SIAM J. MATRIX ANAL. APPL.
Vol. 39, No. 3, pp. 1295–1317

LSMB: MINIMIZING THE BACKWARD ERROR FOR LEAST-SQUARES PROBLEMS*

ERIC HALLMAN† AND MING GU†

Abstract. An iterative method, LSMB, is given for solving $\min_x \|Ax - b\|_2$. LSMB is based on the Golub–Kahan bidiagonalization process and is constructed so that an objective function closely related to the backward error for the least-squares problem is minimized with every iteration. We find that at every step the iterate x_k produced by LSMB is a convex combination of those produced by LSQR (which minimizes $\|r_k\|_2 = \|b - Ax_k\|_2$ over a Krylov subspace) and LSMR (which minimizes $\|A^T r_k\|_2$ over the same subspace). Experiments on test cases from the University of Florida Sparse Matrix Collection show that in practice LSMB performs at least as well as both LSQR and LSMR, although never by more than a small margin. This suggests that LSMB could replace both solvers when stopping rules are based on the backward error.

Key words. least-squares problem, sparse matrix, LSQR, LSMR, Krylov subspace method, Golub–Kahan process, conjugate-gradient method, backward perturbation analysis, stopping criteria, minimum-residual method, iterative method, backward error

AMS subject classifications. 15A06, 65F10, 65F20, 65F50, 93E24

DOI. 10.1137/17M1157106

1. Introduction.

We present an algorithm called LSMB that, given a matrix $A \in \mathbb{R}^{m\times n}$, a vector $b \in \mathbb{R}^m$, and possibly a scalar λ, solves the least-squares problems

$$\min_x \|Ax - b\| \quad \text{or} \quad \min_x \left\| \begin{pmatrix} A \\ \lambda I \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|.$$

Least-squares problems have a wide variety of applications in scientific disciplines, including statistics, signal processing, computer graphics, and systems control; see [1] and [7] for some general background. We assume that we can compute matrix-vector products of the form Av and $A^T u$ for any vectors u and v, but otherwise do not need access to the entries of A. Thus our algorithm is suitable for cases where A is sparse or a fast linear operator.

LSMB is an iterative algorithm closely related to LSQR [20] and to the more recent LSMR [4] in that all three are based on the Golub–Kahan bidiagonalization of A [6]. At the kth iteration, LSQR finds an approximate solution x_k that minimizes the residual norm $\|r_k\| = \|b - Ax_k\|$ over a Krylov subspace, while LSMR minimizes $\|A^T r_k\|$ over the same subspace. In exact arithmetic LSQR and LSMR are equivalent to running the conjugate-gradient method [14] and MINRES [19], respectively, on the normal equations $A^T Ax = A^T b$ or $(A^T A + \lambda^2 I)x = A^T b$.

Both LSQR and LSMR offer cheaply computable upper bounds for the *backward error*: the norm of the smallest perturbation to A (and possibly also b) such that x_k is an exact solution to the perturbed system. As the backward error is typically

*Received by the editors November 15, 2017; accepted for publication (in revised form) by J. L. Barlow June 8, 2018; published electronically August 16, 2018.

http://www.siam.org/journals/simax/39-3/M115710.html

†Department of Mathematics, University of California, Berkeley, CA 94720 (ehallman@berkeley.edu, mgu@math.berkeley.edu).

used to determine when an iterative algorithm may be safely terminated, it is in our best interest to find inexpensive estimates of the backward error that are as accurate as possible. Recent work [15] has found an estimate of the backward error ($\overline{\nu}_0(x_k)$) for LSQR that can be measured in $O(k)$ time and storage and is provably smaller than the existing estimates based on $\|r_k\|$ and $\|A^T r_k\|$, thus allowing the iterative algorithm to be terminated sooner.

LSMB is designed to minimize this new backward error estimate at every iteration. In this way it is similar in spirit to the algorithms GMBACK and MINPERT [17, 18], which are variants of GMRES [22] that minimize the backward error for the unsymmetric system $Ax = b$, and to CGBACK [10], a variant of the conjugate-gradient method that minimizes the energy backward error for the symmetric positive definite system $Ax = b$. We present LSMB as an extension of algorithms LSQR and LSMR and show that all three can be run simultaneously for a minimal $O(1)$ cost beyond that already required by LSMR. In practice we find that the error for LSMB is always less than the minimum of the two estimates provided by LSQR and LSMR, but it turns out that this difference is never greater than a factor of $\sqrt{2}$. This suggests that LSMB will perform about as well as (or marginally better than) running LSQR and LSMR simultaneously.

We also show how to tighten the backward error estimate for LSMB when solving regularized least-squares problems or when the smallest singular value of A is known. These same techniques can be used to find tighter estimates of $\|P_A r_k\|$ (the norm of the projection of the residual onto the span of A) for LSQR. As a result, LSMB will still not substantially outperform both LSQR and LSMR, but the improved estimate of $\|P_A r_k\|$ sometimes allows LSQR to terminate sooner than LSMR on inconsistent problems, particularly low-residual problems.

1.1. Overview. Section 2 reviews the Golub–Kahan bidiagonalization process and introduces a new derivation of LSMR that allows us to run LSQR and LSMR simultaneously at slightly lower cost. Section 3 reviews the backward error and a variety of methods for estimating it. Section 4 derives the main part of LSMB and shows that it will always lie on the line segment between LSQR and LSMR. Section 5 covers backward error estimates and shows how to pick an appropriate point on the line segment. Section 6 shows the results of experiments on a range of overdetermined problems. Section 7 offers our concluding remarks.

1.2. Notation. We denote matrices by capital Roman letters A, B, ..., vectors by lowercase Roman letters u, v, ..., and scalars by lowercase Greek letters α, β, ... Givens rotations are an exception, in which case c and s represent the components of the rotation. The notation $\|v\|$ always means the 2-norm of the vector v, and $\|A\|$ always means the Frobenius norm of the matrix A. The vector e_k is the kth standard basis vector of $\mathbb{R}^n$. The minimum-norm solution to the least-squares problem is denoted by x_*, and $r_* = b - Ax_*$ is the optimal residual.

Underlined scalars ($\underline{\rho}, \underline{\overline{\theta}}$) arise in the process of QR factorizations and will change into related elements ($\rho, \overline{\theta}$) as the factorization progresses. Superscripts ($\overline{Q}, \widehat{R}$) distinguish the QR factorizations used in our algorithm as well as scalars ($\hat{\theta}, \tilde{\rho}$) associated with these QR factorizations. The values $\overline{\nu}_d$ and $\underline{\nu}_d$ are exceptions that denote upper and lower bounds on the backward error estimate ν. Another exception is the scalar $\overline{\phi}_{k+1}$, a legacy from the original LSQR paper [20] and an element that will change into ϕ_{k+1} in the next iteration.

The pseudoinverse and 2-norm condition number of A are denoted by $A^\dagger$ and

cond(A), respectively. The notation $A \succeq B$ means that the matrix $A - B$ is positive semidefinite.

2. The Golub–Kahan process, LSQR, and LSMR. We start with a short summary of the Golub–Kahan bidiagonalization process [6], an iterative method originally designed to estimate the singular values of a matrix A by reducing it to a lower bidiagonal matrix B. We use this process to reproduce the derivation of LSQR from [20] and to provide a new derivation of LSMR. This new derivation will allow us to run LSQR and LSMR simultaneously for the same cost as running LSMR alone.

2.1. Golub–Kahan bidiagonalization. The Golub–Kahan process takes a matrix A and vector b and after k steps produces orthogonal matrices $U_k = (u_1, \ldots, u_k)$ and $V_k = (v_1, \ldots, v_k)$ such that

$$\begin{aligned}\text{Span}(U_k) &= \text{Span}\left\{b, (AA^T)b, \ldots, (AA^T)^{k-1}b\right\} = \mathcal{K}_k(AA^T, b),\\ \text{Span}(V_k) &= \text{Span}\left\{A^Tb, (A^TA)A^Tb, \ldots, (A^TA)^{k-1}A^Tb\right\} = \mathcal{K}_k(A^TA, A^Tb).\end{aligned}$$

The process itself proceeds as follows:

1. Set $\beta_1 u_1 = b$ (i.e., $\beta_1 = \|b\|$, $u_1 = b/\beta_1$) and $\alpha_1 v_1 = A^T u_1$.
2. For $k = 1, 2, \ldots$ compute $\beta_{k+1}u_{k+1} = Av_k - \alpha_k u_k$ and $\alpha_{k+1}v_{k+1} = A^T u_{k+1} - \beta_{k+1}v_k$.

If we additionally define

$$L_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \end{pmatrix}, \quad B_k = \begin{pmatrix} L_k \\ \beta_{k+1}e_k^T \end{pmatrix}, \tag{2.1}$$

we can characterize the process at each iteration by the two relations

$$AV_k = U_{k+1}B_k \quad \text{and} \quad A^TU_{k+1} = V_{k+1}L_{k+1}^T. \tag{2.2}$$

Although the matrices U_k and V_k are orthogonal in exact arithmetic, they will quickly lose their orthogonality in practice unless additional steps are taken to reorthogonalize the vectors u_k and v_k. Some strategies for reorthogonalization are explored in [4]; we do not cover the topic further here, but note that even when U_k and V_k are not exactly orthogonal, all of our error estimates appear to be quite accurate in practice.

In the event that $\alpha_k = 0$ or $\beta_k = 0$, both LSQR and LSMR terminate and yield the least-squares solution x_*. The case $\beta_k = 0$ additionally implies that $Ax_* = b$ (see [4, sects. 3.6 and 4] for details).

2.2. Subproblems for LSQR and LSMR. At every iteration, LSQR and LSMR minimize their respective objective functions over the space $\mathcal{K}_k(A^TA, A^Tb) = \text{Span}(V_k)$, and so their iterates may be written as $x_k = V_k y_k$ for some $y_k \in \mathbb{R}^k$. To distinguish the two algorithms, we denote the iterates for LSQR and LSMR by x_k^C (and y_k^C, r_k^C, etc.) and x_k^M, respectively. The superscripts stand for "conjugate gradient" and "MINRES," following the notation of [19] and others.

Since LSQR was designed to solve the least-squares problem $\min_x \|b - Ax\|$, the iterate was chosen to minimize the residual norm $\|r_k\|$ at every step. The first relation in (2.2) implies that

$$r_k = b - Ax_k = b - AV_k y_k = \beta_1 u_1 - U_{k+1}B_k y_k = U_{k+1}\left(\beta_1 e_1 - B_k y_k\right), \tag{2.3}$$

and because U_k is orthogonal (in exact arithmetic) it follows that

$$\min_{x_k} \|r_k\| = \min_{y_k} \|\beta_1 e_1 - B_k y_k\|. \tag{2.4}$$

Since B_k is lower bidiagonal, we can solve this subproblem efficiently by finding its QR factorization.

Any solution to the least-squares problem also satisfies the relation $A^T r_* = 0$, so LSMR was designed to minimize $\|A^T r_k\|$ at every iteration. This objective function has the advantage of converging to zero for both consistent and inconsistent systems, which is useful when we are considering stopping criteria.

The second relation in (2.2) then implies that

$$A^T r_k = A^T U_{k+1}(\beta_1 e_1 - B_k y_k) = V_{k+1} L_{k+1}^T (\beta_1 e_1 - B_k y_k), \tag{2.5}$$

and so we can derive the algorithm for LSMR by solving the subproblem

$$\min_{x_k} \|A^T r_k\| = \min_{y_k} \left\| L_{k+1}^T (\beta_1 e_1 - B_k y_k) \right\|. \tag{2.6}$$

This subproblem can be solved efficiently using a second QR factorization.

