

Notes 3: **Iterative Methods for Square and Rectangular Systems**

1 Introduction

Some unsymmetric or rectangular systems that arise in optimization are:

- Simplex method: $Bx = b$ and $B^T y = c$, where B is a square basis matrix.
- First-order multiplier estimates: $\min_y \|J^T y - g\|$, where J is the Jacobian for the current set of active constraints, and g is the current objective gradient.

Large linear programs usually require the solution of so many square systems that we could not consider using iterative methods to solve $Bx = b$ and $B^T y = c$ within the Simplex method. However, special circumstances may arise requiring nonstandard approaches.

In general we need to consider iterative methods for the following problems:

- Square systems: $Ax = b$.
- Under-determined compatible systems: $\min \|x\|^2$ subject to $Ax = b$.
- Over-determined systems (least squares): $\min \|Ax - b\|^2$.
- Regularized systems: $\min \|Ax - b\|^2 + \|\delta x\|^2 \equiv \min \left\| \begin{pmatrix} A \\ \delta I \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|^2$,
where δ is a scalar and A may have any shape or rank.

These four problems are equivalent to the symmetric system $\begin{pmatrix} \gamma I & A \\ A^T & \delta I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$ with the parameters $(\gamma, \delta) = (0, 0), (0, -1), (1, 0), (\delta, -\delta)$ respectively. They are also equivalent to the positive definite systems $(A^T A + \gamma^2 I)x = A^T b$ or $(AA^T + \gamma^2 I)y = b$ with $\gamma = 0$ or $-\delta$ respectively, and $x = A^T y$. We may therefore expect the symmetric iterative solvers to apply. However, more effective numerical methods are obtained by working with A directly.

2 Bidiagonalization Methods for Unsymmetric Systems

A square or rectangular matrix A can be reduced to upper bidiagonal form by multiplying alternately on the left and right by certain orthogonal matrices: $U^T A V = B$. This is the starting point for dense singular value decompositions (SVDs) [6]. When A is sparse or a “black box” operator for forming matrix-vector products Av , $A^T u$, the bidiagonalization can be performed iteratively. (Actually, we reduce $\begin{pmatrix} b & A \end{pmatrix}$ to upper bidiagonal form, meaning A is reduced to *lower* bidiagonal form.)

2.1 The Golub-Kahan Process

$\text{Bidiag}(A, b) \rightarrow (B_k, U_{k+1}, V_k)$ or (L_k, U_k, V_k) denotes the following process. Given an $m \times n$ matrix A and a starting vector b , the Golub-Kahan process [5] generates vectors u_k, v_k and positive scalars α_k, β_k ($k = 1, 2, \dots$) according to these steps:

1. Set $\beta_1 u_1 = b$ and $\alpha_1 v_1 = A^T u_1$. (Exit if $\beta_1 = 0$ or $\alpha_1 = 0$.)
2. For $k = 1, 2, \dots$, set

$$\beta_{k+1} u_{k+1} = A v_k - \alpha_k u_k,$$

$$\alpha_{k+1} v_{k+1} = A^T u_{k+1} - \beta_{k+1} v_k.$$

After k steps, the situation is summarized by either of

$$A V_k = U_{k+1} B_k = U_k L_k + \beta_{k+1} u_{k+1} e_k^T, \quad (1)$$

$$A^T U_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T = V_{k+1} L_{k+1}^T, \quad (2)$$

where $U_k = (u_1 \ u_2 \ \dots \ u_k)$, $V_k = (v_1 \ v_2 \ \dots \ v_k)$, L_k is lower bidiagonal, and B_k is also bidiagonal with one extra row:

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

With exact arithmetic the columns of U_k and V_k would be orthonormal for each k until $\alpha_{k+1} = 0$ or $\beta_{k+1} = 0$. In practice, orthonormality is soon lost, but relations (1)–(2) hold to working precision. (A compromise is to use *partial reorthogonalization*, as in the PROPACK package for sparse SVDs [1].)

