LSLQ: AN ITERATIVE METHOD FOR LINEAR LEAST-SQUARES WITH AN ERROR MINIMIZATION PROPERTY*

RON ESTRIN†, DOMINIQUE ORBAN‡, AND MICHAEL A. SAUNDERS§

Abstract. We propose an iterative method named LSLQ for solving linear least-squares problems of any shape. The method is based on the Golub and Kahan (1965) process, where the dominant cost is products with the linear operator and its transpose. In the rank-deficient case, LSLQ identifies the minimum-length least-squares solution. LSLQ is formally equivalent to SYMMLQ applied to the normal equations, so that the current estimate's Euclidean norm increases monotonically while the associated error norm decreases monotonically. We provide lower and upper bounds on the error in the Euclidean norm along the LSLQ iterations. The upper bound translates to an upper bound on the error norm along the LSQR iterations, which was previously unavailable, and provides an error-based stopping criterion involving a transition to the LSQR point. We report numerical experiments on standard test problems and on a full-wave inversion problem arising from geophysics in which an approximate least-squares solution corresponds to an approximate gradient of a relevant penalty function that is to be minimized.

1. Introduction. We propose an iterative method (LSLQ) for solving two ubiquitous problems in computational science—the least-squares problem and the least-norm problem:

$$\text{(LS)} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \ \tfrac{1}{2}\|Ax-b\|^2,$$

$$\text{(LN)} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \ \tfrac{1}{2}\|x\|^2 \quad \text{subject to} \ \ Ax=b,$$

both of which include consistent linear systems $Ax = b$ as a special case. The norm $\|\cdot\|$ is Euclidean and A may be an m-by-n matrix, but we assume more generally that $A : \mathbb{R}^n \to \mathbb{R}^m$ is a linear operator because only operator-vector products of the form Au and A^Tv are required. We often refer to the optimality conditions of (LS), namely the normal equations

$$\text{(NE)} \qquad A^TAx = A^Tb.$$

When $Ax = b$ is consistent, LSLQ identifies a solution of (LN). If $\operatorname{rank}(A) < n$, LSLQ finds the minimum-length solution (MLS) $x_\star = A^\dagger b$, where $A^\dagger$ is the pseudoinverse.

Motivation: monitoring the error. We briefly describe why an iterative method for least squares with an error minimization property is of interest.

Van Leeuwen and Herrmann (2013) describe a penalty method for PDE-constrained optimization in the context of a seismic inverse problem. The penalty objective $\phi_\rho(\mathbf{m}, \mathbf{u})$ depends on the control variable $\mathbf{m}$ and the wavefields $\mathbf{u}$, where $\rho > 0$ is a penalty parameter. For fixed values of ρ and $\mathbf{m}$, the wavefields $\mathbf{u}(\mathbf{m})$ satisfying

*Version of January 11, 2018.

†Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA, USA. E-mail: restrin@stanford.edu.

‡GERAD and Department of Mathematics and Industrial Engineering, École Polytechnique, Montréal, QC, Canada. E-mail: dominique.orban@gerad.ca. Research partially supported by an NSERC Discovery Grant.

§Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA, USA. E-mail: saunders@stanford.edu. Research partially supported by the National Institute of General Medical Sciences of the National Institutes of Health, award U01GM102098.

$\nabla_{\mathbf{u}}\phi_\rho(\mathbf{m}, \mathbf{u}(\mathbf{m})) = 0$ can be found as the solution of a linear least-squares (LS) problem in $\mathbf{u}$. The gradient of ϕ with respect to $\mathbf{m}$ is subsequently expressed as a linear function of $\mathbf{u}(\mathbf{m})$, say

$$\nabla_{\mathbf{m}}\phi_\rho(\mathbf{m}, \mathbf{u}(\mathbf{m})) = G\,\mathbf{u}(\mathbf{m}) - g$$

for a certain matrix G and vector g. Assume now that an inexact solution $\tilde{\mathbf{u}}$ of the LS problem for $\mathbf{u}(\mathbf{m})$ is determined. The error in $\mathbf{u}$ translates directly into an error in the gradient of the penalty function, for

$$\|\nabla_{\mathbf{m}}\phi_\rho(\mathbf{m}, \mathbf{u}) - \nabla_{\mathbf{m}}\phi_\rho(\mathbf{m}, \tilde{\mathbf{u}})\| \leqslant \|G\|\,\|\mathbf{u} - \tilde{\mathbf{u}}\|, \qquad \mathbf{u} \equiv \mathbf{u}(\mathbf{m}). \tag{1}$$

If a derivative-based optimization method is to be used to minimize the penalty function, there is interest in a method to approximate $\mathbf{u}$ in which the error is monotonically decreasing. Indeed, the convergence properties of derivative-based optimization methods are not altered provided the gradient is computed sufficiently accurately in the sense that the left-hand side of (1) is sufficiently small compared to $\|\nabla_{\mathbf{m}}\phi_\rho(\mathbf{m}, \mathbf{u})\|$ (Conn, Gould, and Toint, 2000, §8.4.1.1).

In the following sections, we introduce the LSLQ method. We now comment on the necessity for LSLQ in order to monitor the error reliably. At this stage, it is sufficient to say that LSLQ applied to problem (LS) is equivalent to SYMMLQ (Paige and Saunders, 1975) applied to (NE). LSLQ fits in the category of Krylov-subspace methods based on the Golub and Kahan (1965) process, and in that sense is related to LSQR (Paige and Saunders, 1982a) and LSMR (Fong and Saunders, 2011) (equivalent to CG and MINRES applied to (NE)). As far as error monitoring is concerned, the key advantage that LSLQ inherits from SYMMLQ is that the solution estimate is updated along orthogonal directions. As a consequence, the solution norm increases and the error decreases along the iterations. It happens that both LSQR and LSMR share those properties (Fong and Saunders, 2012, Table 5.2) but with important differences. First, LSLQ's orthogonal updates suggest error lower and upper bounds initially developed for SYMMLQ by Estrin, Orban, and Saunders (2016), and which are the subject of section 4. Second, the error is *minimized* in LSLQ, while it is only monotonic in LSQR and LSMR. In spite of the latter observation, the error along the LSQR and LSMR iterations is typically smaller than for the LSLQ iterations by a few orders of magnitude—see Proposition 1. This is not a contradiction because LSLQ minimizes the error in a transformation of the Krylov subspace. Figure 1 illustrates a typical scenario, where the error is represented along the LSQR, LSMR, and LSLQ iterations on two over-determined problems arising from an animal breeding application (Hegland, 1990, 1993), and where we consider that the solution obtained with a complete orthogonal decomposition is the exact solution.

It appears from Figure 1 that LSQR is more appealing than LSLQ if one is interested in minimizing the error. The difficulty is that LSQR does not lend itself to obvious error lower and upper bounds because it is not naturally formulated in terms of the Euclidean norm and its solution estimate is not updated along orthogonal directions. Estimates of the error in the conjugate gradient (CG) method (Hestenes and Stiefel, 1952) applied to a symmetric and positive definite system have been developed in the literature, an effort led chiefly by Meurant (2005). Those estimates could be applied to LSQR but unfortunately they are only estimates and have not been proved to be lower or upper bounds. Thus it is difficult to terminate the LSQR iterations reliably with a guaranteed error level. Fortunately, SYMMLQ is closely related to CG and it is possible to transition cheaply from a SYMMLQ iterate to a corresponding CG iterate. LSLQ inherits that property and it is possible to transition

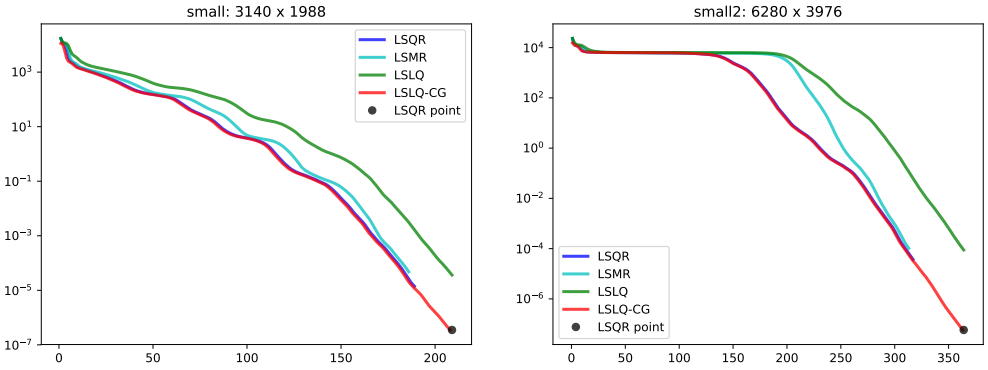

FIG. 1. *Error along the LSQR, LSMR and LSLQ iterations on problems* small *and* small2 *from the animal breeding set. The red curve corresponds to the LSQR iterates generated as a by-product during the LSLQ iterations. The horizontal axis represents the number of iterations (each involving a product with A and a product with A^T).*

to a related LSQR iterate at any iteration. The red curve in Figure 1 represents the error observed at each LSQR point obtained by transitioning from the then-current LSLQ point. Note the high accuracy to which the red and blue curves match; they are essentially superposed. The black dot represents the error observed after transitioning from the final LSLQ iterate to the LSQR point. Note also that because the stopping rule for all methods involves the residual of the normal equations, the curves end at different abscissae.

Our main objective is to exploit the reliable lower and upper bounds on the LSLQ error based on those developed for SYMMLQ by Estrin et al. (2016). The upper bound on the LSLQ errors combined with the tight relationship between LSLQ and LSQR leads to an upper bound on the LSQR error. Thus it becomes possible to end the LSLQ iterations as soon as it becomes apparent that the upper bound on the LSQR error is below a prescribed tolerance.

Both problems used in Figure 1 are rank-deficient and the curves indicate that all methods tested identify the MLS solution. Problem small2 is included in the illustration because it is an example where the error plateaus. We return to this point in section 4.

We do not consider LSMR further here for two reasons. First, it is a consequence of (Hestenes and Stiefel, 1952, Theorem 7:5) that the LSMR error is monotonic but equal to or larger than that of LSQR—see also (Fong and Saunders, 2012, Theorem 2.4). Second, LSMR is a variant of MINRES (Paige and Saunders, 1975) and we know of no result relating the errors along the MINRES iterations on a symmetric positive definite system to those along the SYMMLQ iterations.

Notation. We use Householder notation (A, b, β for matrix, vector, scalar) with the exception of c and s, which denote scalars used to define reflections. Unless specified otherwise, $\|A\|$ and $\|x\|$ denote the Euclidean norm of matrix A and vector x. For rectangular A, we order its singular values according to $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_{\min(m,n)} \geqslant 0$. For symmetric positive definite M, we define the M-norm of u via $\|u\|_M^2 := u^T M u$.

2. Derivation of the method.

In this section, we describe LSLQ using the process/method/implementation framework.