We note that the formulation for LSMR that we derive in (2.6) is slightly different from the one originally derived in [4], which follows from the relation between B_k and L_{k+1} in (2.1):

$$\min_{x_k} \|A^T r_k\| = \min_{y_k} \left\| \alpha_1 \beta_1 e_1 - \begin{pmatrix} B_k^T B_k \\ \alpha_{k+1}\beta_{k+1} e_k^T \end{pmatrix} y_k \right\|. \tag{2.7}$$

Subproblem (2.7) can be solved using the same two QR factorizations as before and will be used in section 5.3 to estimate the norm of the LSMB iterates. The primary difference is that solving (2.7) leads to a recursive formula for the iterates of LSMR, while our solution to (2.6) expresses the iterates for LSMR in the form of the iterates for LSQR plus a correction term, taking advantage of a relationship analogous to that between CG and MINRES [19, sect. 7]. This allows us to run LSQR and LSMR (as well as LSMB) simultaneously for roughly the same cost as running LSMR alone.

2.3. QR factorizations. Because B_k and L_{k+1} are closely related bidiagonal matrices, we can solve subproblems (2.4) and (2.6) efficiently by performing a pair of QR factorizations and solving the resulting upper triangular systems.

These two factorizations can be expressed as the products

$$Q_{k+1} = P_k \dots P_2 P_1,$$
$$\overline{Q}_{k+1} = \overline{P}_k \dots \overline{P}_2 \overline{P}_1,$$

where P_i and $\overline{P}_i$ are Givens rotations operating on rows i and $i+1$ of a given matrix and having significant components $\left(\begin{smallmatrix} c_k & s_k \\ -s_k & c_k \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} \bar{c}_k & \bar{s}_k \\ -\bar{s}_k & \bar{c}_k \end{smallmatrix}\right)$, respectively. As derived in [20], the first QR factorization takes the form

$$Q_{k+1} \begin{pmatrix} B_k & \beta_1 e_1 \end{pmatrix} = \begin{pmatrix} R_k & f_k \\ 0 & \overline{\phi}_{k+1} \end{pmatrix} = \left(\begin{array}{cccc:c} \rho_1 & \theta_2 & & & \phi_1 \\ & \rho_2 & \ddots & & \phi_2 \\ & & \ddots & \theta_k & \vdots \\ & & & \rho_k & \phi_k \\ \hline & & & & \overline{\phi}_{k+1} \end{array}\right). \tag{2.8}$$

The components of R_k and f_k can be computed by the recurrence relation

$$\begin{pmatrix} c_k & s_k \\ -s_k & c_k \end{pmatrix} \begin{pmatrix} \underline{\rho}_k & 0 & \overline{\phi}_k \\ \beta_{k+1} & \alpha_{k+1} & 0 \end{pmatrix} = \begin{pmatrix} \rho_k & \theta_{k+1} & \phi_k \\ 0 & \underline{\rho}_{k+1} & \overline{\phi}_{k+1} \end{pmatrix},$$

with $\underline{\rho}_1 = \alpha_1$ and $\overline{\phi}_1 = \beta_1$. We also use the notation

$$Q_{k+1} L_{k+1} = \underline{R}_{k+1} = \begin{pmatrix} R_k & \theta_{k+1} e_k \\ 0 & \underline{\rho}_{k+1} \end{pmatrix}, \tag{2.9}$$

so that $\underline{R}_{k+1}$ is identical to R_{k+1} except for the final element.

As derived in [4], the second QR factorization takes the form

$$\overline{Q}_{k+1} \underline{R}_{k+1}^T = \overline{R}_{k+1} = \begin{pmatrix} \overline{R}_k & \widehat{\theta}_{k+1} e_k \\ 0 & \widehat{\rho}_{k+1} \end{pmatrix} = \left(\begin{array}{cccc:c} \overline{\rho}_1 & \overline{\theta}_2 & & & \\ & \overline{\rho}_2 & \ddots & & \\ & & \ddots & \overline{\theta}_k & \\ & & & \underline{\rho}_k & \widehat{\theta}_{k+1} \\ \hdashline & & & & \widehat{\rho}_{k+1} \end{array}\right). \tag{2.10}$$

The matrix $\overline{R}_k$ can be computed by the recurrence relation

$$\begin{pmatrix} \overline{c}_k & \overline{s}_k \\ -\overline{s}_k & \overline{c}_k \end{pmatrix} \begin{pmatrix} \underline{\rho}_k & 0 & 0 \\ \theta_{k+1} & \rho_{k+1} & \underline{\rho}_{k+1} \end{pmatrix} = \begin{pmatrix} \overline{\rho}_k & \overline{\theta}_{k+1} & \widehat{\theta}_{k+1} \\ 0 & \underline{\overline{\rho}}_{k+1} & \widehat{\rho}_{k+1} \end{pmatrix}$$

with $\underline{\overline{\rho}}_1 = \rho_1$.

2.4. Solutions for LSQR and LSMR. For LSQR, [20] uses the first QR factorization (2.8) and subproblem (2.4) to arrive at the problem

$$\min_{x_k} \|r_k\| = \min_{y_k} \|\beta_1 e_1 - B_k y_k\| = \min_{y_k} \left\| \begin{pmatrix} f_k \\ \overline{\phi}_{k+1} \end{pmatrix} - \begin{pmatrix} R_k y_k \\ 0 \end{pmatrix} \right\|. \tag{2.11}$$

The solution satisfies $R_k y_k^C = f_k$, which makes most of the entries on the right-hand side zero. It gives the residual norm $\|r_k^C\| = \left|\overline{\phi}_{k+1}\right|$, although y_k^C and r_k^C are not computed explicitly.

For LSMR, we additionally use the second QR factorization (2.10) and subproblem (2.6) to arrive at the problem

$$\min_{x_k} \|A^T r_k\| = \min_{y_k} \left\| \begin{pmatrix} \overline{R}_k & \widehat{\theta}_{k+1} e_k \\ 0 & \widehat{\rho}_{k+1} \end{pmatrix} \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\| \tag{2.12}$$

$$= \min_{y_k} \left\| \begin{pmatrix} \overline{R}_k (f_k - R_k y_k) + \overline{\phi}_{k+1} \widehat{\theta}_{k+1} e_k \\ \widehat{\rho}_{k+1} \overline{\phi}_{k+1} \end{pmatrix} \right\|. \tag{2.13}$$

The solution satisfies $\overline{R}_k R_k y_k^M = \overline{R}_k f_k + \overline{\phi}_{k+1} \widehat{\theta}_{k+1} e_k$, which again makes most of the entries in the final expression zero. In this case $\|A^T r_k^M\| = \widehat{\rho}_{k+1} |\overline{\phi}_{k+1}|$, although y_k^M, r_k^M, and $A^T r_k^M$ are still not computed explicitly.

2.5. Recurrence for x_k^C and x_k^M. If we solve $R_k^T W_k^T = V_k^T$ and $\overline{R}_k^T \overline{W}_k^T = W_k$ by forward substitution, we get the relations

$$x_k^C = V_k y_k^C = V_k R_k^{-1} f_k = W_k f_k = x_{k-1}^C + \phi_k w_k, \tag{2.14}$$

$$x_k^M = V_k y_k^M = V_k R_k^{-1} f_k + \overline{\phi}_{k+1} \widehat{\theta}_{k+1} V_k R_k^{-1} \overline{R}_k^{-1} e_k = x_k^C + \overline{\phi}_{k+1} \widehat{\theta}_{k+1} \overline{w}_k. \tag{2.15}$$

The vectors w_k and $\overline{w}_k$ can be computed by the recurrences

$$\theta_{k+1} w_k + \rho_{k+1} w_{k+1} = v_{k+1},$$
$$\overline{\theta}_k \overline{w}_{k-1} + \overline{\rho}_k \overline{w}_k = w_k,$$

but as mentioned in [4] we can make the algorithm more efficient if we define $h_k = \rho_k w_k$ and $\overline{h}_k = \rho_k \overline{\rho}_k \overline{w}_k$ and compute $h_k, \overline{h}_k$ rather than $w_k, \overline{w}_k$ at each step. We can therefore solve for x_k^C and x_k^M efficiently by defining $x_0^C = 0$, $h_1 = v_1$, $\overline{h}_0 = 0$ and at every iteration computing

$$\overline{h}_k = h_k - \left(\frac{\overline{\theta}_k \rho_k}{\rho_{k-1} \overline{\rho}_{k-1}} \right) \overline{h}_{k-1},$$
$$x_k^C = x_{k-1}^C + (\phi_k / \rho_k) h_k,$$
$$h_{k+1} = v_{k+1} - (\theta_{k+1} / \rho_k) h_k.$$

Upon termination of the algorithm, we can compute x_k^M as

$$x_k^M = x_k^C + \overline{\phi}_{k+1} \widehat{\theta}_{k+1} \left(\frac{1}{\rho_k \overline{\rho}_k} \right) \overline{h}_k.$$

Notably, none of the estimates used as stopping criteria in LSMR requires explicit computation of x_k^M or r_k^M, so although we could compute x_k^M at every step k it is not necessary to do so. This opens up an interesting possibility: rather than choosing ahead of time whether to use LSQR or LSMR for a problem, we can run LSQR while keeping track of the error estimates for both methods, then transfer to LSMR at any point.

2.6. Complexity. Both LSQR and LSMR require the computation of $A^T u$ and Av at each iteration. If A is dense, then this will be the dominant cost, but if A is sparse, then the extra vector operations may be an important consideration in the run time of these algorithms. The QR factorizations can be computed with minimal flops and memory, but computing the rest of the bidiagonalization process and the recurrences costs a further $3m + 5n$ and $3m + 6n$ flops for LSQR and LSMR, respectively.

While the original derivation for LSMR provided a recursive formula for x_k^M, our derivation interprets x_k^M as being equal to x_k^C plus a correction term that changes with every iteration. This gives us a modest improvement in efficiency: while previously running LSQR and LSMR simultaneously would cost an extra $3m + 7n$ flops per iteration, we have reduced the cost to $3m+6n$, the same cost as running LSMR alone. We later show how to extend the algorithm to include LSMB at $O(1)$ extra cost.

3. Motivation for LSMB. LSQR and LSMR offer stopping rules that can be checked cheaply [20, 4]. For dimensionless quantities τ, ϵ, and κ, the rules are as follows:

1. Stop if $\frac{\|r_k\|}{\|A\|\|x_k\| + \tau\|b\|} < \epsilon$.

2. Stop if $\frac{\|A^T r_k\|}{\|A\|\|r_k\|} < \epsilon$.
3. Stop if $\text{cond}(A) > \kappa$.

The first stopping rule is triggered earlier in LSQR than in LSMR since $\|r_k^C\| \leq \|r_k^M\|$ (by design) and $\|x_k^C\| \geq \|x_k^M\|$ (we prove this later). The second stopping rule is triggered earlier in LSMR than in LSQR since $\|A^T r_k^M\| \leq \|A^T r_k^C\|$ and $\|r_k^M\| \geq \|r_k^C\|$. The third stopping rule has a function similar to adding a regularization term, preventing $\|x_k\|$ from growing too large when A is ill-conditioned. The quantities $\|A\|$ and $\text{cond}(A)$ can be estimated by $\|B_k\|$ and $\text{cond}(B_k)$ [4, sect. 3.4].

The first two stopping rules are both estimates of the *backward error*: the smallest possible perturbation to the input data A and b such that x_k is an exact solution for the perturbed system.

