving indefinite least squares problem

YANJUN ZHANG AND HANYU LI†

College of Mathematics and Statistics, Chongqing University, Chongqing 401331, P.R. China

[Received on 30 March 2022]

The indefinite least squares (ILS) problem is a generalization of the famous linear least squares problem. It minimizes an indefinite quadratic form with respect to a signature matrix. For this problem, we first propose an impressively simple and effective splitting (SP) method according to its own structure and prove that it converges ‘unconditionally’ for any initial value. Further, to avoid implementing some matrix multiplications and calculating the inverse of large matrix and considering the acceleration and efficiency of the randomized strategy, we develop two randomized iterative methods on the basis of the SP method as well as the randomized Kaczmarz, Gauss-Seidel and coordinate descent methods, and describe their convergence properties. Numerical results show that our three methods all have quite decent performance in both computing time and iteration numbers compared with the latest iterative method of the ILS problem, and also demonstrate that the two randomized methods indeed yield significant acceleration in term of computing time.

Keywords: indefinite least squares problem; splitting method; randomized method; Kaczmarz; Gauss-Seidel; coordinate descent.

1. Introduction

The indefinite least squares (ILS) problem was first proposed in Chandrasekaran et al. (1998), whose specific form is as follows:

\[
\text{ILS: } \min_{x \in \mathbb{R}^n} (b - Ax)^T J (b - Ax),
\]

where \(A \in \mathbb{R}^{m \times n}\) with \(m \geq n\), \(b \in \mathbb{R}^m\), and \(J\) is the signature matrix defined as

\[
J = \begin{bmatrix}
I_p & 0 \\
0 & -I_q
\end{bmatrix}, \quad p + q = m.
\]

Here and in the sequel, \(G^T\) denotes the transpose of \(G\) and \(I_t\) is the identity matrix of dimension \(t\). Obviously, the ILS problem will reduce to the standard linear least squares problem when \(q = 0\). However, for \(pq > 0\), the problem (1.1) is to minimize an indefinite quadratic form with respect to the signature matrix \(J\) and its normal equation is:

\[
A^T J Ax = A^T J b.
\]

Considering that the Hessian matrix of the problem (1.1) is \(2A^T J A\), the ILS problem has a unique solution if and only if

\[
A^T J A \quad \text{is symmetric and positive definite (SPD).}
\]

†Corresponding author. Email: lihy.hy@gmail.com or hyli@cqu.edu.cn
Throughout this paper, we assume that the above condition is always true.

The ILS problem has found many applications in some fields, such as the total least squares problems (see, e.g., Golub & Van Loan, 1980; Van Huffel & Vandewalle, 1991) and $H^\infty$ smoothing (Hassibi et al., 1993). Extensive works on computations, perturbation analysis, and applications of this problem have been published (see, e.g., Chandrasekaran et al., 1998; Bojanczyk et al., 2003a; Xu, 2004; Liu & Li, 2011; Liu & Zhang, 2013; Liu & Liu, 2014; Li et al., 2014; Li & Wang, 2018; Diao & Zhou, 2019; Song, 2020; Bojanczyk, 2021). In this paper, we mainly focus on its numerical methods. For the small and dense ILS problem (1.1), Chandrasekaran et al. (1998) designed a stable direct method called the QR-Cholesky method, which first performs the QR factorization of $A$, i.e., $A = QR$ with $Q^TQ = I$ and $R$ being an upper triangular matrix, and then solves $(Q^TJQ)y = Q^TJb$ by using the Cholesky factorization. Finally, the solution is returned by $x = R^{-1}y$. Later, Bojanczyk et al. (2003a) devised a method with a lower operation count than the QR-Cholesky method by using the hyperbolic QR factorization. After that, Xu (2004) proposed to apply the hyperbolic QR factorization to the normalized matrix $A$ to make the algorithm be backward stable. More recently, a unified analysis of the above three methods was made in Bojanczyk (2021). For the large and sparse ILS problem, the direct methods are no longer feasible and hence it is necessary to introduce the iterative methods. Specifically, the preconditioned conjugate gradient methods were first considered in Liu & Li (2011) and Liu & Zhang (2013). Then, Liu & Liu (2014) investigated the block SOR method with a relaxation parameter, which was further improved recently by Song (2020) who presented the USSOR method including two parameters.

1.1 Motivation and contributions

In Song (2020), the author first transformed the normal equation (1.2) into a larger linear system and then introduced the USSOR method with two parameters into the new system for solving the problem (1.1). The method is convergent only under certain conditions, and their numerical results show that different parameters will lead to different results and it is difficult to determine the optimal parameters for large-scale problems. Instead, we propose a splitting (SP) method without parameter for solving the ILS problem (1.1) by fully exploiting the structure of the problem itself and show that the new method converges ‘unconditionally’ in theory. For numerical results, it is also uniformly superior to the USSOR method in Song (2020).

Considering that the SP method needs to compute matrix products and inverse and the cost is prohibitive for large-scale matrices, we transform our splitting iterative scheme into two individual consistent linear subsystems. Then, the randomized Kaczmarz (RK) method (Strohmer & Vershynin, 2009) and the randomized Gauss-Seidel (RGS) method (Leventhal & Lewis, 2010) are applied to solve each subsystem and hence we propose the splitting-based RK-RGS (SP-RK-RGS) method for solving the ILS problem (1.1). It is interesting that the proposed joint randomized iterative method only accesses two columns of matrix in each iteration. Furthermore, the method also allows for opportunities to execute in parallel.

Another interesting finding is that when the indices of the two random columns in each iteration of the SP-RK-RGS method are the same, the joint randomized iterative update will reduce to the randomized coordinate descent (RCD) update (Leventhal & Lewis, 2010). Inspired by this result, we design a splitting-based sampling coordinate descent (SP-SCD) method for the ILS problem (1.1), which can accelerate the SP-RK-RGS method.

1.2 Outline

The paper is organized as follows. We propose the SP method and present its convergence analysis in Section 2. The SP-RK-RGS and SP-SCD methods and their convergence analysis are provided in Sections 3 and 4 respectively. We report extensive numerical results in Section 5. Finally, the concluding remarks of
the whole paper are given in Section 6.

1.3 Notation

For a matrix $G = (G(i, j)) \in \mathbb{R}^{m \times n}$, $G(i, j)$, rank$(G)$, $\sigma_{\text{max}}(G)$, $\sigma_{\text{min}}(G)$, $\|G\|_2$, $\|G\|_F$, and $G^\nu$ denote its $i$th row, $j$th column, rank, largest singular value, smallest nonzero singular value, spectral norm, Frobenius norm, and the restriction onto the row indices in the set $\nu$, respectively. If $G$ is a square matrix, i.e., $m = n$, $\lambda(G)$ stands for an eigenvalue of $G$, and $\rho(G) = \max_{1 \leq i \leq n} |\lambda_i(G)|$ represents its spectral radius; if $G \in \mathbb{R}^{n \times n}$ is SPD, we define the energy norm of any vector $x \in \mathbb{R}^n$ as $\|x\|_G := \sqrt{x^T G x}$. For a vector $z \in \mathbb{R}^n$, $z(j)$ represents its $j$th entry. In addition, we use $e(j)$, $\mathbb{E}^{k-1}$, and $\mathbb{E}$ to denote the $j$th column of the identity matrix $I$, the conditional expectation conditioned on the first $k-1$ iterations, and the full expected value, respectively, and let $[m] := \{1, 2, 3, \ldots, m\}$ for an integer $m \geq 1$. Finally, we partition $A$ and $b$ in the ILS problem (1.1) as

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

where $A_1 \in \mathbb{R}^{p \times n}$, $A_2 \in \mathbb{R}^{q \times n}$, $b_1 \in \mathbb{R}^p$ and $b_2 \in \mathbb{R}^q$.

2. SP method for the ILS problem

From the partition form of $A$ defined in (1.4), we have

$$A^T J A = A_1^T A_1 - A_2^T A_2,$$

and hence (1.2) can be equivalently rewritten as the following linear system

$$(A_1^T A_1 - A_2^T A_2)x = A^T J b,$$

which can be rewritten further as

$$A_1^T A_1 x = A_2^T A_2 x + A^T J b.$$
Remark 2.1 The derivation and iterative scheme of the SP method are very simple and concise. Ordinarily, it should be discovered earlier or should have had bad performance. However, we didn’t find it in any literature on ILS problem and indeed find that its performance in solving the problem (1.1) is quite encouraging; see the numerical results in Section 5 for details.

