BIT 35(1995), 588–604.

SOLUTION OF SPARSE RECTANGULAR SYSTEMS USING LSQR AND CRAIG [*]

MICHAEL A. SAUNDERS [†]

Systems Optimization Laboratory, Department of Operations Research
Stanford University, Stanford, CA 94305-4022, USA.
email: mike@SOL-michael.stanford.edu

Dedicated to Professor Åke Björck in honor of his 60th birthday

Abstract.

We examine two iterative methods for solving rectangular systems of linear equations: LSQR for over-determined systems $Ax \approx b$, and Craig's method for under-determined systems $Ax = b$. By including regularization, we extend Craig's method to incompatible systems, and observe that it solves the same damped least-squares problems as LSQR. The methods may therefore be compared on rectangular systems of arbitrary shape.

Various methods for symmetric and unsymmetric systems are reviewed to illustrate the parallels. We see that the extension of Craig's method closes a gap in existing theory. However, LSQR is more economical on regularized problems and appears to be more reliable if the residual is not small.

In passing, we analyze a scaled "augmented system" associated with regularized problems. A bound on the condition number suggests a promising direct method for sparse equations and least-squares problems, based on indefinite LDL^T factors of the augmented matrix.

AMS subject classification: 65F10, 65F20, 65F50, 65F05.

Key words: Conjugate-gradient method, least squares, regularization, Lanczos process, Golub-Kahan bidiagonalization, augmented systems.

1 Introduction.

Many iterative methods are known for solving square and rectangular systems of linear equations. We focus here on LSQR [21, 22] and Craig's method [5, 8, 18, 21], and examine their relationship when a regularization parameter δ is introduced.

LSQR and CRAIG (as we shall denote the implementations) solve compatible systems of the form

$$(1.1)\qquad Ax = b \qquad \text{or} \qquad \min \|x\|^2 \quad \text{subject to} \quad Ax = b,$$

[*]Received December 1994. Revised August 1995. Presented at the 12th Householder Symposium on Numerical Algebra, Lake Arrowhead, California, June 1993.

[†]Partially supported by Department of Energy grant DE-FG03-92ER25117, National Science Foundation grant DMI-9204208, and Office of Naval Research grant N00014-90-J-1242.

where A is an $m \times n$ real matrix and b a real m vector. Typically $m \leq n$, though not necessarily. Both methods are based on the Golub-Kahan bidiagonalization of A [11] with starting vector b. CRAIG is of interest because it is slightly simpler and more efficient. LSQR has an advantage if (1.1) has no solution: it solves the least-squares problem

$$\min \|Ax - b\|^2, \tag{1.2}$$

where typically $m \geq n$, though not necessarily.

1.1 Damping or Regularization

LSQR also solves the damped least-squares problem

$$\min \|Ax - b\|^2 + \|\delta x\|^2 \quad \equiv \quad \min \left\| \begin{pmatrix} A \\ \delta I \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|^2, \tag{1.3}$$

where δ is a small scalar parameter that regularizes the problem if $\operatorname{rank}(A) < n$ or A is ill-conditioned. Almost no additional work or storage are needed to incorporate regularization [22]. (We assume throughout that $\delta \geq 0$ is given. Methods for choosing δ, such as generalized cross-validation, form a separate and important field.)

Note that under-determined systems may be incompatible (e.g., in the case of image reconstruction when there is noise in the measurements). Problem (1.3) covers such cases, and LSQR may be applied. However, our original motivation was to extend CRAIG to incompatible systems in the hope that it might perform better than LSQR when $m \ll n$. For this purpose we study the problem

$$\begin{aligned} & \min \|x\|^2 + \|s\|^2 \quad \text{subject to} \quad Ax + \delta s = b, \\ \equiv \quad & \min \left\| \begin{pmatrix} x \\ s \end{pmatrix} \right\|^2 \quad \text{subject to} \quad \begin{pmatrix} A & \delta I \end{pmatrix} \begin{pmatrix} x \\ s \end{pmatrix} = b. \end{aligned} \tag{1.4}$$

Since this is a compatible system for any $\delta > 0$, Craig's method may be applied. In Section 4.4 we take advantage of the structure of $\begin{pmatrix} A & \delta I \end{pmatrix}$ to develop a specialized version of CRAIG. Some additional work and storage are required, but the method should be reliable if s is not too large compared to x.

Suppose $Ax = b$ has a solution (i.e., the system is compatible). If $\delta > 0$, it is easy to show that a solution of (1.3) does *not* satisfy $Ax = b$. Similarly for a solution of (1.4). However, if δ is rather small, $\|Ax - b\|$ may be negligible.

1.2 Equivalent Problems

If $\delta > 0$, problems (1.3) and (1.4) are both well-defined for any A (with arbitrary dimensions m and n). Somewhat surprisingly, they turn out to be the *same problem.* To see this, define $r = \delta s = b - Ax$ and eliminate s from the first line of (1.4). The extended LSQR and CRAIG algorithms may therefore be compared.

The equivalence of problems (1.3) and (1.4) was observed by Herman *et al.* [14], who used it to solve damped least-squares problems by applying Kaczmarz's method for compatible systems to (1.4). Dax [6] has recently improved the rate of convergence of this SOR-type approach, and extended it to least-squares problems with general linear constraints. Here we explore the equivalence from the viewpoint of conjugate-gradient-like methods.

1.3 Summary

To give some assurance that the equivalent formulation is justified numerically, Section 2 examines the eigenvalues and condition numbers of the systems defining r, s and x, and suggests a promising direct method. Section 3 reviews CG-like methods for symmetric systems. Section 4 discusses methods for unsymmetric or rectangular systems and presents the extended form of CRAIG. An overview is given in Section 5.