2.1. The Golub-Kahan process. LSLQ is based on the Golub and Kahan (1965) process described as Algorithm 1, with A and b as in (LS) or (LN). In line 1, $\beta_1 u_1 = b$ is short for "$\beta_1 = \|b\|$; if $\beta_1 = 0$ then exit; else $u_1 = b/\beta_1$". Similarly for line 2 and the main loop. In exact arithmetic, the algorithm will terminate with $k = \ell \leqslant \min(m, n)$ and either $\alpha_{\ell+1}$ or $\beta_{\ell+1} = 0$.

Algorithm 1 Golub-Kahan Bidiagonalization Process

Require: A, b
1: $\beta_1 u_1 = b$
2: $\alpha_1 v_1 = A^T u_1$
3: **for** $k = 1, 2, \ldots$ **do**
4: $\quad \beta_{k+1} u_{k+1} = A v_k - \alpha_k u_k$
5: $\quad \alpha_{k+1} v_{k+1} = A^T u_{k+1} - \beta_{k+1} v_k$
6: **end for**

We define $U_k := \begin{bmatrix} u_1 & \cdots & u_k \end{bmatrix}$, $V_k := \begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix}$, and

$$L_k := \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \end{bmatrix}, \quad B_k := \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix} = \begin{bmatrix} L_k \\ \beta_{k+1} e_k^T \end{bmatrix}. \tag{2}$$

The situation after k iterations of Algorithm 1 can be summarized as

$$AV_k = U_{k+1} B_k, \tag{3a}$$

$$A^T U_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T = V_{k+1} L_{k+1}^T, \tag{3b}$$

and the identities $U_k^T U_k = I_k$ and $V_k^T V_k = I_k$ are satisfied in exact arithmetic.

2.2. LSLQ: method. By definition, LSLQ applied to (LS) is equivalent to SYMMLQ applied to (NE). The identities (3) yield

$$\begin{aligned} A^T A V_k &= A^T U_{k+1} B_k \\ &= V_k B_k^T B_k + \alpha_{k+1} v_{k+1} e_{k+1}^T B_k \\ &= V_k B_k^T B_k + \alpha_{k+1} \beta_{k+1} v_{k+1} e_k^T \\ &= V_{k+1} H_k, \end{aligned} \tag{4}$$

where

$$H_k := \begin{bmatrix} B_k^T B_k \\ \alpha_{k+1} \beta_{k+1} e_k^T \end{bmatrix}, \tag{5}$$

while lines 1 and 2 of Algorithm 1 yield $A^T b = \alpha_1 \beta_1 v_1$. From here on, we use the shorthand

$$\bar{\alpha}_k := \alpha_k^2 + \beta_{k+1}^2, \quad \text{and} \quad \bar{\beta}_k := \alpha_k \beta_k, \quad k = 1, 2, \ldots \tag{6}$$

As noted by Fong and Saunders (2011), the above characterizes the situation after $k + 1$ steps of the Lanczos (1950) process applied to $A^T A$ with initial vector $A^T b$. For

all $k \geqslant 1$, we denote

$$T_k := B_k^T B_k = \begin{bmatrix} \bar{\alpha}_1 & \bar{\beta}_2 & & \\ \bar{\beta}_2 & \bar{\alpha}_2 & \ddots & \\ & \ddots & \ddots & \bar{\beta}_k \\ & & \bar{\beta}_k & \bar{\alpha}_k \end{bmatrix}, \qquad H_k = \begin{bmatrix} T_k \\ \bar{\beta}_{k+1} e_k^T \end{bmatrix}. \tag{7}$$

Note that T_k is k-by-k and tridiagonal, and H_k is $(k+1)$-by-k.

The k-th iteration of CG applied to (NE) computes $x_k^C = V_k y_k^C$, where y_k^C is the solution of the subproblem

$$T_k y_k^C = \bar{\beta}_1 e_1. \tag{8}$$

The resulting x_k^C can be shown to solve the subproblem

$$\underset{x \in \mathcal{K}_k}{\text{minimize}} \ \|x_\star - x\|_{A^TA}, \tag{9}$$

where $\mathcal{K}_k := \text{Span}\{A^Tb, (A^TA)A^Tb, \ldots, (A^TA)^kA^Tb\}$ is the k-th Krylov subspace associated with A^TA and A^Tb. LSQR (Paige and Saunders, 1982a,b) is equivalent in exact arithmetic. By contrast, the k-th iteration of SYMMLQ applied to (NE) computes y_k^L as the solution of

$$\text{minimize } \tfrac{1}{2}\|y_k^L\|^2 \quad \text{subject to } H_{k-1}^T y_k^L = \bar{\beta}_1 e_1, \tag{10}$$

and sets $x_k^L := V_k y_k^L$. Note that H_{k-1}^T is the first $k-1$ rows of T_k and may be written as $H_{k-1}^T = B_{k-1}^T L_k$. It can be shown that x_k^L solves the subproblem

$$\underset{x \in A^TA\mathcal{K}_{k-1}}{\text{minimize}} \ \|x_\star - x\|. \tag{11}$$

One important distinction between (9) and (11) is that $x_k^C \in \mathcal{K}_k$ while $x_k^L \in (A^TA)\mathcal{K}_{k-1}$, a subset of $\mathcal{K}_k$. By construction, $\|x_\star - x_k\|$ is monotonic along the LSLQ iterates, but as mentioned earlier, it also happens to be monotonic along the LSQR iterates. Somewhat surprisingly, the error is always smaller along the LSQR iterates than along the LSLQ iterates, as formalized by the next result.

PROPOSITION 1. *Let $x_k^C = V_k y_k^C$ and $x_k^L = V_k y_k^L$ with y_k^C and y_k^L defined as in (8) and (10). Then, for all k,*

$$\|x_k^L\| \leqslant \|x_k^C\|,$$
$$\|x_\star - x_k^C\| \leqslant \|x_\star - x_k^L\|.$$

Proof. The result follows from applying (Estrin et al., 2016, Theorem 6) to (NE). ∎

Note first that Proposition 1 holds whether A has full column rank or not. Note also that Proposition 1 does not contradict the definition of LSLQ as minimizing the error because the latter is not minimized over the same subspace as that used during the k-th iteration of LSQR.

In the next section we describe the implementation of LSLQ, and we return to the two errors in section 4.

2.3. LSLQ: implementation. We identify y_k^L by way of an LQ factorization of H_{k-1}^T, which we compute via an implicit LQ factorization of $T_k = B_k^T B_k$. As in LSQR and LSMR we begin with the QR factorization

$$P_k^T \begin{bmatrix} B_k & \beta_1 e_1 \end{bmatrix} = \begin{bmatrix} R_k & g_k \\ 0 & \psi'_{k+1} \end{bmatrix}, \quad R_k := \begin{bmatrix} \gamma_1 & \delta_2 & & \\ & \gamma_2 & \ddots & \\ & & \ddots & \delta_k \\ & & & \gamma_k \end{bmatrix}, \quad g_k = \begin{bmatrix} \psi_1 \\ \vdots \\ \psi_k \end{bmatrix}, \tag{12}$$

where $P_k^T = P_{k,k+1} \dots P_{2,3} P_{1,2}$ is a product of orthogonal reflections. The j-th reflection $P_{j,j+1}$ is designed to zero out the sub-diagonal element β_{j+1} in B_k. With $\bar{\gamma}_1 := \alpha_1$ it may be represented as

$$\begin{array}{c} \\ j \\ j+1 \end{array} \begin{array}{c} \begin{array}{cc} j & j+1 \end{array} \\ \begin{bmatrix} c'_j & s'_j \\ s'_j & -c'_j \end{bmatrix} \end{array} \begin{array}{c} \begin{array}{cc} j & j+1 \end{array} \\ \begin{bmatrix} \bar{\gamma}_j & \\ \beta_{j+1} & \alpha_{j+1} \end{bmatrix} \end{array} = \begin{array}{c} \begin{array}{cc} j & j+1 \end{array} \\ \begin{bmatrix} \gamma_j & \delta_{j+1} \\ & \bar{\gamma}_{j+1} \end{bmatrix} \end{array}, \tag{13}$$

where $\gamma_j = (\bar{\gamma}_j^2 + \beta_{j+1}^2)^{\frac{1}{2}}$, $c'_j = \bar{\gamma}_j / \gamma_j$, $s'_j = \beta_{j+1} / \gamma_j$, and

$$\begin{aligned} \delta_{j+1} &= \quad s'_j \alpha_{j+1}, \\ \bar{\gamma}_{j+1} &= -c'_j \alpha_{j+1}. \end{aligned} \tag{14}$$

The rotations apply to the right-hand side $\beta_1 e_1$ to produce g_k defined by the recurrence

$$\psi'_1 = \beta_1, \quad \psi_k = c'_k \psi'_k, \quad \psi'_{k+1} = s'_k \psi'_k, \quad k = 1, 2, \dots \tag{15}$$

It will be convenient to use the notation $g'_{k+1} = \begin{bmatrix} g_k^T & \psi'_{k+1} \end{bmatrix}^T$.

The QR factors of B_k give the Cholesky factorization $T_k = R_k^T R_k$. To form LQ factors of T_k we take the LQ factorization

$$R_k = \overline{M}_k Q_k, \qquad \overline{M}_k := \begin{bmatrix} \varepsilon_1 & & & \\ \eta_2 & \varepsilon_2 & & \\ & \ddots & \ddots & \\ & & \eta_k & \bar{\varepsilon}_k \end{bmatrix}. \tag{16}$$

Initially, $\bar{\varepsilon}_1 = \gamma_1$ so that $R_1 = \overline{M}_1$. We use the notation of Paige and Saunders (1975) to indicate that $\overline{M}_k$ differs from the leading k-by-k submatrix M_k of $\overline{M}_{k+1}$ in the (k,k)-th element only, which is updated to ε_k once $\delta_{k+1} = \alpha_{k+1}\beta_{k+1}/\gamma_k$ is computed. This results in the plane reflection $Q_{k,k+1}$ defined by

$$\begin{array}{c} \\ k \\ k+1 \end{array} \begin{array}{c} \begin{array}{cc} k & k+1 \end{array} \\ \begin{bmatrix} \bar{\varepsilon}_k & \delta_{k+1} \\ & \gamma_{k+1} \end{bmatrix} \end{array} \begin{array}{c} \begin{array}{cc} k & k+1 \end{array} \\ \begin{bmatrix} c_k & s_k \\ s_k & -c_k \end{bmatrix} \end{array} = \begin{array}{c} \begin{array}{cc} k & k+1 \end{array} \\ \begin{bmatrix} \varepsilon_k & \\ \eta_{k+1} & \bar{\varepsilon}_{k+1} \end{bmatrix} \end{array}, \tag{17}$$

where $\varepsilon_k = (\bar{\varepsilon}_k^2 + \delta_{k+1}^2)^{\frac{1}{2}}$, $c_k = \bar{\varepsilon}_k / \varepsilon_k$, $s_k = \delta_{k+1} / \varepsilon_k$, and

$$\begin{aligned} \eta_{k+1} &= \quad \gamma_{k+1} s_k, \\ \bar{\varepsilon}_{k+1} &= -\gamma_{k+1} c_k. \end{aligned} \tag{18}$$