2.2 Golub-Kahan with Regularization

Let δ be a given scalar (≥ 0 without loss of generality), and define

$$\tilde{A} = \begin{pmatrix} A \\ \delta I \end{pmatrix}, \quad \tilde{b} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \quad (3)$$

During $\text{Bidiag}(A, b)$, orthogonal matrices \tilde{Q}_k may be constructed from $2k - 1$ plane rotations to form the quantities

$$\tilde{Q}_k \begin{pmatrix} B_k \\ \delta I \end{pmatrix} = \begin{pmatrix} \tilde{B}_k \\ 0 \end{pmatrix}, \quad (\tilde{U}_{k+1} \ \tilde{Y}_k) = \begin{pmatrix} U_{k+1} \\ V_k \end{pmatrix} \tilde{Q}_k^T, \quad (4)$$

where \tilde{B}_k (like B_k) is lower bidiagonal with dimensions $(k + 1) \times k$. The following result is obtained straightforwardly from (1)–(4).

Result 2 *If $\text{Bidiag}(A, b) \rightarrow (B_k, U_{k+1}, V_k)$, then $\text{Bidiag}(\tilde{A}, \tilde{b}) \rightarrow (\tilde{B}_k, \tilde{U}_{k+1}, V_k)$.*

In short, the bidiagonalization of $\tilde{A} = \begin{pmatrix} A \\ \delta I \end{pmatrix}$ may be obtained efficiently from the bidiagonalization of A itself. The mechanism is less trivial than in the symmetric case. It motivates the subproblems used next.

Table 3: Subproblems defining y_k and $x_k = V_k y_k$ for three algorithms.

Method		Subproblem	Factorization
LSQR	$\delta = 0$	$\min \ B_k y_k - \beta_1 e_1\ $	$Q_k B_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix}$
	$\delta > 0$	$\min \left\ \begin{pmatrix} B_k \\ \delta I \end{pmatrix} y_k - \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} \right\ $	$Q_k \begin{pmatrix} B_k \\ \delta I \end{pmatrix} = \begin{pmatrix} R_k \\ 0 \\ 0 \end{pmatrix}$
CRAIG	$\delta = 0$	$L_k y_k = \beta_1 e_1$	

Table 4: Definition of W_k and z_k such that $x_k = V_k y_k = W_k z_k$.

		W_k	z_k
LSQR	$\delta = 0$	$V_k R_k^{-1}$	$Q_k \beta_1 e_1 = \begin{pmatrix} z_k \\ \bar{\zeta}_{k+1} \end{pmatrix}$
	$\delta > 0$	$V_k R_k^{-1}$	$Q_k \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} = \begin{pmatrix} z_k \\ \bar{\zeta}_{k+1} \\ q_k \end{pmatrix}$
CRAIG	$\delta = 0$	V_k	$z_k = y_k$

2.3 LSQR and CRAIG

As in the symmetric algorithms, LSQR and CRAIG [10, 11] use certain subproblems to define vectors y_k and solution estimates $x_k = V_k y_k$. Table 3 shows the subproblems and the factorizations needed to solve them. For example, from (1) we have

$$r_k \equiv b - Ax_k = \beta_1 u_1 - AV_k y_k = U_{k+1}(\beta_1 e_1 - B_k y_k), \quad (5)$$

which suggests the first LSQR subproblem. Table 4 shows how the factorizations are used to obtain estimates $x_k = W_k z_k$ that permit updates: $x_k = x_{k-1} + \zeta_k w_k$.

Note that $\text{Bidiag}(A, b)$ is used in all cases. The subproblem that allows LSQR to incorporate regularization was first proposed by Björck [2]. Result 2 helps confirm that the resulting method is equivalent to applying the original LSQR to \tilde{A} and \tilde{b} . (Working backwards, the proof of Result 2 reveals the need for the orthogonal factorization (4), which in turn suggests the subproblem.)

The QR factorizations for LSQR can be computed at negligible cost using one plane rotation for each k when $\delta = 0$, or two rotations when $\delta > 0$, giving upper-bidiagonal factors R_k . The matrix Q_k , a product of the rotations, does not need to be saved. The columns of W_k are obtained by forward substitution: $R_k^T W_k^T = V_k^T$. The following table compares the costs.

	Storage	Work per iteration
LSQR, any δ	$m + 3n$	$3m + 5n$
CRAIG, $\delta = 0$	$m + 2n$	$3m + 4n$

2.4 Preferences

Let $r_k = b - Ax_k$ be the residual for a given x_k , let $d_k = x - x_k$ be the error. LSQR chooses x_k to solve the problem

$$\min_y \|r_k\| \quad \text{such that} \quad x_k = V_k y,$$

so that $\|r_k\|$ decreases and the system may be incompatible. The properties of CRAIG are similar to those of SYMMLQ. The CRAIG point solves

$$\min_t \|d_k\| \quad \text{such that} \quad x_k = A^T U_k t$$

and also

$$\min_y \|x_k\| \quad \text{such that} \quad x_k = V_k y \quad \text{and} \quad U_k^T r_k = 0,$$

so that $\|d_k\|$ decreases, $\|x_k\|$ increases, and the system must be compatible. A benefit is that x_k is formed as a sequence of orthogonal steps.