2 Eigenvalues of Augmented Systems.

The solution of the least-squares problem (1.3) is well known to satisfy the augmented system

$$\begin{pmatrix} I & A \\ A^T & -\delta^2 I \end{pmatrix} \begin{pmatrix} r \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \tag{2.1}$$

where $\delta \geq 0$. If $\delta > 0$, problems (1.3) and (1.4) are both solved by the alternative augmented system

$$\begin{pmatrix} \delta I & A \\ A^T & -\delta I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \tag{2.2}$$

where $r = \delta s$. It is interesting to find that the latter system may be more favorable when A is ill-conditioned. To see this, we regard both systems as special cases of

$$\begin{pmatrix} \alpha I & A \\ A^T & -\frac{\delta^2}{\alpha} I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \tag{2.3}$$

where $\alpha > 0$, $\delta \geq 0$, and $r = \alpha s$. We then extend the analysis of Golub and Björck (see [2, 4]) by expressing the eigenvalues of the augmented system in terms of the singular values of A.

If A has rank $p \leq \min(m, n)$, let its nonzero singular values be σ_i, $i = 1, \ldots, p$, with $\|A\| = \sigma_1$ and $\mathrm{cond}(A) = \sigma_1/\sigma_p$.

RESULT 1. *Let A be an $m \times n$ matrix of rank p with singular values σ_i. Assume $\alpha > 0$ and $\delta \geq 0$. The eigenvalues of the matrix in (2.3) are given by*

$$K_{\alpha\delta} \equiv \begin{pmatrix} \alpha I & A \\ A^T & -\frac{\delta^2}{\alpha} I \end{pmatrix}, \qquad \lambda(K_{\alpha\delta}) = \begin{cases} \frac{1}{2}(\alpha - \frac{\delta^2}{\alpha}) \pm \sqrt{\sigma_i^2 + \frac{1}{4}(\alpha + \frac{\delta^2}{\alpha})^2}, & \\ \alpha & m - p \text{ times}, \\ -\frac{\delta^2}{\alpha} & n - p \text{ times}. \end{cases}$$

When $\delta = 0$ and $p = n$, $\operatorname{cond}(K_{\alpha\delta})$ varies greatly with α: from approximately $\operatorname{cond}(A)^2$ when $\alpha \approx \sigma_1$ to about $\operatorname{cond}(A)$ when $\alpha \approx \sigma_n$. Considerable work has been done on choosing α to improve the numerical performance of direct methods when A is ill-conditioned; see [2, 1, 4]. (In [4], α is chosen to minimize not $\operatorname{cond}(K_{\alpha\delta})$ but a bound on the error in x.) In practice, accurate least-squares solutions may be obtained even if α is not especially close to its optimum value, though a safe choice remains problematical; e.g., see [17].

When $\delta > 0$, the difficulty of choosing α seems to vanish if we set $\alpha = \delta$, since the eigenvalues of $K_{\alpha\delta}$ then simplify and the condition of $K_{\alpha\delta}$ is readily seen.

RESULT 2. *Let A be an $m \times n$ matrix of rank p with singular values σ_i. Assume $\delta > 0$. The eigenvalues of the matrix in (2.2) are given by*

$$K_\delta \equiv \begin{pmatrix} \delta I & A \\ A^T & -\delta I \end{pmatrix}, \qquad \lambda(K_\delta) = \begin{cases} \pm\sqrt{\sigma_i^2 + \delta^2} & i = 1, \dots, p, \\ \delta & m - p \text{ times}, \\ -\delta & n - p \text{ times}. \end{cases}$$

RESULT 3. *If A is square and nonsingular,* $\operatorname{cond}(K_\delta) = \sqrt{(\sigma_1^2 + \delta^2)/(\sigma_n^2 + \delta^2)}$.
If $0 \le \delta \le \sigma_1$, $\operatorname{cond}(K_\delta) \le \sqrt{2}\operatorname{cond}(A)$.
If $\sigma_n < \delta < \sigma_1$, $\operatorname{cond}(K_\delta) \approx \|A\|/\delta$.

RESULT 4. *If A is rectangular, $\operatorname{cond}(K_\delta) = \sqrt{\sigma_1^2 + \delta^2}/\delta$.*
If $\sigma_p \le \delta \le \sigma_1$, $\operatorname{cond}(K_\delta) \le \sqrt{2}\operatorname{cond}(A)$.
If $0 < \delta < \sigma_1$, $\operatorname{cond}(K_\delta) \approx \|A\|/\delta$.

For rectangular A, we see that $\operatorname{cond}(K_\delta) \approx \|A\|/\delta$ regardless of the condition of A. Hence, (2.2) should be a reasonable system on which to base our extension of Craig's method, as long as δ is not too small.

Note that if $\|s\|$ is very large, good accuracy in the combined solution (s, x) may not imply good accuracy in x, although the error should be acceptable if $\|s\| \le 100\|x\|$, for example. In system (2.1), we therefore recommend that δ be large enough to satisfy $\|r\| \le 100\delta\|x\|$.

2.1 A Direct Method

The bounded condition of K_δ suggests a *direct* method for sparse linear equations and least-squares problems, based on factorizations of K_δ and the use of iterative refinement.

In principle, we could apply a stable, sparse factorizer to any of the augmented systems (2.1)–(2.3), as in [1]. A suitable package is MA47 [7], which computes sparse LBL^T factorizations (where B is block-diagonal with blocks of order 1 or 2). Iterative refinement may be used to recover precision if the factorizer is run with a loose stability tolerance to improve the sparsity of L.

More importantly, we focus on the fact that K_δ is "symmetric quasi-definite" when $\delta > 0$, so that Cholesky-type factorizations $PK_\delta P^T = LDL^T$ exist for arbitrary permutations P (with D diagonal but indefinite) [24]. MA47 is able to compute such factors, and they are typically more sparse (but less stable) than LBL^T factors. A stability analysis for solving $K_\delta z = d$ follows from [12], as shown in [10]. The key result is that the error in z is bounded by an *effective condition number* $\operatorname{Econd}(K_\delta)$, which is larger than the usual condition number.

