Numerical Algorithms
https://doi.org/10.1007/s11075-025-02192-9

RESEARCH

Krylov iterative methods for linear least squares problems with linear equality constraints

Haibo Li[1]

Received: 27 January 2025 / Accepted: 28 July 2025

Abstract

We consider the linear least squares problem with linear equality constraints (LSE problem) formulated as $\mathbf{min}_{x\in\mathbb{R}^n}\|\boldsymbol{Ax}-\boldsymbol{b}\|_2$ **s.t.** $\boldsymbol{Cx}=\boldsymbol{d}$. Although there are some classical methods available to solve this problem, most of them rely on matrix factorizations or require the null space of $\boldsymbol{C}$, which limits their applicability to large-scale problems. To address this challenge, we present a novel analysis of the LSE problem from the perspective of operator-type least squares (LS) problems, where the linear operators are induced by $\{\boldsymbol{A}, \boldsymbol{C}\}$. We show that the solution of the LSE problem can be decomposed into two components, each corresponding to the solution of an operator-form LS problem. Building on this decomposed-form solution, we propose two Krylov subspace based iterative methods to approximate each component, thereby providing an approximate solution of the LSE problem. Several numerical examples are constructed to test the proposed iterative algorithm for solving the LSE problems, which demonstrate the effectiveness of the algorithms.

Keywords Linear least squares · Linear equality constraints · Decomposed-form solution · Krylov subspace · Golub-Kahan bidiagonalization · Null space restricted LSQR

Mathematics Subject Classification (2010) 15A09 · 65F10 · 65F20

1 Introduction

The linear least squares problem with equality constraints (LSE problem) arises frequently in various fields such as data fitting, signal processing, control systems and optimization [1–4]. These problems involve minimizing a least squares objective

✉ Haibo Li
haibo.li@unimelb.edu.au

1 School of Mathematics and Statistics, The University of Melbourne, Parkville, Melbourne, VIC 3010, Australia

function while ensuring that a set of linear equality constraints is satisfied. Generally, the LSE problem aims to find the minimizer of the following problem:

$$\min_{x\in\mathbb{R}^n} \|Ax - b\|_2 \;\text{ s.t. }\; Cx = d, \tag{1.1}$$

where $A \in \mathbb{R}^{m\times n}$, and $C \in \mathbb{R}^{p\times n}$. It restricts the solution space to the set of solutions that satisfy both the least squares objective and the linear equality constraints, which is often used in cases where certain relationships between the variables are known a priori and must be preserved. The LSE problem (1.1) has a solution if and only if $Cx = d$ is consistent, and it has a unique solution if and only if $(A^\top, C^\top)^\top$ has full column rank. There is a large amount of work on the analysis of the LSE problem; see e.g. [5–9].

Despite their wide applicability, solving large-scale LSE problems efficiently remains a significant computational challenge. Classical solution approaches typically reduce the constrained LSE problem to an equivalent unconstrained problem by eliminating the constraints. The key strategy of these methods are the constraint substitution technique, which eliminates the constraints by reducing the dimension of the problem. The first one is usually called the null space method [10–13]. This method involves finding a null space basis for the matrix C using a rank-revealing QR factorization [14–16]. The constraints are then incorporated into the LS problem by substituting this basis into the system, leading to a reduced, unconstrained problem of lower dimension. This approach provides numerical stability and is widely used in many practical settings. The second one is usually called the direct elimination method [11]. In this method, a substitution is made directly by expressing certain solution components (those affected by the constraints) in terms of others. This can be accomplished using a pivoted LU factorization or a rank-revealing QR factorization of C [17]. The direct elimination method exhibits good numerical stability and efficiency, particularly when implemented with appropriate matrix factorizations.

In addition to constraint substitution methods, there are some other methods that transform the constrained LS problem to an unconstrained optimization problem. The method based on the Lagrange multiplier formulation [17–19] is often useful. This approach introduces auxiliary variables (Lagrange multipliers) to incorporate the constraints into the optimization process, which constructs an augmented system by combining the linear constraints and the LS problem, and both can be solved simultaneously. This method provides a powerful and general way to enforce equality constraints during the optimization. Techniques like weighting and updating procedures can also be used to enforce constraints progressively, ensuring that the solution satisfies the constraints a posteriori [20–23].

All the above methods, when implemented correctly, can provide a solution with satisfied accuracy. However, in many practical scenarios, the problem size can be very large. In such cases, matrix factorization-based methods become impractical due to their cubic scaling computational complexity. This highlights the need to develop new iterative methods for solving the LSE problem that do not rely on matrix factorizations. The Krylov subspace method is well-known for its effectiveness in solving linear systems, including linear equations and LS problems, where only matrix- vector multiplications are required during the iteration process [17, 24]. However, up to now, there is a lack of Krylov iterative methods specifically for the LSE problem, possibly due to an incomplete understanding of its properties. Establishing connections between the LSE and LS problems could be valuable, as it would aid in the development of efficient Krylov iterative methods for solving the LSE problem.

In this paper, we present a novel analysis of the LSE problem from the perspective of operator-type LS problems. Building on this framework, we propose two Krylov subspace based iterative methods for solving LSE problems. To this end, we construct two linear operators using the matrices $\{A, C\}$ and formulate two LS problems associated with these operators. Using these formulations, we investigate the structure of the solutions to the LSE problem and show that its minimum 2-norm solution can be decomposed into two components, each corresponding to the solution of one of the operator-based LS problems. Building on this connection, we derive two types of decomposed-form solution for the LSE problem. To approximate the solution, it is sufficient to solve the associated operator-form LS problems using the Golub-Kahan bidiagonalization process [25–28]. This approach leads to Krylov subspace based iterative procedures. Consequently, we develop two Krylov iterative methods for the LSE problem, each corresponding to solving one of the decomposed-form solutions. The proposed algorithms do not rely on any matrix factorizations. Instead, they follow an inner-outer iteration structure, where, at each outer iteration, an inner subproblem is approximately solved. We also propose a procedure for constructing LSE problems for testing purposes and present several numerical examples to illustrate the effectiveness of the proposed algorithms.

The paper is organized as follows. In Section 2, we review three commonly used methods for the LSE problem. In Section 3, we analyze the LSE problem from the perspective of operator-type LS problems and derive two types of decomposed-form solution. In Section 4 we proposed two Krylov subspace based iterative algorithms for approximating the decomposed-form solution. Numerical experiments are presented in Section 5, and concluding remarks follow in Section 6.

Throughout the paper, we denote by $\mathcal{N}(\cdot)$ and $\mathcal{R}(\cdot)$ the null space and range space of a matrix or linear operator, respectively, denote by I and $\mathbf{0}$ the identity matrix and zero matrix/vector with orders clear from the context, and denote by span$\{\cdot\}$ the subspace spanned by a group of vectors or columns of a matrix. We use $\mathcal{P}_{\mathcal{S}}$ to denote the orthogonal operator onto a closed linear subspace $\mathcal{S}$.

2 LSE problem and its computation

We review three classical methods for the LSE problem: the null space approach, the method of direct elimination, and the augmented system approach. To simplify the presentation, we assume in this section that C has full row rank.

The null space method was developed and discussed by a number of authors in the 1970s. Suppose the dimension of $\mathcal{N}(C)$ is t and the columns of $Z \in \mathbb{R}^{n\times t}$ form a basis for $\mathcal{N}(C)$. The basic idea is that any vector $x \in \mathbb{R}^n$ satisfying the linear constraint $Cx = d$ can be written as $x = x_0 + Zy$, where x_0 is a particular solution of $Cx = d$.

Let $P \in \mathbb{R}^{n\times n}$ be a permutation matrix representing the pivoting, such that the QR factorization of CP is

$$CP = Q\begin{array}{cc} \begin{pmatrix} R & \mathbf{0} \end{pmatrix} & p, \\ \begin{matrix} p & n-p \end{matrix} & \end{array} \tag{2.1}$$

where $Q \in \mathbb{R}^{p\times p}$ is an orthogonal matrix and R is a nonsingular upper triangular matrix. Now we can get a solution of $Cx = d$:

$$x_0 = P\begin{pmatrix} R^{-1}Q^{\mathsf{T}}d \\ \mathbf{0} \end{pmatrix}\begin{matrix} p \\ n-p \end{matrix} \tag{2.2}$$

Now the LSE problem (1.1) becomes

$$\min_{y\in\mathbb{R}^t} \|AZy - (b - Ax_0)\|_2 \,. \tag{2.3}$$

By solving the above standard LS problem to get the solution $y^\dagger$, we get a solution $x^\dagger = x_0 + Zy^\dagger$ to the LSE problem.

In the null space method, the matrix Q should be stored explicitly or implicitly (by using e.g, Householder transformations), leading to a relatively high memory demands and implied operation counts. The more challenging point is that the matrix Z is usually dense, which makes it inefficient to solve the LS problem (2.6). Also, in recent years, there have been some works about constructing a sparse null space matrix Z, where the QR factorization of C with a threshold pivoting is used; see [29, 30].

The second method is the direct elimination, which involves expressing the dependence of the selected p components of the vector x on the remaining $n-p$ components, and this relationship is then substituted into the LS problem in (1.1). Suppose $P \in \mathbb{R}^{n\times n}$ is a permutation matrix such that $CP = \begin{pmatrix} C_1 \; C_2 \end{pmatrix}$ with $C_1 \in \mathbb{R}^{p\times p}$ being a nonsingular matrix. Let

$$AP = \begin{array}{cc} \begin{pmatrix} A_1 & A_2 \end{pmatrix} & m, \\ \begin{matrix} p & m-p \end{matrix} & \end{array} \qquad x = Py = P\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\begin{matrix} p \\ n-p \end{matrix} \tag{2.4}$$

Now we have the substitution $y_1 = C_1^{-1}(d - C_2y_2)$. Combining this expression with the LS problem (1.1), we have the transformed LS problem

$$\min_{y_2\in\mathbb{R}^{n-p}} \left\| \widetilde{A}y_2 - (b - A_1C_1^{-1}d) \right\|_2 , \tag{2.5}$$

where

$$\widetilde{A} = A_2 - A_1C_1^{-1}C_2 \in \mathbb{R}^{m\times(n-p)}. \tag{2.6}$$

Once we have the solution y_2, then we can compute y_1 and finally get the solution of (1.1) with the expression $x = P\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$. To get P and C_1, usually a QR factorization of C with pivoting should be exploited. For sparse matrices A and C, some strategies have been proposed to give the transformed matrix $\widetilde{A}$ some sparse structure [30, 31], leading to a sparse LS problem (2.5) that can be computed effectively by an iterative solver.

The third method is the augmented system method, which is based on the method of Lagrange multiplier for constrained optimization problem. Consider the following Lagrangian function for the constrained LS problem (1.1):

$$f(x,\lambda)=\frac{1}{2}\|Ax-b\|_2^2+\lambda^{\mathsf{T}}(d-Cx), \quad \lambda\in\mathbb{R}^p. \tag{2.7}$$

Finding the zero root of $\nabla_x f(x,\lambda)$ leads to

$$A^{\mathsf{T}}Ax-A^{\mathsf{T}}b-C^{\mathsf{T}}\lambda=\mathbf{0}.$$

By letting $r=b-Ax$ and using $Cx=d$, we have the following symmetric indefinite linear system:

$$\begin{pmatrix}\mathbf{0} & A^{\mathsf{T}} & C^{\mathsf{T}}\\ A & \mathrm{I} & \mathbf{0}\\ C & \mathbf{0} & \mathbf{0}\end{pmatrix}\begin{pmatrix}x\\ r\\ \lambda\end{pmatrix}=\begin{pmatrix}\mathbf{0}\\ b\\ d\end{pmatrix}. \tag{2.8}$$

If A and C are sparse and have full rank, then (2.8) is a $(m+n+p)\times(m+n+p)$ sparse nonsingular linear system. Based on the above framework, there are several variants of practical algorithms. We do not discuss them in more details, but refer the readers to [30, 32–35].

3 Decomposed-form solution of the LSE problem

In this section, we investigate the structure of the solutions of (1.1) and derive two decomposed-form expressions of the minimum 2-norm solution of (1.1). We consider a more general case, which is formulated as

$$\min_{x\in\mathcal{S}}\|Ax-b\|_2, \quad \mathcal{S}=\{x: x\in\operatorname*{argmin}_{x\in\mathbb{R}^n}\|Cx-d\|_2\}, \tag{3.1}$$

where in $\mathcal{S}$ we use "argmin" to denote all the minimizers of $\min_{x\in\mathbb{R}^n}\|Cx-d\|_2$. To simplify the notation, we sometimes write (3.1) as

$$\min_{x\in\mathbb{R}^n}\|Ax-b\|_2 \quad \text{s.t.} \quad \|Cx-d\|_2=\min, \tag{3.2}$$

provided it does not introduce any ambiguity. In this paper, we also call (3.1) the LSE problem. Note that if $Cx=d$ is a consistent linear system, then (3.1) is equivalent to (1.1). In the rest part of the paper, we focus on the analysis and computation of (3.1). We note that, in what follows, the assumption that C has full row rank is no longer required.