Remark 2.2 In Algorithm 1, computing step 2 to step 4 needs operation counts of about $pn^2 + 2n^3$, $qn^2 + 2n^3$, and $2mn + 2n^2 - 2n$, respectively, and hence gives the total counts of about $mn^2 + 4n^3 + 2mn + 2n^2 - 2n$. Updating $x_{k+1}$ in step 6 requires about $2n^2$ operation counts in each iteration and hence the total operation counts of the SP method are about $mn^2 + 4n^3 + 2mn + 2n^2 - 2n + 2n^2 \cdot T_{SP}$, where $T_{SP}$ is the number of iterations. This cost is almost the same as the one of the USSOR method in Song (2020); see Section 5.1 for the specific cost of the USSOR method. However, the SP method performs better in computing time and iteration numbers, which is confirmed by the numerical experiments in Section 5. The phenomenon also appears in the SP-SCD method. Some possible reasons are given in Section 5.1.

Now, we present the convergence analysis of the SP method.

Theorem 2.1 For the ILS problem (1.1), the SP method, i.e., Algorithm 1, converges for any initial vector $x_0$.

Proof. Since $A^TJA = A_1^TA_1 - A_2^TA_2$ and $A_1^TA_1$ are SPD, it is easy to see that

$$(A_1^TA_1)^{-\frac{1}{2}}(A_1^TA_1 - A_2^TA_2)(A_1^TA_1)^{-\frac{1}{2}} = I - (A_1^TA_1)^{-\frac{1}{2}}A_2^TA_2(A_1^TA_1)^{-\frac{1}{2}}$$

is also SPD, which implies that the eigenvalues of $(A_1^TA_1)^{-\frac{1}{2}}A_2^TA_2(A_1^TA_1)^{-\frac{1}{2}}$ satisfy

$$0 \leq \lambda \left( (A_1^TA_1)^{-\frac{1}{2}}A_2^TA_2(A_1^TA_1)^{-\frac{1}{2}} \right) < 1.$$

Thus, we have that the spectral radius of the iteration matrix of the SP method is less than 1, i.e.,

$$\rho \left( (A_1^TA_1)^{-1}A_2^TA_2 \right) < 1,$$

which concludes the convergence of the SP method for any initial vector $x_0$. □

Remark 2.3 Theorem 2.1 indicates that the SP method can be seen as an ‘unconditionally’ convergent iterative method. Of course, the initial acknowledged condition (1.3) needs to be satisfied.

3. Splitting-based RK-RGS method for the ILS problem

In the SP method, we need to implement the matrix multiplication $A_1^TA_1$ and compute its inverse $(A_1^TA_1)^{-1}$ and $(A_1^TA_1)^{-1}A_2^TA_2$. For the large-scale ILS problem (1.1), and especially when $p \gg q$, the cost is prohibitive. The extreme case on $p, q$ appears in Minkowski spaces (ˇSego, 2009), where $p = m - 1$ and $q = 1$. To reduce the cost, we will transform (2.1) into two subsystems and then adopt the RK and RGS methods to solve them.

We begin by briefly reviewing the RK and RGS methods, which play a foundational role in our proposed method.
3.1 RK method for linear problem

Consider the consistent linear system

\[ X\beta = y, \]  

(3.1)

where \( X \in \mathbb{R}^{t \times l} \) is a full row rank matrix, \( y \in \mathbb{R}^t \), and \( \beta \) is the \( l \)-dimensional unknown vector. Starting from a vector \( \beta_0 \), the RK method repeats the following two steps in each iteration. First, it chooses a row \( i_k \) of \( X \) with probability proportional to the square of its Euclidean norm, i.e.,

\[ \Pr(\text{row } = i_k) = \frac{\|X(i_k)\|^2_2}{\|X\|^2_F}. \]

Then, it projects the current iteration orthogonally onto the solution hyperplane of that row, i.e.,

\[ \beta_{k+1} = \beta_k + \frac{y(i_k) - X(i_k)\beta_k}{\|X(i_k)\|^2_2} (X(i_k)^T). \]

This randomized method was first investigated in Strohmer & Vershynin (2009). Then, Ma et al. (2015) showed that it converges linearly to the least Euclidean norm solution \( \beta_{LN} = X^T(XX^T)^{-1}y \) of (3.1). Specifically, the iteration \( \beta_k \) satisfies the following expected linear rate:

\[ \mathbb{E} \left[ \|\beta_k - \beta_{LN}\|^2 \right] \leq \left( 1 - \frac{\sigma^2_{\text{min}}(X)}{\|X\|^2_F} \right)^k \|\beta_0 - \beta_{LN}\|^2_2. \]

(3.3)

Later, this convergence rate was further accelerated by using various strategies including block strategies (see, e.g., Needell & Tropp, 2014; Necoara, 2019; Du et al., 2020; Zhang & Li, 2021), greedy strategies (see, e.g., Nutini et al., 2016; Bai & Wu, 2018; Niu & Zheng, 2020; Gower et al., 2021; Zhang & Li, 2022), and others (see, e.g., Lin et al., 2015; Liu & Wright, 2016; Jiao et al., 2017). In addition, the RK method was also extended to many other problems such as the inconsistent problems (see, e.g., Zouzias & Freris, 2013; Wang et al., 2015), the ridge regression problems (see, e.g., Hefny et al., 2017; Liu & Gu, 2019), the feasibility problems (see, e.g., De Loera et al., 2017; Morshed et al., 2020, 2021), etc.

3.2 RGS method for linear problem

Consider the consistent linear system

\[ X\beta = y, \]  

(3.4)

where \( X \in \mathbb{R}^{t \times l} \) is a full column rank matrix, \( y \in \mathbb{R}^t \), and \( \beta \) is the \( l \)-dimensional unknown vector. From an initial vector \( \beta_0 \), the RGS method relies on columns rather than rows in each iteration. Specifically, it first chooses a column \( j_k \) of \( X \) with probability proportional to the square of its Euclidean norm, i.e.,

\[ \Pr(\text{column } = j_k) = \frac{\|X(j_k)\|^2_2}{\|X\|^2_F}. \]

(3.5)
TABLE 1. Summary of convergence properties of the RK and RGS methods for the underdetermined system (3.1) and overdetermined system (3.4), where \( \beta_{LN} \) defined in (3.2) denotes the least Euclidean norm solution of (3.1) and \( \beta^* \) defined in (3.6) is the unique solution of (3.4).

| Method | Underdetermined system (3.1): convergence to \( \beta_{LN} \)? | Overdetermined system (3.4): convergence to \( \beta^* \)? |
|--------|-------------------------------------------------------------|---------------------------------------------------------------|
| RK     | Yes (Ma et al., 2015)                                       | Yes (Strohmer & Vershynin, 2009)                              |
| RGS    | No (Ma et al., 2015)                                        | Yes (Leventhal & Lewis, 2010)                                 |

Then, it updates the iteration

\[
\beta_{k+1} = \beta_k + \frac{X^T_{(j_k)} (y - X\beta_k)}{\|X_{(j_k)}\|^2_2} e_{(j_k)}.
\]

This method was proposed by Leventhal and Lewis (Leventhal & Lewis, 2010). They also proved that the RGS method converges to the unique solution

\[
\beta^* = (X^T X)^{-1} X^T y
\]

of (3.4) with the following linear rate:

\[
\mathbb{E}\left[\|\beta_k - \beta^*\|^2_{X^T X}\right] \leq \left(1 - \frac{\sigma_{\min}^2(X)}{\|X\|^2_F}\right)^k \|\beta_0 - \beta^*\|^2_{X^T X}.
\]

Later, Ma et al. (2015) provided a unified analysis of the RK and RGS methods. Their convergence performance in the above two specific settings are listed in Table 1.

3.3 SP-RK-RGS method for the ILS problem

In the SP method, the update formula (2.1) can be equivalently rewritten as

\[
A_1^T A_1 x_{k+1} = \hat{b},
\]

where \( \hat{b} = A_1^T A_2 x_k + A_1^T J b \). Considering the characteristics of the RK and RGS methods introduced above, we adopt them for solving the following two subsystems of (3.7):

\[
A_1^T w = \hat{b},
\]
\[
A_1^T z = w,
\]

in an alternating way. That is, in each iteration, we implement an iteration of the RK method on (3.8) intertwined with an iteration of the RGS method to solve (3.9). Thus, we only need to select two columns for update in each iteration. The specific algorithm is presented in Algorithm 2.