Combining (12) and (16) gives

$$H_{k-1}^T = B_{k-1}^T L_k = \begin{bmatrix} B_{k-1}^T B_{k-1} & \alpha_k \beta_k e_{k-1} \end{bmatrix} = R_{k-1}^T \begin{bmatrix} R_{k-1} & \delta_k e_{k-1} \end{bmatrix}.$$

By construction,

$$R_k = \begin{bmatrix} R_{k-1} & \delta_k e_{k-1} \\ & \gamma_k \end{bmatrix} = \overline{M}_k Q_k = \begin{bmatrix} M_{k-1} & 0 \\ \eta_k e_{k-1}^T & \bar{\varepsilon}_k \end{bmatrix} Q_k$$

and we obtain the LQ factorization

$$H_{k-1}^T = R_{k-1}^T \begin{bmatrix} M_{k-1} & 0 \end{bmatrix} Q_k = \begin{bmatrix} R_{k-1}^T M_{k-1} & 0 \end{bmatrix} Q_k.$$

With the solution of $H_{k-1}^T y_k^L = \bar{\beta}_1 e_1$ in mind, we consider the system $R_k^T t_k = \alpha_1 \beta_1 e_1$ and obtain $t_k := \begin{bmatrix} \tau_1 & \dots & \tau_k \end{bmatrix}^T$ by the recursion

$$\begin{aligned} \tau_1 &:= \alpha_1 \beta_1 / \gamma_1, \\ \tau_j &:= -\tau_{j-1} \delta_j / \gamma_j, \quad j = 2, \dots, k. \end{aligned} \tag{19}$$

We also consider the systems $M_{k-1} z_{k-1} = t_{k-1}$ and $\overline{M}_k \bar{z}_k := t_k$ and obtain $z_{k-1} := \begin{bmatrix} \zeta_1 & \dots & \zeta_{k-1} \end{bmatrix}^T$ and $\bar{z}_k = \begin{bmatrix} z_{k-1}^T & \bar{\zeta}_k \end{bmatrix}^T$ by the recursion

$$\begin{aligned} \zeta_1 &= \tau_1 / \varepsilon_1, \\ \zeta_j &= (\tau_j - \zeta_{j-1} \eta_j) / \varepsilon_j, \quad j = 2, \dots, k-1, \\ \bar{\zeta}_k &= (\tau_j - \zeta_{k-1} \eta_k) / \bar{\varepsilon}_k = \zeta_k / c_k. \end{aligned} \tag{20}$$

Then $y_k^L = Q_k^T \begin{bmatrix} z_{k-1} \\ 0 \end{bmatrix}$ solves (10), while $y_k^C = Q_k^T \bar{z}_k$ solves (8).

Now let $\overline{W}_k := V_k Q_k^T = \begin{bmatrix} w_1 & \dots & w_{k-1} & \bar{w}_k \end{bmatrix} = \begin{bmatrix} W_{k-1} & \bar{w}_k \end{bmatrix}$. Starting with $x_1^L := 0$ and $x_1^C := 0$ we obtain

$$x_k^L = V_k y_k^L = V_k Q_k^T \begin{bmatrix} z_{k-1} \\ 0 \end{bmatrix} = \overline{W}_k \begin{bmatrix} z_{k-1} \\ 0 \end{bmatrix} = W_{k-1} z_{k-1} = x_{k-1}^L + \zeta_{k-1} w_{k-1}, \tag{21}$$

$$x_k^C = V_k Q_k^T \bar{z}_k = \overline{W}_k \bar{z}_k = W_{k-1} z_{k-1} + \bar{\zeta}_k \bar{w}_k = x_k^L + \bar{\zeta}_k \bar{w}_k. \tag{22}$$

Thus, as in SYMMLQ it is always possible to transfer to the CG point. In terms of error, Proposition 1 indicates that transferring is always desirable.

At the next iteration we have $\overline{W}_{k+1} = V_{k+1} Q_{k+1}^T$, where

$$\begin{bmatrix} \bar{w}_k & v_{k+1} \end{bmatrix} \begin{bmatrix} c_k & s_k \\ s_k & -c_k \end{bmatrix} = \begin{bmatrix} w_k & \bar{w}_{k+1} \end{bmatrix}.$$

With $\bar{w}_1 := v_1$ this gives

$$w_k = c_k \bar{w}_k + s_k v_{k+1}, \tag{23a}$$

$$\bar{w}_{k+1} = s_k \bar{w}_k - c_k v_{k+1}. \tag{23b}$$

Because the columns of W_{k-1} and $\overline{W}_k$ are orthonormal in exact arithmetic, we have

$$\|x_k^L\|^2 = \|W_{k-1} z_{k-1}\|^2 = \|z_{k-1}\|^2 = \sum_{j=1}^{k-1} \zeta_j^2 = \|x_{k-1}^L\|^2 + \zeta_{k-1}^2, \tag{24}$$

$$\|x_k^C\|^2 = \|x_k^L\|^2 + \bar{\zeta}_k^2. \tag{25}$$

2.4. Residual estimates. The k-th LSLQ residual is defined as $r_k^L := b - Ax_k^L$. We use the definition of $x_k^L = V_k y_k^L$, (3), (12) and (16) to express it as

$$\begin{aligned}
r_k^L = b - AV_k y_k^L &= U_{k+1}\left(\beta_1 e_1 - B_k y_k^L\right) \\
&= U_{k+1}P_k\left(\beta_1 P_k^T e_1 - \begin{bmatrix} R_k \\ 0 \end{bmatrix} y_k^L\right) \\
&= U_{k+1}P_k\left(g'_{k+1} - \begin{bmatrix} \overline{M}_k Q_k \\ 0 \end{bmatrix} y_k^L\right) \\
&= U_{k+1}P_k\left(g'_{k+1} - \begin{bmatrix} \overline{M}_k \\ 0 \end{bmatrix} \begin{bmatrix} z_{k-1} \\ 0 \end{bmatrix}\right) \\
&= U_{k+1}P_k\left(g'_{k+1} - \begin{bmatrix} M_{k-1} z_{k-1} \\ \eta_k \zeta_{k-1} \\ 0 \end{bmatrix}\right) \\
&= U_{k+1}P_k\left(\begin{bmatrix} g_{k-1} \\ \psi_k \\ \psi'_{k+1} \end{bmatrix} - \begin{bmatrix} t_{k-1} \\ \eta_k \zeta_{k-1} \\ 0 \end{bmatrix}\right),
\end{aligned}$$

where g'_{k+1} is defined in (12) and (15). It is not immediately obvious that $g_{k-1} = t_{k-1}$, but note that (12) yields $\begin{bmatrix} R_{k-1}^T & 0 \end{bmatrix} P_{k-1}^T = B_{k-1}^T$, so that

$$R_{k-1}^T g_{k-1} = \begin{bmatrix} R_{k-1}^T & 0 \end{bmatrix} \begin{bmatrix} g_{k-1} \\ \psi'_k \end{bmatrix} = B_{k-1}^T \beta_1 e_1 = \alpha_1 \beta_1 e_1 = R_{k-1}^T t_{k-1}$$

as long as $\gamma_{k-1} \neq 0$. Therefore, if the process does not terminate, we have $g_{k-1} = t_{k-1}$ as announced. By orthogonality of U_{k+1} and P_k we have

$$\|r_k^L\|^2 = \left\| \begin{bmatrix} 0 \\ \psi_k - \eta_k \zeta_{k-1} \\ \psi'_{k+1} \end{bmatrix} \right\|^2 = (\psi_k - \eta_k \zeta_{k-1})^2 + (\psi'_{k+1})^2. \tag{26}$$

The residual norm for the CG-point can also be computed as

$$r_k^C := b - Ax_k^C = U_{k+1}P_k\left(P_k^T \beta_1 e_1 - \begin{bmatrix} R_k \\ 0 \end{bmatrix} y_k^C\right) = U_{k+1}P_k\left(\begin{bmatrix} g_k \\ \psi'_{k+1} \end{bmatrix} - \begin{bmatrix} R_k \\ 0 \end{bmatrix} y_k^C\right).$$

The top k rows of the parenthesized expression vanish by definition of y_k^C, and there remains

$$\|r_k^C\| = (\beta_1 P_k^T e_1)_{k+1} = |\psi'_{k+1}|.$$

To derive recurrences for the residual norm for (NE), we can use the recurrences derived in Paige and Saunders (1975) for SYMMLQ and CG, which become

$$\begin{aligned}
\|A^T r_k^L\|^2 &= (\gamma_k \epsilon_k)^2 \zeta_k^2 + (\delta_k \eta_{k-1})^2 \zeta_{k-1}^2, \\
\|A^T r_k^C\| &= \alpha_1 \beta_1 s_1 \cdots s_{k-1} s_k / c_k.
\end{aligned}$$

2.5. Norm and condition number estimates. Assuming orthonormality of V_k, (4) yields $V_k^T A^T A V_k = B_k^T B_k$, so that the Poincaré separation theorem ensures $\sigma_{\min}(A) \leqslant \sigma_{\min}(B_k) \leqslant \sigma_{\max}(B_k) \leqslant \sigma_{\max}(A)$ for all k, where $\sigma_{\min}$ denotes the smallest nonzero singular value. Therefore we may use $\|B_k\|$ as an estimate of $\|A\|$ and $\operatorname{cond}(B_k)$