The following theorem about the generalized linear least squares (GLS) problem will be used in the subsequent analysis. We refer to [27, 36] for more details. Note that for a symmetric positive semidefinite matrix $B \in \mathbb{R}^{n\times n}$, if we define $\langle x, x'\rangle_B := x^\mathsf{T} B x$, then $(\mathcal{R}(B), \langle\cdot,\cdot\rangle_B)$ is a finite-dimensional Hilbert space.

Theorem 3.1 *For any $K \in \mathbb{R}^{m\times n}$, $L \in \mathbb{R}^{p\times n}$, and $g \in \mathbb{R}^m$, consider the GLS problem*

$$\min_{x\in\mathbb{R}^n} \|Lx\|_2 \quad \text{s.t.} \quad \|Kx - g\|_2 = \min. \tag{3.3}$$

The following properties hold:

(1) a vector $x \in \mathbb{R}^n$ is a solution of (3.3) if and only if

$$\begin{cases} K^\mathsf{T}(Kx - g) = \mathbf{0}, \\ x^\mathsf{T} M z = 0, \quad \forall\, z \in \mathcal{N}(K), \end{cases} \tag{3.4}$$

where $M = K^\mathsf{T} K + L^\mathsf{T} L$;

(2) there exist a unique solution in $\mathcal{R}(M)$, which is the minimum 2-norm solution of (3.3), given by $x = K_L^\dagger g$, where $K_L^\dagger := (\mathrm{I} - (L\mathcal{P}_{\mathcal{N}(K)})^\dagger L)K^\dagger$ is the L-weighted pseudoinverse of K;

(3) define the linear operator

$$T : \mathcal{X} := (\mathcal{R}(M), \langle\cdot,\cdot\rangle_M) \to (\mathbb{R}^m, \langle\cdot,\cdot\rangle_2), \quad v \mapsto Kv, \tag{3.5}$$

where v and Kv are column vectors under the canonical bases of $\mathbb{R}^n$ and $\mathbb{R}^m$. Then the minimum $\|\cdot\|_{\mathcal{X}}$-norm solution of the least squares problem

$$\min_{v\in\mathcal{X}} \|Tv - g\|_2 \tag{3.6}$$

is the minimum 2-norm solution of (3.3).

The following result characterizes the structure of the solutions of (3.1).

Theorem 3.2 *Let $G = A^\mathsf{T} A + C^\mathsf{T} C$. The minimum 2-norm solution of (3.1) is*

$$x^\dagger = C_A^\dagger d + (\mathcal{P}_{\mathcal{R}(G)} - C_A^\dagger C)A^\dagger b, \tag{3.7}$$

and the set of all the solutions is $x^\dagger + \mathcal{N}(G)$.

Proof First note that $\mathcal{N}(G) = \mathcal{N}(A) \cap \mathcal{N}(C)$. Thus, if x is a solution of (3.1), then $\mathcal{P}_{\mathcal{R}(G)} x$ is also a solution. Conversely, if $x \in \mathcal{R}(G)$ is a solution, then $x + z$ is also a solution for any $z \in \mathcal{N}(G)$. Since $\mathcal{N}(G) \perp \mathcal{R}(G)$, the minimum 2-norm solution of (3.1) must in $\mathcal{R}(G)$. Notice that

$$\|Ax - b\|_2^2 = \|Ax - \mathcal{P}_{\mathcal{R}(A)} b\|_2^2 + \|\mathcal{P}_{\mathcal{R}(A)^\perp} b\|_2^2$$

with $\mathcal{P}_{\mathcal{R}(A)} = AA^\dagger$ and the second term is independent of x. Therefore, the minimizer of (3.1) is identical to the minimizer of

$$\min_{x\in\mathcal{S}} \|A(x - A^\dagger b)\|_2, \quad \mathcal{S} = \{x : x \in \operatorname*{argmin}_{x\in\mathbb{R}^n} \|Cx - d\|_2\}.$$

Using the transformation $\bar{x} = x - A^\dagger b$ and noticing that $Cx - d = C(\bar{x} + A^\dagger b) - d = C\bar{x} - (d - CA^\dagger b)$, the general solution of the above problem is $x = A^\dagger b + \tilde{x}$, where $\tilde{x}$ is the general solution of

$$\min_{\bar{x}\in\mathbb{R}^n} \|A\bar{x}\|_2 \quad \text{s.t.} \quad \|C\bar{x} - (d - CA^\dagger b)\|_2 = \min. \tag{3.8}$$

By Theorem 3.1, the general solution of this problem is

$$\tilde{x} = C_A^\dagger(d - CA^\dagger b) + z, \quad z \in \mathcal{N}(G).$$

Therefore, the general solution of (3.1) is

$$x = A^\dagger b + C_A^\dagger(d - CA^\dagger b) + z = C_A^\dagger d + (\mathrm{I} - C_A^\dagger C)A^\dagger b + z, \quad z \in \mathcal{N}(G). \tag{3.9}$$

Note from Theaorem 3.1 that $\mathcal{R}(C_A^\dagger) \subseteq \mathcal{R}(G)$, which indicates that the projection of the above solution onto $\mathcal{R}(G)$ is $x^\dagger$. Thus, $x^\dagger$ is a solution of (3.1) in $\mathcal{R}(G)$.

It only remains to show that there exists a unique solution of (3.1) in $\mathcal{R}(G)$. To see it, notice from the above transformation that $x \in \mathcal{R}(G)$ is a solution of (3.1) if and only if $x - \mathcal{P}_{\mathcal{R}(G)}A^\dagger b \in \mathcal{R}(G)$ is a solution of (3.8). By Theorem 3.1, (3.8) has a unique solution in $\mathcal{R}(G)$, this implies that (3.1) has a unique solution in $\mathcal{R}(G)$. $\square$

Write

$$x_1^\dagger = C_A^\dagger d, \quad x_2^\dagger = (\mathcal{P}_{\mathcal{R}(G)} - C_A^\dagger C)A^\dagger b. \tag{3.10}$$

The minimum 2-norm solution of (3.1) has the decomposed-form: $x^\dagger = x_1^\dagger + x_2^\dagger$. Note that $x_1^\dagger$ is the minimum 2-norm solution of the GLS problem

$$\min_{x\in\mathbb{R}^n} \|Ax\|_2 \quad \text{s.t.} \quad \|Cx - d\|_2 = \min. \tag{3.11}$$

By Theorem 3.1, $x_1^\dagger$ is also the solution of the operator-form LS problem (3.6), where $K = C$ and $L = A$. We can use the iterative method proposed in [27] to approximate $x_1^\dagger$. Although the expression of $x_2^\dagger$ looks relatively complicated, we will show that it is the minimum 2-norm solution of the following LS problem:

$$\min_{x\in\mathcal{N}(C)} \|Ax - b\|_2. \tag{3.12}$$

The following lemma is useful, which provides the necessary and sufficient condition that the minimum 2-norm solution must satisfy.

Lemma 3.1 *A vector $x \in \mathcal{N}(C)$ is the minimum 2-norm solution of* (3.12) *if and only if*

$$\begin{cases} \mathcal{P}_{\mathcal{N}(C)}(A^{\mathsf{T}}(Ax - b)) = \mathbf{0}, \\ x \perp \mathcal{N}(A) \cap \mathcal{N}(C). \end{cases}$$

Proof Define the linear operator

$$\mathcal{A} : (\mathcal{N}(C), \langle \cdot, \cdot \rangle_2) \to (\mathbb{R}^m, \langle \cdot, \cdot \rangle_2), \quad v \mapsto Av, \tag{3.13}$$

where v and Av are column vectors under the canonical bases of $\mathbb{R}^n$ and $\mathbb{R}^m$. Notice that $\mathcal{X} := (\mathcal{N}(C), \langle \cdot, \cdot \rangle_2)$ is a finite dimensional Hilbert space. Therefore, there exist a unique minimum $\mathcal{X}$-norm solution of $\min_{v \in \mathcal{X}} \|\mathcal{A}v - b\|_2$, which is the minimum 2-norm solution of (3.12). Moreover, $x \in \mathcal{N}(C)$ is the minimum $\mathcal{X}$-norm solution if and only if $\mathcal{A}^*(\mathcal{A}x - b) = \mathbf{0}$ and $x \perp_{\mathcal{X}} \mathcal{N}(\mathcal{A})$, where the orthogonal relation $\perp_{\mathcal{X}}$ in $\mathcal{X}$ is the 2-orthogonal relation in $\mathcal{N}(C)$, and the linear operator $\mathcal{A}^* : (\mathbb{R}^m, \langle \cdot, \cdot \rangle_2) \to (\mathcal{N}(C), \langle \cdot, \cdot \rangle_2)$ is the adjoint of $\mathcal{A}$ defined by the relation $\langle \mathcal{A}v, u \rangle_2 = \langle v, \mathcal{A}^* u \rangle_2$ for any $v \in \mathcal{N}(C)$ and $u \in \mathbb{R}^m$. It is easy to verify that $\mathcal{A}^* v = \mathcal{P}_{\mathcal{N}(C)} A^{\mathsf{T}} v$ under the canonical bases. Thus, $\mathcal{A}^*(\mathcal{A}x - b) = \mathbf{0}$ is equivalent to $\mathcal{P}_{\mathcal{N}(C)}(A^{\mathsf{T}}(Ax - b)) = \mathbf{0}$. Since $\mathcal{N}(\mathcal{A}) = \{x \in \mathcal{N}(C) : Ax = 0\} = \mathcal{N}(A) \cap \mathcal{N}(C)$, it follows that $x \perp_{\mathcal{X}} \mathcal{N}(\mathcal{A})$ is equivalent to $x \perp \mathcal{N}(A) \cap \mathcal{N}(C)$. $\square$

Now we can prove that $x_2^\dagger$ is the minimum 2-norm solution of (3.12).

Theorem 3.3 *Let $x_2^\dagger = (\mathcal{P}_{\mathcal{R}(G)} - C_A^\dagger C) A^\dagger b$, then $x_2^\dagger$ is the minimum 2-norm solution of* (3.12).

Proof By Lemma 3.1, the proof contains the following three steps.

Step 1: prove $x_2^\dagger \in \mathcal{N}(C)$. Using [27, Theorem 3.7], we have the relation $CC_A^\dagger C = C$. It follows that

$$Cx_2^\dagger = (C\mathcal{P}_{\mathcal{R}(G)} - CC_A^\dagger C) A^\dagger b = C(\mathcal{P}_{\mathcal{R}(G)} - \mathrm{I}) A^\dagger b = -C\mathcal{P}_{\mathcal{N}(G)} A^\dagger b = \mathbf{0},$$

where we have used $\mathcal{N}(G) \subseteq \mathcal{N}(C)$.

Step 2: prove $\mathcal{P}_{\mathcal{N}(C)}(A^{\mathsf{T}}(Ax_2^\dagger - b)) = \mathbf{0}$. First we have

$$\begin{aligned} A^{\mathsf{T}}(Ax_2^\dagger - b) &= A^{\mathsf{T}}[A\mathcal{P}_{\mathcal{R}(G)} A^\dagger b - b - AC_A^\dagger C A^\dagger b] \\ &= A^{\mathsf{T}}(AA^\dagger - \mathrm{I})b - A^{\mathsf{T}} A C_A^\dagger C A^\dagger b \\ &= -A^{\mathsf{T}} A C_A^\dagger C A^\dagger b, \end{aligned}$$

where we have used $A\mathcal{P}_{\mathcal{R}(G)} x = Ax - A\mathcal{P}_{\mathcal{N}(G)} x = Ax$ for any $x \in \mathbb{R}^n$, and $\mathrm{I} - AA^\dagger = \mathcal{P}_{\mathcal{R}(A)^\perp} = \mathcal{P}_{\mathcal{N}(A^{\mathsf{T}})}$. Let $w = C_A^\dagger C A^\dagger b$. By Theorem 3.2, w is the minimum 2-norm solution of

$$\min \|Ax\|_2 \quad \text{s.t.} \quad \|Cx - CA^\dagger b\|_2 = \min .$$

Using Theorem 3.2 again, it follows that $w^{\mathsf{T}}Gz = 0$ for any $z \in \mathcal{N}(C)$, which is just

$$w^{\mathsf{T}}(A^{\mathsf{T}}A + C^{\mathsf{T}}C)z = (A^{\mathsf{T}}Aw)^{\mathsf{T}}z = 0$$

for any $z \in \mathcal{N}(C)$, which means that $A^{\mathsf{T}}Aw \perp \mathcal{N}(C)$. This proves $\mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}Aw = 0$, which is the desired result.