DEFINITION 3.1. *Given a matrix A, arbitrary vectors x and b, and flexible parameter $\tau \geq 0$, the* backward error *for the least-squares problem* $\min_x \|Ax - b\|$ *is defined as*

$$\mu(x, \tau) = \min_{E,f} \|E, \tau f\| \ : \ (A + E)^T \left[(A + E)x - (b + f)\right] = 0.$$

Note that x must solve the perturbed problem $\min_x \|(A + E)x - (b + f)\|$.

In most applications there is typically some uncertainty in the input data A and b. If we run an iterative algorithm until the backward error is smaller than this level of uncertainty, then the implication is that we can safely stop because our current solution is as accurate as any we can realistically hope to achieve. Above this threshold, a smaller backward error should in general imply a higher quality solution.

The first two stopping rules for LSQR and LSMR are both upper bounds for $\mu(x, \tau)$. Although these rules are cheap to evaluate, they can be unnecessarily conservative, and so in certain situations LSQR and LSMR may run longer than necessary. For LSMB we attempt to minimize a more recent and tighter estimate of the backward error, leading to a new stopping rule that lets us terminate the algorithm sooner. In the next section we give a brief history of proposed estimates of $\mu(x, \tau)$.

3.1. Estimates of $\mu(x, \tau)$. In 1995, Waldén, Karlson, and Sun [26] showed that

$$\mu(x, \tau) = \min\left\{\omega, \sigma_{\min}\left[A, \ \omega(I - rr^\dagger)\right]\right\},$$

where x is arbitrary, $r = b - Ax$, $r^\dagger = r^T/\|r\|^2$, and ω is defined in [21] as

$$\omega = \min_{E,f} \|E, \tau f\| \ : \ (A + E)x = b + f \tag{3.1}$$

$$= \frac{\tau\|r\|}{\sqrt{1 + \tau^2\|x\|^2}}. \tag{3.2}$$

Since it involves computing the smallest singular value of an $m \times (m + n)$ matrix, this formula is far too expensive to use as a stopping rule for iterative algorithms, but several estimates have been well studied.

- In 1975, Stewart [23] gave two backward perturbations,

$$E_0 = \frac{rx^T}{\|x\|^2}, \ \|E_0\| = \frac{\|r\|}{\|x\|}, \qquad E_1 = \frac{(P_A r)x^T}{\|x\|^2}, \ \|E_1\| = \frac{\|P_A r\|}{\|x\|}, \tag{3.3}$$

  where $P_A r = AA^\dagger r$ is the projection of r onto the column span of A. The perturbation E_0 is the optimal backward perturbation for the consistent problem

$Ax = b$ with $f = 0$, and LSQR performs well with respect to both $\|E_0\|$ and $\|E_1\|$ because it minimizes the norm of both the residual and its projection at every step.

- In 1977, Stewart [24] gave the additional backward perturbation

$$E_2 = -\frac{rr^T A}{\|r\|^2}, \quad \|E_2\| = \frac{\|A^T r\|}{\|r\|}.$$

LSMR, minimizing $\|A^T r\|$ at every step, performs well with respect to $\|E_2\|$.

- In 1997, Karlson and Waldén [16] proposed the estimate

$$\nu(x,\tau) = \omega\|(A^T A + \omega^2 I)^{-1/2} A^T r\|/\|r\| = \frac{\omega}{\|r\|} \left\| \begin{bmatrix} A \\ \omega I \end{bmatrix} \begin{bmatrix} A \\ \omega I \end{bmatrix}^\dagger \begin{bmatrix} r \\ 0 \end{bmatrix} \right\|,$$

where ω is defined in (3.1)–(3.2). Various authors [13, 11, 12, 25] have provided bounds on the accuracy of this estimate, the tightest of which were given in 2012 by Gratton, Jiránek, and Titley-Peloquin [8]; they showed that if r_* is the smallest attainable residual for the least-squares problem, then the bounds

$$1 \le \frac{\mu(x,\tau)}{\nu(x,\tau)} \le \sqrt{2 - \frac{\|r_*\|^2}{\|r\|^2}} \le \sqrt{2}$$

hold for all x and τ. Thus $\nu(x,\tau)$ is always a good estimate of $\mu(x,\tau)$, and the estimate becomes increasingly accurate as x converges to x_*.

- The estimate $\nu(x,\tau)$ is unfortunately still too expensive to use directly as a stopping rule, but in 2010, Jiránek and Titley-Peloquin [15] proposed upper and lower bounds that were cheaper to compute. For a vector $x_k = V_k y_k$ and flexible parameter d, define $t_k = \beta_1 e_1 - B_k y_k$ and

$$\overline{\mu}_d(x_k,\tau) = \min\left\{\omega, \sigma_{\min}\left[L_{k+d+1},\ \omega(I - t_k t_k^\dagger)\right]\right\},$$
$$\underline{\mu}_d(x_k,\tau) = \min\left\{\omega, \sigma_{\min}\left[B_{k+d},\ \omega(I - t_k t_k^\dagger)\right]\right\}.$$

The authors showed that the bounds

$$\underline{\mu}_d(x_k,\tau) \le \mu(x_k,\tau) \le \overline{\mu}_d(x_k,\tau)$$

hold for all $d \ge 0$ in exact arithmetic, becoming progressively tighter as d increases, and recommended estimating the backward error with the corresponding approximations

$$\overline{\nu}_d(x_k,\tau) = \frac{\omega}{\|r_k\|} \left\| (L_{k+d+1}^T L_{k+d+1} + \omega^2 I)^{-1/2} L_{k+d+1}^T \begin{bmatrix} t_k \\ 0_d \end{bmatrix} \right\|, \tag{3.4}$$

$$\underline{\nu}_d(x_k,\tau) = \frac{\omega}{\|r_k\|} \left\| (B_{k+d}^T B_{k+d} + \omega^2 I)^{-1/2} B_{k+d}^T \begin{bmatrix} t_k \\ 0_d \end{bmatrix} \right\|. \tag{3.5}$$

Thus by advancing some number of steps in the bidiagonalization process it is possible to get a more accurate estimate of the backward error at a previous iteration. The authors showed how to compute these bounds in $O(k+d)$ time and memory for LSQR (using some value $d > 0$ can be helpful because the least-squares backward error does not in general decrease monotonically for LSQR).

The authors also provided the bound

$$\|P_A r_k\|^2 \geq \sum_{i=k+1}^{k+d} \phi_i^2,$$

which puts a lower bound on the error $\|E_1\|$ (3.3) and can be computed in $O(d)$ time and memory.

- In 2013, Gratton, Jiránek, and Titley-Peloquin [9] proved the bounds

$$\frac{1}{\sqrt{1+\operatorname{cond}(A)}} \min\left\{\omega\frac{\|P_A r\|}{\|r\|}, \frac{\|A^T r\|}{\|r\|}\right\} \leq \mu(x,\tau) \leq \min\left\{\omega\frac{\|P_A r\|}{\|r\|}, \frac{\|A^T r\|}{\|r\|}\right\}$$

and showed that while the upper bound is typically tight, there exist cases where both LSQR and LSMR produce iterates for which $\mu(x,\tau)$ is much smaller than the upper bound. It is therefore conceivable for ill-conditioned cases that both LSQR and LSMR produce iterates whose backward errors are much larger than the minimum attainable backward error.

The backward error estimate that we use as the basis for LSMB comes from a generalization of the bounds (3.4) and (3.5). These estimates have a structure simple enough that, rather than just using them to estimate the error for LSQR or LSMR, we can attempt to minimize them directly.

4. Algorithm LSMB. After k steps of the bidiagonalization process, we would ideally like to solve

$$\min_{x_k} \frac{\omega_k}{\|r_k\|} \left\|(A^T A + \omega_k^2 I)^{-\frac{1}{2}} A^T r_k\right\|. \tag{4.1}$$

This is difficult because ω_k changes with x_k, and so we instead try solve the simpler problem

$$\min_{x_k} \left\|(A^T A + \tilde{\omega}^2 I)^{-\frac{1}{2}} A^T r_k\right\|, \tag{4.2}$$

where $\tilde{\omega}$ is an arbitrary constant. We therefore end up with a family of algorithms and, interestingly, a bridge between LSQR and LSMR. When $\tilde{\omega} = 0$ we recover LSQR exactly, and as $\tilde{\omega}$ approaches infinity the optimal solution becomes arbitrarily close to the solution produced by LSMR.

This formulation of the problem can also be used to draw a connection between the error estimates from LSQR and LSMR. Using the inequality

$$A^T A + \omega_k^2 I \succeq \min\{1, \omega_k^2/\tilde{\omega}^2\}(A^T A + \tilde{\omega}^2 I),$$

we obtain the upper bound

$$\frac{\omega_k}{\|r_k\|} \left\|(A^T A + \omega_k^2 I)^{-\frac{1}{2}} A^T r_k\right\| \leq \frac{\max\{\omega_k, \tilde{\omega}\}}{\|r_k\|} \left\|(A^T A + \tilde{\omega}^2 I)^{-\frac{1}{2}} A^T r_k\right\|. \tag{4.3}$$

When $\tilde{\omega} = 0$ this upper bound is precisely equal to $\omega_k \|P_A r_k\|/\|r_k\|$, and as $\tilde{\omega}$ approaches infinity the upper bound converges to $\|A^T r_k\|/\|r_k\|$. Our hope is to find some intermediate value of $\tilde{\omega}$ so that the error estimate for LSMB is smaller than either of these upper bounds.

4.1. Subproblem for LSMB. We still cannot solve (4.2) exactly because we do not have the full bidiagonalization of A, but we can express the problem in a simpler form by using the following theorem.

THEOREM 4.1. *For any $\tilde{\omega} \in \mathbb{R}$, there exists a $\tilde{\beta}_{k+2}$ (dependent on $\tilde{\omega}$) such that*

$$V_{k+1}^T (A^T A + \tilde{\omega}^2 I)^{-1} V_{k+1} = (\widetilde{B}_{k+1}^T \widetilde{B}_{k+1} + \tilde{\omega}^2 I)^{-1},$$

where $\widetilde{B}_{k+1} = \begin{pmatrix} L_{k+1} \\ \tilde{\beta}_{k+2} e_{k+1}^T \end{pmatrix}$ and $0 \le \tilde{\beta}_{k+2} \le \beta_{k+2}$. Furthermore, $\sigma_{\min}(\widetilde{B}_{k+1}) \ge \sigma_{\min}(A)$ regardless of $\tilde{\omega}$.