REMARK 3.1 The RK-RGS update in the SP-RK-RGS method is very like the one for the consistent factorized linear system introduced in Ma et al. (2018). The system is in the following form:

\[
X\beta = y, \quad \text{with} \quad X = UV,
\]
implemented the RK method twice in each iteration for the two subsystems of (3.10):

\[ U_A \]

That is, it needs to access both the columns and rows of \( A \). Thus, for our problem (3.7), the RK-RK method has to choose a column and a row of \( A \) simultaneously. On the contrary, our RK-RGS update only needs to access the columns of \( A \).

**Remark 3.2** In Algorithm 2, the main computations of the inner iteration are in step 7 to step 12, which need operation counts of about \( 2n^2 + (2pn + 6p + 2) \cdot T_{RK-RGS} \), where \( T_{RK-RGS} \) is the number of iterations of the inner RK-RGS update. Updating \( x_{k+1} \) from \( x_k \) needs to compute step 5 to step 13, which requires operation counts of about \( 2n^2 + (2pn + 6p + 2) \cdot T_{RK-RGS} \). Here, we assume that \( T_{RK-RGS} \) is always the same for \( k = 0, 1, \ldots \). Then the total operation counts of the SP-RK-RGS method are about

\[ qn^2 + 2mn - n + (2n^2 + (2pn + 6p + 2) \cdot T_{RK-RGS}) \cdot T_{SP-RK-RGS}, \]

where \( T_{SP-RK-RGS} \) is the number of iterations of the outer update of the SP-RK-RGS method. This cost will be much less than the ones of the SP and USSOR methods when \( m > p > n > q \). Furthermore, like the RK-RK method in \cite{Ma et al. (2018)}, the SP-RK-RGS method can also be accelerated in a parallel computing platform.

In the following, we consider the convergence of Algorithm 2. A preliminary result is first presented as follows.

**Theorem 3.1** Let \( x_k + 1 = (A_T^T A_1)^{-1} \hat{b} \) be the unique solution of (3.7), \( w^* = A_1 (A_T^T A_1)^{-1} \hat{b} \) be the least Euclidean norm solution of (3.8), and \( z^* = (A_T^T A_1)^{-1} A_T^T w^* \) be the unique solution of (3.9). Then solving
(3.8) and (3.9) gets the unique solution of (3.7), i.e.,

\[ z^* = x_{k+1}. \]

Proof. The proof is immediate by considering the expressions of \( x_{k+1}, w^*, \) and \( z^* \).

\[ \square \]

Theorem 3.2 For the ILS problem \((1.1)\), the SP-RK-RGS method, i.e., Algorithm [2], converges for any initial vector \( x_0 \).

Proof. Considering Theorems [2] and [3], to prove the convergence of the SP-RK-RGS method, it suffices to show that the sequence \( \{z_t\} \) generated by the inner iteration, i.e., the RK-RGS update, starting from an initial guess \( z_0 = 0 \), converges to \( z^* \) in expectation.

To the above end, we first set

\[ \tilde{z}_t = z_{t-1} + \frac{A_{I(1)}^T (w^*-A_1z_{t-1})}{\|A_1\|^2_F} e_{(j_2)}. \]

Then

\[ E^{-1} \left[ ||z_t - z^*||^2_{A_1^T A_1} \right] = E^{-1} \left[ ||A_1z_t - A_1z^*||^2_2 \right] \]

\[ = E^{-1} \left[ ||A_1z_t - A_1z^* + A_1\tilde{z}_t - A_1\tilde{z}_t||^2_2 \right] \]

\[ = E^{-1} \left[ ||A_1\tilde{z}_t - A_1z^*||^2_2 \right] + E^{-1} \left[ ||A_1z_t - A_1\tilde{z}_t||^2_2 \right] \]

\[ + 2E^{-1} \left[ \langle A_1\tilde{z}_t - A_1z^*, A_1z_t - A_1\tilde{z}_t \rangle \right]. \]

Next, we show that

\[ E^{-1} \left[ \langle A_1\tilde{z}_t - A_1z^*, A_1z_t - A_1\tilde{z}_t \rangle \right] = 0. \]

From Algorithm [2] and the definition of \( \tilde{z}_t \), it follows that

\[ E^{-1} \left[ \langle A_1\tilde{z}_t - A_1z^*, A_1z_t - A_1\tilde{z}_t \rangle \right] \]

\[ = E^{-1} \left[ \langle A_1z_{t-1} - A_1z^*, \frac{A_{I(1)}^T (w^*-A_1z_{t-1})}{\|A_1\|^2_2}, \frac{A_{I(1)}^T (w_t - w^*)}{\|A_1\|^2_2}, A_{I(1)}^T \rangle \right] \]

\[ = E^{-1} \left[ \langle A_1z_{t-1} - A_1z^*, \frac{A_{I(1)}^T (w_t - w^*)}{\|A_1\|^2_2}, A_{I(1)}^T \rangle \right] \]

\[ + E^{-1} \left[ \langle \frac{A_{I(1)}^T (w^*-A_1z_{t-1})}{\|A_1\|^2_2}, \frac{A_{I(1)}^T (w_t - w^*)}{\|A_1\|^2_2}, A_{I(1)}^T \rangle \right] \]

\[ = \sum_{j_2=1}^{n} \frac{\|A_{I(1)}\|^2_2}{\|A_1\|^2_2} \left[ \langle A_1z_{t-1} - A_1z^*, \frac{A_{I(1)}^T (w_t - w^*)}{\|A_1\|^2_2}, A_{I(1)}^T \rangle \right] \]

\[ + \sum_{j_2=1}^{n} \frac{\|A_{I(1)}\|^2_2}{\|A_1\|^2_2} \left[ \langle A_1z_{t-1} - A_1z^*, \frac{A_{I(1)}^T (w^*-A_1z_{t-1})}{\|A_1\|^2_2}, A_{I(1)}^T (w_t - w^*) \rangle \right] \]

\[ = \langle A_1z_{t-1} - A_1z^*, A_{I(1)}^T (w_t - w^*) \rangle + \langle A_{I(1)}^T (w^*-A_1z_{t-1}), A_{I(1)}^T (w_t - w^*) \rangle, \]
which together with the fact that $A_1 z^* = w^*$ yields

$$\mathbb{E}^{t-1} \left[ \langle A_1 \tilde{z}_t - A_1 z^*, A_1 \tilde{z}_t - A_1 \tilde{z} \rangle \right] = \frac{\langle A_1 \tilde{z}_{t-1} - w^*, A_1 A_1^T (w_t - w^*) \rangle}{\|A_1\|^2_F} + \frac{\langle A_1^T (w^* - A_1 \tilde{z}_{t-1}), A_1^T (w_t - w^*) \rangle}{\|A_1\|^2_F} = 0.$$ 

So, the desired result holds. Therefore, (3.11) is reduced to

$$\mathbb{E}^{t-1} \left[ \|z_t - z^*\|^2_{A_1^T A_1} \right] = \mathbb{E}^{t-1} \left[ \|A_1 \tilde{z}_t - A_1 z^*\|^2 \right] + \mathbb{E}^{t-1} \left[ \|A_1 z_t - A_1 \tilde{z}\|^2 \right].$$

Next, we show that $\|A_1 \tilde{z}_t - A_1 z^*\|^2 = \|A_1 z_{t-1} - A_1 z^*\|^2 - \|A_1 \tilde{z}_t - A_1 \tilde{z}_{t-1}\|^2$. From the update formula of $\tilde{z}_t$, we have

$$A_1 (\tilde{z}_t - z_{t-1}) = \frac{A_1^T (w_t - A_1 \tilde{z}_{t-1})}{\|A_1 (j_2)\|^2_F} A_1 (j_2),$$

which implies that $A_1 (\tilde{z}_t - z_{t-1})$ is parallel to $A_1 (j_2)$. Meanwhile,

$$A_1 (\tilde{z}_t - z^*) = A_1 \left( z_{t-1} - z^* + \frac{A_1^T (j_2) (w^* - A_1 \tilde{z}_{t-1})}{\|A_1 (j_2)\|^2} e (j_2) \right),$$

which together with the fact that $A_1 z^* = w^*$ gives

$$A_1 (\tilde{z}_t - z^*) = \left( I - \frac{A_1 (j_2) A_1^T (j_2)}{\|A_1 (j_2)\|^2} \right) A_1 (z_{t-1} - z^*).$$

Further, we can check

$$A_1^T (j_2) A_1 (\tilde{z}_t - z^*) = A_1^T (j_2) \left( I - \frac{A_1 (j_2) A_1^T (j_2)}{\|A_1 (j_2)\|^2} \right) A_1 (z_{t-1} - z^*) = 0.$$ 