Further discussion is given in [14]. While CRAIG is slightly more economical, a conclusion there is that LSQR is suitable for all cases in the sense that it is reliable on square, over-determined, and under-determined systems, with or without regularization.

Note that the solutions $x_k = V_k y_k$ lie in the Krylov subspace $\mathcal{K}(A^T A, A^T b, k)$. The convergence of these bidiagonalization-based methods depends on the eigenvalues of $A^T A$ (the squares of the singular values of A).

2.5 Estimation of Norms

At iteration k of LSQR, estimates of $\|r_k\|$, $\|A^T r_k\|$, $\|x_k\|$, $\|A\|$, and $\text{cond}(A)$ can be obtained at minimal cost. All five items are used in LSQR's stopping rules. To estimate norms, it is often necessary to assume that the columns of U_k and V_k are orthonormal ($U_k^T U_k = I$, $V_k^T V_k = I$). In practice this is rarely true, but the resulting estimates have proved to be remarkably reliable.

For simplicity we assume $\delta = 0$. As shown in [10], the following relations can be derived from (1), (2), and (5) and the QR factorization of B_k in Table 3:

$$\begin{aligned} r_k &= \bar{\zeta}_{k+1} U_{k+1} Q_k^T e_{k+1} \\ \|r_k\| &= \bar{\zeta}_{k+1} = \beta_1 s_1 s_2 \dots s_k \\ A^T r_k &= -(\bar{\zeta}_{k+1} \alpha_{k+1} c_k) v_{k+1} \\ \|A^T r_k\| &= \bar{\zeta}_{k+1} \alpha_{k+1} |c_k|. \end{aligned}$$

With orthogonality assumptions we also have $V_k^T A^T A V_k = B_k^T B_k = R_k^T R_k$, and so from the Courant-Fischer minimax theorem, the eigenvalues of $B_k^T B_k$ are bounded above and below by the largest and smallest nonzero eigenvalues of $A^T A$. The same can be said of the singular values of B_k compared to those of A . It follows for the 2- and F-norms that $\|B_k\| \leq \|A\|$ and $\|R_k^{-1}\| = \|B_k^+\| \leq \|A^+\|$. With $W_k = V_k R_k^{-1}$ we now have

$$1 \leq \|B_k\| \|W_k\| \leq \|A\| \|A^+\| = \text{cond}(A)$$

for the 2- and F-norms. Hence we use the monotonically increasing estimates

$$\|A\|_F \approx \|B_k\|_F, \quad \text{cond}(A) \approx \|B_k\|_F \|W_k\|_F,$$

where $\|B_k\|_F^2 = \|B_{k-1}\|_F^2 + \alpha_k^2 + \beta_{k+1}^2$ and $\|W_k\|_F^2 = \|W_{k-1}\|_F^2 + \|w_k\|^2$ can be updated easily.

To estimate $\|x_k\|$, we make use of the so-called QLP factorization of B_k (see Stewart [16]). The QR factorization of B_k can be followed by an orthogonal transformation from the right that reduces R_k to lower-bidiagonal form: $R_k P_k = \bar{L}_k$. Recall that the least-squares subproblem for defining y_k is solved by $R_k y_k = z_k$. If we define $y_k = P_k p_k$ for some vector p_k , we have $R_k y_k = R_k P_k p_k = \bar{L}_k p_k = z_k$, where p_k is obtained by forward substitution and hence differs from p_{k-1} in just its last element π_k . We now have $\|x_k\|^2 = \|V_k y_k\|^2 \approx \|y_k\|^2 = \|p_k\|^2$ and hence

$$\|x_k\|^2 = \|p_{k-1}\|^2 + \pi_k^2 = \|x_{k-1}\|^2 + \pi_k^2.$$

This construction shows that $\|x_k\|$ increases monotonically for LSQR. The same approach applies to MINRES, and has been implemented in MINRES-QLP [3].

2.6 Stopping Rules

We formulate rules for terminating LSQR in terms of three dimensionless quantities **Atol**, **btol**, and **conlim** specified by a user. The first two rules apply to compatible and incompatible systems respectively. The third rule applies to both.