Algorithm 2 LSLQ

$\beta_1 u_1 = b,\ \ \alpha_1 v_1 = A^T u_1$ *// begin Golub-Kahan process*
$\delta_1 = -1,\ \ \psi_1 = \beta_1$ *// initialize variables*
$\tau_0 = \alpha_1\beta_1,\ \ \zeta_0 = 0$
$c_0 = 1,\ \ s_0 = 0$
$\|A^T r_0^C\| = \alpha_1\beta_1$
$\bar{w}_1 = v_1,\ \ x_1^L = 0$
for $k = 1, 2, \ldots$ **do**
  $\beta_{k+1}u_{k+1} = Av_k - \alpha_k u_k$ *// continue Golub-Kahan process*
  $\alpha_{k+1}v_{k+1} = A^T u_{k+1} - \beta_{k+1}v_k$
  $\gamma_k = (\bar{\gamma}_k^2 + \beta_{k+1}^2)^{\frac{1}{2}},\ \ c_k' = \bar{\gamma}_k/\gamma_k,\ \ s_k' = \bar{\beta}_{k+1}/\gamma_k$ *// continue QR factorization*
  $\delta_{k+1} = s_k'\alpha_{k+1}$
  $\bar{\gamma}_{k+1} = -c_k'\alpha_{k+1}$
  $\tau_k = -\tau_{k-1}\delta_k/\gamma_k$
  $\bar{\varepsilon}_k = -\gamma_k c_{k-1}$ *// continue LQ factorization*
  $\eta_k = \gamma_k s_{k-1}$
  $\varepsilon_k = (\bar{\varepsilon}_k^2 + \delta_{k+1}^2)^{\frac{1}{2}},\ \ c_k = \bar{\varepsilon}_k/\varepsilon_k,\ \ s_k = \delta_{k+1}/\varepsilon_k$
  $\|r_{k-1}^L\| = ((\psi_{k-1}c_k' - \zeta_{k-1}\eta_k)^2 + (\psi_{k-1}s_k')^2)^{\frac{1}{2}}$
  $\psi_k = \psi_{k-1}s_k'$
  $\|r_k^C\| = \psi_k$
  $\zeta_k = (\tau_k - \zeta_{k-1}\eta_k)/\varepsilon_k$ *// optional:* $\bar{\zeta}_k = \zeta_k/c_k$
  $\|A^T r_k^L\| = (\gamma_k^2\epsilon_k^2\zeta_k^2 + \delta_k^2\eta_k^2\zeta_{k-1}^2)^{\frac{1}{2}}$ *// optional:* $\|A^T r_k^C\| = \|A^T r_{k-1}^C\| s_k c_{k-1}/c_k$
  $w_k = c_k\bar{w}_k + s_k v_{k+1}$
  $\bar{w}_{k+1} = s_k\bar{w}_k - c_k v_{k+1}$
  $x_{k+1}^L = x_k^L + \zeta_k w_k$ *// optional:* $x_k^C = x_k^L + \bar{\zeta}_k\bar{w}_k$
  $\|x_{k+1}^L\|^2 = \|x_k^L\|^2 + \zeta_k^2$ *// optional:* $\|x_{k+1}^C\|^2 = \|x_k^C\|^2 + \bar{\zeta}_k^2$
end for

as an estimate of $\mathrm{cond}(A)$ in both the Euclidean and Frobenius norms. In particular, $\|B_{k+1}\|_F^2 = \|B_k\|_F^2 + \alpha_k^2 + \beta_{k+1}^2$.

As in (Fong and Saunders, 2011, Section 3.4), our approximation of $\mathrm{cond}(A)$ rests on the QLP factorization

$$P_k^T B_k Q_k^T = \begin{bmatrix} M_{k-1} & 0 \\ \eta_k e_{k-1}^T & \bar{\epsilon}_k \end{bmatrix}.$$

According to Stewart (1999), the absolute values of the diagonals of the bidiagonal matrix above are tight approximations to the singular values of B_k. Thus we estimate

$$\sigma_{\min}(B_k) \approx \min(\epsilon_1, \ldots, \epsilon_{k-1}, |\bar{\epsilon}_k|), \quad \sigma_{\max}(B_k) \approx \max(\epsilon_1, \ldots, \epsilon_{k-1}, |\bar{\epsilon}_k|),$$

and $\mathrm{cond}(A) \approx \sigma_{\max}(B_k)/\sigma_{\min}(B_k)$, which turns out to be reasonably accurate in practice. If b lies in a subspace spanned by only a few singular vectors of A, iterations will terminate early and $\mathrm{cond}(B_k)$ will be an improving estimate of $\mathrm{cond}(AV_\ell)$.

3. Complete algorithm. The complete procedure is summarized as Algorithm 2. As in (Fong and Saunders, 2011, Theorem 4.2), we can prove the following.

THEOREM 2. *LSLQ returns the MLS solution, i.e., it solves*

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ \|x\| \quad \text{subject to}\ \ x \in \arg\min_y \|Ay - b\|.$$

4. Error estimates. In exact arithmetic, a least-squares solution $x_\star$ is identified after at most $\ell \leqslant \min(m,n)$ iterations, so that $x_\star = x_{\ell+1}^L = \sum_{j=1}^{\ell} \zeta_j w_j$. Because $x_k^L = \sum_{j=1}^{k-1} \zeta_j w_j$, the error may be written as $e_k^L = x_{\ell+1}^L - x_k^L = \sum_{j=k}^{\ell} \zeta_j w_j$. By orthogonality, $\|e_k^L\|^2 = \sum_{j=k}^{\ell} \zeta_j^2$. A possible stopping condition is

$$\|x_{k+1}^L - x_{k-d}^L\|^2 = \left(\sum_{j=k-d}^{k} \zeta_j^2 \right)^{\frac{1}{2}} \leqslant \varepsilon \|x_{k+1}^L\| \quad (k > d), \tag{27}$$

where $d \in \mathbb{N}$ is a delay and $0 < \varepsilon < 1$ is a tolerance. The left-hand side of (27) is a lower bound on the error $\|e_{k-d}^L\|$.

As we illustrate in section 6, (27) is not a robust stopping criterion because the lower bound may sometimes underestimate the actual error by several orders of magnitude. In the following sections, we develop a more robust estimate defined by an upper bound.

4.1. Upper bound on the LSLQ error. Estrin et al. (2016) develop an upper bound on the Euclidean error along SYMMLQ iterations for a symmetric positive semidefinite system. The bound leads to an upper bound on the error along CG iterations. We now translate those estimates to the present scenario and obtain upper bounds on the error along LSLQ and LSQR iterations for (LS) or (38). We begin with an upper bound on the LSLQ error. By orthogonality, $\|x_\star - x_k^L\|^2 = \|x_\star\|^2 - \|x_k^L\|^2$, and because $\|x_k^L\|^2$ can be computed, an upper bound on the error will follow from an upper bound on $\|x_\star\|^2$. Assume temporarily that $m \geqslant n$ and that A has full column rank, so that $A^T A$ is nonsingular. We may express

$$\|x_\star\|^2 = b^T A (A^T A)^{-2} A^T b = b^T A f(A^T A) A^T b,$$

where $f(\xi) := \xi^{-2}$ is defined for all $\xi \in (0, \sigma_1^2]$, and where we define $f(A^T A) := P f(\Sigma^T \Sigma) P^T$ with $A = Q \Sigma P^T$ the SVD of A. In other words, if p_i is the i-th column of P and σ_i is the i-th largest singular value of A,

$$f(A^T A) = \sum_{i=1}^{n} f(\sigma_i^2) p_i p_i^T.$$

We have from line 2 of Algorithm 1 and (6) that $A^T b = \bar{\beta}_1 v_1$ and therefore

$$\|x_\star\|^2 = \bar{\beta}_1^2 \sum_{i=1}^{n} f(\sigma_i^2) \mu_i^2, \qquad \mu_i := p_i^T v_1, \ i = 1, \ldots, n.$$

When A is rank-deficient, $A^T A$ is positive semidefinite and singular, but (NE) remains consistent. In addition, the MLS solution of (LS) lies in Range(A^T). Let r be the smallest integer in $\{1, \ldots, n\}$ such that $\sigma_{r+1} = \cdots = \sigma_n = 0$ and $\sigma_r > 0$. Then $\operatorname{rank}(A) = r = \dim \operatorname{Range}(A^T)$ and the smallest nonzero eigenvalue of $A^T A$ is σ_r^2. By the Rayleigh-Ritz theorem,

$$\sigma_r^2 = \min \left\{ \|Av\|^2 \mid v \in \operatorname{Range}(A^T), \ \|v\| = 1 \right\}.$$

Note that each $v_i \in \operatorname{Range}(A^T)$ and that (4) implies $T_k = V_k^T A^T A V_k$ in exact arithmetic. Hence, for all $u \in \mathbb{R}^k$ with $\|u\| = 1$, we have $\|V_k u\| = 1$ and $u^T T_k u = \|A V_k u\|^2 \geqslant \sigma_r^2 > 0$, and each T_k is uniformly positive definite, despite the fact that $A^T A$ is singular.

Thus, in the rank-deficient case, $A^TA = \sum_{i=1}^{r} \sigma_i^2 p_i p_i^T$. The only difference with the full-rank case is that the sum occurs over all nonzero singular values of A. Therefore, we need only redefine

$$f(\xi) := \begin{cases} \xi^{-2} & \text{if } x > 0 \\ 0 & \text{if } x = 0. \end{cases} \tag{28}$$

Because each x_k^L and each $x_k^C \in \text{Range}(A^T)$, the LSLQ and LSQR iterations occur in $\text{Range}(A^T)$ exactly as if they were applied to the r-by-r positive-definite system

$$P_r^T A^T A P_r \bar{x} = P_r^T A^T b,$$

where $P_r = \begin{bmatrix} p_1 & \dots & p_r \end{bmatrix}$ and $x_\star = P_r \bar{x}$. A consequence of the above discussion is that

$$\|x_\star\|^2 = \bar{\beta}_1^2 \sum_{i=1}^{r} f(\sigma_i^2)\mu_i^2, \qquad \mu_i := p_i^T v_1, \ i = 1, \dots, n.$$

Golub and Meurant (1997) explain that the main insight is to view the previous sum as the Riemann-Stieltjes integral

$$\sum_{i=1}^{r} f(\sigma_i^2)\mu_i^2 = \int_{\sigma_r}^{\sigma_1} f(\sigma^2)\, \mathrm{d}\mu(\sigma), \tag{29}$$

where the piecewise constant Stieltjes measure μ is defined as

$$\mu(\sigma) := \begin{cases} 0 & \text{if } \sigma < \sigma_r \\ \sum_{j=i}^{r} \mu_j^2 & \text{if } \sigma_i \leqslant \sigma < \sigma_{i+1} \\ \sum_{j=1}^{r} \mu_j^2 & \text{if } \sigma \geqslant \sigma_1. \end{cases}$$

Approximations to the integral via Gauss-related quadrature rules yield corresponding approximations to $\|x_\star\|^2$.

Our main result leading to an upper bound estimate follows from a Gauss-Radau approximation of (29) with a fixed quadrature node in $(0, \sigma_r^2)$. We begin with a paraphrase of (Estrin et al., 2016, Theorem 2).

PROPOSITION 3. *Suppose $f : \mathbb{R} \to \mathbb{R}$ is such that $f^{(2j+1)}(\xi) < 0$ for all $\xi \in (\sigma_r^2, \sigma_1^2)$ and all $j \geqslant 0$. Fix $\sigma_{est} \in (-\sigma_r, \sigma_r)$, $\sigma_{est} \neq 0$. Let T_k be the tridiagonal generated after k steps of Algorithm 1 and $\varpi_k \in \mathbb{C}$ be chosen so that the smallest eigenvalue of*

$$\widetilde{T}_k := \begin{bmatrix} T_{k-1} & \bar{\beta}_k e_{k-1} \\ \bar{\beta}_k e_{k-1}^T & \alpha_k^2 + \varpi_k^2 \end{bmatrix}$$

is precisely σ_{est}^2. Then,

$$b^T A f(A^T A) A^T b \leqslant \bar{\beta}_1^2 e_1^T f(\widetilde{T}_k) e_1.$$

In particular, Proposition 3 applied to f defined in (28) provides an upper bound on $\|x_\star\|^2$.