Proof. Let $\overline{U}$ and $\overline{V}$ be any matrices such that $[U_{k+2},\ \overline{U}]$ and $[V_{k+1},\ \overline{V}]$ are square and orthogonal. Based on the relations in (2.2), we find that

$$\begin{aligned} A[V_{k+1},\ \overline{V}] &= [U_{k+2},\ \overline{U}][U_{k+2},\ \overline{U}]^T [U_{k+2} B_{k+1},\ A\overline{V}] \\ &= [U_{k+2},\ \overline{U}] \begin{bmatrix} B_{k+1} & U_{k+2}^T A \overline{V} \\ 0 & \overline{U}^T A \overline{V} \end{bmatrix} \\ &= [U_{k+2},\ \overline{U}] \begin{bmatrix} B_{k+1} & L_{k+2} V_{k+2}^T \overline{V} \\ 0 & \overline{U}^T A \overline{V} \end{bmatrix} \\ &= [U_{k+2},\ \overline{U}] \begin{bmatrix} B_{k+1} & e_{k+2} \tilde{v}^T \\ 0 & \overline{U}^T A \overline{V} \end{bmatrix}, \end{aligned}$$

where $\tilde{v} = \alpha_{k+2} \overline{V}^T v_{k+2}$. It follows that

$$[V_{k+1},\ \overline{V}]^T (A^T A + \tilde{\omega}^2 I)[V_{k+1},\ \overline{V}] = \begin{bmatrix} B_{k+1}^T B_{k+1} + \tilde{\omega}^2 I & \beta_{k+2} e_{k+1} \tilde{v}^T \\ \beta_{k+2} \tilde{v} e_{k+1}^T & \widetilde{A} \end{bmatrix},$$

where $\widetilde{A} = \overline{V}^T A^T \overline{U}\overline{U}^T A \overline{V} + \tilde{v}\tilde{v}^T + \tilde{\omega}^2 I$. If we then use the formula for 2×2 block matrix inversion

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} (A_{11} - A_{12} A_{22}^{-1} A_{21})^{-1} & * \\ * & * \end{bmatrix},$$

we can conclude that

$$\begin{aligned} V_{k+1}^T (A^T A + \tilde{\omega}^2 I)^{-1} V_{k+1} &= \begin{bmatrix} I \\ 0 \end{bmatrix}^T \begin{bmatrix} B_{k+1}^T B_{k+1} + \tilde{\omega}^2 I & \beta_{k+2} e_{k+1} \tilde{v}^T \\ \beta_{k+2} \tilde{v} e_{k+1}^T & \widetilde{A} \end{bmatrix}^{-1} \begin{bmatrix} I \\ 0 \end{bmatrix} \\ &= \left(B_{k+1}^T B_{k+1} + \tilde{\omega}^2 I - \beta_{k+2}^2 \left(\tilde{v}^T \widetilde{A}^{-1} \tilde{v} \right) e_{k+1} e_{k+1}^T \right)^{-1} \\ &= \left(L_{k+1}^T L_{k+1} + \tilde{\omega}^2 I + \beta_{k+2}^2 \left(1 - \tilde{v}^T \widetilde{A}^{-1} \tilde{v} \right) e_{k+1} e_{k+1}^T \right)^{-1} \\ &= \left(\widetilde{B}_{k+1}^T \widetilde{B}_{k+1} + \tilde{\omega}^2 I \right)^{-1}, \end{aligned}$$

where

$$\widetilde{B}_{k+1} = \begin{pmatrix} L_{k+1} \\ \beta_{k+2} \sqrt{1 - \tilde{v}^T \widetilde{A}^{-1} \tilde{v}}\ e_{k+1}^T \end{pmatrix}.$$

The third equality follows from the fact that $B_{k+1}^T B_{k+1} = L_{k+1}^T L_{k+1} + \beta_{k+2}^2 e_{k+1} e_{k+1}^T$, and the expression under the radical is nonnegative because $\widetilde{A} \succeq \tilde{v}\tilde{v}^T$.

As for the claim that $\sigma_{\min}(\widetilde{B}_{k+1}) \geq \sigma_{\min}(A)$, we can conclude from the first part of the theorem that

$$\sigma_{\max}(\widetilde{B}_{k+1}^T \widetilde{B}_{k+1} + \tilde{\omega}^2 I)^{-1} \leq \sigma_{\max}(A^T A + \tilde{\omega}^2 I)^{-1}.$$

It follows that

$$\sigma_{\min}(\widetilde{B}_{k+1}^T \widetilde{B}_{k+1} + \tilde{\omega}^2 I) \geq \sigma_{\min}(A^T A + \tilde{\omega}^2 I),$$

and therefore

$$\sigma_{\min}(\widetilde{B}_{k+1}) \geq \sigma_{\min}(A). \qquad \square$$

The R factor from the QR factorization of $\widetilde{B}_{k+1}$ is identical to that from B_{k+1} (2.8) except for the bottom right entry. Combining this factorization with Theorem 4.1 and relation (2.5), we arrive at the following identity.

THEOREM 4.2. *For arbitrary $\tilde{\omega} \in \mathbb{R}$ and $x_k = V_k y_k$,*

$$\left\| (A^T A + \tilde{\omega}^2 I)^{-\frac{1}{2}} A^T r_k \right\| = \left\| (\underline{\widetilde{R}}_{k+1}^T \underline{\widetilde{R}}_{k+1} + \tilde{\omega}^2 I)^{-\frac{1}{2}} \underline{R}_{k+1}^T \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\|, \tag{4.4}$$

where $\underline{\widetilde{R}}_{k+1} = \begin{pmatrix} R_k & \theta_{k+1} e_k \\ 0 & \tilde{c}^{-1} \underline{\rho}_{k+1} \end{pmatrix}$ for some $\tilde{c}$ (dependent on $\tilde{\omega}$) such that $c_{k+1} \leq \tilde{c} \leq 1$, and $\sigma_{\min}(\underline{\widetilde{R}}_{k+1}) \geq \sigma_{\min}(A)$.

For flexible parameters $\tilde{c}$ and $\tilde{\omega}$, we therefore write the LSMB subproblem as

$$\min_{y_k} \left\| (\underline{\widetilde{R}}_{k+1}^T \underline{\widetilde{R}}_{k+1} + \tilde{\omega}^2 I)^{-\frac{1}{2}} \underline{R}_{k+1}^T \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\|, \quad x_k = V_k y_k. \tag{4.5}$$

4.2. Upper and lower bounds. Without a full bidiagonalization of A we cannot compute $\tilde{c}$ exactly, but we can put upper and lower bounds on $\tilde{c}$ that let us compute upper and lower bounds, respectively, on $\nu(x_k, \tau)$. The case $\tilde{c} = 1$ corresponds to the upper bound $\overline{\nu}_0(x_k, \tau)$ from (3.4), and the case $\tilde{c} = c_{k+1}$ corresponds to the lower bound $\underline{\nu}_1(x_k, \tau)$ from (3.5).

If we happen to have a lower bound σ_{est} on $\sigma_{\min}(A)$ (in particular, if we are solving a regularized problem), then we can use the fact that

$$\sigma_{\min} \begin{pmatrix} R_k & \theta_{k+1} \\ 0 & \tilde{c}^{-1} \underline{\rho}_{k+1} \end{pmatrix} \geq \sigma_{\text{est}} \tag{4.6}$$

to put a second upper bound on $\tilde{c}$ that holds for all $\tilde{\omega}$ and that may provide a much tighter upper bound on the backward error. The paper [3, sect. 4.1] uses a similar result to put an upper bound on the error $\|x - x_*\|$ of LSQR, and given a lower bound σ_{est} shows how to cheaply compute the largest $\tilde{c}_{\text{est}}$ satisfying (4.6).

Since we are mainly interested in putting an upper bound on the backward error, we propose the following choice for $\tilde{c}$: given a lower bound on $\sigma_{\min}(A)$, compute $\tilde{c}_{\text{est}}$ according to the procedure in [3] and fix $\tilde{c} = \min\{\tilde{c}_{\text{est}}, 1\}$. We also note that by fixing $\tilde{\omega} = 0$ in (4.4), any upper bound on $\tilde{c}$ allows us to compute an upper bound on $\|P_A r_k\|$. Although $\tilde{c} = 1$ gives the trivial bound $\|P_A r_k\| \leq \|r_k\|$, any nonzero value of σ_{est} will lead to a value of $\tilde{c}$ small enough to guarantee that our estimate of $\|P_A r_k\|$ converges to zero.

If we continued the bidiagonalization process another d steps, we could tighten the upper and lower bounds on $\tilde{c}$ enough to retroactively minimize $\overline{\nu}_d(x_k, \tau)$ (3.4) and $\underline{\nu}_d(x_k, \tau)$ (3.5). We consider this unnecessary because we could just as easily try to minimize the same objective function over the larger subspace $\text{Span}(V_{k+d})$ (i.e., $\overline{\nu}_0(x_{k+d}, \tau)$), which would presumably yield a smaller error. We therefore consider just the case $d = 0$.

4.3. QL factorization. By applying the QR factorization from (2.10), we can rewrite subproblem (4.5) as

$$\min_{y_k} \left\| (\widetilde{R}_{k+1}\widetilde{R}_{k+1}^T + \tilde{\omega}^2 I)^{-\frac{1}{2}} \underline{\overline{R}}_{k+1} \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\|, \tag{4.7}$$

where $\widetilde{R}_{k+1} = \left[\begin{smallmatrix} \overline{R}_k & \tilde{c}^{-1}\widehat{\theta}_{k+1}e_k \\ 0 & \tilde{c}^{-1}\widehat{\rho}_{k+1} \end{smallmatrix}\right]$. We aim to find the QL factorization

$$\widehat{Q}_{k+1} \begin{bmatrix} \widetilde{R}_{k+1}^T \\ \tilde{\omega} I \end{bmatrix} = \begin{bmatrix} \widehat{R}_{k+1}^T \\ 0 \end{bmatrix}, \tag{4.8}$$

since such a factorization would imply that $\widetilde{R}_{k+1}\widetilde{R}_{k+1}^T + \tilde{\omega}^2 I = \widehat{R}_{k+1}\widehat{R}_{k+1}^T$, leaving us with the minimization problem

$$\min_{y_k} \left\| \widehat{R}_{k+1}^{-1} \underline{\overline{R}}_{k+1} \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\|. \tag{4.9}$$

The matrix $\widehat{Q}_{k+1}$ can be expressed as a product of $2k+1$ Givens rotations. We illustrate the first two such rotations for the case $k = 2$ below, where $*$ represents a nonzero element and $\bullet$ represents an element that is affected by the current rotation:

$$\begin{bmatrix} * & & \\ * & * & \\ & \bullet & \bullet \\ * & & \\ & * & \\ & & \bullet \end{bmatrix} \mapsto \begin{bmatrix} * & & \\ * & * & \\ & \bullet & \bullet \\ * & & \\ & * & \\ & \bullet & 0 \end{bmatrix}, \qquad \begin{bmatrix} * & & \\ * & * & \\ & * & * \\ * & & \\ & \bullet & \\ & \bullet & 0 \end{bmatrix} \mapsto \begin{bmatrix} * & & \\ * & * & \\ & * & * \\ * & & \\ & \bullet & \\ & 0 & 0 \end{bmatrix}.$$

In this way the Givens rotations alternate between acting on rows j and $j+k+1$ and acting on rows $j+k$ and $j+k+1$. If we wanted to compute this QL factorization in full, we would need to start from scratch with every iteration, which would require $O(k)$ flops and memory. To solve problem (4.9), however, we only need to compute the first rotation in (4.8):

$$\begin{bmatrix} \tilde{c}^{-1}\widehat{\theta}_{k+1} & \tilde{c}^{-1}\widehat{\rho}_{k+1} \\ 0 & \tilde{\omega} \end{bmatrix} \mapsto \begin{bmatrix} \tilde{c}^{-2}\widehat{\theta}_{k+1}\widehat{\rho}_{k+1}/\delta & \delta \\ \tilde{c}^{-1}\widehat{\theta}_{k+1}\tilde{\omega}/\delta & 0 \end{bmatrix},$$

where we define $\delta = \sqrt{\tilde{c}^{-2}\widehat{\rho}_{k+1}^2 + \tilde{\omega}^2}$ for convenience. From this we can conclude that

$$\widehat{R}_{k+1} = \begin{pmatrix} \widehat{R} & (\tilde{c}^{-2}\widehat{\theta}_{k+1}\widehat{\rho}_{k+1}/\delta)e_k \\ 0 & \delta \end{pmatrix} \tag{4.10}$$

for some upper bidiagonal matrix $\widehat{R}$. Inserting this expression into (4.9) allows us to solve the LSMB subproblem:

$$\begin{aligned} &\min_{y_k} \left\| \widehat{R}_{k+1}^{-1} \underline{\overline{R}}_{k+1} \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\| \\ &= \min_{y_k} \left\| \begin{pmatrix} \widehat{R} & (\tilde{c}^{-2}\widehat{\theta}_{k+1}\widehat{\rho}_{k+1}/\delta)e_k \\ 0 & \delta \end{pmatrix}^{-1} \begin{pmatrix} \overline{R}_k & \widehat{\theta}_{k+1}e_k \\ 0 & \widehat{\rho}_{k+1} \end{pmatrix} \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\| \\ &= \min_{y_k} \left\| \begin{pmatrix} \widehat{R} & (\tilde{c}^{-2}\widehat{\theta}_{k+1}\widehat{\rho}_{k+1}/\delta)e_k \\ 0 & \delta \end{pmatrix}^{-1} \begin{pmatrix} \overline{R}_k(f_k - R_k y_k) + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}e_k \\ \overline{\phi}_{k+1}\widehat{\rho}_{k+1} \end{pmatrix} \right\| \\ &= \left|\overline{\phi}_{k+1}\right| \widehat{\rho}_{k+1}/\delta, \end{aligned}$$

where the last equality follows because the bottom entry cannot be altered but all others can be made zero. The minimum is attained when

$$\begin{pmatrix} \overline{R}_k(f_k - R_k y_k) + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}e_k \\ \overline{\phi}_{k+1}\widehat{\rho}_{k+1} \end{pmatrix} = \begin{pmatrix} \widehat{R} & (\tilde{c}^{-2}\widehat{\theta}_{k+1}\widehat{\rho}_{k+1}/\delta)e_k \\ 0 & \delta \end{pmatrix} \begin{pmatrix} 0 \\ \overline{\phi}_{k+1}\widehat{\rho}_{k+1}/\delta \end{pmatrix},$$

and comparing the top entries shows that the solution y_k satisfies

$$\overline{R}_k(f_k - R_k y_k) + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}e_k = \overline{\phi}_{k+1}\widehat{\theta}_{k+1}\left(\frac{\tilde{c}^{-2}\widehat{\rho}_{k+1}^2}{\delta^2}\right)e_k. \tag{4.11}$$

4.4. LSMB is a convex combination of LSQR and LSMR. Let y_k be the solution to (4.11), and let $x_k = V_k y_k$. By rearranging (4.11) and using identities (2.14) and (2.15) as well as the fact that $\delta = \sqrt{\tilde{c}^{-2}\widehat{\rho}_{k+1}^2 + \tilde{\omega}^2}$, we find that

$$\begin{aligned} x_k &= V_k R_k^{-1} f_k + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}\left(\frac{\tilde{\omega}^2}{\delta^2}\right) V_k R_k^{-1}\overline{R}_k^{-1} e_k \\ &= x_k^C + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}\left(\frac{\tilde{\omega}^2}{\delta^2}\right)\overline{w}_k \\ &= \left(\frac{\tilde{c}^{-2}\widehat{\rho}_{k+1}^2}{\delta^2}\right) x_k^C + \left(\frac{\tilde{\omega}^2}{\delta^2}\right) x_k^M \\ &= (1-\gamma)x_k^C + \gamma x_k^M, \end{aligned}$$

where

$$\gamma = \frac{\tilde{\omega}^2}{\delta^2}. \tag{4.12}$$

From the definition of δ it is clear that γ always lies in the interval $[0, 1]$. Thus for any fixed $\tilde{\omega}$ and $\tilde{c}$, the iterate x_k that results from solving (4.11) is a convex combination of the iterates produced by LSQR and LSMR.

4.5. Pseudocode for LSMB.

1. (Initialize)

$$\begin{gathered} \beta_1 u_1 = b, \quad \alpha_1 v_1 = A^T u_1, \quad \overline{\phi}_1 = \beta_1, \quad \underline{\rho}_1 = \alpha_1, \quad \rho_0 = 1, \quad \overline{\rho}_0 = 1, \\ \overline{c}_0 = 1, \quad \overline{s}_0 = 0, \quad h_1 = v_1, \quad \overline{h}_0 = 0, \quad x_0^C = 0. \end{gathered}$$

2. For $k = 1, 2, 3, \ldots$ repeat steps 3–8.
3. (Continue the bidiagonalization)

$$\beta_{k+1}u_{k+1} = Av_k - \alpha_k u_k, \qquad \alpha_{k+1}v_{k+1} = A^T u_{k+1} - \beta_{k+1}v_k.$$

4. (Construct and apply rotation P_k)

$$\begin{gathered} \rho_k = (\underline{\rho}_k + \beta_{k+1}^2)^{\frac{1}{2}}, \quad c_k = \underline{\rho}_k/\rho_k, \quad s_k = \beta_{k+1}/\rho_k, \quad \theta_{k+1} = s_k\alpha_{k+1}, \\ \underline{\rho}_{k+1} = c_k\alpha_{k+1}, \quad \phi_k = c_k\overline{\phi}_k, \quad \overline{\phi}_{k+1} = -s_k\overline{\phi}_k. \end{gathered}$$

5. (Construct and apply rotation $\overline{P}_k$)

$$\begin{gathered} \overline{\theta}_k = \overline{s}_{k-1}\rho_k, \quad \underline{\overline{\rho}}_k = \overline{c}_{k-1}\rho_k, \quad \overline{\rho}_k = (\underline{\overline{\rho}}_k^2 + \theta_{k+1}^2)^{\frac{1}{2}}, \\ \overline{c}_k = \underline{\overline{\rho}}_k/\overline{\rho}_k, \quad \overline{s}_k = \theta_{k+1}/\overline{\rho}_k. \end{gathered}$$

6. (Update $\overline{h}, h, x_k^C$)

$$\begin{aligned}\overline{h}_k &= h_k - (\overline{\theta}_k \rho_k / (\rho_{k-1}\overline{\rho}_{k-1}))\overline{h}_{k-1},\\ x_k^C &= x_{k-1}^C + (\phi_k/\rho_k)h_k,\\ h_{k+1} &= v_{k+1} - (\theta_{k+1}/\rho_k)h_k.\end{aligned}$$

7. (Fix $\tilde{c}, \tilde{\omega}$) Upper and lower bounds for $\tilde{c}$ were discussed in section 4.2, and methods for choosing $\tilde{\omega}$ are discussed in the following section.
8. (Compute the update for x_k)

$$\begin{aligned}\widehat{\theta}_{k+1} &= \overline{s}_k \underline{\rho}_{k+1},\\ \widehat{\rho}_{k+1} &= \overline{c}_k \underline{\rho}_{k+1},\\ \delta &= (\tilde{c}^{-2}\widehat{\rho}_{k+1}^2 + \tilde{\omega}^2)^{\frac{1}{2}},\\ x_k &= x_k^C + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}\left(\frac{\tilde{\omega}^2}{\delta^2}\right)\left(\frac{1}{\rho_k\overline{\rho}_k}\right)\overline{h}_k.\end{aligned}$$

Note that x_k does not have to be computed explicitly until a stopping rule is triggered and the algorithm terminates.

5. Error estimates. Here we derive estimates for $\|r_k\|$, $\|P_A r_k\|$, $\|A^T r_k\|$, and $\|x_k\|$ for use in stopping rules, as well as upper and lower bounds on the backward error. The estimates apply whenever x_k is a convex combination of x_k^C and x_k^M. Although we have up to this point described LSMB in terms of the parameters $\tilde{\omega}$ and $\tilde{c}$, for the purpose of these error estimates it is simpler to parametrize x_k in terms of γ (4.12).

5.1. Estimate of $\|A^T r_k\|$. Substituting (4.12) into (4.11) shows that y_k satisfies the relation

$$\overline{R}_k(f_k - R_k y_k) = -\gamma\overline{\phi}_{k+1}\widehat{\theta}_{k+1}e_k. \tag{5.1}$$

Assuming that V_{k+1} is exactly orthogonal, we can combine the above result with the relations from (2.12) to get that

$$\|A^T r_k\| = \left\| \begin{pmatrix} \overline{R}_k(f_k - R_k y_k) + \overline{\phi}_{k+1}\widehat{\theta}_{k+1}e_k \\ \widehat{\rho}_{k+1}\overline{\phi}_{k+1} \end{pmatrix} \right\| = \left\| \begin{bmatrix} (1-\gamma)\overline{\phi}_{k+1}\widehat{\theta}_{k+1} \\ \widehat{\rho}_{k+1}\overline{\phi}_{k+1} \end{bmatrix} \right\|. \tag{5.2}$$

5.2. Estimates of $\|r_k\|$ and $\|P_A r_k\|$. Assuming that U_{k+1} is exactly orthogonal, we combine the result in (5.1) with the relations from (2.11) to find that

$$\|r_k\| = \left\| \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\| = \left\| \begin{pmatrix} -\gamma\overline{\phi}_{k+1}\widehat{\theta}_{k+1}\overline{R}_k^{-1}e_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\|,$$

which can be computed efficiently by performing a third QR factorization,

$$\tilde{Q}_k\overline{R}_k^T = \underline{\tilde{R}}_k = \begin{pmatrix} \tilde{R}_{k-1} & \tilde{\theta}_k e_{k-1} \\ 0 & \underline{\tilde{\rho}}_k \end{pmatrix}. \tag{5.3}$$

5.2.1. Pseudocode for computing $\|r_k\|$.

1. (Initialize) $\underline{\tilde{\rho}}_0 = 1$.
2. For $k = 1, 2, 3, \ldots$ repeat steps 3–4.

3. (Construct and apply rotation $\tilde{P}_{k-1}$)

$$\begin{aligned} \tilde{\rho}_{k-1} &= (\underline{\tilde{\rho}}_{k-1}^2 + \overline{\theta}_k^2)^{\frac{1}{2}}, \\ \tilde{c}_{k-1} &= \underline{\tilde{\rho}}_{k-1}/\tilde{\rho}_{k-1}, \qquad \tilde{s}_{k-1} = \overline{\theta}_k/\tilde{\rho}_{k-1}, \\ \tilde{\theta}_k &= \tilde{s}_{k-1}\overline{\rho}_k, \qquad\qquad \underline{\tilde{\rho}}_k = \tilde{c}_{k-1}\overline{\rho}_k. \end{aligned}$$

4. (Form $\|r_k\|$)

$$\|r_k\| = \left((\gamma\overline{\phi}_{k+1}\widehat{\theta}_{k+1}/\underline{\tilde{\rho}}_k)^2 + \overline{\phi}_{k+1}^2\right)^{\frac{1}{2}}. \tag{5.4}$$

To estimate $\|P_A r_k\|$, we evaluate (4.5) with $\tilde{\omega} = 0$ to get

$$\|P_A r_k\| = \left\| (\widetilde{\underline{R}}_{k+1}^T \widetilde{\underline{R}}_{k+1})^{-\frac{1}{2}} \underline{R}_{k+1}^T \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\| = \left\| \begin{pmatrix} f_k - R_k y_k \\ \tilde{c} \cdot \overline{\phi}_{k+1} \end{pmatrix} \right\|,$$

which can be evaluated in the same manner. In particular, $\|P_A r_k^C\| = \tilde{c} \cdot |\overline{\phi}_{k+1}|$. When $\tilde{c} = 1$ this estimate will be equal to $\|r_k\|$, but with a nonzero lower bound $\sigma_{\min}(A)$ we can put an upper bound on $\|P_A r_k\|$ that converges to zero.