Step 3: prove $x_2^{\dagger} \perp \mathcal{N}(A) \cap \mathcal{N}(C)$. This is obvious by noticing that $x_2^{\dagger} \in \mathcal{R}(G)$ and $\mathcal{R}(G) \perp \mathcal{N}(A) \cap \mathcal{N}(C)$. □

From the above proof, we know that $x_2^{\dagger}$ is the minimum $\mathcal{X}$-norm solution of operator-form LS problem $\min_{\mathcal{X}} \|\mathcal{A}x - b\|_2$ with $\mathcal{A}$ defined in (3.13). Therefore, we have

$$x_2^{\dagger} = \mathcal{A}^{\dagger}b =: A_{\mathcal{N}(C)}^{\dagger}b. \tag{3.14}$$

Note that $A_{\mathcal{N}(C)}^{\dagger}$ is essentially the matrix form of $\mathcal{A}^{\dagger}$ under the canonical bases of $\mathbb{R}^n$ and $\mathbb{R}^m$, which depends both on A and $\mathcal{N}(C)$.

Based on Theorem 3.3, we will propose an iterative method for the LS problem (3.12) to approximate $x_2^{\dagger}$. Before this, let us investigate several properties of the matrix $A_{\mathcal{N}(C)}^{\dagger}$, which will be used to derive another decomposed-form solution of (3.1).

Proposition 3.1 *The following two equalities hold:*

$$\begin{cases} (\mathrm{I} - A_{\mathcal{N}(C)}^{\dagger}A)C^{\dagger} = C_A^{\dagger} \\ (\mathcal{P}_{\mathcal{R}(C)} - C_A^{\dagger}C)A^{\dagger} = A_{\mathcal{N}(C)}^{\dagger}. \end{cases} \tag{3.15}$$

Proof The second equality is directly derived from Theorem 3.3. Now we prove the first equality. By Theorem 3.1, for any $y \in \mathbb{R}^n$, $C_A^{\dagger}y$ is the 2-minimum solution of

$$\min \|Ax\|_2 \quad \text{s.t.} \quad \|Cx - y\|_2 = \min,$$

which has the same solution as

$$\min \|Ax\|_2 \quad \text{s.t.} \quad \|C(x - C^{\dagger}y)\|_2 = \min.$$

Let $\bar{x} = x - C^{\dagger}y$. The above problem becomes

$$\min \|A\bar{x} + AC^{\dagger}y\|_2 \quad \text{s.t.} \quad \|C\bar{x}\|_2 = \min,$$

which has the minimum 2-norm solution $\bar{x}^{\dagger} = -A_{\mathcal{N}(C)}^{\dagger}AC^{\dagger}y$, and a general solution is $\bar{x} = \bar{x}^{\dagger} + z$ with $z \in \mathcal{N}(A) \cap \mathcal{N}(C)$. Therefore, a general solution of the original problem is

$$x = C^{\dagger}y + \bar{x}^{\dagger}y + z.$$

Note that $\mathcal{P}_{\mathcal{N}(C)}C^{\dagger}y = (\mathrm{I} - C^{\dagger}C)C^{\dagger}y = \mathbf{0}$. Thus, $C^{\dagger}y \perp \mathcal{N}(C)$ and $C^{\dagger}y \perp z$. Combining with $\bar{x}^{\dagger} \perp z$ we have $C^{\dagger}y + \bar{x}^{\dagger} \perp z$. Therefore, $C^{\dagger}y + \bar{x}^{\dagger}y = (\mathrm{I} -$

$A^{\dagger}_{\mathcal{N}(C)}A)C^{\dagger}y$ is the minimum 2-norm solution of the original problem. Since y is arbitrary, we finally get $(\mathrm{I} - A^{\dagger}_{\mathcal{N}(C)}A)C^{\dagger} = C^{\dagger}_{A}$. □

From the above result, we obtain the following decomposed-form solution of (3.1).

Corollary 3.1 *The minimum 2-norm solution of* (3.1) *has the form*

$$x^{\dagger} = C^{\dagger}d + A^{\dagger}_{\mathcal{N}(C)}(b - AC^{\dagger}d). \tag{3.16}$$

Proof Using Theorem 3.1 and Proposition 3.1, we have

$$\begin{aligned} x^{\dagger} &= C^{\dagger}_{A}d + A^{\dagger}_{\mathcal{N}(C)}b = (\mathrm{I} - A^{\dagger}_{\mathcal{N}(C)}A)C^{\dagger}d + A^{\dagger}_{\mathcal{N}(C)}b \\ &= A^{\dagger}_{\mathcal{N}(C)}(b - AC^{\dagger}d) + C^{\dagger}d, \end{aligned}$$

which is the desired result. □

By Theorem 3.2 and Corollary 3.1, we can give two approaches for computing $x^{\dagger}$.

The first approach

(1) Solve the GLS problem (3.11) to get $x_1^{\dagger} = C^{\dagger}_{A}d$;
(2) Solve the LS problem (3.12) to get $x_2^{\dagger} = A^{\dagger}_{\mathcal{N}(C)}b$;
(3) Compute $x^{\dagger} = x_1^{\dagger} + x_2^{\dagger}$.

The second approach

(1) Solve the LS problem $\min_x \|Cx - d\|_2$ to get the minimum 2-norm solution $\tilde{x}_1^{\dagger} = C^{\dagger}d$;
(2) Let $\tilde{b} = b - A\tilde{x}_1^{\dagger}$. Solve the LS problem $\min_{x \in \mathcal{N}(C)} \|Ax - \tilde{b}\|_2$ to get the minimum 2-norm solution $\tilde{x}_2^{\dagger} = A^{\dagger}_{\mathcal{N}(C)}\tilde{b}$;
(3) Compute $x^{\dagger} = \tilde{x}_1^{\dagger} + \tilde{x}_2^{\dagger}$.

In the next section, we will propose two Krylov subspace based iterative methods for solving (3.1), which correspond to the above two approaches, respectively.

4 Krylov iterative methods for the LSE problem

From the previous section, we find that for solving the LSE problem, we need to compute $C^{\dagger}_{A}$ or $A^{\dagger}_{\mathcal{N}(C)}$. We first propose the iterative methods for such computations based on the Krylov subspace, then we give two iterative algorithms for the LSE problem.

4.1 Iterative method for computing $C_A^\dagger$

Based on Theorem 3.1, the author in [27] proposes a Krylov iterative method for approximating $C_A^\dagger d$ for a vector $d \in \mathbb{R}^p$. The idea is to apply the Golub-Kahan bidiagonalization (GKB) to solve the operator-form LS problem

$$\min_{x\in\mathcal{X}} \|Tx - d\|_2, \tag{4.1}$$

where $\mathcal{X} = (\mathcal{R}(G), \langle\cdot,\cdot\rangle_G)$ and $T : \mathcal{X} \to (\mathbb{R}^m, \langle\cdot,\cdot\rangle_2),\ x \mapsto Cx$ under the canonical bases. Applying the GKB to $\{T, d\}$ we get the recursive relations

$$\begin{cases} \beta_1 u_1 = d \\ \alpha_i v_i = T^* u_i - \beta_i v_{i-1} \\ \beta_{i+1} u_{i+1} = T v_i - \alpha_i u_i, \end{cases} \tag{4.2}$$

where $T^* : (\mathbb{R}^m, \langle\cdot,\cdot\rangle_2) \to \mathcal{X}$ is the adjoint operator of T defined by the relation $\langle Tx, y\rangle_2 = \langle x, T^* y\rangle_G$ for any $x \in \mathcal{X}$ and $y \in \mathbb{R}^m$. It has been shown in [27] that the matrix form of T^* is $G^\dagger C$. The positive scalars α_i and β_i are computed such that $\|v_i\|_{\mathcal{X}} = \|u_i\|_2 = 1$. Note that $v_0 := \mathbf{0}$ for the initial step.

After k steps, the above GKB process generates two Krylov subspaces and projects the LS problem (4.1) onto the Krylov subspaces to get a k-dimensional LS problem. The solution of the k-dimensional LS problem can be updated step by step from the previous one, which converges to $C_A^\dagger d$ as k increases. This leads to the following Algorithm 1 for iteratively approximating $C_A^\dagger d$. Please refer to [27] for more details.

Algorithm 1 Generalized LSQR (gLSQR) for computing $C_A^\dagger d$.

```
Input: $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times n}$, $d \in \mathbb{R}^p$
1: Compute $\beta_1 = \|d\|_2,\ u_1 = d/\beta_1\ \beta_1\tilde{u}_1 = b$
2: Compute $s = G^\dagger C^\top u_1,\ \alpha_1 = (s^\top G s)^{1/2},\ v_1 = s/\alpha_1$          ▷ $G = A^\top A + C^\top C$
3: Set $x_0 = \mathbf{0}$, $w_1 = v_1$, $\bar{\phi}_1 = \beta_1$, $\bar{\rho}_1 = \alpha_1$
4: for $i = 1, 2, \dots$ until convergence, do
5:     $r = Cv_i - \alpha_i u_i$
6:     $\beta_{i+1} = \|r\|_2,\ u_{i+1} = r/\beta_{i+1}$
7:     $s = G^\dagger C^\top u_{i+1} - \beta_{i+1} v_i$
8:     $\alpha_{i+1} = (s^\top G s)^{1/2},\ v_{i+1} = s/\alpha_{i+1}$
9:     $\rho_i = (\bar{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$
10:    $c_i = \bar{\rho}_i/\rho_i$
11:    $s_i = \beta_{i+1}/\rho_i$
12:    $\theta_{i+1} = s_i \alpha_{i+1}$
13:    $\bar{\rho}_{i+1} = -c_i \alpha_{i+1}$
14:    $\phi_i = c_i \bar{\phi}_i$
15:    $\bar{\phi}_{i+1} = s_i \bar{\phi}_i$
16:    $x_i = x_{i-1} + (\phi_i/\rho_i) w_i$
17:    $w_{i+1} = v_{i+1} - (\theta_{i+1}/\rho_i) w_i$
18: end for
Output: Approximation to $C_A^\dagger d$
```

In Algorithm 1, the main computational bottleneck is the need to compute $G^\dagger(C^\top u_i)$ at each iteration. For large-scale matrices, it is generally impractical to obtain $G^\dagger$ directly. In this case, using the relation

$$G^\dagger(C^\top u_i) = \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \|Gx - C^\top u_i\|_2, \tag{4.3}$$

we can compute $G^\dagger(C^\top u_i)$ by iteratively solving the above LS problem. Furthermore, by noticing that $G^\dagger(C^\top u_i)$ is the minimum 2-norm solution of the LS problem

$$\min_{x\in\mathbb{R}^n} \left\| \begin{pmatrix} C \\ A \end{pmatrix} x - \begin{pmatrix} u_i \\ \mathbf{0} \end{pmatrix} \right\|_2, \tag{4.4}$$

we can use the LSQR algorithm [37] to approximate $G^\dagger(C^\top u_i)$ without explicitly forming G. If $\begin{pmatrix} C \\ A \end{pmatrix}$ is sparse and has full column rank, and its sparse QR factorization is not difficult to compute, then we can compute the solution of (4.4) directly.

4.2 Iterative method for computing $A^\dagger_{\mathcal{N}(C)}$

Now we consider how to design a GKB based method to approximate $A_{\mathcal{N}(C)}b$ for a $b \in \mathbb{R}^m$. First, suppose an orthonormal basis of the null space $\mathcal{N}(C)$ is $\{w_1, \dots, w_t\}$. Let $W = (w_1, \dots, w_t) \in \mathbb{R}^{n\times t}$. Using Theorem 3.3, if follows that $A^\dagger_{\mathcal{N}(C)}b = Wf$, where $f \in \mathbb{R}^t$ is the minimum 2-norm solution of the LS problem

$$\min_{f\in\mathbb{R}^t} \|(AW)f - b\|_2 \tag{4.5}$$

To solve (4.5) iteratively, we apply the GKB to $\{AW, b\}$, which leads to the following recursive relations:

$$\begin{cases} \delta_1 p_1 = b \\ \gamma_i \tilde{q}_i = (AW)^\top p_i - \delta_i \tilde{q}_{i-1} \\ \delta_{i+1} p_{i+1} = (AW)\tilde{q}_i - \gamma_i p_i, \end{cases} \tag{4.6}$$

where the positive scalars are computed such that $\|p_i\|_2 = \|q_i\|_2 = 1$, and we set $q_0 := \mathbf{0}$ for the initial step.