Then $A_1 (\tilde{z}_t - z^*)$ is orthogonal to $A_1 (j_2)$ and hence the vector $A_1 (\tilde{z}_t - z_{t-1})$ is perpendicular to the vector $A_1 (\tilde{z}_t - z^*)$. Thus, by the Pythagorean theorem, we get the desired result

$$\|A_1 \tilde{z}_t - A_1 z^*\|_2 = \|A_1 z_{t-1} - A_1 z^*\|^2_2 - \|A_1 \tilde{z}_t - A_1 \tilde{z}_{t-1}\|^2_2.$$ 

Substituting it into (3.12) leads to

$$\mathbb{E}^{t-1} \left[ \|z_t - z^*\|^2_{A_1^T A_1} \right] = \mathbb{E}^{t-1} \left[ \|A_1 z_{t-1} - A_1 z^*\|^2 \right] - \mathbb{E}^{t-1} \left[ \|A_1 \tilde{z}_t - A_1 \tilde{z}_{t-1}\|^2 \right] + \mathbb{E}^{t-1} \left[ \|A_1 \tilde{z}_t - A_1 \tilde{z}_t\|^2 \right].$$

Thus, by using (3.13), the update rule of $z_t$, and $E^{-1} = E^{-1}_w E^{-1}_t$, we have

$$
E^{-1} \left[ \| z_t - z^* \|_{A_t^{-1}}^2 \right] = \| A_1 z_t - A_1 z^* \|_2^2 - E^{-1}_w \left[ \| A_T^{T(j_2)} (w^* - A_1 z_{t-1}) \|_2 \right]^2 \\
+ E^{-1}_t \left[ \| A_T^{T(j_2)} (w_t - w^*) \|_2 \right]^2 \\
= \| A_1 z_t - A_1 z^* \|_2^2 - \sum_{j_2=1}^n \| A_1 \|_F^2 \left( \frac{A_T^{T(j_2)} (w^* - A_1 z_{t-1})}{\| A_1 \|_2} \right)^2 \\
+ E^{-1}_w E^{-1}_t \left[ \frac{\| A_T^{T(j_2)} (w_t - w^*) \|_2^2}{\| A_1 \|_2^2} \right] \\
= \| A_1 z_t - A_1 z^* \|_2^2 - \frac{\| A_T^{T} (w^* - A_1 z_{t-1}) \|_2^2}{\| A_1 \|_F^2} + E^{-1}_w \left[ \frac{\| A_T^{T} (w_t - w^*) \|_2^2}{\| A_1 \|_F^2} \right].
$$

Further, noting $A_1 z^* = w^*$ and $\| A_T^{T} (A_1 z_t - A_1 z^*) \|_2^2 \geq \sigma_{\text{min}}^2 (A_1) \| A_1 z_t - A_1 z^* \|_2^2$, we get

$$
E^{-1} \left[ \| z_t - z^* \|_{A_t^{-1}}^2 \right] \leq \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right) \| A_1 z_{t-1} - A_1 z^* \|_2^2 + E^{-1}_w \left[ \frac{\| A_T^{T} (w_t - w^*) \|_2^2}{\| A_1 \|_F^2} \right] \\
\leq \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right) \| z_{t-1} - z^* \|_{A_t^{-1}}^2 + \frac{\sigma_{\text{max}}^2 (A_1)}{\| A_1 \|_F^2} E^{-1}_w \left[ \| w_t - w^* \|_2^2 \right],
$$

which together with a result derived from the convergence property of the RK method discussed in (3.3), i.e.,

$$
E \left[ \| w_t - w^* \|_2^2 \right] \leq \left( 1 - \frac{\sigma_{\text{max}}^2 (A_1)}{\| A_1 \|_F^2} \right) \| w^* \|_2^2,
$$

and the law of total expectation, implies

$$
E \left[ \| z_t - z^* \|_{A_t^{-1}}^2 \right] \\
\leq \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right) E \left[ \| z_{t-1} - z^* \|_{A_t^{-1}}^2 \right] + \frac{\sigma_{\text{max}}^2 (A_1)}{\| A_1 \|_F^2} \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right) \| w^* \|_2^2 \\
\leq \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right)^2 E \left[ \| z_{t-2} - z^* \|_{A_t^{-1}}^2 \right] + 2 \frac{\sigma_{\text{max}}^2 (A_1)}{\| A_1 \|_F^2} \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right) \| w^* \|_2^2 \\
\leq \ldots \leq \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right)^t \| z^* \|_{A_t^{-1}}^2 + \| z^* \|_{A_t^{-1}}^2 \left( 1 - \frac{\sigma_{\text{min}}^2 (A_1)}{\| A_1 \|_F^2} \right)^t \| w^* \|_2^2.
$$

This completes the proof.
4. SP-SCD method for the ILS problem

In Algorithm 2, if we set the two columns in each iteration to be the same, i.e., \( j_1 = j_2 = j \), then the inner iteration, i.e., the RK-RGS update, reduces to the RCD update. Specifically,

\[
\begin{align*}
 z_{t+1} &= z_t + \frac{A^T_{1(j)} (w_{t+1} - A_1 z_t)}{\|A_1(j)\|^2_2} e(j) \\
 &= z_t + \frac{A^T_{1(j)} \left( w_t + \hat{b}^{(t)} - \tilde{A}_1^{(j)} w_{t} - A_1 z_t \right)}{\|A_1(j)\|^2_2} e(j) \\
 &= z_t + \frac{\tilde{b}^{(j)} - A^T_{1(j)} A_1 z_t}{\|A_1(j)\|^2_2} e(j),
\end{align*}
\]

where \( \hat{b}^{(j)} - A^T_{1(j)} A_1 z_t \) is the \( j \)-th coordinate of the gradient and \( \left( A^T_{1(j)} A_1 \right)_{(j,j)} \) is its Lipschitz constant. Hence, the above formula can be seen as the CD update for \( \min_1 \frac{1}{2} z^T A^T_1 A_1 z - \tilde{b}^T z \), which has the same solution as the positive definite linear system (3.7) (Leventhal & Lewis, 2010). By the way, the relationship between the RK, RGS and RCD methods was discussed in Hefny et al. (2017) in detail. In particular, the RK and RGS methods can be viewed as different variants of the RCD method.

Based on the above discussions and inspired by De Loera et al. (2017) and Haddock & Ma (2021), similar to the SP-RK-RGS method, we propose the SP-SCD method for solving the ILS problem (1.1). That is, the inner iteration in the SP-RK-RGS method is replaced by the sampling coordinate descent (SCD) update. The specific algorithm is summarized in Algorithm 3.

Remark 4.1 Unlike the SP-RK-RGS method or the SP-RCD method (it is the immediate result of the SP-RK-RGS method with \( j_1 = j_2 = j \)), the probability utilized in the SP-SCD method is adaptive. Specifically, the probability used in Algorithm 3, i.e., \( p(\tau, \beta) \), depends on the value of \( \tilde{A}_{1(i,\tau,\beta)}(\tau, \beta) \), which gives the largest residual value among \( \left( \hat{b}^{(s)} - \tilde{A}_1^{(s)} \beta \right)^2 \) where \( s \in \tau \). In particular, if \( \alpha_t = 1 \), the probability reduces to

\[
p(\tau, \beta) = \frac{\tilde{A}_{1(j,\beta)}(\beta)}{\sum_{\beta \in [p]} \tilde{A}_{1(j,\beta)}(\beta)} = \frac{\|A_{1(j)}\|^2_2}{\|A_1\|^2_2},
\]

which is a fixed probability equivalent to the one of the RGS method listed in (3.5). If \( \alpha_t = n \), the probability reduces to

\[
p(\tau, \beta) = 1,
\]

which is equivalent to grasping the index corresponding to the largest magnitude entry of the residual vector as used in the Motzkin method (Motzkin & Schoenberg, 1954). If \( \tilde{A}_{1(i,j)} = \bar{A}_{1(j,j)} \) for any \( i, j \in [n] \), the
Algorithm 3 SP-SCD method for the ILS problem (1.1).