5.3. Estimates of $\|x_k\|$ and ω_k. In order to estimate $\|x_k\|$, we use the derivation from LSMR [4]:

$$\begin{aligned} \|A^T r_k\| &= \|V_{k+1} L_{k+1}^T (\beta_1 e_1 - B_k y_k)\| \\ &= \|\alpha_1\beta_1 e_1 - L_{k+1}^T B_k y_k\| \\ &= \left\| \alpha_1\beta_1 e_1 - \underline{R}_{k+1}^T \begin{bmatrix} R_k y_k \\ 0 \end{bmatrix} \right\| \\ &= \left\| \begin{bmatrix} z_k \\ \overline{\zeta}_{k+1} \end{bmatrix} - \begin{bmatrix} \overline{R}_k R_k y_k \\ 0 \end{bmatrix} \right\|, \end{aligned}$$

where, like f_k (2.8), the leading elements of z_k do not change between iterations. We can therefore express any iterate x_k for LSMB in the form

$$\begin{aligned} x_k &= V_k R_k^{-1} f_k + \gamma\overline{\phi}_{k+1}\widehat{\theta}_{k+1} V_k R_k^{-1} \overline{R}_k^{-1} e_k \\ &= x_k^M - (1-\gamma) V_k R_k^{-1} \overline{R}_k^{-1} e_k \\ &= V_k R_k^{-1} \overline{R}_k^{-1} z_k + (\gamma - 1) V_k R_k^{-1} \overline{R}_k^{-1} e_k \\ &= V_k R_k^{-1} \overline{R}_k^{-1} \left(z_k + (\gamma - 1)\overline{\phi}_{k+1}\widehat{\theta}_{k+1} e_k \right), \end{aligned}$$

and as in LSMR [4, sect. 3.3], we can use the QR factorization $\tilde{Q}_k \overline{R}_k^T = \tilde{R}_k$ (5.3) and make a fourth QR factorization $\hat{Q}_k(\underline{\overline{R}}_k \tilde{Q}_k^T) = \hat{R}_k$ to get the relation

$$\begin{aligned} x_k &= V_k R_k^{-1} \overline{R}_k^{-1} \left(z_k + (\gamma - 1)\overline{\phi}_{k+1}\widehat{\theta}_{k+1} e_k \right) \\ &= V_k \overline{Q}_k^T \underline{\overline{R}}_k^{-T} \tilde{Q}_k^T \tilde{R}_k^{-T} \left(\tilde{R}_k^T \tilde{z}_k + (\gamma - 1)\overline{\phi}_{k+1}\widehat{\theta}_{k+1} e_k \right) \\ &= V_k \overline{Q}_k^T \hat{Q}_k^T \hat{R}_k^{-T} \left(\hat{R}_k^T \hat{z}_k + (\gamma - 1)(\overline{\phi}_{k+1}\widehat{\theta}_{k+1}/\tilde{\rho}_k) e_k \right) \\ &= V_k \overline{Q}_k^T \hat{Q}_k^T \left(\hat{z}_k + (\gamma - 1)(\overline{\phi}_{k+1}\widehat{\theta}_{k+1}/\tilde{\rho}_k\hat{\rho}_k) e_k \right), \end{aligned}$$

where $\tilde{z}_k$ and $\hat{z}_k$ satisfy the relations $\tilde{R}_k^T \tilde{z}_k = z_k$ and $\hat{R}_k^T \hat{z}_k = \tilde{z}_k$. Only the final entry of $\tilde{R}_k$ and the bottom 2×2 submatrix of $\hat{R}_k$ change with each iteration, and so by assuming orthogonality of V_k we arrive at the estimate

$$\|x_k\| = \left\| \left[\hat{z}_{k-2}, \quad \hat{\underline{\zeta}}_{k-1}, \quad \hat{\underline{\zeta}}_k + (\gamma - 1)(\overline{\phi}_{k+1} \widehat{\theta}_{k+1} / (\tilde{\rho}_k \hat{\rho}_k)) \right] \right\|, \tag{5.5}$$

where the leading entries of $\hat{z}_{k-2}$ do not change and so $\|\hat{z}_{k-2}\|$ is monotonically increasing. It does not necessarily follow that $\|x_k\|$ is monotonically increasing, but the following theorem shows that $\|x_k^M\|$ is also a monotonically increasing lower bound for $\|x_k\|$.

THEOREM 5.1. *$\|x_k\|$ is increasing along the line segment from x_k^M to x_k^C.*

Proof. It is shown in [14, Thm. 7:2b] that x_k^M is a convex combination of x_{k-1}^M and x_k^C and in [5, Thm. 2.3] that $\|x_k^M\|$ is monotonically increasing. The claim follows from the fact that norms are convex. ∎

The estimates of $\|r_k\|$ (5.4) and $\|x_k\|$ (5.5) then allow us to cheaply estimate $\omega_k = \tau \|r_k\| / \sqrt{1 + \tau^2 \|x_k\|^2}$. Since $\|r_k\|$ increases along the line segment from x_k^C to x_k^M and $\|x_k\|$ decreases along the same segment, we conclude the following.

COROLLARY 5.2. *ω_k is increasing along the line segment from x_k^C to x_k^M.*

5.4. Estimate of $\boldsymbol{\nu(x_k, \tau)}$. By using the full QL factorization outlined in (4.8) and the expression in (4.9) we get the estimate

$$\tilde{\nu}(x_k, \tau) = \frac{\omega_k}{\|r_k\|} \left\| (\underline{\widetilde{R}}_{k+1}^T \underline{\widetilde{R}}_{k+1} + \omega^2 I)^{-\frac{1}{2}} \underline{R}_{k+1}^T \begin{pmatrix} f_k - R_k y_k \\ \overline{\phi}_{k+1} \end{pmatrix} \right\| \tag{5.6}$$

$$= \frac{\omega_k}{\|r_k\|} \left\| \widehat{R}_{k+1}^{-1} \begin{pmatrix} (1 - \gamma) \overline{\phi}_{k+1} \widehat{\theta}_{k+1} e_k \\ \overline{\phi}_{k+1} \widehat{\rho}_{k+1} \end{pmatrix} \right\|, \tag{5.7}$$

where γ depends on $\tilde{\omega}$ as in (4.12). This estimate can be computed in $O(k)$ time and memory, and requires storing all previous values of $\overline{\theta}_k$ and $\overline{\rho}_k$. As mentioned in section 4.2, the estimate will be a lower bound on $\nu(x_k, \tau)$ if $\tilde{c} = c_{k+1}$ and an upper bound if $\tilde{c} = \min\{\tilde{c}_{\text{est}}, 1\}$. In the case $\tilde{c} = c_{k+1}$, we can get an even cheaper lower bound by computing only the final entry of the vector in (5.7). This $O(1)$ lower bound requires no extra storage and is still often tight in practice. For the remainder of the paper, we denote this upper bound and the O(1) lower bound by $\overline{\nu}(x_k, \tau)$ and $\underline{\nu}(x_k, \tau)$, respectively.

5.5. Choosing a value for $\boldsymbol{\tilde{\omega}}$. Assuming that we have chosen a value of $\tilde{c}$ that guarantees an upper bound on the backward error, we would ideally like to choose $\tilde{\omega}$ (or, equivalently, γ) so that the estimate (5.7) is minimized. This is still difficult to do because $\widehat{R}_{k+1}$ changes with $\tilde{\omega}$, but there are a few reasonable choices we could make. Since Corollary 5.2 implies that $\omega_k^C \le \omega_k \le \omega_k^M$ for all choices of $\tilde{\omega}$, both ω_k^C and ω_k^M may be reasonable options.

We could also choose $\tilde{\omega}$ such that $\omega_k = \tilde{\omega}$, in which case we could compute the error estimate (5.7) in $O(1)$ time. Rearranging (4.12) shows that

$$\tilde{\omega}^2 = \tilde{c}^{-2} \widehat{\rho}_{k+1}^2 \frac{\gamma}{1 - \gamma}, \tag{5.8}$$

so substituting this value for $\tilde{\omega}$, using the definition $\omega_k = \tau \|r_k\| / \sqrt{1 + \tau^2 \|x_k\|^2}$, squaring both sides, and rearranging terms leads to the problem

$$\tilde{c}^{-2} \widehat{\rho}_{k+1}^2 (1 + \tau^2 \|x_k\|^2) \gamma = \tau^2 \|r_k\|^2 (1 - \gamma). \tag{5.9}$$

Using (5.5) and (5.4) to express $\|x_k\|^2$ and $\|r_k\|^2$ in terms of γ yields a cubic equation. This equation always has at least one real solution with $\gamma \in [0,1]$, and in our experiments we have not found a scenario for which there was more than one real solution.

In practice, our choice does not much matter. Given our suggested bounds on $\tilde{c}$, it turns out that no choice of $\tilde{\omega}$ will yield a solution that substantially outperforms both LSQR and LSMR. To see this, we combine (5.7) with the expression for $\widehat{R}_{k+1}$ from (4.10) to find that for any $\tilde{\omega}$ and resulting iterate x_k,

$$\begin{aligned}
\tilde{\nu}(x_k,\tau) &= \frac{\omega_k}{\|r_k\|}\left\|\widehat{R}_{k+1}^{-1}\begin{pmatrix}(1-\gamma)\overline{\phi}_{k+1}\widehat{\theta}_{k+1}e_k\\ \overline{\phi}_{k+1}\widehat{\rho}_{k+1}\end{pmatrix}\right\| \\
&\geq \frac{\omega_k}{\|r_k\|}\left(\frac{|\overline{\phi}_{k+1}|\widehat{\rho}_{k+1}}{\sqrt{\tilde{c}^{-2}\widehat{\rho}_{k+1}^2+\omega_k^2}}\right) \\
&= \frac{|\overline{\phi}_{k+1}|}{\|r_k\|}\left(\frac{\widehat{\rho}_{k+1}\tilde{c}\cdot\omega_k}{\sqrt{\widehat{\rho}_{k+1}^2+\tilde{c}^2\cdot\omega_k^2}}\right) \\
&\geq \frac{|\overline{\phi}_{k+1}|}{\sqrt{2}\|r_k\|}\min\{\widehat{\rho}_{k+1},\tilde{c}\cdot\omega_k\} \\
&= \frac{1}{\sqrt{2}}\min\left\{\frac{|\overline{\phi}_{k+1}|\widehat{\rho}_{k+1}}{\|r_k\|},\tilde{c}\frac{\tau|\overline{\phi}_{k+1}|}{\sqrt{1+\tau^2\|x_k\|^2}}\right\} \\
&\geq \frac{1}{\sqrt{2}}\min\left\{\frac{\|A^Tr_k^M\|}{\|r_k^M\|},\frac{\tilde{c}\cdot|\overline{\phi}_{k+1}|}{\|r_k^C\|}\omega_k^C\right\}.
\end{aligned}$$

The first term is an upper bound on the error for LSMR. If we have chosen $\tilde{c}$ so that $\tilde{c}=\min\{c_{\text{est}},1\}$, then the second term is an upper bound on $\omega_k^C\|P_Ar_k^C\|/\|r_k^C\|$, which is an upper bound on the error for LSQR. Therefore, our backward error estimates for LSMB will at best be a factor of $\sqrt{2}$ smaller than the better of the estimates from LSQR and LSMR. Furthermore, if we are solving an inconsistent least-squares problem and have no lower bound on $\sigma_{\min}(A)$ (i.e., we use $\tilde{c}=1$), the estimate $\omega_k^C(\tilde{c}\cdot|\overline{\phi}_{k+1}|)/\|r_k^C\|$ will not converge to zero. In this situation, the estimate $\overline{\nu}(x_k,\tau)$ will become arbitrarily close to $\|A^Tr_k^M\|/\|r_k^M\|$ as the error decreases to zero.