Using the property of GKB, after k steps, it generates two groups of 2-orthonormal vectors $\{p_i\}_{i=1}^{k+1}$ and $\{\tilde{q}_i\}_{i=1}^{k+1}$. Then we can approximate the solution of (4.5) in the subspace $\operatorname{span}\{\tilde{q}_i\}_{i=1}^{k}$ as k grows from 1 to t. This approach is equivalent to applying the standard LSQR algorithm to (4.5). Therefore, to get a good approximation to $A^\dagger_{\mathcal{N}(C)}b$, we can search a solution of (3.12) in the subspace $\operatorname{span}\{W\tilde{q}_i\}_{i=1}^{k}$ at the k-th iteration. Let $q_i = W\tilde{q}_i$. Note that $\mathcal{P}_{\mathcal{N}(C)} = WW^{\mathsf{T}}$. From the recursions (4.6), we get

$$\begin{cases} \delta_1 p_1 = b \\ \gamma_i q_i = \mathcal{P}_{\mathcal{N}(C)} A^{\mathsf{T}} p_i - \delta_i q_{i-1} \\ \delta_{i+1} p_{i+1} = A q_i - \gamma_i p_i, \end{cases} \tag{4.7}$$

where $\|q_i\|_2 = 1$. The following result demonstrates that this iterative process is essentially an operator-type GKB.

Proposition 4.1 *Let the linear operator be defined as* (3.13). *Then the iterative process* (4.7) *is equivalent to the GKB applied to* $\{\mathcal{A}, b\}$.

Proof From the proof of Lemma 3.1 we know that $\mathcal{A}^* v = \mathcal{P}_{\mathcal{N}(C)} A^{\mathsf{T}} v$ for any $v \in \mathbb{R}^m$ under the canonical bases. Therefore, the second recursive relation in (4.7) is equivalent to $\gamma_i q_i = \mathcal{A}^* p_i - \delta_i q_{i-1}$. Now we can find that (4.7) is just the recursions of the operator-type GKB applied to $\{\mathcal{A}, b\}$ under the canonical bases. □

Proposition 4.1 implies that the outputs of the above iterative process do not depend on the choice of the 2-orthonormal basis of $\mathcal{N}(C)$, i.e. it will generate the same vectors $\{p_i, q_i\}$ and scalars $\{\gamma_i, \delta_i\}$, regardless of the particular 2-orthonormal basis $\{w_i\}_{i=1}^t$ chosen for $\mathcal{N}(C)$.

Now we can give the practical computational approach of this iterative process. Since all the constructed vectors q_i are restricted in $\mathcal{N}(C)$, we name this process the *Null Space Restricted GKB* (NSR-GKB). The pseudocode of NSR-GKB is shown in Algorithm 2.

Note that $\mathcal{P}_{\mathcal{N}(C)} = \mathrm{I}_n - C^{\dagger}C$. In the computation of NSR-GKB, the orthonormal basis of $\mathcal{N}(C)$ is not required, where instead at each step we need to compute $C^{\dagger}CA^{\mathsf{T}}p_i$, which is the most costly part. Write $\tilde{v}_i = CA^{\mathsf{T}}p_i$, which is easy to compute. To get a good approximation to $C^{\dagger}\tilde{v}_i$, we can iteratively compute the minimum 2-norm solution of the LS problem

$$\min_{x \in \mathbb{R}^n} \|Cx - \tilde{v}_i\|_2, \tag{4.8}$$

which can be done efficiently by using the LSQR algorithm. In this case, NSR-GKB has the nested inner-outer iteration structure. If C has a special structure such that its

Algorithm 2 Null Space Restricted GKB (NSR-GKB).

```
Input: $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times n}$, $b \in \mathbb{R}^m$
1: Compute $\delta_1 = \|b\|_2,\ p_1 = b/\delta_1$,
2: Compute $s = \mathcal{P}_{\mathcal{N}(C)} A^{\mathsf{T}} p_1,\ \gamma_1 = \|s\|_2,\ q_1 = s/\gamma_1$
3: for $i = 1, 2, \dots, k$, do
4:     $r = Aq_i - \gamma_i p_i$,
5:     $\delta_{i+1} = \|r\|_2,\ p_{i+1} = r/\delta_{i+1}$
6:     $s = \mathcal{P}_{\mathcal{N}(C)} A^{\mathsf{T}} p_{i+1} - \delta_{i+1} q_i$
7:     $\gamma_{i+1} = \|s\|_2,\ q_{i+1} = s/\gamma_{i+1}$
8: end for
Output: $\{\gamma_i, \delta_i\}_{i=1}^{k+1}$, $\{p_i, q_i\}_{i=1}^{k+1}$
```

rank-revealing QR factorization is relatively easy to compute, we can first get the QR factorization of C and then compute $C^{\dagger}\tilde{v}_i$ directly.

The following result characterizes the structures of the two Krylov subspaces generated by NSR-GKB.

Proposition 4.2 *For the NSR-GKB process, the generated vectors $\{q_i\}_{i=1}^{k} \subset \mathcal{N}(C)$ constitute a 2-orthonormal basis of the Krylov subspace*

$$\mathcal{K}_k(\mathcal{P}_{\mathcal{N}(C)}A^{\top}A, \mathcal{P}_{\mathcal{N}(C)}A^{\top}b) = \mathrm{span}\{(\mathcal{P}_{\mathcal{N}(C)}A^{\top}A)^i\mathcal{P}_{\mathcal{N}(C)}A^{\top}b\}_{i=0}^{k-1}, \tag{4.9}$$

and $\{p_i\}_{i=1}^{k} \subset \mathbb{R}^m$ constitute a 2-orthonormal basis of the Krylov subspace

$$\mathcal{K}_k(A\mathcal{P}_{\mathcal{N}(C)}A^{\top}, b) = \mathrm{span}\{(A\mathcal{P}_{\mathcal{N}(C)}A^{\top})^i b\}_{i=0}^{k-1}. \tag{4.10}$$

Proof To get more insights into the NSR-GKB process, here we give two proofs.

The first proof is based on the property of GKB for linear compact operators [26]. By Proposition 4.1, the NSR-GKB is essentially the operator-type GKB of $\{\mathcal{A}, b\}$, where the underlying Hilbert spaces are $\mathcal{X} := (\mathcal{N}(C), \langle\cdot,\cdot\rangle_2)$ and $\mathbb{R}^m$. Therefore, under the canonical bases, the generated vectors satisfy $q_i \in \mathcal{N}(C)$ and $p_i \in \mathbb{R}^m$, and $\{q_i\}_{i=1}^{k}$ and $\{p_i\}_{i=1}^{k}$ are 2-orthonormal bases of the Krylov subspaces $\mathcal{K}_k(\mathcal{A}^*\mathcal{A}, \mathcal{A}^*b)$ and $\mathcal{K}_k(\mathcal{A}\mathcal{A}^*, b)$, respectively. Since $\mathcal{A}^*y = \mathcal{P}_{\mathcal{N}(C)}A^{\top}y$ for any $y \in \mathbb{R}^m$, we have

$$(\mathcal{A}^*\mathcal{A})^i\mathcal{A}^*b = (\mathcal{P}_{\mathcal{N}(C)}A^{\top}A)^i\mathcal{P}_{\mathcal{N}(C)}A^{\top}b,$$

and

$$(\mathcal{A}\mathcal{A}^*)^i b = (A\mathcal{P}_{\mathcal{N}(C)}A^{\top})^i b.$$

The desired result immediately follows.

The second proof uses the recursions (4.6), which is based on a 2-orthonormal basis $\{w_i\}_{i=1}^{t}$ for $\mathcal{N}(C)$. The standard GKB process of $\{(AW), b\}$ with recursions (4.6) generates two 2-orthonormal basis $\{\tilde{q}_i\}_{i=1}^{k}$ and $\{p_i\}_{i=1}^{k}$ for the two Krylov subspaces

$$\begin{aligned}
&\mathcal{K}_k((AW)^{\top}AW, (AW)^{\top}b) = \mathrm{span}\{((AW)^{\top}AW)^i(AW)^{\top}b\}_{i=0}^{k-1},\\
&\mathcal{K}_k(AW(AW)^{\top}, b) = \mathrm{span}\{(AW(AW)^{\top})^i b\}_{i=0}^{k-1},
\end{aligned}$$

respectively. Using $q_i = W\tilde{q}_i$, $WW^{\top} = \mathcal{P}_{\mathcal{N}(C)}$ and noticing that

$$\begin{aligned}
&W((AW)^{\top}AW)^i(AW)^{\top}b\\
&= W(W^{\top}A^{\top}AW)^iW^{\top}A^{\top}b = (WW^{\top}A^{\top}A)^iWW^{\top}A^{\top}b\\
&= (\mathcal{P}_{\mathcal{N}(C)}A^{\top}A)^i\mathcal{P}_{\mathcal{N}(C)}A^{\top}b,
\end{aligned}$$

we immediately obtain (4.9). Similarly, we can obtain (4.10). □

Since the dimensions of $(\mathcal{N}(C), \langle\cdot,\cdot\rangle_2)$ and $\mathbb{R}^m$ are t and m, respectively, Proposition 4.2 implies that NSR-GKB will eventually terminate at most $\min\{t, m\}$ steps. The "terminate step" of NSR-GKB is defined as

$$k_t = \min\{k : \alpha_{k+1}\beta_{k+1} = 0\}, \tag{4.11}$$

which means that γ_i or δ_i equals zero at the current step and thereby the Krylov subspace can not expand any longer. Suppose NSR-GKB does not terminate before the k-th iteration, that is, $\gamma_i\delta_i \neq 0$ for $1 \leq i \leq k$. Then the k-step NSR-GKB process generates two 2-orthonormal matrices $Q_k = (q_1, \ldots, q_k) \in \mathbb{R}^{n\times k}$ and $P_k = (p_1, \ldots, p_k) \in \mathbb{R}^{m\times k}$ that satisfy the following matrix-form relations:

$$\begin{cases} \beta_1 Q_{k+1} e_1 = b \\ A P_k = Q_{k+1} B_k \\ \mathcal{P}_{\mathcal{N}(C)} A^{\mathsf{T}} Q_{k+1} = P_k B_k^{\mathsf{T}} + \gamma_{k+1} q_{k+1} e_{k+1}^{\mathsf{T}}, \end{cases} \tag{4.12}$$

where e_1 and e_{k+1} are the first and $(k+1)$-th columns of the identity matrix of order $k+1$, and the bidiagonal matrix

$$B_k = \begin{pmatrix} \gamma_1 & & & \\ \delta_2 & \gamma_2 & & \\ & \delta_3 & \ddots & \\ & & \ddots & \gamma_k \\ & & & \delta_{k+1} \end{pmatrix} \in \mathbb{R}^{(k+1)\times k} \tag{4.13}$$

has full column rank. We remark that it may happen that $\delta_{k+1} = 0$, meaning that NSR-GKB terminates at the k-th step with $q_{k+1} = \mathbf{0}$.

Now we seek the approximation to $A^{\dagger}_{\mathcal{N}(C)} b$ by computing the solution of (3.12) in the Krylov subspace $\operatorname{span}\{Q_k\} \subset \mathcal{N}(C)$. For any $x \in \operatorname{span}\{Q_k\}$, let $x = Q_k y$ with $y \in \mathbb{R}^k$. Using the relations (4.12), we get

$$\min_{x=Q_k y} \|Ax - b\|_2 = \min_{y\in\mathbb{R}^k} \|P_{k+1}(B_k y - \beta_1 e_1)\|_2 = \min_{y\in\mathbb{R}^k} \|B_k y - \beta_1 e_1\|_2.$$

Therefore, at the k-th iteration, we only need to solve the following k-dimensional subproblem to get the approximation:

$$x_k = Q_k y_k, \quad y_k = \operatorname*{argmin}_{y\in\mathbb{R}^k} \|B_k y - \beta_1 e_1\|_2. \tag{4.14}$$

Note that before NSR-GKB terminates, B_k has full column rank and the LS problem $\operatorname{argmin}_y \|B_k y - \beta_1 e_1\|_2$ always has the unique solution $y_k = B_k^{\dagger}\beta e_1$. As the iteration proceeds, x_k will gradually approximate the true solution of (3.12). The following result shows that at the terminate step, we will get the exact solution of (3.12).

Theorem 4.1 *Suppose that NSR-GKB terminates at step k_t. Then the iterative solution $x_{k_t} = A^{\dagger}_{\mathcal{N}(C)}b$, which is the exact minimum 2-norm solution of* (3.12).

Proof Since $x_{k_t} \in \mathrm{span}\{Q_k\} \subset \mathcal{N}(C)$, we only need to verify that x_{k_t} satisfies the two relations of Lemma 3.1.