When A is square and nonsingular, we find from [10] that the effective condition of K_δ is about $(\|A\|/\delta)\operatorname{cond}(K_\delta)$, and hence from Result 3,

$$\operatorname{Econd}(K_\delta) \approx \begin{cases} (\|A\|/\delta)\operatorname{cond}(A) & 0 < \delta < \sigma_n, \\ (\|A\|/\delta)^2 & \delta > \sigma_n. \end{cases}$$

Iterative refinement may be used to restore precision as before, and also to eliminate the effect of δ. For example, if we really want to solve $Ax = b$, we could apply refinement to the system

$$\begin{pmatrix} & A \\ A^T & \end{pmatrix}\begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}, \tag{2.4}$$

using LDL^T factors of K_δ to solve for corrections (with some convenient δ and c). In exact arithmetic, refinement will converge with any $\delta > 0$, and the convergence is rapid if $\delta < 0.5\sigma_n$, say. In practice, convergence to a solution of (2.4) seems to occur reliably if $\operatorname{cond}(A) < 1/\sqrt{\epsilon}$ (and $\delta < 0.5\sigma_n$), where ϵ is the machine precision.

When A is rectangular, $\operatorname{Econd}(K_\delta) \approx (\|A\|/\delta)^2$, and similar comments apply. The approach has been pursued elsewhere [23], with promising results.

3 Iterative Methods for Symmetric Systems.

To provide further background, we review three methods for solving symmetric systems $Bx = b$. As described in [20], the methods CGM, MINRES and SYMMLQ are based on the Lanczos process [16] for tridiagonalizing B. A helpful framework for viewing such methods has been suggested by Paige [19]:

> An iterative *process* generates certain quantities from the data. At each iteration a *subproblem* is defined, suggesting how those quantities may be combined to give a new estimate of the solution. Different subproblems define different *methods* for solving the original problem. Different ways of solving a subproblem lead to different *implementations* of the associated method.

Typically the subproblems may be solved efficiently and stably (though stability questions are sometimes overlooked). The numerically difficult aspects are usually introduced by the *process*.

The framework also applies to eigenvalue problems, but for $Bx = b$ we emphasize the additional idea of taking *orthogonal steps*, and the ability to *transfer* from one method to another (see Section 3.4).

3.1 The Lanczos Process

Let $\operatorname{Tridiag}(B, b) \to (T_k, V_k)$ denote the following process. Given a symmetric matrix B and a starting vector b, the Lanczos process generates vectors v_k and positive scalars α_k, β_k $(k = 1, 2, \ldots)$ such that after k steps,

$$BV_k = V_k T_k + \beta_{k+1} v_{k+1} e_k^T = V_{k+1} H_k, \tag{3.1}$$

where e_k is the kth unit vector, $V_k = (\, v_1 \;\; v_2 \;\; \ldots \;\; v_k \,)$, T_k is tridiagonal, and H_k is also tridiagonal with one extra row:

$$T_k = \begin{pmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \beta_3 & \\ & \ddots & \ddots & \ddots \\ & & \beta_k & \alpha_k \end{pmatrix}, \qquad H_k = \begin{pmatrix} T_k \\ \beta_{k+1} e_k^T \end{pmatrix}.$$

The starting condition is $\beta_1 v_1 = b$, and with exact arithmetic the columns of V_k would be orthonormal for each k until $\beta_{k+1} = 0$.

3.2 Lanczos with Shifts

Let δ be a given scalar. The next result is well known and follows directly from (3.1). We state it to motivate the unsymmetric case in Section 4.2.

RESULT 5. *If* Tridiag$(B, b) \to (T_k, V_k)$, Tridiag$(B + \delta I, b) \to (T_k + \delta I, V_k)$.

3.3 CGM, MINRES and SYMMLQ

At each iteration, CGM, MINRES and SYMMLQ use various subproblems to define vectors y_k, from which estimates of x could be obtained in the form $x_k = V_k y_k$. Table 3.1 shows the subproblems and the factorizations needed to solve them.

Since y_k may differ from y_{k-1} in all elements, each method computes certain quantities W_k and z_k, allowing the solution estimates to be *updated* in the form $x_k = W_k z_k = x_{k-1} + \zeta_k w_k$. Table 3.2 shows the possibilities.

Key numerical facts are that (3.1) holds to working precision, $V_k(\beta_1 e_1) = b$ exactly for all k, and the subproblems can be solved accurately. Residual norms may be estimated as the algorithms proceed (to provide stopping criteria). In practice, the residuals do eventually become small even when the columns of V_k are not reorthogonalized, but the convergence of each algorithm awaits full explanation.

3.4 Preferences

If A is positive definite, so is T_k; CGM can then obtain Cholesky factors of T_k for all k. MINRES uses the QR factorization of H_k, and is applicable to any symmetric B, including cases where $Bx = b$ is incompatible. SYMMLQ uses the same QR factorization (disguised as the LQ factorization of H_k^T), and is again applicable to any symmetric B, except that $Bx = b$ must be compatible.

Let $r_k = b - Bx_k$ be the residual vector for a given x_k, and let $d_k = x - x_k$ be the error in the solution. The different algorithms have numerous properties.

Method	Subproblem	Factorization	Estimate of x
CGM	$T_k y_k = \beta_1 e_1$	$T_k = L_k D_k L_k^T$	$x_k^C = V_k y_k$
MINRES	$\min \lVert H_k y_k - \beta_1 e_1 \rVert$	$Q_k H_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix}$	$x_k^M = V_k y_k$
SYMMLQ	$\min \lVert y_{k+1} \rVert$ s.t. $H_k^T y_{k+1} = \beta_1 e_1$	$H_k^T Q_k^T = \begin{pmatrix} L_k & 0 \end{pmatrix}$	$x_k^L = V_{k+1} y_{k+1}$

Table 3.1: Subproblems defining y_k and $x_k = V_k y_k$ for three algorithms.