1: Input: $A, J, b$, and initial estimate $x_0$.
2: Set $\bar{A}_1 = A_1^T A_1$.
3: Set $\bar{A}_2 = A_2^T A_2$.
4: Set $\bar{b} = A^T J b$.
5: for $k = 0, 1, 2, \ldots$ until convergence, do 
6:   Compute $\tilde{b} = \bar{A}_2 x_k + \bar{b}$.
7:   Set $\beta_0 = 0$.
8: for $t = 0, 1, 2, \ldots$ until convergence, do
9:   Generate a positive integer $\alpha_t \in [n]$ at random.
10: Choose an index subset $\tau_t$ of size $\alpha_t$ from among $[n]$ with probability
11:   $p(\tau_t, \beta_t) = \frac{\bar{A}_1(\tilde{s}(\tau_t, \beta_t), s(\tau_t, \beta_t))}{\sum_{\tau \in (\mathbb{Z}_n^\alpha \cap [n])} \bar{A}_1(\tilde{s}(\tau, \beta_t), s(\tau, \beta_t))}$, \hfill (4.1)
12: where $s(\tau, \beta_t) = \arg\max_{s \in \tau} \left( \tilde{b}(s) - \bar{A}_1(s, \beta_t) \right)^2$.
13:   Set $j_t = s(\tau_t, \beta_t)$.
14:   Update $\beta_{t+1} = \beta_t + \frac{\tilde{b}(j_t) - \bar{A}_1(\tilde{s}(j_t), \beta_t)}{\bar{A}_1(j_t, j_t)} \epsilon(j_t)$.
15: end for
16: end for
17: Set $x_{k+1} = \beta_{t+1}$. 


probability reduces to
\[ p(\tau, \beta_k) = \frac{1}{(\eta)} , \]
which is a uniform probability equivalent to the strategy discussed in \cite{DeLoera2017}. In the numerical experiments in Section 5, we mainly consider the last strategy.

Now, we present the convergence analysis for the SP-SCD method.

**Theorem 4.1** For the ILS problem \((1.1)\), the SP-SCD method, i.e., Algorithm 3, converges for any initial vector \(x_0\).

**Proof.** Considering Theorem 2.1 and the assumption \(\beta^* = x_{k+1}\), where \(\beta^*\) is the unique solution of the rewritten form of (3.7), i.e., \(A_1\beta = \hat{b}\), to prove the convergence of the SP-SCD method, we only need to show that the sequence \(\{\beta_k\}\) generated by the inner iteration, i.e., the SCD update, starting from an initial guess \(\beta_0 = 0\), converges to \(\beta^*\) in expectation.

First, from Algorithm 3 we have
\[
\beta_k - \beta^* = \beta_{k-1} - \beta^* + \frac{\hat{b}(j_{k-1}) - \bar{A}_{1}(j_{k-1}) \beta_{k-1}}{A_{1}(j_{k-1}, j_{k-1})}e_{(j_{k-1})} = \beta_{k-1} - \beta^* - \frac{e^T_{(j_{k-1})}(\bar{A}_1 \beta_{k-1} - \hat{b})}{A_{1}(j_{k-1}, j_{k-1})}e_{(j_{k-1})},
\]
which together with the fact \(\bar{A}_1 \beta^* = \hat{b}\) yields
\[
\beta_k - \beta^* = \left( I - \frac{e^T_{(j_{k-1})}e_{(j_{k-1})}}{A_{1}(j_{k-1}, j_{k-1})} \bar{A}_1 \right) (\beta_{k-1} - \beta^*).
\]

Thus, taking the square of the energy norm on both sides, by some algebra, we get
\[
\|\beta_k - \beta^*\|^2_{A_1} = (\beta_k - \beta^*)^T \bar{A}_1 (\beta_k - \beta^*)
\]
\[
= (\beta_{k-1} - \beta^*)^T \left( I - \frac{\bar{A}_1 e_{(j_{k-1})} e_{(j_{k-1})}}{A_{1}(j_{k-1}, j_{k-1})} \right) \bar{A}_1 \left( I - \frac{e^T_{(j_{k-1})}e_{(j_{k-1})}}{A_{1}(j_{k-1}, j_{k-1})} \bar{A}_1 \right) (\beta_{k-1} - \beta^*)
\]
\[
= (\beta_{k-1} - \beta^*)^T \left( \frac{\bar{A}_1 e_{(j_{k-1})}}{A_{1}(j_{k-1}, j_{k-1})} \right) (\beta_{k-1} - \beta^*)
\]
\[
= \|\beta_{k-1} - \beta^*\|^2_{A_1} - \frac{(\beta_{k-1} - \beta^*)^T \bar{A}_1 e_{(j_{k-1})} e_{(j_{k-1})} \bar{A}_1 (\beta_{k-1} - \beta^*)}{A_{1}(j_{k-1}, j_{k-1})}
\]
\[
= \|\beta_{k-1} - \beta^*\|^2_{A_1} - \frac{(e^T_{(j_{k-1})} \bar{A}_1 (\beta_{k-1} - \beta^*))^2}{A_{1}(j_{k-1}, j_{k-1})}
\]
\[
= \|\beta_{k-1} - \beta^*\|^2_{A_1} - \frac{(\bar{A}_1 (j_{k-1}) \beta_{k-1} - \hat{b}(j_{k-1}))^2}{A_{1}(j_{k-1}, j_{k-1})}.
\]
Now, taking expectation of both sides (with respect to $\tau_{-1}$) conditioned on $\beta_{-1}$, we obtain

$$\mathbb{E}_{\tau_{-1}}[\|\beta_t - \beta^*\|_{\hat{A}_1}^2] = \|\beta_t - \beta^*\|_{\hat{A}_1}^2 - \mathbb{E}_{\tau_{-1}}\left[\frac{(\hat{A}_1^{(j,-1)}\beta_{j-1} - \hat{b}^{(j-1)})^2}{A_1^{(j,-1)}}\right]$$

$$= \|\beta_{t-1} - \beta^*\|_{\hat{A}_1}^2 - \sum_{\tau \in (\alpha_{t-1})} p(\tau, \beta_{t-1}) \cdot \frac{(\hat{A}_1^{(j,-1)}\beta_{j-1} - \hat{b}^{(j-1)})^2}{A_1^{(j,-1)}}$$

$$= \|\beta_{t-1} - \beta^*\|_{\hat{A}_1}^2 - \sum_{\tau \in (\alpha_{t-1})} \frac{\hat{A}_1^{(s(\tau, \beta_{t-1}), s(\tau, \beta_{t-1}))}}{\sum_{\upsilon \in (\alpha_{t-1})} \hat{A}_1^{(s(\upsilon, \beta_{t-1}), s(\upsilon, \beta_{t-1}))}} \sum_{\tau \in (\alpha_{t-1})} \|\hat{A}_1^\tau \beta_{t-1} - \hat{b}^\tau\|_m^2$$

which together with

$$\xi_j = \frac{\sum_{\tau \in (\alpha_{t-1})} \left\| \hat{A}_1^\tau \beta_j - \hat{b}^\tau \right\|_2^2}{\sum_{\tau \in (\alpha_{t-1})} \left\| \hat{A}_1^\tau \beta_j - \hat{b}^\tau \right\|_m^2}$$

leads to

$$\mathbb{E}_{\tau_{-1}}[\|\beta_t - \beta^*\|_{\hat{A}_1}^2]$$

$$= \|\beta_{t-1} - \beta^*\|_{\hat{A}_1}^2 - \sum_{\upsilon \in (\alpha_{t-1})} \frac{1}{\sum_{\alpha_{t-1}} \hat{A}_1^{(s(\upsilon, \beta_{t-1}), s(\upsilon, \beta_{t-1}))}} \cdot \frac{1}{\xi_{t-1}} \cdot \sum_{\tau \in (\alpha_{t-1})} \left\| \hat{A}_1^\tau \beta_{t-1} - \hat{b}^\tau \right\|_2^2$$

$$= \|\beta_{t-1} - \beta^*\|_{\hat{A}_1}^2 - \sum_{\upsilon \in (\alpha_{t-1})} \frac{1}{\sum_{\alpha_{t-1}} \hat{A}_1^{(s(\upsilon, \beta_{t-1}), s(\upsilon, \beta_{t-1}))}} \cdot \frac{1}{\xi_{t-1}} \cdot \frac{\binom{n}{\alpha_{t-1}}}{n} \cdot \left\| \hat{A}_1^\tau \beta_{t-1} - \hat{b} \right\|_2^2.$$
Further, noting $\bar{A}_1 \beta^* = \hat{b}$ and $\bar{A}_1 = A_1^T A_1$, we have