Write x_{k_t} as $x_{k_t} = Q_{k_t}y_{k_t}$ and use the relation $Ax_{k_t} - b = Q_{k_t+1}(B_{k_t}y_t - \beta_1 e_1)$. We have

$$\begin{aligned}
\mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}(Ax_{k_t} - b) &= \mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}Q_{k_t+1}(B_{k_t}y_{k_t} - \delta_1 e_1) \\
&= (P_{k_t}B_{k_t}^{\mathsf{T}} + \gamma_{k_t+1}q_{k_t+1}e_{k+1}^{\mathsf{T}})(B_{k_t}y_{k_t} - \delta_1 e_1) \\
&= P_{k_t}(B_{k_t}^{\mathsf{T}}B_{k_t}y_{k_t} - B_{k_t}^{\mathsf{T}}\delta_1 e_1) + \gamma_{k_t+1}\delta_{k_t+1}v_{k_t+1}e_{k_t}^{\mathsf{T}}y_{k_t} \\
&= \gamma_{k_t+1}\delta_{k_t+1}v_{k_t+1}e_{k_t}^{\mathsf{T}}y_{k_t} \\
&= \mathbf{0},
\end{aligned}$$

since $\gamma_{k_t+1}\delta_{k_t+1} = 0$ and $B_{k_t}^{\mathsf{T}}B_{k_t}y_{k_t} = B_{k_t}^{\mathsf{T}}\delta_1 e_1$ due to $y_{k_t} = \mathrm{argmin}_y \|B_{k_t}y - \beta_1 e_1\|_2$. This verifies the first relation of Lemma 3.1. By Proposition 4.2 we have $x_{k_t} \in \mathcal{P}_{\mathcal{N}(C)}\mathcal{R}(A^{\mathsf{T}}) = \mathcal{P}_{\mathcal{N}(C)}\mathcal{N}(A)^{\perp}$. Let $x_{k_t} = \mathcal{P}_{\mathcal{N}(C)}w$ with $w \in \mathcal{N}(A)^{\perp}$. For any $y \in \mathcal{N}(A) \cap \mathcal{N}(C)$, we have

$$\langle x_{k_t}, y\rangle_2 = \langle \mathcal{P}_{\mathcal{N}(C)}w, y\rangle_2 = \langle w, \mathcal{P}_{\mathcal{N}(C)}y\rangle_2 = \langle w, y\rangle_2 = 0.$$

This verifies the second relation of Lemma 3.1. □

In the practical computation, we do not need to compute $B_k^{\dagger}$ to get x_k at each iteration. Instead, by exploiting the bidiagonal structure of B_k, we can design a recursive procedure to update x_k based on the Givens QR factorization of B_k. This procedure follows a very similar approach proposed in [37, Section 4.1], and we omit the derivation. Combining the NSR-GKB process and the update procedure, we get the following Algorithm 3 for approximating $A^{\dagger}_{\mathcal{N}(C)}b$. This algorithm is named the *Null Space Restricted LSQR*(NSR-LSQR). We remark that for notational simplicity, some notations in Algorithm 3 are the same as those in Algorithm 1, but the readers can easily find the differences between them.

To check the convergence condition of NSR-LSQR, here we give a stopping criterion. The idea is based on Theorem 3.3 and Proposition 4.1, which implies that NSR-LSQR is a Krylov subspace iterative method applied to the operator-type LS problem $\min_{x\in\mathcal{X}} \|\mathcal{A}x - b\|_2$. For the standard LS problem $\min_x \|Ax - b\|_2$, a commonly used stopping criterion is $\frac{\|A^{\mathsf{T}}r_k\|_2}{\|A\|_2\|r_k\|_2}$, where $r_k = Ax_k - b$ (here we use r_k and x_k to denote the quantities computed by LSQR without ambiguity); see [37, Section 6]. Similarly, for the NSR-LSQR algorithm, we use the following relative residual norm for the stopping criterion:

$$\frac{\|\mathcal{A}^{*}r_k\|_2}{\|\mathcal{A}\|_2\|r_k\|_2} = \frac{\|\mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}r_k\|_2}{\|r_k\|_2} \le \mathtt{tol}, \tag{4.15}$$

Algorithm 3 Null Space Restricted LSQR (NSR-LSQR) for computing $A^{\dagger}_{\mathcal{N}(C)}b$.

```
Input: $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times n}$, $b \in \mathbb{R}^m$
 1: (Initialization)
 2: Compute $\delta_1 p_1 = b$, $\gamma_1 q_1 = \mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}p_1$        ▷ Step 1–2 of NSR-GKB
 3: Set $x_0 = \mathbf{0}$, $z_1 = q_1$, $\bar{\phi}_1 = \delta_1$, $\bar{\rho}_1 = \gamma_1$
 4: for $i = 1, 2, \dots$ until convergence, do
 5:     (Applying the NSR-GKB process)
 6:     $\delta_{i+1}p_{i+1} = Aq_i - \gamma_i p_i$
 7:     $\gamma_{i+1}q_{i+1} = \mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}p_{i+1} - \delta_{i+1}q_i$
 8:     (Applying the Givens QR factorization to $B_k$)
 9:     $\rho_i = (\bar{\rho}_i^2 + \delta_{i+1}^2)^{1/2}$
10:     $c_i = \bar{\rho}_i/\rho_i$
11:     $s_i = \beta_{i+1}/\rho_i$
12:     $\theta_{i+1} = s_i\gamma_{i+1}$
13:     $\bar{\rho}_{i+1} = -c_i\gamma_{i+1}$
14:     $\phi_i = c_i\bar{\phi}_i$
15:     $\bar{\phi}_{i+1} = s_i\bar{\phi}_i$
16:     (Updating the solution)
17:     $x_i = x_{i-1} + (\phi_i/\rho_i)z_i$
18:     $w_{i+1} = v_{i+1} - (\theta_{i+1}/\rho_i)z_i$
19: end for
Output: Approximation to $A^{\dagger}_{\mathcal{N}(C)}b$
```

where $r_k = Ax_k - b$, and $\|\mathcal{A}\|_2 := \max\limits_{\substack{v\in\mathcal{N}(C)\\ v\neq\mathbf{0}}} \frac{\|\mathcal{A}v\|_2}{\|v\|_2}$. From the proof of Theorem 4.1 we know that $\mathcal{A}^*r_k$ would be zero when the exact solution is obtained. Furthermore, at each iteration, we also have

$$\|\mathcal{A}^*r_k\|_2 = \|\mathcal{P}_{\mathcal{N}(C)}A^{\mathsf{T}}(Ax_k - b)\|_2 = \|\gamma_{k+1}\delta_{k+1}v_{k+1}e_k^{\mathsf{T}}y_k\|_2 = \gamma_{k+1}\delta_{k+1}|e_k^{\mathsf{T}}y_k|.$$

This means that $\|\mathcal{A}^*r_k\|_2$ can be quickly obtained with almost no additional cost. The following result provides an approach for estimating $\|\mathcal{A}\|_2$.

Proposition 4.3 *Suppose $\{w_i\}_{i=1}^t$ is an arbitrary 2-orthonormal basis of $\mathcal{N}(C)$ and $W = (w_1, \dots, w_t) \in \mathbb{R}^{n\times t}$. Then it holds that*

$$\|\mathcal{A}\|_2 = \sigma_{\max}(AW), \tag{4.16}$$

which is the largest singular value of AW.

Proof For any $v \in \mathcal{N}(C)$, there exist a unique $y \in \mathbb{R}^t$ such that $v = Wy$, and $\|v\|_2 = \|y\|_2$. Therefore, we have

$$\|\mathcal{A}\|_2 = \max_{\substack{v\in\mathcal{N}(C)\\ v\neq\mathbf{0}}} \frac{\|\mathcal{A}v\|_2}{\|v\|_2} = \max_{\substack{y\in\mathbb{R}^t\\ y\neq\mathbf{0}}} \frac{\|AWy\|_2}{\|Wy\|_2} = \max_{\substack{y\in\mathbb{R}^t\\ y\neq\mathbf{0}}} \frac{\|AWy\|_2}{\|y\|_2} = \|AW\|_2 = \sigma_{\max}(AW).$$

The proof is completed. □

Algorithm 4 Krylov Iterative Decomposed Solver-I (KIDS-I) for (3.1).

```
Input: $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times n}$, $b \in \mathbb{R}^{m}$, $d \in \mathbb{R}^{p}$
1: Compute $x_1^{\dagger} = C_A^{\dagger} d$ by Algorithm 1
2: Compute $x_2^{\dagger} = A_{\mathcal{N}(C)}^{\dagger} b$ by Algorithm 3
3: Compute $x^{\dagger} = x_1^{\dagger} + x_2^{\dagger}$
Output: Approximate solution of (3.1)
```

Note from Proposition 4.3 that $\|\mathcal{A}\|_2 = \sigma_{\max}(AW)$ does not depend on the choice of the 2-orthonormal basis of $\mathcal{N}(C)$. To estimate $\sigma_{\max}(AW)$, a very practical approach is to apply the GKB based SVD algorithm [25]. Combining (4.6) and (4.7), we can find that NSR-GKB generates the same $\{\gamma_i, \delta_i\}$ as that generated by the GKB of AW, whenever which 2-orthonormal basis $\{w_i\}_{i=1}^{t}$ is used. Therefore, we can use the largest singular value of B_k generated by NSR-GKB to approximate $\sigma_{\max}(AW)$, and it will not take too many iterations to get an accurate estimate.

4.3 Two Krylov iterative methods for the LSE problem

Based on Algorithm 1 and Algorithm 3, we give two Krylov subspace based iterative algorithms for the LSE problem, which correspond to the two approaches at the end of Section 3, respectively. The first algorithm named *Krylov Iterative Decomposed Solver-I* (KIDS-I) is shown in Algorithm 4, and the second algorithm named *Krylov Iterative Decomposed Solver-II* (KIDS-II) is shown in Algorithm 5.

We give a brief comparison between the above two algorithms. Generally, for large-scale problems, both the two algorithms have a nested inner-outer iteration structure: for KIDS-II, we need to solve (4.8) at each iteration of Algorithm 3, while for KIDS-II, we need to solve (4.4) and (4.8) at each iteration of Algorithm 1 and Algorithm 3, respectively. In KIDS-I, the computation of $x_1^{\dagger}$ and $x_2^{\dagger}$ can be performed simultaneously. However, in KIDS-II, the three steps (corresponding to lines 1–3 in Algorithm 5) must be performed sequentially.

5 Numerical experiments

We present several numerical examples to illustrate the performance of the two proposed algorithms for the LSE problems. All the experiments are conducted in MATLAB R2023b with double precision.

Algorithm 5 Krylov Iterative Decomposition Solver-II (KIDS-II) for (3.1).

```
Input: $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times n}$, $b \in \mathbb{R}^{m}$, $d \in \mathbb{R}^{p}$
1: Compute $\tilde{x}_1^{\dagger} = C^{\dagger} d$ by solving $\min_x \|Cx - d\|_2$
2: Compute $\tilde{b} = b - A\tilde{x}_1^{\dagger}$
3: Compute $\tilde{x}_2^{\dagger} = A_{\mathcal{N}(C)}^{\dagger} \tilde{b}$ by Algorithm 3
4: Compute $x^{\dagger} = \tilde{x}_1^{\dagger} + \tilde{x}_2^{\dagger}$
Output: Approximate solution of (3.1)
```

5.1 Small- and medium-scale problems

It is worth noting that much of the existing literature on LSE problems lacks numerical results, primarily due to the difficulty of constructing nontrivial test problems, particularly for large-scale matrices. Based on the analysis of the LSE problems in Section 3, we propose the following procedure to construct LSE problems for testing purposes.

Construct test problems By Theorem 3.2 and Theorem 3.3, the minimum 2-norm solution of (3.1) is $x^\dagger = x_1^\dagger + x_2^\dagger$, where $x_1^\dagger$ is the minimum 2-norm solutions of (3.3) with $K = C$ and $L = A$, and $x_2^\dagger$ is the minimum 2-norm solutions of (3.12). With the help of the approach for constructing test problems for the GLS problems (see [27, Section 5]), we can construct a test LSE problem using the following steps:

(1) Choose two matrices $A \in \mathbb{R}^{m\times n}$ and $C \in \mathbb{R}^{p\times n}$. Compute $G = A^\mathsf{T}A + C^\mathsf{T}C$.
(2) Compute a matrix $B \in \mathbb{R}^{n\times t}$ whose columns form a basis for $\mathcal{N}(C)$.
(3) Construct a vector $w_1 \in \mathcal{R}(G)$. Compute

$$x_1^\dagger = w_1 - B(B^\mathsf{T}GB)^{-1}B^\mathsf{T}Gw_1. \tag{5.1}$$

(4) Choose a vector $z_1 \in \mathcal{R}(C)^\perp$ and let $d = Cx_1^\dagger + z_1$.
(5) Construct a vector $w_2 \in \mathbb{R}^t$ such that $w_2 \perp \mathcal{N}(AB)$, and choose a vector $z_2 \in \mathcal{R}(AB)^\perp$.
(6) Let $x_2^\dagger = Bw_2$ and $b = Ax_2^\dagger + z_2$.
(7) Compute $x^\dagger = x_1^\dagger + x_2^\dagger$.

Note that the fourth step ensures that $x_1^\dagger = C_A^\dagger d$, while the sixth step ensures that $x_2^\dagger = A_{\mathcal{N}(C)}^\dagger b$. By this construction, the minimum 2-norm solution of (3.1) is $x^\dagger$.