	W_k	z_k	Estimate of x
CGM	$V_k L_k^{-T}$	$L_k D_k z_k = \beta_1 e_1$	$x_k^C = W_k z_k$
MINRES	$V_k R_k^{-1}$	$Q_k \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} = \begin{pmatrix} z_k \\ \bar{\zeta}_{k+1} \end{pmatrix}$	$x_k^M = W_k z_k$
SYMMLQ	$V_{k+1} Q_k^T \begin{pmatrix} I_k \\ 0 \end{pmatrix}$	$L_k z_k = \beta_1 e_1$	$x_k^L = W_k z_k$

Table 3.2: Definition of W_k and z_k such that $x_k = V_k y_k = W_k z_k$.

For example, the MINRES point x_k^M solves the problem "$\min_y \lVert r_k \rVert$ such that $x_k = V_k y$", so that $\lVert r_k \rVert$ decreases monotonically, and there is no difficulty if the system is incompatible.

In contrast, SYMMLQ's point x_k^L solves "$\min_t \lVert d_k \rVert$ such that $x_k = AV_k t$" [9, 15], so that $\lVert d_k \rVert$ decreases. It also solves "$\min_y \lVert x_k \rVert$ such that $x_k = V_k y$ and $V_k^T r_k = 0$", so that $\lVert x_k \rVert$ increases, and the system must be compatible.

Note that SYMMLQ accumulates x_k^L as a sequence of theoretically *orthogonal* steps. The columns of V_k and W_k are not orthonormal in practice, but at least $\lVert w_k \rVert \approx 1$ for SYMMLQ. On ill-conditioned systems, forming $W_k z_k$ should involve less cancellation error than in MINRES (and perhaps even in CGM), where the columns of W_k could be very large.

Note also that $\lVert r_k \rVert$ is often much larger for SYMMLQ than for the other methods. Since the residual norms can be estimated cheaply, SYMMLQ has provision for *transferring* to the CGM point upon termination if the residual is then smaller. Thus, if $\lVert r_{k+1}^C \rVert < \lVert r_k^L \rVert$, SYMMLQ takes a final step of the form $x_{k+1}^C = x_k^L + \bar{\zeta}_{k+1} \bar{w}_{k+1}$, where the last two items are already known. MINRES could transfer cheaply to the same CGM point if so desired.

4 Iterative Methods for Rectangular Systems.

The preceding thoughts carry over to the methods we are interested in for solving problems (1.2)–(1.4). As described in [21, 22], LSQR and CRAIG are based on the Golub-Kahan bidiagonalization procedure [11], which we should now call a *process*.

4.1 The Golub-Kahan Process

Let $\mathrm{Bidiag}(A, b) \rightarrow (B_k, U_{k+1}, V_k)$ or (L_k, U_k, V_k) denote the following process. Given a general matrix A and a starting vector b, the Golub-Kahan process generates vectors u_k, v_k and positive scalars α_k, β_k $(k = 1,\ 2,\ \ldots)$ such that after k steps,

$$\begin{aligned} AV_k &= U_{k+1}B_k = U_kL_k + \beta_{k+1}u_{k+1}e_k^T, \\ A^TU_{k+1} &= V_kB_k^T + \alpha_{k+1}v_{k+1}e_{k+1}^T = V_{k+1}L_{k+1}^T, \end{aligned} \tag{4.1}$$

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

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

The starting condition is $\beta_1 u_1 = b$, and with exact arithmetic the columns of U_k and V_k would be orthonormal for each k until $\alpha_{k+1} = 0$ or $\beta_{k+1} = 0$.

4.2 Golub-Kahan with Regularization

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

$$\tilde{A} = \begin{pmatrix} A \\ \delta I \end{pmatrix}, \qquad \tilde{b} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \qquad \hat{A} = (\, A \;\; \delta I\,). \tag{4.2}$$

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

$$\tilde{Q}_k \begin{pmatrix} B_k \\ \delta I \end{pmatrix} = \begin{pmatrix} \tilde{B}_k \\ 0 \end{pmatrix}, \qquad (\, \tilde{U}_{k+1} \;\; \tilde{Y}_k\,) = \begin{pmatrix} U_{k+1} & \\ & V_k \end{pmatrix} \tilde{Q}_k^T, \tag{4.3}$$

where $\tilde{B}_k$ (like B_k) is lower bidiagonal with dimensions $(k+1) \times k$. Alternatively, orthogonal matrices $\hat{Q}_k$ may be constructed from $2k-1$ plane rotations to form

$$(\, L_k \;\; \delta I\,)\hat{Q}^T = (\, \hat{L}_k \;\; 0\,), \qquad (\, \hat{V}_k \;\; \hat{Y}_k\,) = \begin{pmatrix} V_k & \\ & U_k \end{pmatrix} \hat{Q}_k^T, \tag{4.4}$$

where $\hat{L}_k$ (like L_k) is lower bidiagonal and $k \times k$. The following results are obtained straightforwardly from (4.1)–(4.4).

RESULT 6. *If* $\text{Bidiag}(A, b) \to (B_k, U_{k+1}, V_k)$, $\text{Bidiag}(\tilde{A}, \tilde{b}) \to (\tilde{B}_k, \tilde{U}_{k+1}, V_k)$.

RESULT 7. *If* $\text{Bidiag}(A, b) \to (L_k, U_k, V_k)$, $\text{Bidiag}(\hat{A}, b) \to (\hat{L}_k, U_k, \hat{V}_k)$.