$$\mathbb{E}_{\xi_{t-1}}\left[\|\beta_t - \beta^*\|_{A_1}^2\right] = \|\beta_{t-1} - \beta^*\|_{A_1}^2 - \frac{1}{\sum_{u \in \left(\frac{n}{\alpha_{t-1}}\right)} A_1(s(u, \beta_{t-1}), s(u, \beta_{t-1}))} \cdot \frac{1}{\xi_{t-1}} \cdot \frac{n}{\xi_{t-1}} \cdot \|A_1 (\beta_{t-1} - \beta^*)\|_2^2$$

$$\leq \|\beta_{t-1} - \beta^*\|_{A_1}^2 - \frac{1}{\sum_{u \in \left(\frac{n}{\alpha_{t-1}}\right)} A_1(s(u, \beta_{t-1}), s(u, \beta_{t-1}))} \cdot \frac{1}{\xi_{t-1}} \cdot \frac{n}{\xi_{t-1}} \cdot \sigma^2_{\min}(A_1) \|\beta_{t-1} - \beta^*\|_2^2$$

$$\leq \left(1 - \frac{1}{\sum_{u \in \left(\frac{n}{\alpha_{t-1}}\right)} A_1(s(u, \beta_{t-1}), s(u, \beta_{t-1}))} \cdot \frac{1}{\xi_{t-1}} \cdot \frac{n}{\xi_{t-1}} \cdot \sigma^2_{\min}(A_1)\right) \|\beta_{t-1} - \beta^*\|_{A_1}^2.$$ 

Thus, by the law of total expectation, we can obtain

$$\mathbb{E}\left[\|\beta_t - \beta^*\|_{A_1}^2\right] \leq \prod_{j=0}^{t-1} \left(1 - \frac{1}{\sum_{u \in \left(\frac{n}{\alpha_j}\right)} A_1(s(u, \beta_{j}), s(u, \beta_{j}))} \cdot \frac{1}{\xi_j} \cdot \frac{n}{\xi_j} \cdot \sigma^2_{\min}(A_1)\right) \|\beta^*\|_{A_1}^2,$$

which concludes the proof. \qed

**Remark 4.2** Similar to the SP and SP-RK-RGS methods, from Theorem 4.1 it follows that the SP-SCD method also converges ‘unconditionally’.

### 5. Experimental results

In this section, we compare the latest iterative method for the ILS problem, i.e., the USSOR method, with our proposed methods, i.e., the SP, SP-RK-RGS, and SP-SCD methods, in terms of the computing time in seconds (denoted as “CPU”) and the number of iterations (denoted as “IT”). Here, the CPU and IT are arithmetical average quantities with respect to 10 repeated trials of each method. We also use CPU-inner and IT-inner to represent respectively the total inner computing time and iteration numbers of the inner iterations of the SP-RK-RGS and SP-SCD methods. Furthermore, to see the advantage of our proposed SP, SP-RK-RGS and SP-SCD methods over the USSOR method more intuitively, we also present the computing time speed-up of our methods against the USSOR method, which are defined as

$$\text{speed-up-1} = \frac{\text{CPU of USSOR}}{\text{CPU of SP}}, \quad \text{speed-up-2} = \frac{\text{CPU of USSOR}}{\text{CPU of SP-RK-RGS}},$$

and

$$\text{speed-up-3} = \frac{\text{CPU of USSOR}}{\text{CPU of SP-SCD}}.$$

All the computations are obtained by using MATLAB (version R2017a) on a personal computer with 3.00 GHz CPU (Intel(R) Core(TM) i7-9700), 16.0 GB memory, and Windows 10 operating system.
In addition, all the experiments start from an initial vector $x_0 = 0$, and terminate once the relative residual (RR) at $x_k$, defined by

$$RR = \frac{||A^T (Ax_k - b)||_2}{||A^T b||_2},$$

is less than $10^{-6}$, or the number of outer iterations exceeds 20000.

5.1 Computational complexities

Before showing the specific experimental results, we first discuss the computational complexities of the USSOR method listed in Algorithm 4 and our proposed methods, i.e., the SP, SP-RK-RGS and SP-SCD methods.

**Algorithm 4** USSOR method for the ILS problem (1.1) (Song, 2020).

1. Give an initial vector $x^0$, and parameters $\omega$ and $\hat{\omega}$.
2. Set $\bar{b}_1 = A_1^T b_1, R = A_1^T A_2, P = (A_1^T A_1)^{-1}$ and $\tau = \omega + \hat{\omega} - \omega \hat{\omega}$.
3. Compute $\bar{\delta}_1^0 = A_1^T (\bar{b}_1 - A_1 x^0), \delta_2^0 = b_2 - A_2 x^0$.
4. for $k = 1, 2, \ldots$ until convergence, do
5. \hspace{1cm} $\delta_1^{k+1} = \tau \bar{\delta}_1^k + (1 - \omega) \delta_2^k + \omega \tau R P (\delta_1^k - \bar{b}_1) + (1 - \tau) \bar{\delta}_1^k$.
6. \hspace{1cm} $x^{k+1} = (1 - \tau) x^k + P (\tau \bar{b}_1 - \omega (1 - \hat{\omega}) \bar{\delta}_1^k - \omega \hat{\omega} \delta_2^{k+1})$.
7. \hspace{1cm} $\delta_2^{k+1} = (1 - \tau) (A_2 x^k + \bar{\delta}_2^k) - A_2 x^{k+1} + \tau b_2$.
8. end for

In Algorithm 4, the steps 2 and 3 need operation counts of about $mn^2 + 2n^3 + 2pn + 3 - n$ and $2mn + 2pn - n$, respectively, and hence give the total counts of about $mn^2 + 2n^3 + 2mn + 4pn + 3 - 2n$. Determining $\delta_1^{k+1}, x^{k+1}$ and $\delta_2^{k+1}$ needs to compute step 5 to step 7, which requires operation counts of about $2qn + 4n^2 + 3q + 3n + 3, 2n^2 + 6n + 3$, and $4qn + 3q + 1$, respectively, and hence gives the total counts of about $6qn + 6n^2 + 6q + 9n + 7$. Then the total operation counts of the USSOR method are about

$$mn^2 + 2n^3 + 2mn + 4pn + 3 - 2n + (6qn + 6n^2 + 6q + 9n + 7) \cdot T_{USSOR},$$

where $T_{USSOR}$ is the number of iterations of the USSOR method.

For the SP and SP-RK-RGS methods, from Remarks 2.2 and 3.2, we know that they require operation counts of about $mn^2 + 4n^3 + 2mn + 2n^2 - 2n + 2n^2 \cdot T_{SP}$ and $qn^2 + 2mn - n + (2n^2 + (2n + 6p + 2) \cdot T_{RK-RGS}) \cdot T_{SP-RK-RGS}$, respectively. The differences among them and the cost of the USSOR method are also introduced in Remarks 2.2 and 3.2 respectively.

For the SP-SCD method, since the sampling probability (4.1) is computationally prohibitive, we rewrite Algorithm 3 as Algorithm 5 and apply it to the specific experiments. For simplicity for analyzing the computational complexity, we assume that $\alpha = \alpha_{t+1} = \alpha$ for $t = 0, 1, 2, \ldots$ and the inner SCD update has the same iteration numbers $T_{SCD}$ for $k = 0, 1, 2, \ldots$. In this case, the total cost of Algorithm 5 is about

$$mn^2 + 2mn - n + (2n^2 + (2n + 2\alpha + 4) \cdot T_{SCD}) \cdot T_{SP-SCD},$$

where $T_{SP-SCD}$ is the outer iteration numbers of the SP-SCD method. This cost is almost the same as the ones of the USSOR and SP methods. However, the SP-SCD method performs best in numerical experiments. This is because the total operation counts of various methods given above are only approximate and may be far from the accurate ones. One of the contributing factors is the actual iteration numbers. This implies
that the above distinguishing on complexities of the four methods is quite wild and hence may only provide limited suggestions for practical applications.