S1 Stop if $\|r_k\| \leq \mathbf{Atol}\|A\|\|x_k\| + \mathbf{btol}\|b\|$.

S2 Stop if $\frac{\|A^T r_k\|}{\|A\| \|r_k\|} \leq \mathbf{Atol}$.

S3 Stop if $\text{cond}(A) \geq \mathbf{conlim}$.

Rules S1 and S2 are based on allowable perturbations in the data. The user may therefore set **Atol** and **btol** according to the (known or estimated) accuracy of the data. Rule S3 represents an attempt to regularize ill-conditioned systems.

To justify S1, note that x_k is the exact solution of the perturbed system

$$\left(A + \frac{\alpha}{\alpha + \beta} \frac{r_k x_k^T}{x_k^T x_k} \right) x_k = b - \frac{\beta}{\alpha + \beta} r_k$$

for any positive α and β . The perturbations will be “small enough” if their norms satisfy

$$\frac{\alpha}{\alpha + \beta} \frac{\|r_k\|}{\|x_k\|} \leq \mathbf{Atol}\|A\| \quad \text{and} \quad \frac{\beta}{\alpha + \beta} \|r_k\| \leq \mathbf{btol}\|b\|.$$

With the definitions $\alpha \equiv \mathbf{Atol}\|A\|\|x_k\|$ and $\beta \equiv \mathbf{btol}\|b\|$, both inequalities reduce to $\|r_k\| \leq \alpha + \beta$, as required in S1.

To justify S2, note that x_k and r_k are the exact solution and residual for the perturbed least-squares problem

$$\min \|(A + E_k)x - b\|, \quad E_k \equiv -\frac{r_k r_k^T A}{\|r_k\|^2}, \quad \|E_k\|_2 = \|A^T r_k\| / \|r_k\|.$$

Hence the perturbation to A is sufficiently small if S2 is satisfied.

Stopping rule S3 is based on the following arguments. Suppose that A has the singular value decomposition USV^T with $S = \text{diag}(\sigma_j)$, $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n > 0$. As k increases, our estimate $\text{cond}(A) \approx \|B_k\|_F \|W_k\|_F$ tends to level off near the values of the ordered sequence $\sigma_1/\sigma_1, \sigma_1/\sigma_2, \dots, \sigma_1/\sigma_n$, with varying numbers of iterations near each level. This suggests rule S3 as a means of regularizing ill-conditioned problems, as in the discretization of ill-posed problems (e.g., [7]). For example, if the singular values of A were known to be of order 1, 0.9, 10^{-3} , 10^{-6} , 10^{-7} , the effect of the two smallest singular values could probably be suppressed by setting $\text{conlim} = 10^4$.

3 Arnoldi-based Methods for Square Systems

For square unsymmetric systems $Ax = b$, it may seem desirable to develop a method in which the approximate solutions $x_k = V_k y_k$ lie in the Krylov subspace $\mathcal{K}(A, b, k)$ rather than LSQR's $\mathcal{K}(A^T A, A^T b, k)$. Indeed this is possible, but at the cost of steadily increasing work and storage.

3.1 The Arnoldi Process

Given a general matrix A and a starting vector b , the Arnoldi process generates vectors v_k and scalars β_k and h_{ik} as follows:

1. Set $\beta_1 v_1 = b$. (Exit if $\beta_1 = 0$.)
2. For $k = 1, 2, \dots$,
 - Compute $w = Av_k$
 - For $i = 1, 2, \dots, k$, set

$$h_{ik} = v_i^T w$$

$$w = w - h_{ik} v_i.$$
 - Set $\beta_{k+1} v_{k+1} = w$. (Exit if $\beta_{k+1} = 0$.)