In short, the bidiagonalizations of $\tilde{A} = \left(\begin{smallmatrix} A \\ \delta I \end{smallmatrix}\right)$ and $\hat{A} = (\, A \;\; \delta I \,)$ 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.

4.3 LSQR and CRAIG

As in the symmetric algorithms, LSQR and CRAIG use certain subproblems to define vectors y_k and solution estimates $x_k = V_k y_k$. Table 4.1 shows the subproblems and the factorizations needed to solve them. Table 4.2 shows how the factorizations are used to obtain updatable estimates $x_k = W_k z_k$.

Note that $\text{Bidiag}(A, b)$ is used in all cases. The subproblem that allows LSQR to incorporate regularization was first proposed by Björck [3]. Result 6 helps to confirm that the resulting algorithm is equivalent to applying the original LSQR to $\tilde{A}$ and $\tilde{b}$. (Working backwards, the proof of Result 6 reveals the need for the orthogonal factorization (4.3), which in turn suggests the subproblem.)

Similarly, the subproblem that allows CRAIG to incorporate regularization was originally just "written down", but Result 7 and its derivation now confirm that the resulting algorithm is equivalent to applying the original CRAIG to $\hat{A}$ and b.

4.4 The Extended CRAIG Algorithm

At last we have enough background to state the extended Craig-type algorithm for solving the regularized least-squares problem (1.3) in its equivalent form (1.4), namely

$$\min \|x\|^2 + \|s\|^2 \quad \text{subject to} \quad Ax + \delta s = b.$$

At stage k of $\text{Bidiag}(A, b)$, we solve the subproblem

$$\min \|y_k\|^2 + \|t_k\|^2 \quad \text{subject to} \quad L_k y_k + \delta t_k = \beta_1 e_1, \tag{4.5}$$

using the LQ factorization in (4.4):

$$(\, L_k \;\; \delta I \,)\hat{Q}_k^T = (\, \hat{L}_k \;\; 0 \,), \qquad \hat{L}_k z_k = \beta_1 e_1, \qquad \begin{pmatrix} y_k \\ t_k \end{pmatrix} = \hat{Q}_k^T \begin{pmatrix} z_k \\ 0 \end{pmatrix}. \tag{4.6}$$

We then define solution estimates (x_k, s_k) as follows:

$$\begin{pmatrix} x_k \\ s_k \end{pmatrix} \equiv \begin{pmatrix} V_k y_k \\ U_k t_k \end{pmatrix} = \begin{pmatrix} V_k & \\ & U_k \end{pmatrix} \begin{pmatrix} y_k \\ t_k \end{pmatrix} = (\, \hat{V}_k \;\; \hat{Y}_k \,) \begin{pmatrix} z_k \\ 0 \end{pmatrix} = \hat{V}_k z_k. \tag{4.7}$$

Since s_k is not really needed, we use the top of half of $\hat{V}_k z_k$ to obtain $x_k = W_k z_k$, where W_k is defined in Table 4.2. Note that $\hat{Y}_k$ is not needed either, but computation of W_k from V_k and $\hat{Q}_k$ requires additional work and storage. (In contrast, LSQR does not need $\tilde{U}_{k+1}$ or $\tilde{Y}_k$ in (4.3), so incurs little cost with regularization.) Table 4.3 compares the 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	$\delta = 0$	$L_k y_k = \beta_1 e_1$	
	$\delta > 0$	$\min \|y_k\|^2 + \|t_k\|^2$ s.t. $L_k y_k + \delta t_k = \beta_1 e_1$	$(\, L_k \;\; \delta I \,) \hat{Q}_k^T = (\, \hat{L}_k \;\; 0 \,)$

Table 4.1: Subproblems defining y_k and $x_k = V_k y_k$ for four algorithms.

		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	$z_k = y_k$
	$\delta > 0$	$(\, V_k \;\; 0 \,) \hat{Q}_k^T \begin{pmatrix} I_k \\ 0 \end{pmatrix}$	$\hat{L}_k z_k = \beta_1 e_1$

Table 4.2: Definition of W_k and z_k such that $x_k = V_k y_k = W_k z_k$.

	Storage	Work per iteration
LSQR, any δ	$m + 3n$	$3m + 5n$
CRAIG, $\delta = 0$	$m + 2n$	$3m + 4n$
CRAIG, $\delta > 0$	$m + 3n$	$3m + 8n$

Table 4.3: Comparison of algorithm costs with and without regularization.

4.5 Residuals

The solution of (4.5) satisfies

$$\begin{pmatrix} \delta I & L_k \\ L_k^T & -\delta I \end{pmatrix} \begin{pmatrix} t_k \\ y_k \end{pmatrix} = \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix}. \tag{4.8}$$

From (4.1) and (4.8), we find that

$$\begin{pmatrix} \delta I & A \\ A^T & -\delta I \end{pmatrix} \begin{pmatrix} s_k \\ x_k \end{pmatrix} = \begin{pmatrix} b + \eta_k \beta_{k+1} u_{k+1} \\ 0 \end{pmatrix}, \tag{4.9}$$

where η_k is the last element of y_k. Since $\hat{Q}_k$ is computed as a product of plane rotations,

$$\hat{Q}_k^T = \left(Q_{1,k+1} Q_{k+2,k+1}\right) \left(Q_{2,k+2} Q_{k+3,k+2}\right) \cdots \left(Q_{k-1,2k-1} Q_{2k,2k-1}\right) Q_{k,2k},$$

we have

$$\eta_k = e_k^T y_k = e_k^T \hat{Q}_k^T \begin{pmatrix} z_k \\ 0 \end{pmatrix} = e_k^T Q_{k,2k} \begin{pmatrix} z_k \\ 0 \end{pmatrix} = c_k e_k^T z_k = c_k \zeta_k,$$

where c_k is the cosine defining $Q_{k,2k}$, and ζ_k is the last element of z_k. Thus, CRAIG may terminate when $|c_k \zeta_k \beta_{k+1}|$ is suitably small. (The analogous quantity when $\delta = 0$ is $|\zeta_k \beta_{k+1}|$ [21].)