In the numerical experiments, we construct four test examples. For the first example, we set $A = D_1$, which is the scaled discretization of the first-order differential operator:

$$D_1 = \begin{pmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{pmatrix} \in \mathbb{R}^{(n-1)\times n}. \tag{5.2}$$

The matrix $C \in \mathbb{R}^{2324\times 4486}$ named lp_bnl2 comes from linear programming problems and is sourced from the SuiteSparse Matrix Collection [38]. Let $w_1 = (1, \cdots, 1)^\mathsf{T} \in \mathbb{R}^n$. We use the MATLAB built-in function `null.m` to compute a basis matrix B for $\mathcal{N}(C)$, and we let w_2 be the first column of $(AB)^\mathsf{T}$. To obtain vectors z_1 and z_2, we compute the projections of the random vectors `randn(p,1)` and `randn(m,1)` onto $\mathcal{R}(C)^\perp$ and $\mathcal{R}(AB)^\perp$, respectively.

For the second example, we set $A = D_2$, which is the scaled discretization of the second-order differential operator:

$$D_2 = \begin{pmatrix} -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \end{pmatrix} \in \mathbb{R}^{(n-2)\times n}, \tag{5.3}$$

and the matrix $C \in \mathbb{R}^{5190\times 9690}$ named r05 comes from linear programming problems, taken from [38]. We use almost the similar setting as the above for constructing the LSE problem, where the only difference is that we construct $w_1 \in \mathbb{R}^n$ by evaluating the function $f(t) = t$ on a uniform grid over the interval [0, 1], that is, $w_1(k) = \frac{k-1}{n-1}$ for $k = 1, \ldots, n$.

For the third example, we choose the matrix $M \in \mathbb{R}^{3534\times 3534}$ named cage9 from [38], which arises from the directed weighted graph problem. Then we set $A = M(:, 1 : 2500)$ and $C = M(:, 2501 : 3534)$. Then we construct the LSE problem using almost the same setting as the above, where the only difference is that we construct $w_1 \in \mathbb{R}^n$ by evaluating the function $f(t) = t^2$ on a uniform grid over the interval $[-1, 1]$, that is, $w_k(k) = \left(\frac{2(k-1)}{n-1} - 1\right)^2$ for $k = 1, \ldots, n$. We use cage9-I and cage9-II to denote A and C, respectively.

For the fourth example, we choose the matrix $M \in \mathbb{R}^{9728\times 9728}$ named pf2177 from [38], which arises from the optimization problem. Then we set $A = M(:, 1 : 6500)$ and $C = M(:, 6501 : 9728)$. Then we construct the LSE problem using almost the same setting as the above, where the only difference is that we construct $w_1 \in \mathbb{R}^n$ by evaluating the function $f(t) = \sin(2t) + 3\cos(t)$ on a uniform grid over the interval $[-\pi, \pi]$, that is, $w(k) = \sin\left(\frac{4\pi(k-1)}{n-1} - 2\pi\right) - 3\cos\left(\frac{2\pi(k-1)}{n-1} - \pi\right)$ for $k = 1, \ldots, n$. We use pf2177-I and pf2177-II to denote A and C, respectively.

Several properties of the matrices in the four test examples are listed in Table 1.

Experimental results In this experiment, we demonstrate the convergence behavior and the final accuracy of the approximate solutions computed by KIDS-I and KIDS-II. For comparison, we also compute two solutions using the null space method and the direct elimination method, denoted as "NS" and "DE", respectively. For the KIDS-I algorithm, at the k-th step, we compute the approximations to $x_1^\dagger$ and $x_2^\dagger$, respectively, which are denoted by x_{1k} and x_{2k}. We then compute the k-th approximate solution of

Table 1 Properties of the test examples

	A			C		
Example	name	$m \times n$	$\kappa(A)$	name	$p \times n$	$\kappa(C)$
1	D_1	4485×4486	2855.90	lp_bnl2	2324×4486	7765.31
2	D_2	9688×9690	1.68×10^7	r05	5190×9690	121.82
3	cage9-I	2500×3534	3.93	cage9-II	1034×3534	12.48
4	pf2177-I	6500×9728	134.84	pf2177-II	3228×9728	44.00

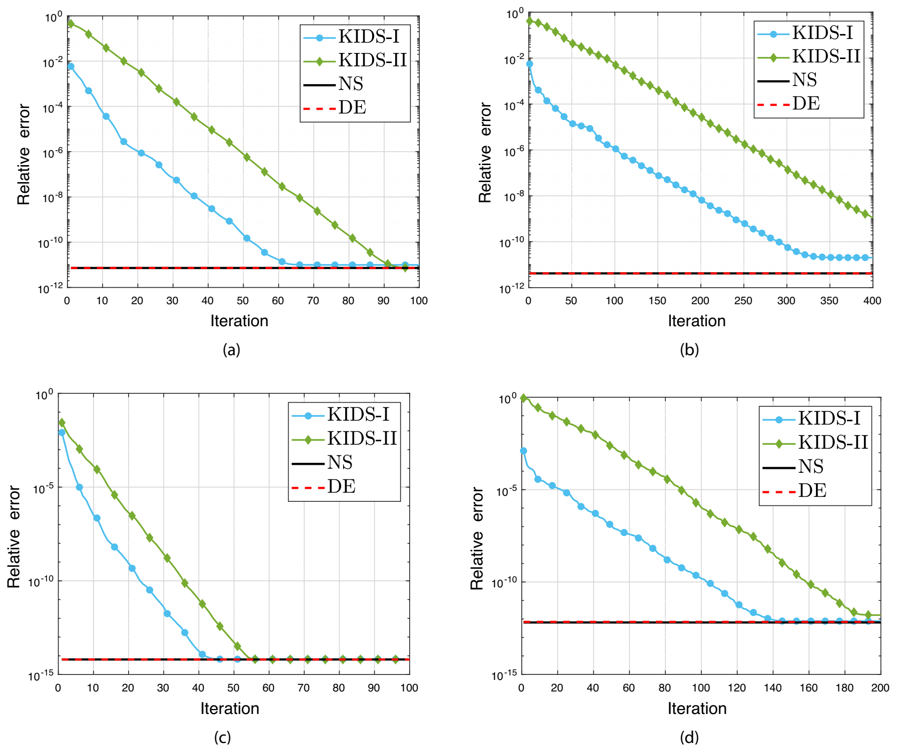

Fig. 1 The convergence history of KIDS-I and KIDS-II with respect to the true solution, where all the inner iterations are computed accurately. (a) $\{D_1$, lp_bnl2$\}$; (b)$\{D_2$, r05$\}$; (c) $\{$cage9-I, cage9-II$\}$; (d) $\{$pf2177-I, pf2177-II$\}$

(1.1) as $x_k = x_{1k} + x_{2k}$. For the KIDS-II algorithm, we first compute $\tilde{x}_1^{\dagger}$, which is the solution to the LS problem $\min_x \|Cx - d\|_2$. Then we apply Algorithm 3 to compute the approximations to $\tilde{x}_2^{\dagger}$, where we denote the k-th approximation by $\tilde{x}_{2k}$. The k-th approximate solution of (1.1) by KIDS-II is $x_k = \tilde{x}_1^{\dagger} + \tilde{x}_{2k}$. In this experiment, all the inner iterations are computed accurately.

Figure 1 shows the convergence history of the two algorithms with respect to the true solution. Table 2 lists the relative errors of the solutions at the final iterations

Table 2 Relative errors of the solutions at the final iterations of KIDS-I and KIDS-II, and the relative errors of the solutions computed by NS and DE

Example	KIDS-I	KIDS-II	NS	DE
1	9.92×10^{-12}	7.29×10^{-12}	7.26×10^{-12}	7.26×10^{-12}
2	2.04×10^{-11}	1.10×10^{-9}	4.18×10^{-12}	4.18×10^{-12}
3	6.55×10^{-15}	6.37×10^{-15}	6.37×10^{-15}	6.37×10^{-15}
4	7.64×10^{-13}	1.62×10^{-12}	6.46×10^{-13}	6.94×10^{-13}

of KIDS-I and KIDS-II, as well as the relative errors of the solutions computed by NS and DE. We have three key findings. First, for both algorithms, all the approximate solutions eventually converge to the exact solution of the LSE problem, with accuracy that is almost the same as, or slightly lower than, the solutions obtained by the NS or DE methods. Second, both KIDS-I and KIDS-II exhibit a linear convergence rate for the four test problems. Since the two algorithms are based on the operator-form GKB process, we hypothesize that the convergence rate may share similarities with the LSQR algorithm. However, a more detailed investigation is needed in the future to confirm this. Third, compared with KIDS-II, KIDS-I requires fewer iterations to achieve a solution with a given accuracy. However, it is not yet clear whether this is a general property of the algorithms.

In Figure 2 we plot the curve corresponding to x_k computed by KIDS-I at the final iteration, alongside the true solution $x^{\dagger}$. It is important to note that the curves corresponding to the true solutions are not smooth, as the operations used in constructing the test problems can lead to oscillating vectors. We remark that constructing a smooth true solution based on the proposed procedure for generating a test LSE problem is quite challenging. From the figure, we observe that the computed solutions closely match the true solution. These results demonstrate the effectiveness of the proposed algorithms in iteratively solving LSE problems.

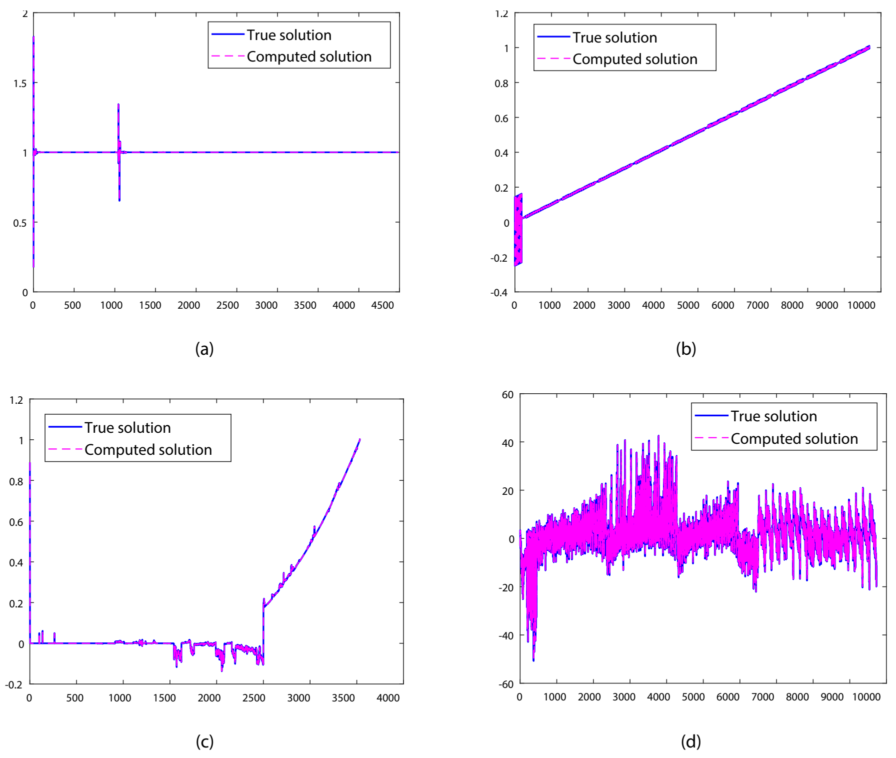

Fig. 2 Curves for the true and computed solutions obtained by KIDS-I at the final iteration. (a) $\{D_1$, lp_bnl2$\}$; (b)$\{D_2$, r05$\}$; (c) {cage9-I, cage9-II}; (d) {pf2177-I, pf2177-II}

In this experiment, we investigate how the inaccuracy in computing the inner iterations of both KIDS-I and KIDS-II affects the final accuracy of the approximate solutions. For KIDS-I, at each iteration, we use LSQR with stopping tolerance τ to iteratively solving (4.4) and (4.8) for approximating $x_1^\dagger$ and $x_2^\dagger$, respectively. For KIDS-II, we first compute an exact solution $\tilde{x}_1^\dagger$, and then use LSQR with stopping tolerance τ to iteratively solving (4.8) for approximating $\tilde{x}_2^\dagger$. The stopping tolerance value τ for LSQR are set to 10^{-10} and 10^{-8}. For simplicity, we only present the results for the first and third examples, as the results for the other two examples are similar. From Figure 3, we observe that the value of τ significantly impacts the final accuracy of x_k, with the accuracy being approximately on the order of $\mathcal{O}(\tau)$. On the other hand, the convergence rate is not affected very much. It is important to investigate how the final accuracy of the computed solution is influenced by the value of τ, especially because, for large-scale problems, it is not feasible to compute the inner iterations accurately. This aspect should be explored further in future work.