(This is called the Modified Gram-Schmidt version of Arnoldi, and leads to the MGS-GMRES algorithm for solving $Ax = b$.) As for the symmetric Lanczos process, each matrix $V_k = (v_1 \ v_2 \ \dots \ v_k)$ would have orthonormal columns with exact arithmetic. After k steps, the situation may be summarized as

$$AV_k = V_{k+1} H_k, \tag{6}$$

where H_k is now a $(k+1) \times k$ Hessenberg matrix

$$H_k = \begin{pmatrix} h_{11} & h_{12} & \dots & \dots & h_{1k} \\ \beta_2 & h_{22} & \dots & \dots & h_{2k} \\ & \beta_3 & \dots & \dots & h_{3k} \\ & & \ddots & \vdots & \vdots \\ & & & \beta_k & h_{kk} \\ & & & & \beta_{k+1} \end{pmatrix}.$$

3.2 GMRES

If we define $x_k = V_k y_k$ for some y_k , (6) gives the residual

$$r_k \equiv b - Ax_k = \beta_1 v_1 - AV_k y_k = V_{k+1}(\beta_1 e_1 - H_k y_k), \quad (7)$$

which suggests the least-squares subproblem $\min \|H_k y_k - \beta_1 e_1\|$ for defining y_k (as in MINRES and LSQR). This is the basis for the *generalized minimum residual* method GMRES [13]. For each k , a single plane rotation maintains the QR factorization

$$Q_k H_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix}, \quad Q_k(\beta_1 e_1) = \begin{pmatrix} z_k \\ \bar{\zeta}_{k+1} \end{pmatrix},$$

from which the least-squares solution can be obtained when necessary by back-substitution: $R_k y_k = z_k$. Before that we have

$$r_k = V_{k+1} Q_k^T \left(\begin{pmatrix} z_k \\ \bar{\zeta}_{k+1} \end{pmatrix} - \begin{pmatrix} R_k \\ 0 \end{pmatrix} y_k \right) = \bar{\zeta}_{k+1} V_{k+1} Q_k^T e_{k+1},$$

so the estimate $\|r_k\| \approx |\bar{\zeta}_{k+1}|$ is available without y_k or x_k (and as for MINRES and LSQR it is a reliable estimate in practice, even if the columns of V_k are not reorthogonalized).

Note that the Arnoldi process requires all columns of V_k to be retained. Whenever $\|r_k\|$ is judged suitably small (or when k reaches a pre-defined limit m), the simplest form of GMRES terminates by solving $R_k y_k = z_k$ and forming $x_k = V_k y_k$ directly. There is no need to form x_1, x_2, \dots, x_{k-1} .

3.3 Restarted GMRES

For large systems, the iteration limit m is necessarily quite small (perhaps only 10 or 20). The simplest approach is to build a loop around GMRES analogous to iterative refinement for linear systems. The resulting algorithm is called GMRES(m):

1. Given x_0 , form $r = b - Ax_0$.
2. If $\|r\|$ is sufficiently small, exit with $x = x_0$.
3. Run the Arnoldi process for m iterations, starting with $\beta_1 v_1 = r$.
4. Solve the least-squares problem $\min \|H_m y_m - \beta_1 e_1\|$ and form $\Delta x = V_m y_m$.
5. Set $x_0 \leftarrow x_0 + \Delta x$ and return to step 1.

A difficulty with GMRES(m) is the need to choose m .

If A is positive definite (the symmetric part $(A + A^T)/2$ is positive definite), it can be proved that GMRES(m) converges for any $m \geq 1$ [12].

If A is diagonalizable—i.e., it has a full set of eigenvalues and can be written as $A = XDX^{-1}$ with D diagonal (but perhaps complex)—the *rate of convergence* can be bounded in terms of the condition number of X . This is most useful if A is normal or nearly normal ($A^T A \approx A A^T$), in which case $\text{cond}(X) \approx 1$.

For more general matrices A , GMRES is in danger of *stagnation*, in the sense that progress may become negligible for any practical value of m . (This is in contrast to

LSQR and Craig, which may converge slowly at times but will *always get there* if allowed to iterate long enough, and they do so using a constant amount of storage. The same is true of the symmetric methods CG, MINRES, SYMMLQ.) Google seems to have a non-monotonic memory:

Google search	LSQR	GMRES
April 2005	4,250	48,700
April 2006	35,000	254,000
April 2008	22,000	86,000
April 2009	18,000	92,000

but by any measure, GMRES is a remarkably popular method for solving unsymmetric square $Ax = b$ in spite of its apparent shortcomings. One reason is that it uses products Av but no transpose products $A^T u$ (which may not be available). A further saving grace is the availability of effective preconditioners for many practical cases.

4 Preconditioning

It is a cliché to say that iterative methods need good preconditioners, but this is the main hope for minimizing the total iterations required, and for keeping m low for GMRES(m).