4.6 Transferring to the LSQR Point

Just as (1.4) is equivalent to (1.3) when $\delta \neq 0$, we see that the extended CRAIG subproblem (4.5) is equivalent to the regularized least-squares problem

$$\min \left\| \begin{pmatrix} L_k \\ \delta I \end{pmatrix} y_k - \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} \right\|^2, \tag{4.10}$$

which is very similar to the LSQR subproblem in Table 4.1:

$$\min \left\| \begin{pmatrix} B_k \\ \delta I \end{pmatrix} y_k - \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix} \right\|^2. \tag{4.11}$$

This in turn is equivalent to the minimum-length problem

$$\min \|y_k\|^2 + \|t_{k+1}\|^2 \quad \text{subject to} \quad B_k y_k + \delta t_{k+1} = \beta_1 e_1, \tag{4.12}$$

which we can easily solve using the LQ factorization (4.6) already available. A final step from the CRAIG point to the corresponding LSQR point can therefore be taken if desired, just as SYMMLQ can transfer to a final CGM point.

4.7 Preferences

Let $r_k = b - Ax_k$ be the residual for a given x_k, let $d_k = x - x_k$ be the error, and first suppose that there is no regularization.

LSQR chooses x_k to solve the problem "$\min_y \|r_k\|$ such that $x_k = V_k y$", 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 "$\min_t \|d_k\|$ such that $x_k = A^T U_k t$" and also "$\min_y \|x_k\|$ such that $x_k = V_k y$ and $U_k^T r_k = 0$", 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.

Similar properties hold when regularization is introduced. However, for CRAIG it is estimates of the combined vector $\binom{x}{s}$ that are formed via orthogonal steps. We cannot expect good accuracy in the final estimate of x if $\|x\| \ll \|s\|$.

5 The Broad Picture.

For completeness we state the following results, linking the regularized LSQR and CRAIG algorithms to hypothetical ones based on the symmetric Lanczos process. The equivalences are algebraic, and they generalize known results for the case $\delta = 0$. Result 9 closes a gap in existing theory.

5.1 Equivalence to CGM on Positive Definite Systems

RESULT 8. *The* LSQR *iterates x_k are the same as the* CGM *iterates x_k for the problem $(A^TA + \delta^2 I)x = A^Tb$.*

RESULT 9. *The Extended* CRAIG *iterates x_k are related to the* CGM *iterates for the problem $(AA^T + \delta^2 I)t = b$ according to $x_k = A^T t_k$.*

5.2 Equivalence to CGM Subproblem for Indefinite Systems

As noted in Section 2, the augmented system

$$K_\delta \begin{pmatrix} s \\ x \end{pmatrix} = \tilde{b}, \qquad K_\delta = \begin{pmatrix} \delta I & A \\ A^T & -\delta I \end{pmatrix}, \quad \tilde{b} = \begin{pmatrix} b \\ 0 \end{pmatrix} \tag{5.1}$$

is of interest when $\delta > 0$. From the Golub-Kahan process (4.1) we have

$$K_\delta \begin{pmatrix} U_{k+1} & \\ & V_k \end{pmatrix} = \begin{pmatrix} U_{k+1} & \\ & V_k \end{pmatrix} \begin{pmatrix} \delta I & B_k \\ B_k^T & -\delta I \end{pmatrix} + \begin{pmatrix} 0 \\ \alpha_{k+1} v_{k+1} \end{pmatrix} e_{k+1}^T$$

and

$$K_\delta \begin{pmatrix} U_k & \\ & V_k \end{pmatrix} = \begin{pmatrix} U_k & \\ & V_k \end{pmatrix} \begin{pmatrix} \delta I & L_k \\ L_k^T & -\delta I \end{pmatrix} + \begin{pmatrix} \beta_{k+1} u_{k+1} \\ 0 \end{pmatrix} e_k^T;$$

which are equivalent to the Lanczos process $\mathrm{Tridiag}(K_\delta, \tilde{b}) \to (\tilde{T}_{2k+1}, \tilde{V}_{2k+1})$ and $(\tilde{T}_{2k}, \tilde{V}_{2k})$ respectively, where

$$\tilde{T}_{2k+1} = \begin{pmatrix} \delta & \alpha_1 & & & & \\ \alpha_1 & -\delta & \beta_2 & & & \\ & \beta_2 & \delta & \alpha_2 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \alpha_k & -\delta & \beta_{k+1} \\ & & & & \beta_{k+1} & \delta \end{pmatrix} = \begin{pmatrix} \tilde{T}_{2k} & & 0 \\ & & \beta_{k+1} \\ 0 & \beta_{k+1} & \delta \end{pmatrix}$$

and

$$\tilde{V}_{2k+1} = \begin{pmatrix} u_1 & & u_2 & & \dots & u_k & & u_{k+1} \\ & v_1 & & v_2 & \dots & & v_k & \end{pmatrix} = \begin{pmatrix} \tilde{V}_{2k} & u_{k+1} \\ & 0 \end{pmatrix}.$$

RESULT 10. *Subproblem k in* LSQR *is equivalent to subproblem $2k+1$ in* CGM *for the indefinite system (5.1).*

RESULT 11. *Subproblem k in Extended* CRAIG *is equivalent to subproblem $2k$ in* CGM *for the indefinite system (5.1).*

5.3 Alternative Implementations

The equivalence of problems (1.3) and (1.4) reveals that there are two distinct ways of solving the regularized least-squares problem, based on distinct orthogonal factorizations. We may call these two *implementations*:

LS1 Form $\tilde{Q}\begin{pmatrix} A \\ \delta I \end{pmatrix} = \begin{pmatrix} R \\ 0 \end{pmatrix}$, set $\begin{pmatrix} c \\ d \end{pmatrix} = \tilde{Q}\begin{pmatrix} b \\ 0 \end{pmatrix}$, and solve $Rx = c$.