### Algorithm 5 SP-SCD method for ILS problem (1.1).

1: Input: \( A, J, b \), and initial estimate \( x_0 \).
2: Set \( \bar{A}_1 = A^T_1 A_1, \bar{A}_2 = A^T_2 A_2 \), and \( \bar{b} = A^T J b \).
3: for \( k = 0, 1, 2, \ldots \) until convergence, do
   4: Compute \( \hat{b} = \bar{A}_2 x_k + \bar{b} \).
   5: Set \( \beta_0 = 0 \) and \( r_0 = \hat{b} - \bar{A}_1 \beta_0 \).
6: for \( t = 0, 1, 2, \ldots \) until convergence, do
   7: Generate a positive integer \( \alpha_t \in \{ n \} \) at random.
   8: Choose an index subset of size \( \alpha_t \), \( \tau_t \), uniformly at random from among \( \{ n \} \).
   9: Set \( j_t = \arg \max \{ r(s) \} \).
   10: Update \( r_{t+1} = r_t - \frac{r_s}{A_{t}(j_t)} \bar{A}_1(j_t) \).
   11: Update \( \beta_{t+1} = \beta_t + \frac{r_s}{A_{t}(j_t)} e_j \).
12: end for
13: Set \( x_{k+1} = \beta_{t+1} \).
14: end for

5.2 Examples from Song (2020)

Specifically, we set \( A_1 = \text{rand}(p,n) \), \( A_2 = 7 \times \text{eye}(q,n) \), \( b_1 = \text{rand}(p,1) \), and \( b_2 = \text{rand}(q,1) \). For the optimal parameters of the USSOR method, we obtain them according to Theorem 3.1 in Song (2020). Numerical results on different \( p, q \) and \( n \) are reported in Tables 2 and 3. From these two tables, we can find that our proposed three methods outperform the USSOR method in terms of the iteration numbers and computing time, and the computing time speed-up is at least 3.5294 (see speed-up-1 in Table 2 for the 43000 \( \times \) 13000 matrix). Meanwhile, the SP-RK-RGS and SP-SCD methods are more efficient than the SP method in computing time, and the efficiency of the SP-SCD method is the most remarkable. This is probably mainly because the inner iteration of the SP-SCD method needs fewer iteration numbers and less running time compared with the one of the SP-RK-RGS method.

5.3 Examples from Minkowski spaces

In this case, \( p = m - 1 \) and \( q = 1 \). We consider the same setting as in Section 5.2. That is, \( A_1 = \text{rand}(p,n) \), \( A_2 = 7 \times \text{eye}(1,n) \), \( b_1 = \text{rand}(p,1) \), and \( b_2 = \text{rand}(1,1) \). The optimal parameters of the USSOR method are also computed according to Theorem 3.1 in Song (2020). We report the numerical results on different \( p \) and \( n \) in Tables 4 and 5 which show the similar results obtained in Section 5.2. That is, the SP, SP-RK-RGS and SP-SCD methods outperform the USSOR method in both iteration numbers and CPU time, and the SP-RK-RGS and SP-SCD methods have better performance in computing time.

6. Concluding remarks

In this paper, we propose three ‘unconditionally’ convergent iterative methods, i.e., the SP, SP-RK-RGS, and SP-SCD methods, to solve the ILS problem (1.1). Numerical results show that they all have
TABLE 2. Numerical results of the methods on $p = 30000$ and $q = n$.

| $m \times n$         | 43000 $\times$ 13000 | 44000 $\times$ 14000 | 45000 $\times$ 15000 |
|----------------------|-----------------------|-----------------------|-----------------------|
| USSOR                |                       |                       |                       |
| $\tau$               | 1.0458                | 1.0544                | 1.0645                |
| $\omega$             | 0.5000                | 0.5000                | 0.5000                |
| $\hat{\omega}$       | 1.0917                | 1.1087                | 1.1291                |
| IT                   | 3                     | 4                     | 4                     |
| CPU                  | 1974.0                | 2504.4                | 2938.5                |
| SP                   |                       |                       |                       |
| IT                   | 3.5294                | 4.0452                | 3.5395                |
| CPU                  | 559.3                 | 619.1                 | 830.2                 |
| speed-up-1           | 3.5294                | 4.0452                | 3.5395                |

TABLE 3. Numerical results of the methods on $p = 40000$ and $q = n$.

| $m \times n$         | 53000 $\times$ 13000 | 54000 $\times$ 14000 | 55000 $\times$ 15000 |
|----------------------|-----------------------|-----------------------|-----------------------|
| USSOR                |                       |                       |                       |
| $\tau$               | 1.0206                | 1.0230                | 1.0258                |
| $\omega$             | 0.5000                | 0.5000                | 0.5000                |
| $\hat{\omega}$       | 1.0412                | 1.0460                | 1.0516                |
| IT                   | 3                     | 3                     | 3                     |
| CPU                  | 2581.8                | 2948.1                | 3392.9                |
| SP                   |                       |                       |                       |
| IT                   | 1                     | 1                     | 1                     |
| CPU                  | 4.3095                | 3.8689                | 3.6597                |
| speed-up-1           | 4.3095                | 3.8689                | 3.6597                |

quite decent performance, and the two randomized methods are particularly efficient in computing time. A future work is to consider the splitting-based randomized iterative methods for the large-scale ILS problem with equality constraints (see, e.g., [Bojanczyk et al., 2003b; Liu & Wang, 2010; Mastronardi & Van Dooren, 2000]).
Table 4. Numerical results of the methods in Minkowski spaces with $p = 50000$ and $q = 1$.

|          | $m \times n$ | 50001 $\times$ 13000 | 50001 $\times$ 14000 | 50001 $\times$ 15000 |
|----------|---------------|------------------------|------------------------|------------------------|
| USSOR    | $\tau$       | 1.0040                 | 1.0041                 | 1.0043                 |
|          | $\omega$     | 0.5000                 | 0.5000                 | 0.5000                 |
|          | $\hat{\omega}$ | 1.0080               | 1.0083                 | 1.0085                 |
|          | IT           | 2                      | 2                      | 2                      |
|          | CPU          | 680.7                  | 864.4                  | 1048.8                 |
| SP       | IT           | 1                      | 1                      | 1                      |
|          | CPU          | 566.9453               | 731.0016               | 881.0750               |
|          | speed-up-1   | 1.2006                 | 1.1825                 | 1.1904                 |
| SP-RK-RGS| IT-inner     | $1.0920 \times 10^5$  | $1.1956 \times 10^5$  | $1.1491 \times 10^5$  |
|          | IT           | 1                      | 1                      | 1                      |
|          | CPU-inner    | 237.4219               | 265.5109               | 264.4844               |
|          | CPU          | 466.9094               | 530.9078               | 560.6984               |
|          | speed-up-2   | 1.4579                 | 1.6282                 | 1.8705                 |
| SP-SCD   | IT-inner     | 6749.2                 | 7315.2                 | 7398.8                 |
|          | IT           | 1                      | 1                      | 1                      |
|          | CPU-inner    | 11.7109                | 15.6125                | 16.3172                |
|          | CPU          | 198.2734               | 231.1078               | 265.1656               |
|          | speed-up-3   | 3.4331                 | 3.7402                 | 3.9553                 |

Table 5. Numerical results of the methods in Minkowski spaces with $p = 60000$ and $q = 1$.

|          | $m \times n$ | 60001 $\times$ 13000 | 60001 $\times$ 14000 | 60001 $\times$ 15000 |
|----------|---------------|------------------------|------------------------|------------------------|
| USSOR    | $\tau$       | 1.0031                 | 1.0032                 | 1.0033                 |
|          | $\omega$     | 0.5000                 | 0.5000                 | 0.5000                 |
|          | $\hat{\omega}$ | 1.0063               | 1.0064                 | 1.0066                 |
|          | IT           | 2                      | 2                      | 2                      |
|          | CPU          | 739.6                  | 901.1                  | 1152.7                 |
| SP       | IT           | 1                      | 1                      | 1                      |
|          | CPU          | 633.1                  | 768.9                  | 1019.6                 |
|          | speed-up-1   | 1.1681                 | 1.1719                 | 1.1305                 |
| SP-RK-RGS| IT-inner     | $1.0131 \times 10^5$  | $1.0289 \times 10^5$  | $1.1229 \times 10^5$  |
|          | IT           | 1                      | 1                      | 1                      |
|          | CPU-inner    | 236.9672               | 245.5719               | 279.6359               |
|          | CPU          | 511.2844               | 567.4719               | 687.4875               |
|          | speed-up-2   | 1.4465                 | 1.5879                 | 1.6766                 |
| SP-SCD   | IT-inner     | 5584.7                 | 6046.7                 | 6026.5                 |
|          | IT           | 1                      | 1                      | 1                      |
|          | CPU-inner    | 13.1734                | 14.0109                | 10.6609                |
|          | CPU          | 238.0406               | 273.4203               | 310.2516               |
|          | speed-up-3   | 3.1068                 | 3.2956                 | 3.7153                 |
Funding

The National Natural Science Foundation of China (No. 11671060); The Natural Science Foundation of Chongqing, China (No. cstc2019jcyj-msxmX0267).