In this experiment, we explore how the solution accuracy of $\tilde{x}_1^\dagger = \operatorname{argmin}_x \|Cx - d\|_2$ influences the final accuracy of $x^\dagger$. To obtain an approximate $\tilde{x}_1^\dagger$, we apply the

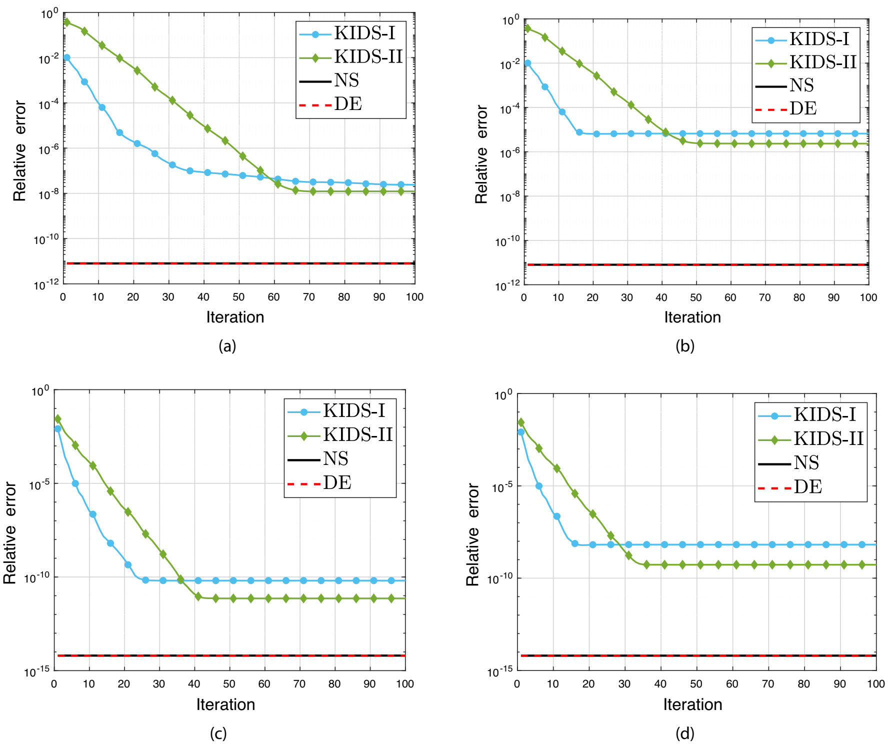

Fig. 3 The convergence history of KIDS-I and KIDS-II with respect to the true solution, where the inner iterations are approximated by solving (4.4) and (4.8) by LSQR with stopping tolerance τ. (a) $\{D_1$, lp_bnl2$\}$, $\tau = 10^{-10}$; (b) $\{D_1$, lp_bnl2$\}$, $\tau = 10^{-8}$; (c) {cage9-I, cage9-II}, $\tau = 10^{-10}$; (d) {cage9-I, cage9-II}, $\tau = 10^{-8}$

LSQR algorithm to solve $\min_x \|Cx - d\|_2$ with a stopping tolerance set to τ_1. The inner iteration is approximated by solving (4.8) using the LSQR algorithm, with the stopping tolerance set to τ_2. We only show the experimental results for the first example, as the results for the other examples are similar. First, we set $\tau_1 = 10^{-10}$ and $\tau_1 = 10^{-8}$, respectively, and set $\tau_2 = 0$, meaning that we compute the inner iteration accurately. The convergence history is shown in Fig. 4a. We observe that an inaccurate $\tilde{x}_1^\dagger$ affects the final accuracy of $x^\dagger$, even when the inner iterations are computed accurately. Second, we set $\tau_1 = \tau_2 = 10^{-10}$ and $\tau_1 = \tau_2 = 10^{-8}$, respectively. The convergence history is shown in Fig. 4b. We observe that when the inner iterations are performed with the same accuracy as $\tilde{x}_1^\dagger$, then the final accuracy of $x^\dagger$ is comparable to the accuracy achieved when the inner iterations are computed accurately. Since the solution error of $\tilde{x}_1^\dagger$ may be amplified in the subsequent computation, analyzing the impact of the inaccuracy in $\tilde{x}_1^\dagger$ on the final accuracy of $x^\dagger$ is more complex than simply analyzing the inner iterations. This also implied that KIDS-II can be more susceptible to computational errors than KIDS-I. A systematic comparison of the two algorithms and their susceptibility to computational errors will be explored in future work.

5.2 Large-scale problems

We propose the following steps to construct a large-scale LSE problem. To clearly indicate the dimensions of each block matrix, we use $\mathbf{1}_r$ to denote a vector of size $r \times 1$ with all entries equal to 1, and use $\mathbf{0}_r$ to denote either a zero matrix of size $r \times r$ or a zero vector of size $r \times 1$, depending on the context.

(1) Given positive integers n, r_1, r_2 and r_3, satisfying $r_1 + r_2 + r_3 = n$. Let

$$A = \begin{pmatrix} I_{r_1} & & \\ & \Sigma_A & \\ & & \mathbf{0}_{r_3} \end{pmatrix} D, \quad C = \begin{pmatrix} \mathbf{0}_{r_1} & & \\ & \Sigma_C & \\ & & I_{r_3} \end{pmatrix} D,$$

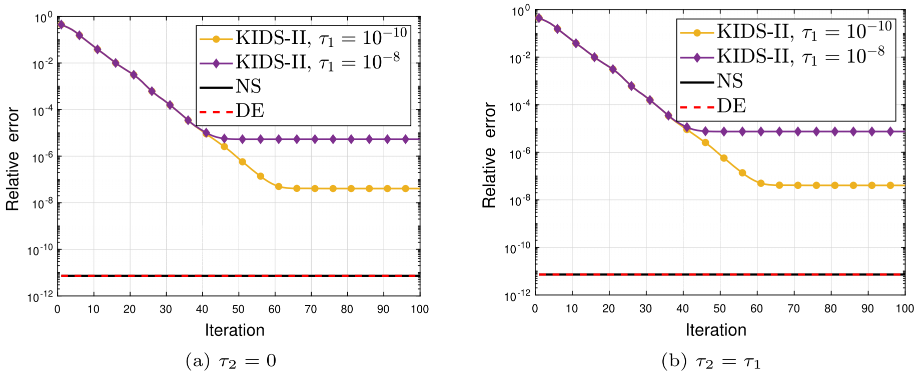

Fig. 4 The convergence history of KIDS-II with respect to the true solution, where $\tilde{x}_1^\dagger = \operatorname{argmin}_x \|Cx - d\|_2$ is computed by LSQR with stopping tolerance τ_1, and the inner iteration is approximated by solving (4.8) by LSQR with stopping tolerance τ_2. The test example is $\{D_1, \text{lp\_bnl2}\}$

where $\Sigma_A = \mathrm{diag}(a)$ with $a = \texttt{linspace(0.99, 0.01, r}_2\texttt{)}$, $\Sigma_C = (I_{r_2} - \Sigma_A^2)^{1/2}$, and $D = \mathrm{diag}(d)$ with $d = \texttt{linspace(1, 100, n)}$.

(2) Compute the following matrix $B \in \mathbb{R}^{n\times r_1}$ whose columns span $\mathcal{N}(C)$:

$$B = D^{-1}\begin{pmatrix} I_{r_1} \\ \mathbf{0}_{r_2} \\ \mathbf{0}_{r_3} \end{pmatrix}.$$

(3) Let $G = A^{\mathsf{T}}A + C^{\mathsf{T}}C$ and $z_1 = (\mathbf{0}_{r_1}^{\mathsf{T}}, \mathbf{1}_{r_2}^{\mathsf{T}}, \mathbf{0}_{r_3}^{\mathsf{T}})^{\mathsf{T}}$. Compute $x_1^\dagger = Gz_1$ and $d = Cx_1^\dagger$.

(4) Let $z_2 = \texttt{linspace(100, 1, r}_1\texttt{)}^{\mathsf{T}}$. Compute $x_2^\dagger = Bz_2$ and $b = Ax_2^\dagger$.

(5) Compute $x^\dagger = x_1^\dagger + x_2^\dagger$.

Using Theorem 3.2 and Theorem 3.3, we can verify that the above construction ensures that $x_1^\dagger$ is the minimum 2-norm solution of (3.3) with $K = C$ and $L = A$, and $x_2^\dagger$ is the minimum 2-norm solutions of (3.12). In the experiment, we fix $r_1 = 200$, $r_2 = 300$, while varying the value of r_3, which leads to $n = 6000, 8000, 10000, 12000, 14000, 16000, 18000$.

We use the above large-scale problems to evaluate the performance of KIDS-I and KIDS-II on large sparse LSE problems, comparing them with the NS and DE methods. The comparison of the relative errors of the solutions are shown in Table 3, where the corresponding total iterations of KIDS-I and KIDS-II are shown in parentheses. The corresponding running time of both the methods are listed in Table 4. For $n = 6000, 8000, 10000, 12000, 14000$, we test both the direct methods and the iterative methods, where all inner iterations are computed either accurately (i.e., $\tau = 0$) or approximately by solving (4.4) and (4.8) with LSQR using a stopping tolerance of $\tau = 10^{-12}$. For $n = 16000, 18000$, we only apply KIDS-I and KIDS-II with $\tau = 10^{-12}$.

As shown in Table 3, both KIDS-I and KIDS-II with $\tau = 0$ achieve very low relative errors across all tested problem sizes. The relative errors of KIDS-I are only slightly higher than those of the two direct methods, which typically serve as baselines for evaluating the accuracy of numerical algorithms. Meanwhile, the relative errors of KIDS-II are consistently slightly higher than those of KIDS-I for all tested problem sizes. Since KIDS-I and KIDS-II with $\tau = 0$ rely on matrix factorizations to compute the inner iterations, their running time grows in a manner similar to that of the direct methods NS and DE. This trend is clearly illustrated in Table 4, where all four methods exhibit comparable total running times for problem sizes $n = 6000, 8000, 10000, 12000, 14000$.

As shown in both Tables 3 and 4, when the inner iterations are computed approximately with $\tau = 10^{-12}$, the relative errors of KIDS-I and KIDS-II are higher compared to those obtained with $\tau = 0$. However, the running time is significantly reduced, particularly for large-scale problems since matrix factorizations are no longer used. This makes them much faster than the direct methods on large sparse LSE problems. These results demonstrate that relaxing the inner iteration precision offers a practical trade-off between solution accuracy and computational efficiency. We remark that for our large sparse matrices, solving (4.4) and (4.8) using LSQR converges very quickly, often within just a few iterations. However, for real-world large-scale problems, the inner iterations may require more time. Even so, they can still be more efficient than direct methods, especially if the inner iteration precision is moderately relaxed.

Table 3 Comparison of the relative errors of the solutions at the final iterations (indicated in parentheses) of KIDS-I and KIDS-II, along with the relative errors of the solutions computed by NS and DE

n	KIDS-I		KIDS-II		NS	DE
	$\tau = 0$	$\tau = 10^{-12}$	$\tau = 0$	$\tau = 10^{-12}$		
6000	$5.46 \times 10^{-16}(100)$	$2.64 \times 10^{-14}(80)$	$8.17 \times 10^{-15}(100)$	$2.12 \times 10^{-11}(60)$	8.76×10^{-17}	6.33×10^{-17}
8000	$5.79 \times 10^{-16}(90)$	$2.04 \times 10^{-14}(80)$	$4.22 \times 10^{-15}(90)$	$1.32 \times 10^{-11}(60)$	9.83×10^{-17}	6.79×10^{-17}
10000	$6.19 \times 10^{-16}(90)$	$1.76 \times 10^{-14}(80)$	$2.52 \times 10^{-15}(90)$	$5.66 \times 10^{-12}(60)$	8.43×10^{-17}	6.50×10^{-17}
12000	$5.53 \times 10^{-16}(90)$	$1.73 \times 10^{-14}(80)$	$1.86 \times 10^{-15}(90)$	$4.41 \times 10^{-12}(40)$	7.84×10^{-17}	6.36×10^{-17}
14000	$6.63 \times 10^{-16}(90)$	$1.99 \times 10^{-14}(80)$	$1.25 \times 10^{-15}(90)$	$2.32 \times 10^{-12}(40)$	1.28×10^{-16}	6.65×10^{-17}
16000	–	$1.35 \times 10^{-14}(80)$	–	$1.44 \times 10^{-12}(40)$	–	–
18000	–	$1.07 \times 10^{-14}(80)$	–	$1.05 \times 10^{-12}(40)$	–	–

Table 4 Comparison of running times (in seconds) for KIDS-I, KIDS-II, NS, and DE

n	KIDS-I		KIDS-II		NS	DE
	$\tau = 0$	$\tau = 10^{-12}$	$\tau = 0$	$\tau = 10^{-12}$		
6000	98.24	0.55	103.75	0.52	97.10	78.65
8000	244.69	0.58	242.86	0.65	227.14	181.77
10000	451.59	0.83	464.38	1.14	418.68	360.21
12000	702.43	0.88	732.80	0.85	681.10	593.02
14000	1186.49	0.95	1161.88	0.84	1073.51	954.86
16000	–	1.02	–	0.99	–	–
18000	–	1.10	–	1.09	–	–

Test for operator-form matrices A and C In this experiment, we construct two large-scale linear operators $A \in \mathbb{R}^{(n-2)\times n}$ and $C \in \mathbb{R}^{r\times n}$ in a matrix-free fashion, i.e., without explicitly forming the matrices. Both operators are implemented as function handles that compute matrix-vector products efficiently, which is critical for large-scale problems where explicit matrices are infeasible.