LS2 Form $(\, A \;\; \delta I \,)\hat{Q}^T = (\, L \;\; 0 \,)$, solve $Lz = b$, and form $\begin{pmatrix} x \\ s \end{pmatrix} = \hat{Q}^T\begin{pmatrix} z \\ 0 \end{pmatrix}$.

Similarly, there are two possible implementations of LSQR (when $\delta > 0$) and two options for extending CRAIG, based on two ways of solving the subproblems in Table 4.1:

LSQR1 Use QR factors of $\begin{pmatrix} B_k \\ \delta I \end{pmatrix}$, as in the standard LSQR.

LSQR2 Use LQ factors of $(\, B_k \;\; \delta I \,)$.

CRAIG1 Use QR factors of $\begin{pmatrix} L_k \\ \delta I \end{pmatrix}$.

CRAIG2 Use LQ factors of $(\, L_k \;\; \delta I \,)$, as in Section 4.4.

Experiments with Matlab indicate that the LSQR1 matrix is typically better conditioned than the other three shown. Hence, the standard LSQR implementation is probably the most reliable (and fortunately also the most efficient).

6 Numerical Tests.

We have compared LSQR and the extended CRAIG method on a range of regularized least-squares problems in which A has one of the forms

$$A = Y\begin{pmatrix} D \\ 0 \end{pmatrix}Z, \qquad YDZ, \qquad \text{or} \qquad Y(D\ \ 0)Z,$$

where D is a diagonal matrix containing specified singular values, and Y and Z are Householder matrices. (The problems are regularized versions of those in [21].)

Figure 6.1 shows how $\|A^T r_k - \delta^2 x_k\|$ and the error $\|x - x_k\|$ varied on a particular under-determined problem with A reasonably ill-conditioned ($m = 100$, $n = 200$, $\mathrm{cond}(A) \approx 10^6$, $\|A\| = 1$, $\|b\| \approx 1$, $\|x\| \approx 1$, $\|r\| \approx 5 \times 10^{-4}$, $\delta = 10^{-6}$, machine precision $\epsilon \approx 10^{-16}$).

The plots for LSQR seem considerably preferable at first sight, and suggest that transferring to the LSQR point may often be desirable. Since CRAIG minimizes the error in the combined vector (x, s), the error in x itself may not be monotonic. The effect is exaggerated by the fact that $\|s\| = \|r\|/\delta \approx 500 \gg \|x\|$.

In general, CRAIG performed well on systems of all shapes if $\|r\|$ was not too large and δ was not too small (say $\|r\|/\delta < 10^4$). After transfer to the LSQR point, the final solution agreed closely with LSQR. Under more extreme conditions, it was apparent that $\|x_k\|$ may exceed the exact $\|x\|$ by a large factor, and cancellation can occur during transfer to the final LSQR point. This is a definite disadvantage.

It was observed that LSQR also performed reliably on systems of all shapes (including ones that were strongly under-determined), with no apparent restriction on $\|r\|$ or δ. We checked for cancellation in the LSQR update $x_k = (V_k R_k^{-1})z_k = x_{k-1} + \zeta_k w_k$ by monitoring ζ_k, $\|w_k\|$ and their product (at no cost, since $\|w_k\|$ is already needed for one of LSQR's stopping criteria). We found that $\|w_k\|$ was indeed often large, but the corresponding ζ_k was invariably small. (The largest change to x_k was always for $k = 1$!) Thus, we have not yet found cases to illustrate CRAIG's potential advantage. Future tests on the examples proposed in [13] may be more revealing.

7 Conclusions.

Since Craig's method is efficient and reliable for compatible systems $Ax = b$, it has long seemed desirable to extend it to least-squares problems. The extension given here is equivalent to applying the existing method to a compatible system $Ax + \delta s = b$. It should therefore retain good numerical properties when $\|s\|/\|x\|$ is not too large. (Thus, $\|b - Ax\|$ should not be very large, and δ should not be too small.) Incompatible under-determined systems are likely candidates.

The extended method is slightly more expensive than LSQR. A potential advantage exists on problems for which the LSQR iterates are much larger than the solution: $\|x_k\| \gg \|x\|$. However, we have not yet found examples of such problems.