REFERENCES

BAI, Z. Z. & WU, W. T. (2018) On greedy randomized Kaczmarz method for solving large sparse linear systems. *SIAM J. Sci. Comput.*, **40**, A592–A606.

BOJANCZYK, A. W., HIGHAM, N. J. & Patel, H. (2003a) Solving the indefinite least squares problem by hyperbolic QR factorization. *SIAM J. Matrix Anal. Appl.*, **24**, 914–931.

BOJANCZYK, A. W., HIGHAM, N. J. & Patel, H. (2003b) The equality constrained indefinite least squares problem: theory and algorithms. *BIT Numer. Math.*, **43**, 505–517.

BOJANCZYK, A. W. (2021) Algorithms for indefinite linear least squares problems. *Linear Algebra Appl.*, **623**, 104–127.

CHANDRASEKARAN, S., GU, M. & SAYED, A. H. (1998) A stable and efficient algorithm for the indefinite linear least-squares problem. *SIAM J. Matrix Anal. Appl.*, **20**, 354–362.

DE LOERA, J. A., HADDOCK, J. & NEEDELL, D. (2017) A sampling Kaczmarz-Motzkin algorithm for linear feasibility. *SIAM J. Sci. Comput.*, **39**, S66–S87.

DIAO, H. A. & ZHOU, T. Y. (2019) Backward error and condition number analysis for the indefinite linear least squares problem. *Int. J. Comput. Math.*, **96**, 1603–1622.

DU, K., SI, W. T. & SUN, X. H. (2020) Randomized extended average block Kaczmarz for solving least squares. *SIAM J. Sci. Comput.*, **42**, A3541–A3559.

GOLUB, G. H. & VAN LOAN, C. F. (1980) An analysis of the total least squares problem. *SIAM J. Numer. Anal.*, **17**, 883–893.

GOWER, R. M., MOLITOR, D., MOORMAN, J. & NEEDELL, D. (2021) On adaptive sketch-and-project for solving linear systems. *SIAM J. Matrix Anal. Appl.*, **42**, 954–989.

HADDOCK, J. & MA, A. (2021) Greed works: an improved analysis of sampling Kaczmarz-Motzkin. *SIAM J. Math. Data Sci.*, **3**, 342–368.

HASSIBI, B., SAYED, A. H. & KAILATH, T. (1993) Recursive linear estimation in Krein spaces. I. Theory. *Proceedings of 32nd IEEE Conference on Decision and Control*. IEEE, IEEE, pp. 3489–3494.

HEFNY, A., NEEDELL, D. & RAMDAS, A. (2017) Rows versus columns: randomized Kaczmarz or Gauss-Seidel for ridge regression. *SIAM J. Sci. Comput.*, **39**, S528–S542.

JIAO, Y. L., JIN, B. T. & LU, X. L. (2017) Preasymptotic convergence of randomized Kaczmarz method. *Inverse Problems*, **33**, 125012.

LEVENTHAL, D. & LEWIS, A. S. (2010) Randomized methods for linear constraints: convergence rates and conditioning. *Math. Oper. Res.*, **35**, 641–654.

LI, H. Y., WANG, S. X. & YANG, H. (2014) On mixed and componentwise condition numbers for indefinite least squares problem. *Linear Algebra Appl.*, **448**, 104–129.

LI, H. Y. & WANG, S. X. (2018) On the partial condition numbers for the indefinite least squares problem. *Appl. Numer. Math.*, **123**, 200–220.

LIN, C., ZANG, J. F. & QING, A. Y. (2015) Extended Kaczmarz algorithm with projection adjustment. *2015 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization*. IEEE, IEEE, pp. 1–3.

LIU, J. & WRIGHT, S. J. (2016) An accelerated randomized Kaczmarz algorithm. *Math Comp.*, **85**, 153–178.

LIU, Q. H. & LI, X. J. (2011) Preconditioned conjugate gradient methods for the solution of indefinite least squares problems. *Calcolo*, **48**, 261–271.

LIU, Q. H. & LIU, A. J. (2014) Block SOR methods for the solution of indefinite least squares problems. *Calcolo*, **51**, 367–379.
LIU, Q. H. & WANG, M. H. (2010) Algebraic properties and perturbation results for the indefinite least squares problem with equality constraints. *Int. J. Comput. Math.*, 87, 425–434.

LIU, Q. H. & ZHANG, F. D. (2013) Incomplete hyperbolic Gram-Schmidt-based preconditioners for the solution of large indefinite least squares problems. *J. Comput. Appl. Math.*, 250, 210–216.

LIU, Y. & GU, C. Q. (2019) Variant of greedy randomized Kaczmarz for ridge regression. *Appl. Numer. Math.*, 143, 223–246.

MA, A., NEEDELL, D. & RAMDAS, A. (2015) Convergence properties of the randomized extended Gauss-Seidel and Kaczmarz methods. *SIAM J. Matrix Anal. Appl.*, 36, 1590–1604.

MA, A., NEEDELL, D. & RAMDAS, A. (2018) Iterative methods for solving factorized linear systems. *SIAM J. Matrix Anal. Appl.*, 39, 104–122.

MASTRONARDI, N. & VAN DOOREN, P. (2014) An algorithm for solving the indefinite least squares problem with equality constraints. *BIT Numer. Math.*, 54, 201–218.

MASTRONARDI, N. & VAN DOOREN, P. (2015) A structurally backward stable algorithm for solving the indefinite least squares problem with equality constraints. *IMA J. Numer. Anal.*, 35, 107–132.

MORSHEID, M. S., ISLAM, M. S. & NOOR-E-ALAM, M. (2020) Accelerated sampling Kaczmarz Motzkin algorithm for the linear feasibility problem. *J. Global Optim.*, 77, 361–382.

MORSHEID, M. S., ISLAM, M. S. & NOOR-E-ALAM, M. (2021) Sampling Kaczmarz-Motzkin method for linear feasibility problems: generalization and acceleration. *Math. Program.*, 1–61.

MOTZKIN, T. S. & SCHOLENBERG, I. J. (1954) The relaxation method for linear inequalities. *Canad. J. Math.*, 6, 393–404.

NECOARA, I. (2019) Faster randomized block Kaczmarz algorithms. *SIAM J. Matrix Anal. Appl.*, 40, 1425–1452.

NEEDELL, D. & TROPP, J. A. (2014) Paved with good intentions: analysis of a randomized block Kaczmarz method. *Linear Algebra Appl.*, 441, 199–221.

NIU, Y. Q. & ZHENG, B. (2020) A greedy block Kaczmarz algorithm for solving large-scale linear systems. *Appl. Math. Lett.*, 104, 106294.

NUTINI, J., SEPEHRY, B., VIRANI, A., LARADJI, I., SCHMIDT, M. & KOEKPE, H. (2016) Convergence rates for greedy Kaczmarz algorithms. *32nd Conference on Uncertainty in Artificial Intelligence*. AUAI Press, AUAI Press.

ŠEGO, V. (2009) Two-Sided Hyperbolic Singular Value Decomposition. *Ph.D. thesis*, Department of Mathematics, University of Zagreb.

SONG, J. (2020) USSOR method for solving the indefinite least squares problem. *Int. J. Comput. Math.*, 97, 1781–1791.

STROHMER, T. & VERSHYNNIN, R. (2009) A randomized Kaczmarz algorithm with exponential convergence. *J. Fourier Anal. Appl.*, 15, 262–278.

VAN HUFFEL, S. & VANDEWALLE, J. (1991) *The Total Least Squares Problem: Computational Aspects and Analysis*. Philadelphia: SIAM.

WANG, C., AGASKAR, A. & LU, Y. M. (2015) Randomized Kaczmarz algorithm for inconsistent linear systems: an exact MSE analysis. *2015 International Conference on Sampling Theory and Applications*. IEEE, IEEE, pp. 498–502.

XU, H. G. (2004) A backward stable hyperbolic QR factorization method for solving indefinite least squares problem. *J. Shanghai Univ.*, 8, 391–396.

ZHANG, Y. J. & LI, H. Y. (2021) Block sampling Kaczmarz-Motzkin methods for consistent linear systems. *Calcolo*, 58, 1–20.

ZHANG, Y. J. & LI, H. Y. (2022) Greedy Motzkin-Kaczmarz methods for solving linear systems. *Numer. Linear Algebra Appl.*, 29, e2429.

ZOUZIAS, A. & FERRIS, N. M. (2013) Randomized extended Kaczmarz for solving least squares. *SIAM J. Matrix Anal. Appl.*, 34, 773–793.