Note that the Poincaré separation theorem ensures that the smallest eigenvalue of each T_{k-1} is at least σ_r^2 and that the Cauchy interlace theorem guarantees that the smallest eigenvalue of $\widetilde{T}_k$ is smaller than or equal to that of T_{k-1}. Thus it is possible to choose ϖ_k satisfying the requirements of Proposition 3.

We now comment on the surprising fact that $\varpi_k \in \mathbb{C}$ in Proposition 3. To avoid forming T_k and $\widetilde{T}_k$ explicitly, we would prefer to pick a nonzero $\sigma_{\text{est}} \in (0, \sigma_r)$ and seek ϖ_k such that σ_{est} is the smallest singular value of

$$\widetilde{B}_k = \begin{bmatrix} L_k \\ \varpi_k e_k^T \end{bmatrix}. \tag{30}$$

The fact that $\varpi_k \in \mathbb{C}$ is a departure from the computations of Estrin et al. (2016), who establish that the last diagonal of $\widetilde{T}_k$ is real: $\alpha_k^2 + \varpi_k^2 \in \mathbb{R}$. In order for ϖ_k^2 to be real, ϖ_k must be either real or purely imaginary. In a numerical implementation of (30), although it is possible to avoid computations in complex arithmetic, we do observe corrections ϖ_k such that the last diagonal is strictly less than α_k^2, i.e., such that ϖ_k is purely imaginary.

An alternative strategy that avoids complex numbers altogether is to pick a nonzero $\sigma_{\text{est}} \in (0, \sigma_r)$ and seek ω_k such that σ_{est} is the smallest singular value of

$$\widetilde{R}_k = \begin{bmatrix} R_{k-1} & \delta_k e_{k-1} \\ & \omega_k \end{bmatrix}. \tag{31}$$

Note that $\widetilde{R}_k$ differs from R_k, the R factor in the QR factors of B_k, in the (k, k)-th entry only. In addition, if $\widetilde{R}_k$ is the Cholesky factor of $\widetilde{T}_k$, its diagonals are guaranteed to be real and positive and the smallest eigenvalue of $\widetilde{T}_k$ will be σ_{est}^2.

As earlier, the Poincaré separation theorem guarantees that the singular values of each R_{k-1}, which are the same as those of B_{k-1}, lie between σ_r and σ_1, and the Cauchy interlace theorem for singular values guarantees that it is indeed possible to choose ω_k so that the smallest singular value (31) is σ_{est}. We may now restate Proposition 3 with the above in mind.

THEOREM 4. *Suppose $f : \mathbb{R} \to \mathbb{R}$ is such that $f^{(2j+1)}(\xi) < 0$ for all $\xi \in (\sigma_r^2, \sigma_1^2)$ and all $j \geqslant 0$. Fix $\sigma_{est} \in (0, \sigma_r)$. Let B_k be the bidiagonal generated after k steps of Algorithm 1 and $\omega_k > 0$ be chosen so that the smallest singular value of (31) is precisely σ_{est}. Then,*

$$b^T A f(A^T A) A^T b \leqslant \bar{\beta}_1^2 e_1^T f(\widetilde{R}_k^T \widetilde{R}_k) e_1.$$

In order to determine ω_k, we follow Golub and Kahan (1965) and embed $\widetilde{R}_k$ into a larger symmetric matrix to change the singular value problem into an eigenvalue problem. Indeed,

$$\begin{bmatrix} 0 & \widetilde{R}_k \\ \widetilde{R}_k^T & 0 \end{bmatrix} \tag{32}$$

has eigenvalues $\pm\sigma_i(\widetilde{R}_k)$. Define

$$Y_{2k-2} := \begin{bmatrix} 0 & \gamma_1 & & & & & \\ \gamma_1 & 0 & \delta_2 & & & & \\ & \delta_2 & 0 & \gamma_2 & & & \\ & & \gamma_2 & 0 & \delta_3 & & \\ & & & \delta_3 & 0 & \ddots & \\ & & & & \ddots & \ddots & \gamma_{k-1} \\ & & & & & \gamma_{k-1} & 0 \end{bmatrix}, \quad \widetilde{Y}_{2k} := \begin{bmatrix} Y_{2k-2} & \delta_k e_{2k-2} & \\ \delta_k e_{2k-2}^T & 0 & \omega_k \\ & \omega_k & 0 \end{bmatrix}.$$

Note that $\widetilde{Y}_{2k}$ is a symmetric permutation of (32) and therefore shares the same eigenvalues. If σ_{est} is an eigenvalue of $\widetilde{Y}_{2k}$ and $h^{(2k)} = \begin{bmatrix}\theta_1 & \ldots & \theta_{2k}\end{bmatrix}^T$ is a corresponding eigenvector, then $(\widetilde{Y}_{2k} - \sigma_{\text{est}} I)h^{(2k)} = 0$; that is,

$$\begin{bmatrix} Y_{2k-2} - \sigma_{\text{est}} I & \delta_k e_{2k-2} & \\ \delta_k e_{2k-2}^T & -\sigma_{\text{est}} & \omega_k \\ & \omega_k & -\sigma_{\text{est}} \end{bmatrix} \begin{bmatrix} h_{2k-2}^{(2k)} \\ \theta_{2k-1} \\ \theta_{2k} \end{bmatrix} = 0.$$

Necessarily, $\theta_{2k-1} \neq 0$ because otherwise $h^{(2k)} = 0$ entirely. Thus we may fix $\theta_{2k-1} = 1$. The first block equation reads $(Y_{2k-2} - \sigma_{\text{est}} I)h_{2k-2}^{(2k)} = -\delta_k e_{2k-2}$. Let θ_{2k-2} be the last entry of $h_{2k-2}^{(2k)}$, which can be computed by updating the QR factors of Y_{2k-2} as in (Estrin et al., 2016).

In order to compute ω_k, note that the last two equations,

$$\begin{bmatrix} \delta_k & -\sigma_{\text{est}} & \omega_k \\ & \omega_k & -\sigma_{\text{est}} \end{bmatrix} \begin{bmatrix} \theta_{2k-2} \\ 1 \\ \theta_{2k} \end{bmatrix} = 0,$$

imply that $\omega_k = \sqrt{\sigma_{\text{est}}^2 - \sigma_{\text{est}} \delta_k \theta_{2k-2}}$.

With ω_k computed, we have $\widetilde{R}_k^T \widetilde{R}_k = \widetilde{T}_k$. We are now interested in efficiently computing the upper bound

$$\|x_\star\|^2 \leqslant \bar{\beta}_1^2 e_1^T f(\widetilde{R}_k^T \widetilde{R}_k) e_1 = \bar{\beta}_1^2 e_1^T (\widetilde{R}_k^T \widetilde{R}_k)^{-2} e_1. \tag{33}$$

The LQ factorization $\widetilde{R}_k = \widetilde{M}_k \widetilde{Q}_k$ provides the LQ factorization $\widetilde{T}_k = \widetilde{R}_k^T \widetilde{M}_k \widetilde{Q}_k$, which in turn yields

$$\|x_\star\|^2 \leqslant \left\| \bar{\beta}_1 \widetilde{M}_k^{-1} \widetilde{R}_k^{-T} e_1 \right\|^2 = \|\widetilde{M}_k^{-1} \tilde{t}_k\|^2 = \|\tilde{z}_k\|^2,$$

where we define $\tilde{t}_k$ and $\tilde{z}_k$ from $\widetilde{R}_k^T \tilde{t}_k = \bar{\beta}_1 e_1$ and $\widetilde{M}_k \tilde{z}_k = \tilde{t}_k$ as in (Estrin et al., 2016).

We determine the LQ factorization $\widetilde{R}_k = \widetilde{M}_k \widetilde{Q}_k$ from

$$\widetilde{R}_k = \begin{bmatrix} R_{k-1} & \delta_k e_{k-1} \\ & \omega_k \end{bmatrix} = \begin{bmatrix} M_{k-1} & \\ \widetilde{\eta}_k e_{k-1}^T & \widetilde{\varepsilon}_k \end{bmatrix} \begin{bmatrix} Q_{k-1} & \\ & 1 \end{bmatrix}.$$

Thus $\widetilde{Q}_k = Q_k$ and $\widetilde{M}_k$ differs from M_k in the $(k, k-1)$-th and (k,k)-th entries only, which become

$$\widetilde{\eta}_k = \omega_k s_{k-1}, \qquad \widetilde{\varepsilon}_k = -\omega_k c_{k-1}.$$

Recalling the definition of t_k in (19) and z_{k-1} in (20) we observe that

$$\tilde{t}_k = \begin{bmatrix} t_{k-1} \\ \tilde{\tau}_k \end{bmatrix} \quad \text{and} \quad \tilde{z}_k = \begin{bmatrix} z_{k-1} \\ \widetilde{\zeta}_k \end{bmatrix}, \tag{34}$$

where

$$\widetilde{\tau}_k = -\tau_{k-1} \delta_k / \omega_k = \tau_k \gamma_k / \omega_k \quad \text{and} \quad \widetilde{\zeta}_k = (\widetilde{\tau}_k - \widetilde{\eta}_k \zeta_{k-1}) / \widetilde{\varepsilon}_k. \tag{35}$$

From (24) and orthogonality of W_k we now have

$$\|x_\star - x_k^L\|^2 = \|x_\star\|^2 - \|x_k^L\|^2 \leqslant \|z_{k-1}\|^2 + \widetilde{\zeta}_k^2 - \|z_{k-1}\|^2 = \widetilde{\zeta}_k^2. \tag{36}$$

4.2. Upper bound on the LSQR error. Obtaining an upper bound on the LSQR error is of interest for two reasons. First, LSLQ may transfer to the LSQR point at any iteration using a simple vector operation—see (22). Second, LSQR always produces a smaller error, as formalized by Proposition 1.

Based on Proposition 1, we wish to use the upper bound (36) and the transition (22) to the LSQR point to terminate LSLQ early and obtain an iterate with an error below a prescribed level. Evidently the same upper bound (36) could be used, but Estrin et al. (2016) provide the improved bound

$$\|x_\star - x_k^C\|^2 \leqslant \widetilde{\zeta}_k^2 - \bar{\zeta}_k^2, \tag{37}$$

where $\bar{\zeta}_k$ is defined in (20) and $\widetilde{\zeta}_k$ is in (35).