5.6. Stopping criteria and convergence. Given dimensionless quantities τ,ϵ, and κ, and $\tilde{c}=\min\{\tilde{c}_{\text{est}},1\}$, we propose the following stopping rules:

1. Stop if $\frac{\overline{\nu}(x_k,\tau)}{\|A\|}<\epsilon$.
2. Stop if $\text{cond}(A)>\kappa$.

Since $\|r_k^C\|$, $\|P_Ar_k^C\|$, and $\|A^Tr_k^M\|$ are monotonically decreasing, we expect the estimate for the first stopping rule to decrease monotonically or at least nearly so. If one wants to solve the problem so that A and b have relative backward errors at most α and β, respectively, then $\epsilon=\alpha$ and $\tau=\frac{\alpha\|A\|}{\beta\|b\|}$ are reasonable choices. If the value of b is known exactly, then $\tau=\infty$ is an acceptable option, in which case $\omega=\|r\|/\|x\|$.

As for convergence, both LSQR and LSMR have been proven to converge to the minimum-norm solution to the least-squares problem. Since LSMB produces only convex combinations of the iterates from LSQR and LSMR, it must also converge to the minimum-norm solution.

5.7. Regularized least squares. LSMB can be extended to the regularized least-squares problem

$$\min_x \left\| \begin{pmatrix} A \\ \lambda I \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|$$

by defining $\overline{A} = \left(\begin{smallmatrix} A \\ \lambda I \end{smallmatrix}\right)$ and $\overline{b} = \left(\begin{smallmatrix} b \\ 0 \end{smallmatrix}\right)$ and attempting to minimize

$$\overline{\mu}(x,\tau) = \min_{E,f} \|E, \tau f\| \ : \ (\overline{A}+E)^T \left[(\overline{A}+E)x - (\overline{b}+f)\right] = 0.$$

Note that x solves the perturbed problem $\min_x \|(A+E_1)x-(b+f)\|^2+\|(\lambda I+E_2)x\|^2$, where $E = \left(\begin{smallmatrix} E_1 \\ E_2 \end{smallmatrix}\right)$. Although it may be more natural to minimize a structured backward error with the constraint $E_2 = \Delta\lambda I$, we allow E_2 to be arbitrary because doing so ensures that the iterates produced by LSMB will still be convex combinations of the iterates from LSQR and LSMR, thus preserving the connection between the two methods. If we use the QR factorization $Q_{2k+1}\left(\begin{smallmatrix} B_k \\ \lambda I \end{smallmatrix}\right) = \left(\begin{smallmatrix} R_k \\ 0 \end{smallmatrix}\right)$ defined in [4, sect. 5], then we arrive at a subproblem with structure nearly identical to (4.5) and proceed as in the unregularized case. The norms $\|r_k\|$ and $\|x_k\|$ will change, but can still be estimated cheaply. We omit the details, but note that regularization allows us to obtain tighter backward error estimates for LSQR for two reasons: First, the inclusion of any regularization term will guarantee that $|\overline{\phi}_{k+1}|$ (and thus our estimate of $\|P_A r_k^C\|$) converges to zero. Second, since $\sigma_{\min}(\overline{A}) \geq \lambda$, we can put a nontrivial upper bound on $\tilde{c}$. Thus regularization (or, in general, having a good estimate of $\sigma_{\min}(A)$) may potentially make LSQR competitive with LSMR on inconsistent systems.

6. Numerical experiments. We tested our algorithm on overdetermined problems with matrices from the University of Florida Sparse Matrix Collection [2].

6.1. Problems with $\sigma_{\min}(A)$ unknown. For overdetermined problems we followed the procedure of Fong and Saunders [4]; problems were downloaded from the `LPnetlib` group in MATLAB, and a sparse matrix A was generated by the command `A = (Problem.A)'`. To simulate a right preconditioner we scaled the columns of A to have unit 2-norm.

For a given $m \times n$ matrix A, the right-hand vector was generated according to the following procedure:

1. `x = (1:n)';`
2. `b = A*x + delta*randn(m,1)*norm(x)*sqrt(m/(m-n));`

with $\delta = 10^{-4}$ to generate high-residual problems and $\delta = 10^{-10}$ for low-residual problems. We ran all problems to a maximum of 20K iterations, or until $\overline{\nu}(x_k,\infty)/\|A\|_F$ was smaller than machine precision. We tested problems with regularization parameters $\lambda = 0$ and $\lambda = 10^{-4}$, and in both cases used $\sigma_{\text{est}} = \lambda$ as a lower bound for $\sigma_{\min}(\overline{A})$ (note that the bound $\sigma_{\text{est}} = 0$ fixes the trivial upper bound $\tilde{c} = 1$). In general we observed the following behavior:

1. The upper bound $\overline{\nu}(x_k,\tau)$ appeared to be monotonically decreasing in all cases (Figure 6.1). As expected, based on our conclusions from section 5.5, it was always approximately equal to the smaller of $\|A^T r_k^M\|/\|r_k^M\|$ and $\omega_k^C(\tilde{c}\cdot|\overline{\phi}_{k+1}|)/\|r_k^C\|$ (our upper bound on $\omega_k^C\|P_A r_k^C\|/\|r_k^C\|$).
2. Since all test problems were inconsistent, $\|A^T r_k^M\|$ always eventually converged to zero, while $\|r_k^C\|$ did not. Typically, $\|r_k^C\|$ would remain smaller than $\|A^T r_k^M\|$ until the residual became very close to the minimum residual $\|r_*\|$ (Figures 6.2 and 6.4).

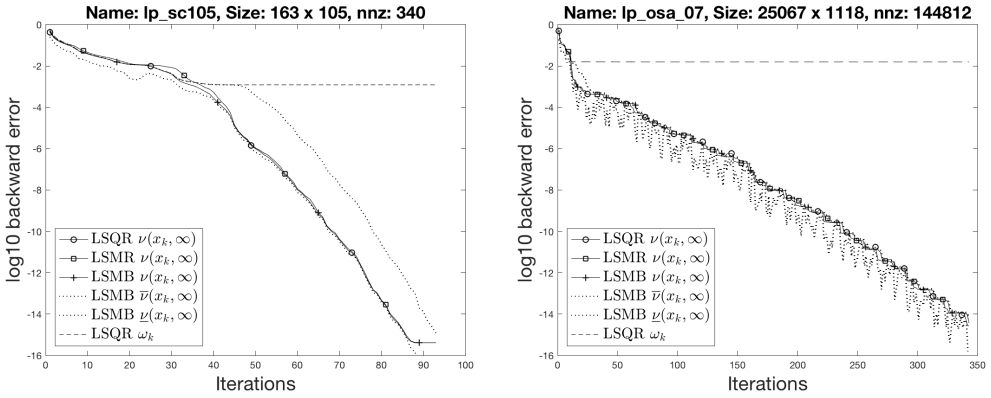

FIG. 6.1. *The error estimate $\overline{\nu}(x_k,\infty)$ seems to be monotonically decreasing. The backward errors from LSQR, LSMR, and LSMB are typically close to each other. While the lower bound $\underline{\nu}(x_k,\infty)$ is often accurate, it can sometimes oscillate.*

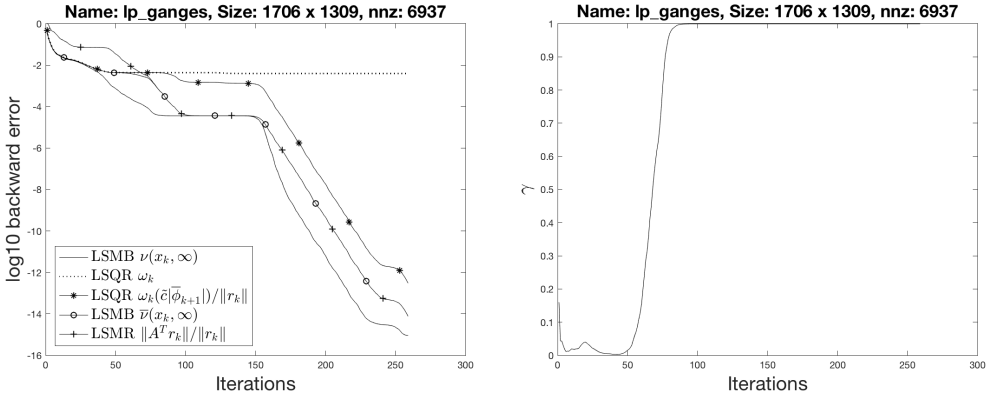

FIG. 6.2. *A high-residual problem with $\lambda = 10^{-4}$. The error estimate $\overline{\nu}(x_k,\infty)$ from LSMB is nearly equal to the smaller of the error estimates $\|A^T r_k^M\|/\|r_k^M\|$ from LSMR and $\omega_k^C(\tilde{c}|\overline{\phi}_{k+1}|)/\|r_k^C\|$ from LSQR. The value of γ is typically very close to 0 or 1, depending on whether LSQR or LSMR provides the better estimate.*

3. For regularized problems, the estimate $\omega_k^C(\tilde{c}\cdot|\overline{\phi}_{k+1}|)/\|r_k^C\|$ reliably converged to zero (Figures 6.2 and 6.3). For high-residual problems this estimate tended to be larger than both $\omega_k^C\|P_A r_k^C\|/\|r_k^C\|$ and $\|A^T r_k^M\|/\|r_k^M\|$. Low-residual problems showed a much more interesting pattern: in most cases, the estimate would closely track $\omega_k^C\|P_A r_k^C\|/\|r_k^C\|$ until the error was small (around 10^{-8} or 10^{-9}), plateau, then closely track $\|A^T r_k^M\|/\|r_k^M\|$ thereafter (Figure 6.3).
4. When chosen to satisfy the equality $\tilde{\omega} = \omega_k$, the value of γ was typically very close to 0 when the LSQR estimates were smaller, and very close to 1 when the LSMR estimates were smaller (Figure 6.2). The iterates from LSMB therefore tended to be close to those from LSQR in earlier iterations and close to those from LSMR later on. A notable exception was in the low-residual regularized cases (Figure 6.3), where γ appeared to converge to a value near 0.5! It appears that in low-residual cases where λ is relatively small compared to the singular values of A, both $\tilde{c}^{-1}\widehat{\rho}$ and ω converge to values near λ, and as a result (5.9) will have a solution with γ near 0.5 when τ is large. We leave

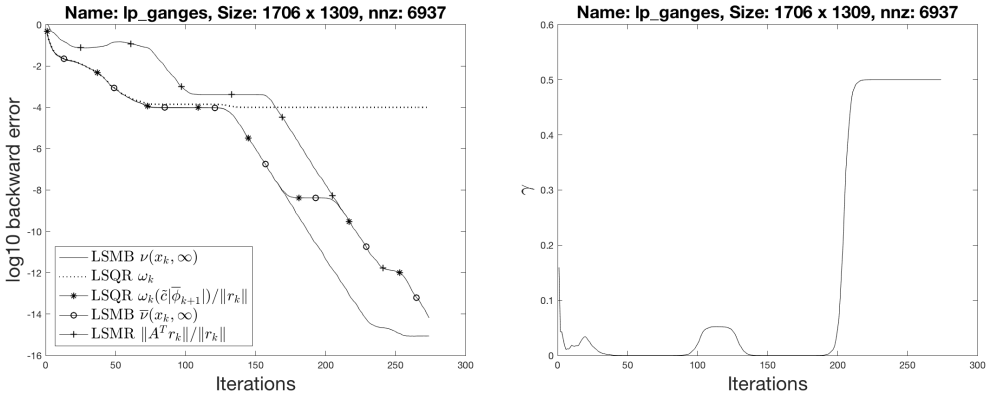

FIG. 6.3. *A low-residual problem with* $\lambda = 10^{-4}$. *LSQR outperforms LSMR for longer, but the estimates from the two algorithms eventually converge. Correspondingly,* γ *seems to converge to a value near* 0.5 *in these cases.*

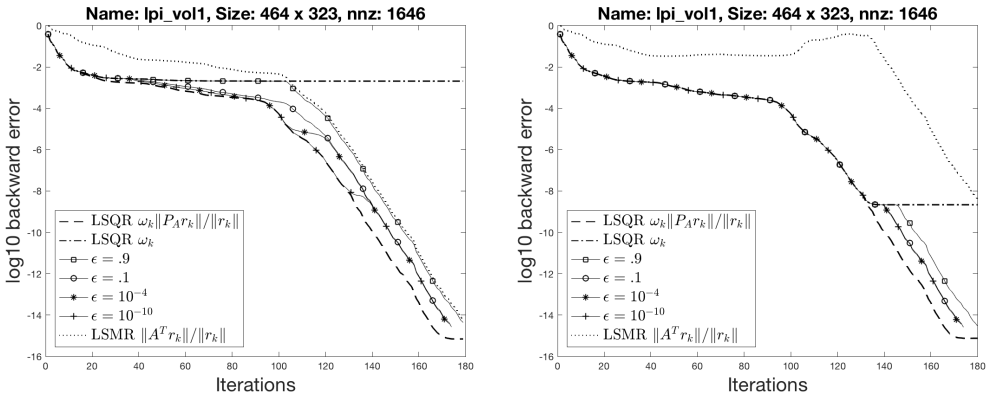

FIG. 6.4. *Estimating* $\|P_Ar_k^C\|$ *for high-residual and low-residual problems. On low-residual problems in particular, even conservative lower bounds on* $\sigma_{\min}(A)$ *may allow LSQR to terminate before LSMR.*

the details for a future investigation.