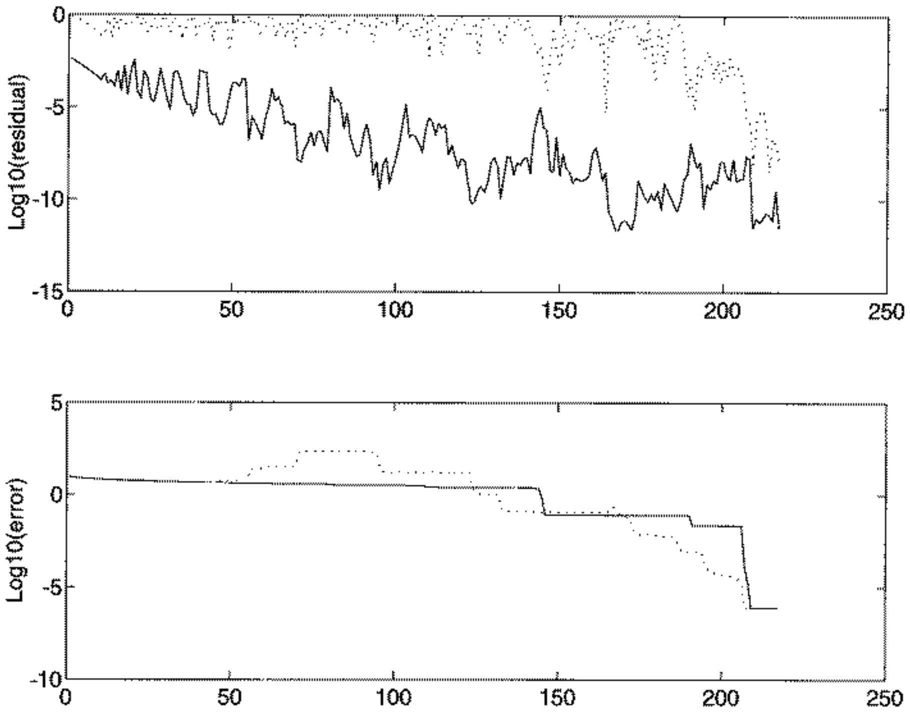

Figure 6.1: Residuals $\|A^T r_k - \delta^2 x_k\|$ for normal equations $(A^T A + \delta^2 I)x = A^T b$, and corresponding errors $\|x - x_k\|$. LSQR —, CRAIG $\cdots$.

A benefit of this research has been to observe that LSQR's reliable performance on over-determined systems seems to hold for under-determined systems also (with or without regularization).

A further benefit has been to focus on the augmented system (2.2) and the fact that $\text{cond}(K_\delta) \approx \|A\|/\delta$, suggesting a direct method for sparse equations and least-squares problems based on indefinite Cholesky factors (see Section 2).

The presentation has followed Paige [19] (and [20]) in emphasizing the separation of the Lanczos and Golub-Kahan processes from the subproblems used to define particular solution methods. It also illustrates the parallels between the algorithms for symmetric and unsymmetric equations, and unifies the bidiagonalizations of A, $\left(\begin{smallmatrix} A \\ \delta I \end{smallmatrix}\right)$ and $(\, A \;\; \delta I\,)$.

Acknowledgements.

I owe special thanks to Chris Paige for encouraging this work and for laying the foundations long ago. His comments on the manuscript were equally valuable. I am also grateful to Åke Björck, Bernd Fischer, Gene Golub and the referee for their kind help at critical moments.

REFERENCES

1. M. Arioli, I. S. Duff and P. P. M. de Rijk, *On the augmented system approach to sparse least-squares problems*, Numer. Math., 55 (1989), pp. 667–684.
2. Å. Björck, *Iterative refinement of linear least squares solutions I*, BIT, 7 (1967), pp. 257–278.
3. 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.
4. Å. Björck, *Pivoting and stability in the augmented system method*, in D. F. Griffiths and G. A. Watson (eds.), *Numerical Analysis 1991: Proceedings of the 14th Dundee Conference*, Pitman Research Notes in Mathematics 260, Longman Scientific and Technical, Harlow, Essex, 1992.
5. J. E. Craig, *The N-step iteration procedures*, J. Math. and Phys., 34, 1 (1955), pp. 64–73.
6. A. Dax, *On row relaxation methods for large constrained least-squares problems*, SIAM J. Sci. Comp., 14 (1993), pp. 570–584.
7. I. S. Duff and J. K. Reid, *Exploiting zeros on the diagonal in the direct solution of indefinite sparse symmetric linear systems*, ACM Trans. Math. Softw., to appear.
8. D. K. Faddeev and V. N. Faddeeva, *Computational Methods of Linear Algebra*, Freeman, London, 1963.
9. R. W. Freund, *Über einige CG-ähnliche Verfahren zur Lösung linearer Gleichungssysteme*, Ph.D. Thesis, Universität Würzburg, FRG, 1983.
10. P. E. Gill, M. A. Saunders and J. R. Shinnerl, *On the stability of Cholesky factorization for quasi-definite systems*, SIAM J. Mat. Anal., 17(1) (1996), to appear.
11. G. H. Golub and W. Kahan, *Calculating the singular values and pseudoinverse of a matrix*, SIAM J. Numer. Anal., 2 (1965), pp. 205–224.
12. G. H. Golub and C. F. Van Loan, *Unsymmetric positive definite linear systems*, Linear Alg. and its Appl., 28 (1979), pp. 85–98.
13. P. C. Hansen, *Test matrices for regularization methods*, SIAM J. Sci. Comput., 16(2) (1995), pp. 506–512.
14. G. T. Herman, A. Lent and H. Hurwitz, *A storage-efficient algorithm for finding the regularized solution of a large, inconsistent system of equations*, J. Inst. Math. Appl., 25 (1980), pp. 361–366.
15. D. P. O'Leary, Private communication, 1990.
16. C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45 (1950), pp. 255–282.
17. P. Matstoms, *Sparse QR factorization in MATLAB*, ACM Trans. Math. Software, 20(1) (1994), pp. 136–159.
18. C. C. Paige, *Bidiagonalization of matrices and solution of linear equations*, SIAM J. Numer. Anal., 11 (1974), pp. 197–209.
19. C. C. Paige, *Krylov subspace processes, Krylov subspace methods and iteration polynomials*, in J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, eds., *Proceedings of the Cornelius Lanczos International Centenary Conference, Raleigh, NC, Dec. 1993*, SIAM, Philadelphia, 1994, pp. 83–92.
20. C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12(4) (1975), pp. 617–629.

21. C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8(1) (1982), pp. 43–71.

22. C. C. Paige and M. A. Saunders, *Algorithm 583. LSQR: Sparse linear equations and least squares problems*, ACM Trans. Math. Software, 8(2) (1982), pp. 195–209.

23. M. A. Saunders, *Cholesky-based methods for sparse least squares: The benefits of regularization*, Report SOL 95-1, Dept. of Operations Research, Stanford University, California, USA, 1995.

24. R. J. Vanderbei, *Symmetric quasi-definite matrices*, SIAM J. Optim., 5(1) (1995), pp. 100–113.