5. Regularization. LSLQ may be adapted to solve the regularized least-squares problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \tfrac{1}{2} \left\| \begin{bmatrix} A \\ \lambda I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|^2, \tag{38}$$

where $\lambda \geqslant 0$ is a given regularization parameter. The optimality conditions (NE) become

$$(A^T A + \lambda^2 I)x = A^T b. \tag{39}$$

If we run Algorithm 1 on A only, we will produce the factorization

$$\begin{bmatrix} A \\ \lambda I \end{bmatrix} V_k = \begin{bmatrix} U_{k+1} & \\ & V_k \end{bmatrix} \begin{bmatrix} B_k \\ \lambda I \end{bmatrix}, \tag{40}$$

which we can compare to the factorization achieved when running Algorithm 1 on the entire regularized system,

$$\begin{bmatrix} A \\ \lambda I \end{bmatrix} V_k = \hat{U}_{k+1} \hat{B}_k = \hat{U}_{k+1} \begin{bmatrix} \hat{\alpha}_1 & & \\ \hat{\beta}_2 & \ddots & \\ & \ddots & \hat{\alpha}_k \\ & & \hat{\beta}_{k+1} \end{bmatrix}. \tag{41}$$

Note that V_k will remain unchanged, as can be seen from the equivalence between the Golub-Kahan process and the Lanczos process on the normal equations (Saunders, 1995). Given $\hat{B}_k$, we could run the non-regularized LSLQ algorithm (using $\hat{\alpha}$ and $\hat{\beta}$ instead of α and β) to obtain all of the desired iterates and estimates. The idea is therefore to compute B_k via Golub-Kahan on (A, b), cheaply compute each $\hat{\alpha}_k$ and $\hat{\beta}_k$ and use them in place of α_k and β_k in the rest of the algorithm. For $k = 3$, the

factorization proceeds according to

$$
\begin{bmatrix} \alpha_1 & & \\ \beta_2 & \alpha_2 & \\ & \beta_3 & \alpha_3 \\ & & \beta_4 \\ \lambda & & \\ & \lambda & \\ & & \lambda \end{bmatrix} \rightarrow \begin{bmatrix} \alpha_1 & & \\ \hat{\beta}_2 & \hat{\alpha}_2 & \\ & \beta_3 & \alpha_3 \\ & & \beta_4 \\ & \hat{\lambda}_2 & \\ & \lambda & \\ & & \lambda \end{bmatrix} \rightarrow \begin{bmatrix} \alpha_1 & & \\ \hat{\beta}_2 & \hat{\alpha}_2 & \\ & \beta_3 & \alpha_3 \\ & & \beta_4 \\ & & \\ & \lambda_2 & \\ & & \lambda \end{bmatrix}
$$

$$
\rightarrow \begin{bmatrix} \alpha_1 & & \\ \hat{\beta}_2 & \hat{\alpha}_2 & \\ & \hat{\beta}_3 & \hat{\alpha}_3 \\ & & \beta_4 \\ & & \\ & & \hat{\lambda}_3 \\ & & \lambda \end{bmatrix} \rightarrow \begin{bmatrix} \alpha_1 & & \\ \hat{\beta}_2 & \hat{\alpha}_2 & \\ & \hat{\beta}_3 & \hat{\alpha}_3 \\ & & \beta_4 \\ & & \\ & & \\ & & \lambda_3 \end{bmatrix} \rightarrow \begin{bmatrix} \alpha_1 & & \\ \hat{\beta}_2 & \hat{\alpha}_2 & \\ & \hat{\beta}_3 & \hat{\alpha}_3 \\ & & \hat{\beta}_4 \\ & & \\ & & \\ & & \end{bmatrix} \tag{42}
$$

We use β_{k+1} to zero out λ_k, which transforms α_{k+1} into $\hat{\alpha}_{k+1}$ and introduces a nonzero $\hat{\lambda}_{k+1}$ above λ in the next column. We then use a second reflection to zero out $\hat{\lambda}_{k+1}$ using λ, which produces λ_{k+1}. With $\lambda_1 = \lambda$, the recurrences for $k \geqslant 2$ are

$$
\begin{aligned}
\hat{\beta}_{k+1} &= (\beta_{k+1}^2 + \lambda_k^2)^{\frac{1}{2}}, \\
c_k^L &= \beta_{k+1} / \hat{\beta}_{k+1}, \\
s_k^L &= \lambda_k / \hat{\beta}_{k+1}, \\
\hat{\alpha}_{k+1} &= c_k^L \alpha_{k+1}, \\
\hat{\lambda}_{k+1} &= s_k^L \alpha_{k+1}, \\
\lambda_{k+1} &= (\lambda^2 + \hat{\lambda}_{k+1}^2)^{\frac{1}{2}}.
\end{aligned} \tag{43}
$$

With $\lambda > 0$, the operator of (38) has full column rank, i.e., $r = n$, and satisfies $\sigma_n \geqslant \lambda$. Theorem 4 then states that we should select $\sigma_{\text{est}} \in (0, \lambda)$.

6. Numerical experiments. In the experiments reported here, the exact solution of (LS) was computed as the MLS solution using a complete orthogonal decomposition of A via the `Factorize` package (Davis, 2013). The horizontal axis in plots represents iterations, each involving a product with A and a product with A^T. LSLQ is implemented in the Julia language (julialang.org) and is available as part of the `Krylov.jl` suite of iterative methods (Orban, 2017). Subsection 6.1 and subsection 6.2 document our results on problems from the animal breeding test set and on the seismic inversion problem described in section 1, respectively. Although all test problems are over-determined, the solvers apply to systems of any shape. We have observed qualitatively similar results for square and underdetermined systems.

6.1. Problems from the animal breeding test set. In this section, we use test problems from the animal breeding collection of Hegland (1990, 1993). These over-determined problems have rank-deficiency 1, come in two flavors and sizes, and have accompanying right-hand sides. In the first flavor, a single parameter is fitted per animal, while in the second flavor, two parameters are fitted per animal and A has twice as many rows and columns. The nonzero columns of A are scaled to have unit Euclidean norm.

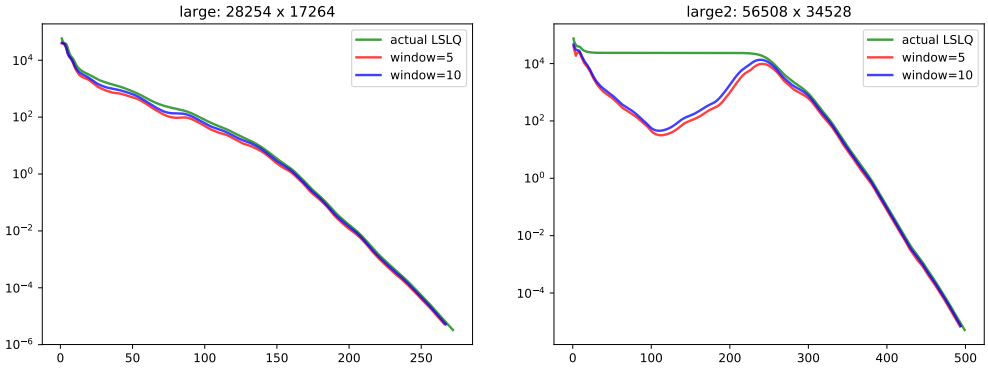

FIG. 2. *Error along the LSLQ iterations on problems* large *and* large2 *from the animal breeding set. The red and blue curves show the lower bounds with* $d = 5$ *and* $d = 10$.

We found that generating the problems from the original archive requires a small amount of corrections to the programs and several compilation steps. Because we feel that the problems from this set are generally useful as least-squares test problems, we have created an archive containing the problems as well as the MLS solutions corresponding to the scaled problems in Rutherford-Boeing format (Duff, Grimes, and Lewis, 1997). Our repository can be accessed at github.com/optimizers/animal (Orban, 2016).

We begin with an illustration of the non-robust lower bound (27) based on a delay d. Figure 2 plots the actual LSLQ error along with the lower bound with delay (window size) $d = 5$ and 10 iterations for problems large and large2 (larger versions of the problems used in Figure 1). The behavior seen is typical. As in the left-hand plot, the lower bound tends to follow the exact error curve tightly when the latter is strictly decreasing. But as the right-hand plot shows, it tends to underestimate the actual error by several orders of magnitude when the latter plateaus, and requires a fair number of iterations to recover, rendering the stopping test unreliable by itself. In both plots, the stopping test used is (27) with $\varepsilon = 10^{-10}$. The curves for $d = 5$ and 10 are almost the same.

Figure 3 illustrates the behavior of our upper bound (36) on problems large and large2 with regularization: a typical scenario for rank-deficient problems whose smallest nonzero singular value is unknown. For a given value $\lambda \neq 0$, the smallest singular value of the regularized A is $\sigma_n = |\lambda|$. Estrin et al. (2016) show numerically that the upper bound is tighter when $|\sigma_{\text{est}}|$ is closer to $|\sigma_n|$, but they do not consider the effect of regularization. To simplify the discussion, we consider only positive values of λ. For each value of $\lambda > 0$, we set $\sigma_{\text{est}} := (1 - 10^{-10})\,\lambda$ and measure the error with respect to the solution of the regularized problem.

We observe from Figure 3 that increasing λ (and hence σ_{est}) substantially improves the quality of the upper bound. The reason may be that $\widetilde{T}_k$ is moved further away from singularity. In the case of large2 with $\lambda = 10^{-2}$, the upper bound is exceptionally tight after about 100 iterations. As λ decreases, the upper bound deteriorates, although it remains a potentially useful bound as long as $\lambda \neq 0$.

In Figure 4, we compute the bound (37) on the error along the LSQR iterates or, equivalently, along the LSQR points obtained by transitioning from a corresponding LSLQ point. As with LSLQ, the quality of the LSQR upper bound deteriorates when A, or its regularization, approaches rank-deficiency. The LSQR bound appears

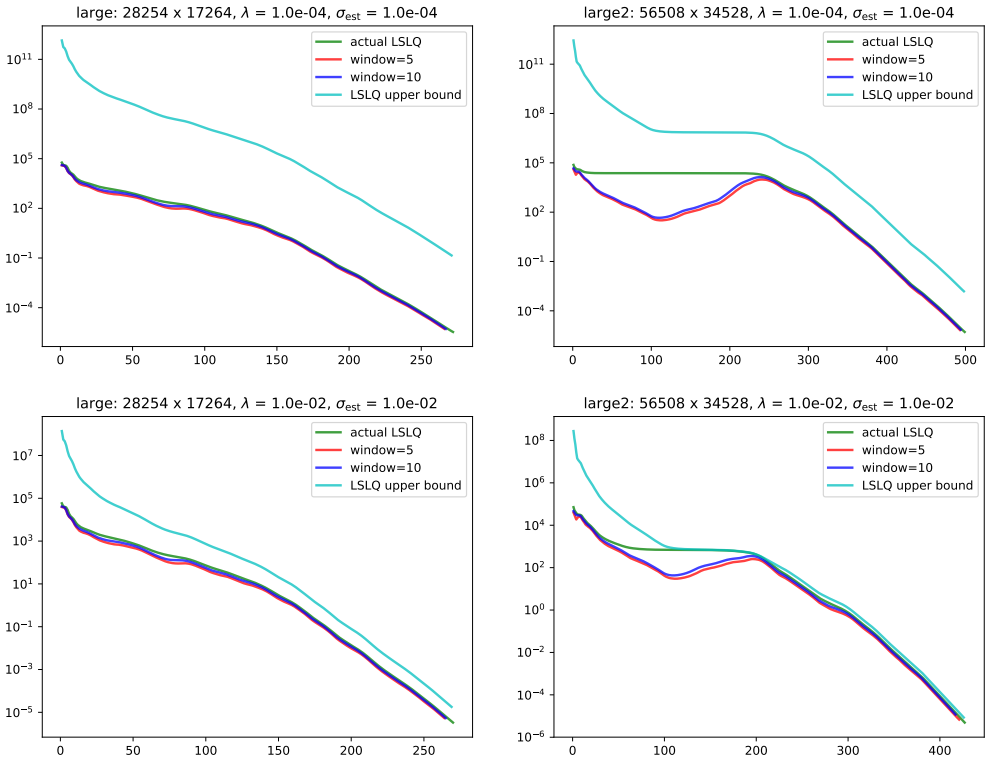

FIG. 3. *Error along the LSLQ iterations on problems* large *and* large2 *with regularization. The red and blue curves show the lower bounds with* $d = 5$ *and* $d = 10$*. The cyan curve shows the upper bounds for* $\lambda = 10^{-4}$ *(top) and* $\lambda = 10^{-2}$ *(bottom).*

somewhat looser than the LSLQ bound, although Estrin et al. (2016) note that it could be tightened by incorporating an additional term along a moving window to the right-hand side of (37).