The operator-form matrix A is a modified version of the discrete second-order finite difference operator defined in (5.3), where the diagonal entries are replaced by 5, i.e.,

$$(Ax)_i = x_i - 5x_{i+1} + x_{i+2}, \quad i = 1, \dots, n-2,$$

where $x \in \mathbb{R}^n$ is an input vector. The matrix A is implemented as a function handle `Afun`(x, mode) supporting the following two modes:

- `mode = 'notransp'` computes the forward multiplication $y = Ax$, returning a vector $y \in \mathbb{R}^{n-2}$ with components

$$y_i = x_i - 5x_{i+1} + x_{i+2}, \quad i = 1, \dots, n-2.$$

- `mode = 'transp'` computes the transpose multiplication $y = A^\mathsf{T}x$, where $x \in \mathbb{R}^{n-2}$ and the output $y \in \mathbb{R}^n$ is given by

$$\begin{cases} y_1 = x_1,\, y_2 = -5x_1 + x_2, \\ y_i = x_{i-2} - 5x_{i-1} + x_i, \quad i = 3, \dots, n-2, \\ y_{n-1} = -5x_{n-2} + x_{n-3}, \\ y_n = x_{n-2}. \end{cases}$$

The operator-form matrix $C \in \mathbb{R}^{r\times n}$ is constructed as follows. We set $r = n - t$ with $t = 500$, and construct the operator $C : \mathbb{R}^n \to \mathbb{R}^r$ as

$$C = MP^\mathsf{T},$$

where $P \in \mathbb{R}^{n\times n}$ is an orthogonal matrix generated by the Matlab built-in function `gallery('orthog', n)`, and $M = \begin{pmatrix} I_r & \mathbf{0} \end{pmatrix} \in \mathbb{R}^{r\times n}$. Thus, for any vector $x \in \mathbb{R}^n$ we have

$$Cx = M(P^{\mathsf{T}}x) = (P^{\mathsf{T}}x)_{1:r},$$

and for any $y \in \mathbb{R}^r$ we have

$$C^{\mathsf{T}}y = P\begin{pmatrix} y \\ \mathbf{0} \end{pmatrix}.$$

Thus, the matrix C is implemented as a function handle `Cfun`(x, mode) with the following two modes:

- `mode = 'notransp'` computes the forward product

$$y = Cx = (P^{\mathsf{T}}x)_{1:r}.$$

- `mode = 'transp'` computes the transpose product

$$y = C^{\mathsf{T}}x = P\begin{pmatrix} x \\ \mathbf{0} \end{pmatrix}.$$

From the above construction, an explicit basis for $\mathcal{N}(C)$ is given by the columns of the matrix

$$B = P\begin{pmatrix} \mathbf{0} \\ I_d \end{pmatrix} \in \mathbb{R}^{n\times t}.$$

We set $n = 10000$ and follow the approach outlined in Section 5.1 to construct an LSE problem. For this problem, methods based on matrix factorizations are not applicable; instead, iterative methods that rely solely on matrix-vector products should be used.

Figure 5 presents the experimental results, where 20 iterations are performed for both KIDS-I and KIDS-II. In each case, the inner iterations are approximated by solving

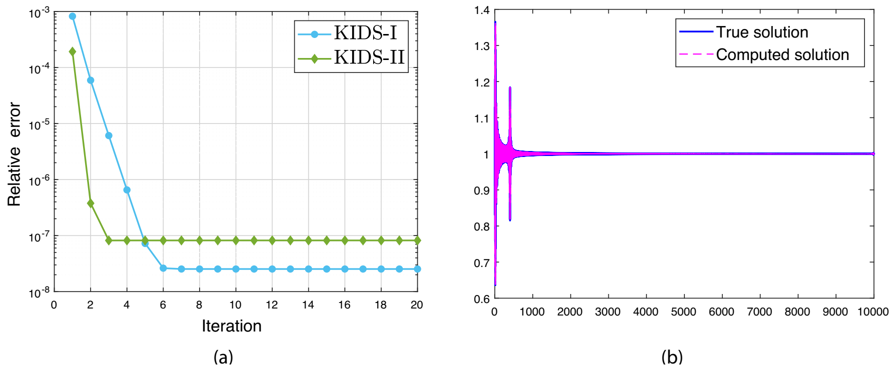

Fig. 5 Test results for the LSE problem with operator-form matrices. (a) Convergence history of KIDS-I and KIDS-II with respect to the true solution. (b) Curves for the true and computed solutions obtained by KIDS-I at the final iteration

(4.4) and (4.8) using LSQR, with a stopping tolerance of $\tau = 10^{-8}$. It can be observed that the convergence is very fast, possibly because the dominant components of the true solution lies in a low-dimensional Krylov subspace generated by KIDS-I and KIDS-II. We note, however, that for a general LSE problem, the convergence may be much slower than in this example. Constructing a suitable example that clearly demonstrates such slow convergence is not a straightforward task. The final accuracy of the solutions computed by both KIDS-I and KIDS-II are influenced by the value of τ, consistent with the results demonstrated in Section 5.1. We also plot the curve corresponding to x_k computed by KIDS-I at the final iteration, alongside the true solution $x^{\dagger}$. These results demonstrate the effectiveness of the proposed algorithms for LSE problems with operator-form matrices.

6 Conclusion and outlook

In this paper, we have introduced a novel approach to solving the LSE problems by reformulating them as operator-type LS problems. This perspective allows us to decompose the solution of the LSE problem into two components, each corresponding to a simpler operator-based LS problem. We have derived two types of decomposed-form solutions, and building on this decomposition, we have developed two Krylov subspace based iterative methods that efficiently approximate the solution without matrix factorizations. The two proposed algorithms, named KIDS-I and KIDS-II, follow a nested inner-outer structure, where the inner subproblem can be computed iteratively. We have proposed an approach to construct the LSE problems for testing purposes, and used several test examples to demonstrate the effectiveness of the algorithms.

The primary computational bottleneck of the proposed algorithms lies in the inner iteration. In future work, we plan to explore additional theoretical and computational strategies to improve the efficiency of this step and thereby accelerate the overall performance of the proposed algorithms.

Acknowledgements The authors are grateful to the three anonymous referees who provided invaluable comments about the initial versions of this manuscript.

Author Contributions HL: Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. No funding is related to this research.

Data Availability The source code and data used in this work are available at https://github.com/Machealb/LSE_solver.

Declarations

Competing interests The authors declare no competing interests.

References

1. Damm, T., Stahl, D.: Linear least squares problems with additional constraints and an application to scattered data approximation. Linear Algebra Appl. **439**(4), 933–943 (2013)
2. Fear, B.: Visualizing statistical models and concepts (2004)
3. Zhu, Y., Li, X.R.: Recursive least squares with linear constraints **7**(3), 287–311 (2007)
4. Pisinger, G., Zimmermann, A.: Bivariate least squares approximation with linear constraints. BIT Numer. Math. **47**, 427–439 (2007)
5. Wei, M.: Algebraic properties of the rank-deficient equality-constrained and weighted least squares problems. Linear Algebra Appl. **161**, 27–43 (1992)
6. Gulliksson, M.: Backward error analysis for the constrained and weighted linear least squares problem when using the weighted QR factorization. SIAM J. Matrix Anal. Appl. **16**(2), 675–687 (1995)
7. Ding, J., Hang, W.: New perturbation results for equality-constrained least squares problems. Linear Algebra Appl. **272**(1–3), 181–192 (1998)
8. Gulliksson, M., Wedin, P.-Å.: Perturbation theory for generalized and constrained linear least squares. Numerical linear algebra with applications **7**(4), 181–195 (2000)
9. Gulliksson, M., Jin, X.-Q., Wei, Y.-M.: Perturbation bounds for constrained and weighted least squares problems. Linear Algebra Appl. **349**(1–3), 221–232 (2002)
10. Hanson, R.J., Lawson, C.L.: Extensions and applications of the householder algorithm for solving linear least squares problems. Math. Comput. **23**(108), 787–812 (1969)
11. Lawson, C.L., Hanson, R.J.: Solving Least Squares Problems. SIAM, Philadelphia (1995)
12. Schittkowski, K., Stoer, J.: A factorization method for the solution of constrained linear least squares problems allowing subsequent data changes. Numer. Math. **31**, 431–463 (1978)
13. Stoer, J.: On the numerical solution of constrained least-squares problems. SIAM J. Numer. Anal. **8**(2), 382–411 (1971)
14. Chan, T.F.: Rank revealing QR factorizations. Linear Algebra Appl. **88**, 67–82 (1987)
15. Chan, T.F., Hansen, P.C.: Some applications of the rank revealing QR factorization. SIAM J. Sci. Stat. Comput. **13**(3), 727–741 (1992)
16. Gu, M., Eisenstat, S.C.: Efficient algorithms for computing a strong rank-revealing QR factorization. SIAM J. Sci. Comput. **17**(4), 848–869 (1996)
17. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. The Johns Hopkins University Press, Baltimore (2013)
18. Golub, G.: Numerical methods for solving linear least squares problems. Numer. Math. **7**, 206–216 (1965)
19. Heath, M.T.: Some extensions of an algorithm for sparse linear least squares problems. SIAM J. Sci. Stat. Comput. **3**(2), 223–237 (1982)
20. Van Loan, C.: On the method of weighting for equality-constrained least-squares problems. SIAM J. Numer. Anal. **22**(5), 851–864 (1985)
21. Barlow, J.L., Handy, S.L.: The direct solution of weighted and equality constrained least-squares problems. SIAM J. Sci. Stat. Comput. **9**(4), 704–716 (1988)
22. Gulliksson, M.: Iterative refinement for constrained and weighted linear least squares. BIT Numer. Math. **34**, 239–253 (1994)
23. Stewart, G.W.: On the weighting method for least squares problems with linear equality constraints. BIT Numer. Math. **37**(4), 961–967 (1997)
24. Liesen, J., Strakos, Z.: Krylov Subspace Methods: Principles and Analysis. Oxford University Press, Oxford (2013)
25. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis **2**(2), 205–224 (1965)

26. Caruso, N.A., Novati, P.: Convergence analysis of LSQR for compact operator equations. Linear Algebra Appl. **583**, 146–164 (2019)
27. Li, H.: A new interpretation of the weighted pseudoinverse and its applications. SIAM J. Matrix Anal. Appl. **46**(2), 934–956 (2025)
28. Li, H.: Characterizing GSVD by singular value expansion of linear operators and its computation. SIAM J. Matrix Anal. Appl. **46**(1), 439–465 (2025)
29. Scott, J., Tůma, M.: A null-space approach for large-scale symmetric saddle point systems with a small and non zero (2, 2) block. Numerical Algorithms **90**(4), 1639–1667 (2022)
30. Scott, J., Tůma, M.: Solving large linear least squares problems with linear equality constraints. BIT Numer. Math. **62**(4), 1765–1787 (2022)
31. Scott, J., Tůma, M.: Solving mixed sparse-dense linear least-squares problems by preconditioned iterative methods. SIAM J. Sci. Comput. **39**(6), 2422–2437 (2017)
32. Björck, Å.: A general updating algorithm for constrained linear least squares problems. SIAM J. Sci. Stat. Comput. **5**(2), 394–402 (1984)
33. Zhdanov, A.I.: The method of augmented regularized normal equations. Comput. Math. Math. Phys. **52**, 194–197 (2012)
34. Zhdanov, A.I., Gogoleva, S.Y.: Solving least squares problems with equality constraints based on augmented regularized normal equations. Appl. Math. E-Notes **15**, 218–224 (2015)
35. Scott, J., Tůma, M.: A computational study of using black-box QR solvers for large-scale sparse-dense linear least squares problems. ACM Trans. Math. Softw. (TOMS) **48**(1), 1–24 (2022)
36. Eldén, L.: A weighted pseudoinverse, generalized singular values, and constrained least squares problems. BIT Numer. Math. **22**, 487–502 (1982)
37. Paige, C.C., Saunders, M.A.: LSQR: An algorithm for sparse linear equations and sparse least squares. ACM Trans. Math. Software **8**, 43–71 (1982)
38. Davis, T.A., Hu, Y.: The university of florida sparse matrix collection. ACM Trans. Math. Software (TOMS) **38**(1), 1–25 (2011)