5. The backward errors from LSQR, LSMR, and LSMB were typically very close, with the error from LSMB never larger than the error from LSMR (Figure 6.1). The error from LSQR could sometimes be smaller, typically in cases where the true value of $\omega_k^C\|P_Ar_k^C\|/\|r_k^C\|$ was significantly smaller than both $\omega_k^C(\tilde{c}\cdot|\overline{\phi}_{k+1}|)/\|r_k^C\|$ and $\|A^Tr_k^M\|/\|r_k^M\|$.
6. The $O(1)$ estimate for the lower bound on $\nu(x_k,\infty)$ (5.4) was often quite accurate (Figure 6.1). In some cases it was very close to the upper bound, allowing us to estimate the true backward error accurately and at only $O(1)$ cost. In other cases, however, it could oscillate wildly or underestimate the true backward error by several orders of magnitude. The lower bound therefore does not appear to be a reliable estimate of the backward error for considering stopping criteria.

6.2. Problems where $\sigma_{\min}(A)$ is known. For a handful of the smaller test matrices, we computed $\sigma_{\min}(A)$ in order to estimate $\omega_k^C\|P_Ar_k^C\|/\|r_k^C\|$ without having

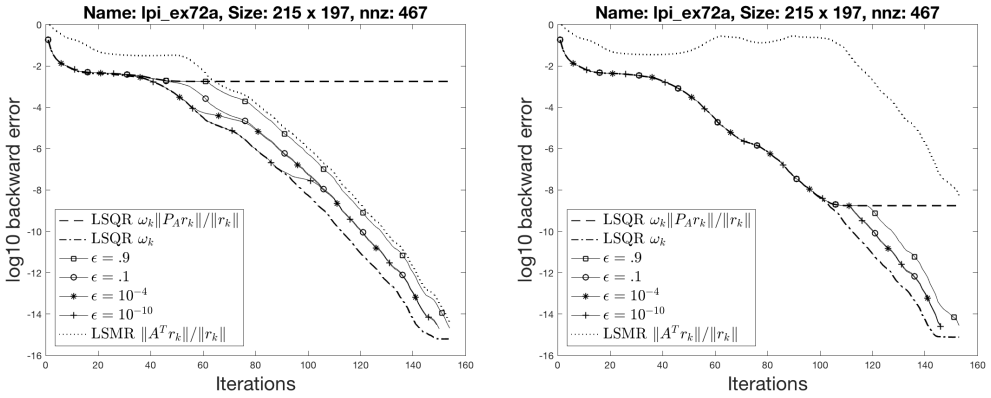

FIG. 6.5. *Estimating $\|P_A r_k^C\|$ for a rank-deficient problem. Even if the smallest singular values are nonzero, setting σ_{est} near the smallest "nonnegligible" value appears to produce a useful upper bound on the backward error.*

to add a regularization term. We generated right-hand vectors b in the same manner as before using $\delta = 10^{-4}$ and $\delta = 10^{-10}$, and we set $\sigma_{\text{est}} = (1-\epsilon)\sigma_{\min}(A)$, with values of ϵ including $\{.9, .1, 10^{-4}, 10^{-10}\}$. In general we observed the following behavior:

1. For low-residual problems, even conservative values of σ_{est} allowed the LSQR estimate to converge before the estimate $\|A^T r_k^M\|/\|r_k^M\|$ from LSMR. For high-residual problems results were more mixed. In some cases all of the LSQR estimates were smaller than the LSMR estimate (Figure 6.4), but in some cases the reverse held.
2. In general, tighter estimates of $\sigma_{\min}(A)$ yielded tighter estimates of $\|P_A r_k^C\|$ (Figures 6.4 and 6.5). The difference was more pronounced for low-residual problems, where the estimates were very accurate early on and lost accuracy as the number of iterations increased. Estimates of $\|P_A r_k^C\|$ obtained using the tightest bounds on $\sigma_{\min}(A)$ remained accurate for longer.
3. Several rank-deficient matrices were tested as well. These matrices were well-conditioned except for having a small number of singular values on the order of machine precision. Using values of σ_{est} near $\sigma_{\min}(A)$ offered no benefit, but setting σ_{est} to be near the smallest "nonnegligible" singular value of A gave some interesting results (Figure 6.5): estimates no longer gave upper bounds on $\|P_A r_k^C\|$, but they continued to provide accurate upper bounds on $\nu(x_k, \infty)$! A likely explanation is that the estimates are upper bounds on $\|P_{A'} r_k\|$, where A' is a perturbed matrix close to A.

7. Conclusion. We have shown how to implement LSQR simultaneously with LSMR at minimal extra cost, and have shown how to estimate the projection $\|P_A r_k\|$ cheaply when solving regularized least-squares problems or when a lower bound on $\sigma_{\min}(A)$ is available. This improved estimate may allow LSQR to terminate sooner than LSMR on inconsistent problems, particularly when the residual is small.

In the case of LSQR, the estimate $\overline{\nu}_0(x_k\,\tau)$ (3.4) was known to be an upper bound on $\nu(x_k, \tau)$ and smaller than both $\|r_k^C\|$ and $\|A^T r_k^C\|$, and could be computed in $O(k)$ time and memory. We showed that there exists a point on the line segment between the iterates x_k^C (from LSQR) and x_k^M (from LSMR) for which this estimate could be computed in $O(1)$ time, and we showed how to tighten the error estimate in regularized cases or when a lower bound on $\sigma_{\min}(A)$ is available.

In all of our test cases the backward error estimate $\overline{\nu}(x_k, \tau)$ for LSMB appeared to be monotonically decreasing, suggesting that measuring the backward error for past iterates of LSQR should no longer be necessary. As it turns out, however, $\overline{\nu}(x_k, \tau)$ will never be smaller than the corresponding estimates of LSQR and LSMR by more than a factor of $\sqrt{2}$. Although LSMB may terminate marginally sooner in some cases, it may be simpler just to run LSQR and LSMR simultaneously.

A MATLAB implementation of LSMB is available at https://math.berkeley.edu/~ehallman/lsmb/, as is a version that runs LSQR and LSMR simultaneously without producing the LSMB iterates.

Acknowledgments. We are grateful to the editors and the two anonymous referees for their many comments and suggestions which have greatly improved the presentation of this paper, and for pointing out the paper [10] to us.

REFERENCES

[1] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996, https://doi.org/10.1137/1.9781611971484.

[2] T. A. Davis and Y. Hu, *The University of Florida Sparse Matrix Collection*, ACM Trans. Math. Softw., 38 (2011), 1.

[3] R. Estrin, D. Orban, and M. A. Saunders, *LSLQ: An iterative method for linear least squares with an error minimization property*, SIAM J. Matrix Anal. Appl., to appear.

[4] D. C.-L. Fong and M. Saunders, *LSMR: An iterative algorithm for sparse least-squares problems*, SIAM J. Sci. Comput., 33 (2011), pp. 2950–2971, https://doi.org/10.1137/10079687X.

[5] D. C.-L. Fong and M. Saunders, *CG versus MINRES: An empirical comparison*, Sultan Qaboos Univ. J. Sci., 17 (2012), pp. 44–62.

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

[7] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.

[8] S. Gratton, P. Jiránek, and D. Titley-Peloquin, *On the accuracy of the Karlson–Waldén estimate of the backward error for linear least squares problems*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 822–836, https://doi.org/10.1137/110825467.

[9] S. Gratton, P. Jiránek, and D. Titley-Peloquin, *Simple backward error bounds for linear least-squares problems*, Linear Algebra Appl., 439 (2013), pp. 78–89.

[10] S. Gratton, P. Jiránek, and X. Vasseur, *Minimizing the Backward Error in the Energy Norm with Conjugate Gradients*, Tech. Report TR/PA/10/45, CERFACS, Toulouse, France, 2010.

[11] J. F. Grcar, *Optimal Sensitivity Analysis of Linear Least Squares*, Tech. Report LBNL-52434, Lawrence Berkeley National Laboratory, Berkeley, CA, 2003.

[12] J. F. Grcar, M. Saunders, and Z. Su, *Estimates of Optimal Backward Perturbations for Linear Least Squares Problems*, Tech. Report SOL-2007-1, Department of Management Science and Engineering, Stanford University, Stanford, CA, 2007.

[13] M. Gu, *Backward perturbation bounds for linear least squares problems*, SIAM J. Matrix Anal. Appl., 20 (1998), pp. 363–372, https://doi.org/10.1137/S0895479895296446.

[14] M. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.

[15] P. Jiránek and D. Titley-Peloquin, *Estimating the backward error in LSQR*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2055–2074, https://doi.org/10.1137/090770655.

[16] R. Karlson and B. Waldén, *Estimation of backward perturbation bounds for the linear least squares problem*, BIT, 37 (1997), pp. 862–869.

[17] E. M. Kasenally, *GMBACK: A generalised minimum backward error algorithm for nonsymmetric linear systems*, SIAM J. Sci. Comput., 16 (1995), pp. 698–719, https://doi.org/10.1137/0916042.

[18] E. M. Kasenally and V. Simoncini, *Analysis of a minimum perturbation algorithm for nonsymmetric linear systems*, SIAM J. Numer. Anal., 34 (1997), pp. 48–66, https://doi.org/10.1137/S0036142994266844.

[19] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*,

SIAM J. Numer. Anal., 12 (1975), pp. 617–629, https://doi.org/10.1137/0712047.
[20] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[21] J. L. Rigal and J. Gaches, *On the compatibility of a given solution with the data of a linear system*, J. ACM, 14 (1967), pp. 543–548.
[22] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869, https://doi.org/10.1137/0907058.
[23] G. Stewart, *An inverse perturbation theorem for the linear least squares problem*, SIGNUM Newsl., 10 (1975), pp. 39–40.
[24] G. Stewart, *Research, development, and LINPACK*, in Mathematical Software III, Publ. Math. Res. Center 39, Academic Press, New York, 1977, pp. 1–14.
[25] Z. Su, *Computational Methods for Least Squares Problems and Clinical Trials*, Ph.D. thesis, Stanford University, Stanford, CA, 2005.
[26] B. Waldén, R. Karlson, and J.-G. Sun, *Optimal backward perturbation bounds for the linear least squares problem*, Numer. Linear Algebra Appl., 2 (1995), pp. 271–286.