The next experiment illustrates the upper bounds for rank-deficient problems when we have knowledge of σ_r. A sparse SVD reveals that the smallest nonzero singular value after scaling is approximately $\sigma_r = \sigma_{n-1} \approx 0.0498733$ for problem small and $\sigma_r = \sigma_{n-1} \approx 0.00499044$ for small2. In each case, we set $\sigma_{\text{est}} = (1 - 10^{-10})\,\sigma_{n-1}$. In practice, one may need to underestimate further in order to account for inaccurate σ_r.

As the error bounds in Figure 5 are quite tight, it seems important to supply an estimate of σ_r in rank-deficient problems if such knowledge is available. In Figure 5, LSLQ stops as soon as the upper bound on the LSQR error falls below $10^{-10}\|x_k^C\|$.

6.2. The seismic inverse problem. The least-squares problem arising from the PDE-constrained optimization problem described in section 1 has the form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; \tfrac{1}{2} \left\| \begin{bmatrix} \rho A \\ P \end{bmatrix} x - \begin{bmatrix} \rho q \\ d \end{bmatrix} \right\|^2, \tag{44}$$

where $\rho = 0.1$ is fixed, A is a square 5-point stencil discretization of a Helmholtz operator, P is a sampling operator (some rows of the identity), and q and d are fixed vectors. We experimented with a case in which $n = 83,600$ and P has 248 rows. The columns of the operator were not scaled as in the previous section, as that reduced

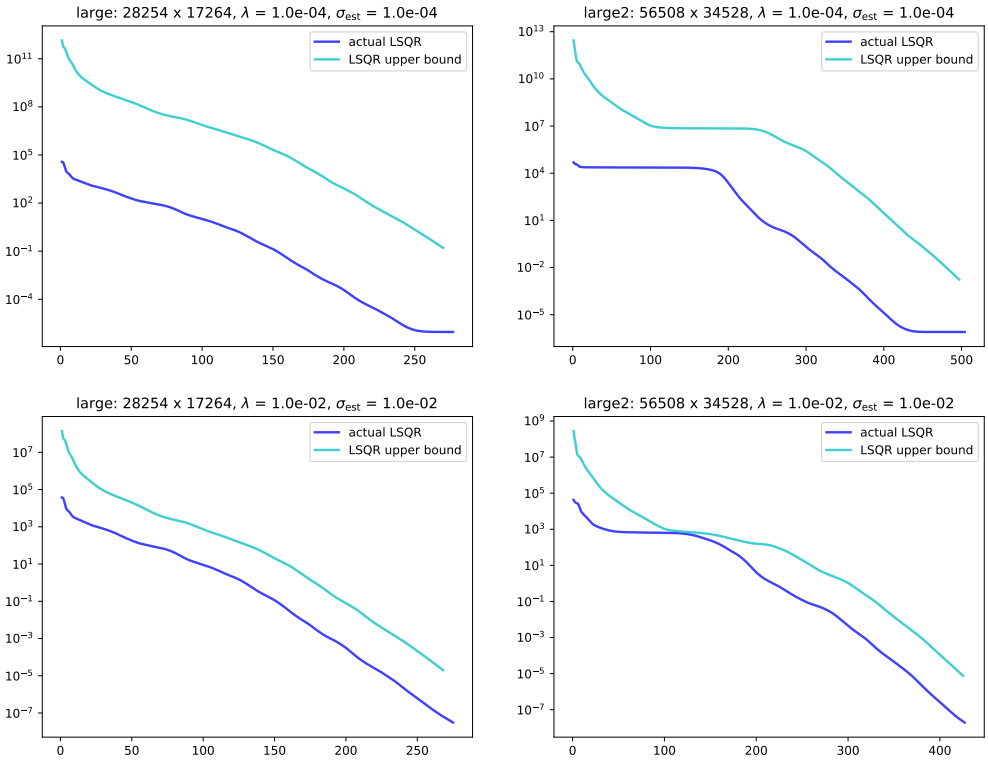

FIG. 4. *Error along the LSQR iterations on problems* large *and* large2 *with regularization. The cyan curve shows the upper bounds for* $\sigma_{est} = 10^{-4}$ *(top) and* $\sigma_{est} = 10^{-2}$ *(bottom).*

the performance of LSLQ. A complete orthogonal decomposition, used to compute the exact solution, reveals that the operator of (44) has full rank but its smallest nonzero singular value is $O(10^{-6})$. A partial sparse SVD suggests that there are several small singular values. To obtain upper error bounds, it was necessary to set $\sigma_{\text{est}} = 10^{-7}$ to avoid domain errors in computing the square root in the expression for ω_k preceding (33). The left plots of Figure 6 illustrate the upper and lower bounds on the error and the large number of iterations needed to decrease the error by a factor of 10^{10}. The bounds on the LSLQ and LSQR errors nonetheless track the exact errors quite accurately, with the upper bound on the LSQR error overestimating by one or two orders of magnitude. Though the factor 10^{10} is far too demanding in practice, it illustrates that many iterations are likely when there are many tiny singular values. The situation is similar when the problem is regularized and the error is measured with respect to the exact solution of the original, unregularized, problem. The right plots of Figure 6 show the bounds in the presence of modest regularization λ when the error is computed with respect to the exact solution of the regularized problem. Dramatically fewer iterations are needed to achieve a corresponding decrease in the error. Note the remarkable tightness of the LSLQ and LSQR bounds, with the LSQR upper bound consistently overestimating by about one order of magnitude. The improved performance on the regularized problem suggests that a regularized optimization approach, such as that of Arreckx and Orban (2016), could be appropriate.

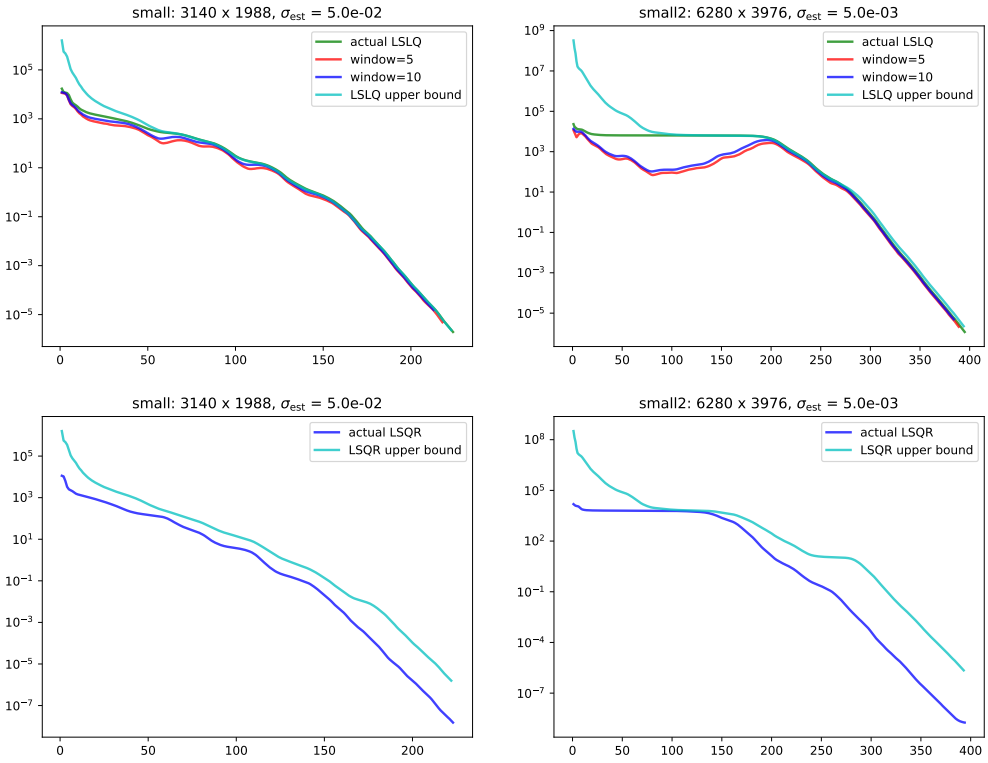

FIG. 5. *Error along the LSLQ and LSQR iterations on problems* **small** *and* **small2** *without regularization. Both problems have rank-deficiency* 1.

7. Discussion. LSLQ is an iterative method for the least-squares and least-norm problems (LS) and (LN) with the attractive property that it ensures monotonic reduction in the Euclidean error $\|x - x_k\|_2$. In deriving it we have completed the triad of solvers LSQR, LSMR, LSLQ for problem (LS) based on the Golub and Kahan (1965) process. They are mathematically equivalent to the symmetric solvers CG, MINRES, SYMMLQ on (NE) but are numerically more reliable when A is ill-conditioned.

Although the Euclidean error for LSQR is provably better at each iterate, it is possible to develop cheaply computable lower and upper bounds on the error for LSLQ. The intimate relationship between the methods, analogous to that between CG and SYMMLQ (Estrin et al., 2016), provides a corresponding upper bound on the LSQR error at each iteration. Such an upper bound was not previously available. It may be used in a stopping criterion to terminate LSLQ and transfer to the LSQR point.

Strakoš and Tichý (2002) justify the adequacy of A-norm error estimates for CG by way of a finite-precision arithmetic analysis. The upper bounds described in the present paper assume exact arithmetic and orthogonality of the Golub-Kahan bases. In the numerical experiments, our aim has been to observe if the theoretical upper bounds remain upper bounds in practice. They appear to do so up to the point of convergence, as they do for CG and SYMMLQ. We conclude that a future finite-precision analysis is justified.