For square systems $Ax = b$, there is a choice of left, right, or split preconditioning. We need matrices C_1, C_2 such that $C_1 C_2 \approx A$ and systems $C_i z = v$ and possibly $C_i^T z = u$ can be solved efficiently ($i = 1, 2$). The iterative method is then applied to $C_1^{-1} A C_2^{-1} y = C_1^{-1} b$, and the solution is recovered by solving $C_2 x = y$. The choice of C_i inevitably depends on each application.

In the absence of other knowledge, the *very least* that one should do is apply row and column scaling. The general aim would be to ensure that all rows and all columns have essentially the same norm. Some efficient scaling algorithms are described in [8] for symmetric, square, and rectangular systems.

More generally, incomplete Cholesky or LU factorizations of A are commonly used, in which the “incomplete” factors are more sparse than the exact factors.

For least-squares problems, we must use $C_1 = I$ to avoid altering the problem. Again, the *very least* one should do to help LSQR is to apply column scaling. In this case, it is important to make the 2-norms of all columns of $A C_2^{-1}$ equal. Thus, C_2 should be a diagonal matrix whose j th diagonal is $\|a_j\|_2$, where a_j is the corresponding column of A .

More generally, C_2 should approximate the R part of a QR factorization of A (because the exact R would give convergence in one iteration—the perfect preconditioner). It is more likely that we could compute sparse rectangular LU factors $P_1 A P_2 = \begin{pmatrix} L_1 & \\ & I \end{pmatrix} \begin{pmatrix} U \\ 0 \end{pmatrix}$, and then $C_2 = L_1 U P_2^T$ or $C_2 = U P_2^T$ may be effective.

5 Other Methods for Unsymmetric Systems

LSQR remains the primary iterative solver for over-determined (least-squares) problems, but should be kept in mind for square and under-determined systems.

Some other important methods for square unsymmetric systems are CGS [15], QMR [4], and Bi-CGSTAB [17]. Intriguing test cases are given by Nachtigal et al. [9] to show that LSQR, GMRES, and CGS are fundamentally different methods that can outperform each other (in iteration counts) by factors as large as \sqrt{n} or n .

References

- [1] PROPACK software for SVD of sparse matrices. <http://soi.stanford.edu/~rmunk/PROPACK/>.
- [2] Å. Björck. A bidiagonalization algorithm for solving ill-posed systems of linear equations. Report LITH-MAT-R-80-33, Dept. of Mathematics, Linköping University, Linköping, Sweden, 1980.
- [3] S.-C. Choi. *Iterative Methods for Singular Linear Equations and Least-Squares Problems*. PhD thesis, iCME, Stanford University, 2006.
- [4] R. W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-Hermitian linear systems. In R. Beauwens and P. de Groen, editors, *Iterative Methods in Linear Algebra*, pages 151–154. Elsevier Science Publishers, 1992.
- [5] G. H. Golub and W. Kahan. Calculating the singular values and pseudoinverse of a matrix. *SIAM J. Numer. Anal.*, 2:205–224, 1965.
- [6] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. The Johns Hopkins University Press, Baltimore, third edition, 1996.
- [7] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. SIAM, Philadelphia, 1998.
- [8] O. E. Livne and G. H. Golub. Scaling by binormalization. Technical Report SCCM-03-12, SCCM Program, Stanford University, 2003. <http://sccm.stanford.edu/pub/sccm/sccm03-12.pdf>.
- [9] N. M. Nachtigal, S. C. Reddy, and L. N. Trefethen. How fast are nonsymmetric matrix iterations? *SIAM J. Matrix Anal. Appl.*, 13(3):778–795, 1992.
- [10] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8(1):43–71, 1982.
- [11] C. C. Paige and M. A. Saunders. Algorithm 583; LSQR: Sparse linear equations and least-squares problems. *ACM Trans. Math. Software*, 8(2):195–209, 1982.
- [12] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, second edition, 2003.
- [13] Y. Saad and M. H. Schultz. GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. and Statist. Comput.*, 7:856–869, 1986.
- [14] M. A. Saunders. Solution of sparse rectangular systems using LSQR and CRAIG. *Nordisk Tidskr. Informationsbehandling (BIT)*, 35:588–604, 1995.
- [15] P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. and Statist. Comput.*, 10:36–52, 1989.
- [16] G. W. Stewart. The QLP approximation to the singular value decomposition. *SIAM J. Sci. Comput.*, 20(4):1336–1348, 1999.
- [17] H. A. van der Vorst. BI-CGSTAB: a fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. and Statist. Comput.*, 13(2):631–644, 1992.