Fong and Saunders (2012, Table 5.1) summarize the monotonicity of various quantities related to the LSQR and LSMR iterations. Table 1 is similar but focuses on LSQR and LSLQ.

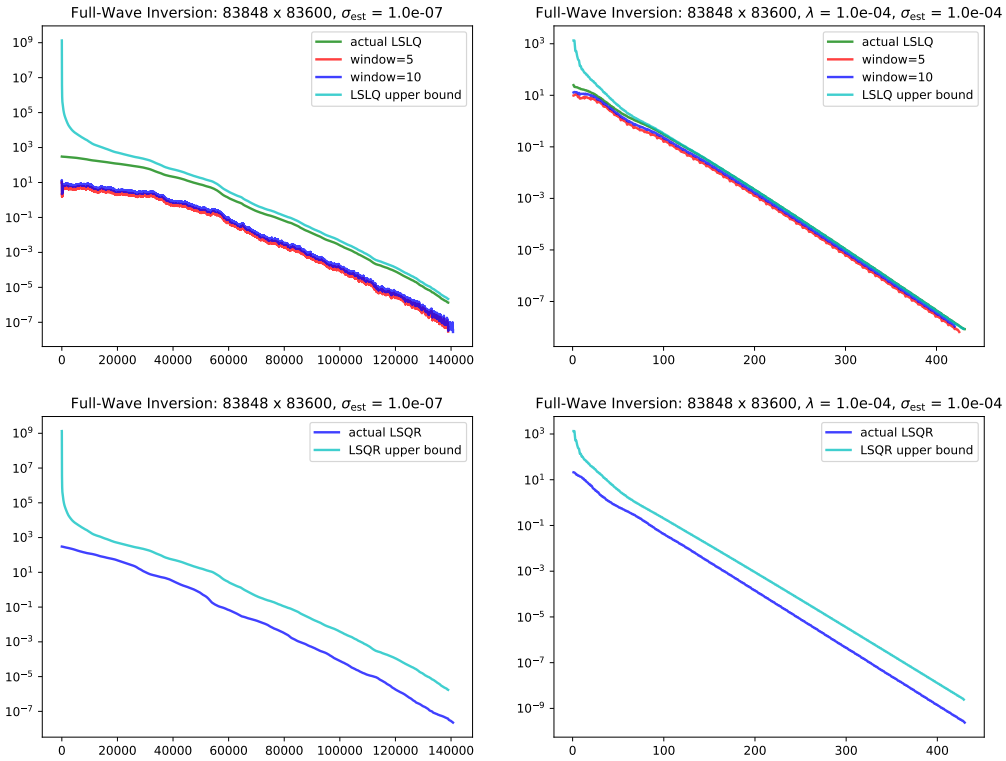

FIG. 6. *Error along the LSLQ and LSQR iterations on the seismic inverse problem without regularization (left) and with regularization (right).*

TABLE 1
Comparison of LSQR and LSLQ properties on a linear least-square problems $\min \|Ax - b\|$.

	LSQR	LSLQ
$\|x_k\|$	↗ (F, 2011, Theorem 3.3.1)	↗ (PS, 1975), ⩽ LSQR (Proposition 1)
$\|x_\star - x_k\|$	↘ (F, 2011, Theorem 3.3.2)	↘ (PS, 1975), ⩾ LSQR (Proposition 1)
$\|r_\star - r_k\|$	↘ (F, 2011, Theorem 3.3.3)	not-monotonic
$\|r_k\|$	↘	not-monotonic
$\|A^T r_k\|$	not-monotonic	not-monotonic
	x_k converges to MLS on column-rank-deficient problems	

↗ monotonically increasing ↘ monotonically decreasing
F (Fong, 2011), PS (Paige and Saunders, 1975)

Saunders, Simon, and Yip (1988) develop the USYMLQ method based on an orthogonal tridiagonalization process that applies to square systems. USYMLQ only applies to consistent systems and, analogous to SYMMLQ, reduces the Euclidean error monotonically. Because the orthogonal tridiagonalization process reduces to the Lanczos (1950) process in the symmetric case, USYMLQ applied to (NE) must be equivalent to SYMMLQ applied to (NE), and therefore to LSLQ applied to (LS), in exact arithmetic. However, applying USYMLQ to (NE) would perform redundant work and require two products with $A^T A$ per iteration.

7.1. A generalization. LSLQ may be generalized to the solution of symmetric quasi-definite systems (Vanderbei, 1995) of the form

$$\begin{bmatrix} M & A \\ A^T & -N \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{45}$$

where $M = M^T$ and $N = N^T$ are positive definite. Indeed (45) represents the optimality conditions of

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \tfrac{1}{2} \left\| \begin{bmatrix} A \\ I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_E^2, \tag{46}$$

where $E = \text{blkdiag}(M^{-1}, N)$. Under the assumption that solves with M and N can be performed cheaply, which is the case in certain optimization schemes and fluid flow simulations (Orban and Arioli, 2017), it suffices to replace the Golub-Kahan process (Algorithm 1) with its preconditioned variant, stated as (Orban and Arioli, 2017, Algorithm 4.2), and to set the regularization parameter $\lambda = 1$.

Note that (45) also represents the optimality conditions of the *least-norm* problem

$$\underset{x \in \mathbb{R}^n,\, s \in \mathbb{R}^m}{\text{minimize}} \ \tfrac{1}{2}(\|r\|_M^2 + \|x\|_N^2) \quad \text{subject to } Mr + Ax = b. \tag{LN2}$$

We may construct a companion method to LSLQ that solves (LN2) by implicitly applying SYMMLQ to the normal equations of the second kind, which in this case are

$$(AN^{-1}A^T + M)r = b, \qquad Nx = A^T r. \tag{NE2}$$

This variant, let us call it LNLQ, is to LSLQ as the method of Craig (1955) is to LSQR. Following the same reasoning as Saunders (1995) and Orban and Arioli (2017), it appears possible to show that applying SYMMLQ to (45) with preconditioner $\text{blkdiag}(M, N)$ is equivalent to applying LSLQ to (46) and LNLQ to (LN2) simultaneously. If so, SYMMLQ applied to (45) would perform twice the work by solving the two equivalent problems (NE) and (NE2) simultaneously, making a solver for (LN2) worthwhile. An implementation of LNLQ is the subject of ongoing work.

Acknowledgements. We are grateful to Tristan van Leeuwen for supplying code that allowed us to generate instances of the seismic inverse problem. We are also deeply grateful to the referees for their insightful recommendations.

References.

S. Arreckx and D. Orban. A regularized factorization-free method for equality-constrained optimization. Cahier du GERAD G-2016-65, GERAD, Montréal, QC, Canada, 2016.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*, volume 1 of *MPS/SIAM Series on Optimization*. SIAM, Philadelphia, USA, 2000. DOI: 10.1137/1.9780898719857.

J. E. Craig. The N-step iteration procedures. *Journal of Mathematics and Physics*, 34 (1):64–73, 1955.

T. A. Davis. Algorithm 930: FACTORIZE: An object-oriented linear system solver for matlab. *ACM Trans. Math. Softw.*, 39(4):28:1–28:18, July 2013. DOI: 10.1145/2491491.2491498.

I. S. Duff, R. G. Grimes, and J. G. Lewis. The Rutherford-Boeing sparse matrix collection. Technical Report RAL-TR-97-031, Rutherford Appleton Laboratory, Chilton, OX, UK, 1997.

R. Estrin, D. Orban, and M. A. Saunders. Euclidean-norm error bounds for CG via SYMMLQ. Cahier du GERAD G-2016-70, GERAD, Montréal, QC, Canada, 2016.

D. C.-L. Fong. Minimum-residual methods for sparse least-squares using golub-kahan bidiagonalization, December 2011.

D. C.-L. Fong and M. A. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. *SIAM J. Sci. Comput.*, 33(5):2950–2971, 2011. DOI: 10.1137/10079687X.

D. C.-L. Fong and M. A. Saunders. CG versus MINRES: An empirical comparison. *SQU Journal for Science*, 17(1):44–62, 2012.

G. H. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *SIAM J. Numer. Anal.*, 2(2):205–224, 1965. DOI: 10.1137/0702016.

G. H. Golub and G. Meurant. Matrices, moments and quadrature II; how to compute the norm of the error in iterative methods. *BIT Num. Math.*, 37(3):687–705, 1997. DOI: 10.1007/BF02510247.

M. Hegland. *On the computation of breeding values*, pages 232–242. Springer Berlin Heidelberg, Berlin, Heidelberg, 1990. ISBN 978-3-540-46597-3. DOI: 10.1007/3-540-53065-7_103.

M. Hegland. Description and use of animal breeding data for large least squares problems. Technical Report TR/PA/93/50, CERFACS, Toulouse, France, 1993.

M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. N.B.S.*, 49(6):409–436, 1952.

C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. N.B.S. B*, 45:225–280, 1950.

G. Meurant. Estimates of the l2 norm of the error in the conjugate gradient algorithm. *Numerical Algorithms*, 40(2):157–169, 2005. DOI: 10.1007/s11075-005-1528-0.

D. Orban. optimizers/animal: Initial release, December 2016. URL https://github.com/optimizers/animal.

D. Orban. `Krylov.jl` suite of iterative methods, July 2017. URL https://github.com/JuliaSmoothOptimizers/Krylov.jl.

D. Orban and M. Arioli. *Iterative Methods for Symmetric Quasi-Definite Linear Systems*. Number 03 in Spotlights. SIAM, Philadelphia, PA, 2017. ISBN 978-1-611974-72-0.

C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975. DOI: 10.1137/0712047.

C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8(1):43–71, 1982a. DOI: 10.1145/355984.355989.

C. C. Paige and M. A. Saunders. Algorithm 583; LSQR: Sparse linear equations and least-squares problems. *ACM Trans. Math. Softw.*, 8(2):195–209, 1982b. DOI: 10.1145/355993.356000.

M. A. Saunders. Solution of sparse rectangular systems using LSQR and CRAIG. *BIT Num. Math.*, 35:588–604, 1995. DOI: 10.1007/BF01739829.

M. A. Saunders, H. D. Simon, and E. L. Yip. Two conjugate-gradient-type methods for unsymmetric linear equations. *SIAM J. Numer. Anal.*, 25(4):927–940, 1988. DOI: 10.1137/0725052.

G. W. Stewart. The QLP approximation to the singular value decomposition. *SIAM J. Sci. Comput.*, 20(4):1336–1348, 1999. DOI: 10.1137/S1064827597319519.

Z. Strakoš and P. Tichý. On error estimation in the conjugate gradient method and why it works in finite precision. *Elec. Trans. Numer. Anal.*, 13, 2002.

T. Van Leeuwen and F. J. Herrmann. A penalty method for PDE-constrained op-

timization. Technical Report TR-EOAS-2013-6, University of British Columbia, 2013.

R. J. Vanderbei. Symmetric quasi-definite matrices. *SIAM J. Optim.*, 5(1):100–113, 1995. DOI: 10.1137/0805005.