SOLUTION OF SPARSE LINEAR EQUATIONS USING CHOLESKY FACTORS OF AUGMENTED SYSTEMS[*]

ALAN GEORGE[†] AND MICHAEL A. SAUNDERS[‡]

Abstract. Cholesky factorizations have reached a high peak of efficiency for solving sparse symmetric systems, largely because their Analyze and Factor phases do not conflict. We explore the possibility of using Cholesky packages to solve sparse *unsymmetric* systems $Ax = b$ by computing indefinite $\mathrm{LDL^T}$ factors of a *regularized augmented system* twice the size of A.

The same approach applies to rectangular systems (as an alternative to sparse Cholesky on A^TA or sparse QR on A). Here we focus on square systems and compare with sparse LU factorization, for which Analyze and Factor *do* conflict. On data from the Harwell-Boeing collection, we find that LU factors of A are usually more efficient, but the new approach performs reasonably on sparse banded systems such as those arising from 2-D and 3-D finite-difference grids.

Key words. sparse linear equations, direct methods, unsymmetric matrices, indefinite matrices, Cholesky factors, augmented systems, regularization

AMS subject classifications. 65F05, 65F50, 65N22

1. Introduction.

We focus on direct methods for solving general unsymmetric systems of linear equations

$$Ax = b, \tag{1.1}$$

where A is a sparse, nonsingular matrix of order n (with $n \le 10^6$, say). Square systems of this kind are typically handled by sparse LU software (notably MA48 [DR93] and SuperLU [DGL99, DGLL]). Row and column orderings can usually be found to preserve sparsity and stability in the LU factors and to make effective use of the hardware on present-day machines. The main disadvantage is that for any given A, the storage and computation requirements cannot be predicted with certainty.

For *symmetric positive-definite* systems, sparse Cholesky factorizations have also reached a high peak of efficiency (e.g., [DR95, Rot96]). Symmetric orderings can usually be found to preserve sparsity; stability is guaranteed, and the storage requirements can be determined in advance. To capitalize on these advantages, our aim is to solve unsymmetric systems (1.1) in a way that allows use of black-box Cholesky packages.

Of course, sparse Cholesky is often effective on the normal equations $A^TAx = A^Tb$. However, iterative refinement with the Cholesky factors may not give full precision in the computed x [Bjo67]. Also, A^TA or its Cholesky factors are sometimes unacceptably dense, even if A has no obviously dense rows. Sparse QR or LU packages would give better precision, but they tend to be inefficient in the same cases as sparse Cholesky.

We seek improved sparsity (and full precision) by applying sparse Cholesky to *regularized augmented systems* $K_{\delta\delta}z = d$, as described below.

[*]This is a revision of Report SOL 99-1 [Sau99]. First presented at the 7th Householder Symposium, Pontresina, Switzerland, 1996. This research was partially supported by National Science Foundation grant DMI-9500668, Office of Naval Research grants N00014-96-1-0274 and N00014-98-1-0299-A00002, and the Natural Science and Engineering Research Council of Canada.

[†]Dept of Computer Science, University of Waterloo, Ontario, Canada N2L 3G1 (jageorge @sparse3.uwaterloo.ca).

[‡]Dept of Management Science and Engineering, Stanford University, Stanford, CA 94305-4026 (saunders@stanford. edu).

2. The RAS Method. Consider the following *regularized normal equations*, where δ is a scalar and c is some vector, possibly zero:

$$(A^TA + \delta^2 I)x = A^Tb - \delta c, \qquad (AA^T + \delta^2 I)s = Ac + \delta b. \tag{2.1}$$

If δ is small (and A is square and nonsingular), we may regard these systems as perturbations of $Ax = b$ and $A^Ts = c$. More generally, they are equivalent to the least-squares problems

$$\min_x \left\| \begin{pmatrix} A \\ \delta I \end{pmatrix} x - \begin{pmatrix} b \\ -c \end{pmatrix} \right\|_2^2, \qquad \min_s \left\| \begin{pmatrix} A^T \\ \delta I \end{pmatrix} s - \begin{pmatrix} c \\ b \end{pmatrix} \right\|_2^2, \tag{2.2}$$

which may be solved simultaneously via the following *regularized augmented system*:

$$\begin{pmatrix} \delta I & A \\ A^T & -\delta I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix} \qquad \equiv \qquad K_{\delta\delta} z = d. \tag{2.3}$$

We make use of the Cholesky-type factorization

$$PK_{\delta\delta}P^T = LDL^T, \tag{2.4}$$

where P is a permutation, L is unit lower triangular, and D is diagonal but indefinite. Although $K_{\delta\delta}$ is indefinite, it is *symmetric quasi-definite* [Van95] and in exact arithmetic its $\mathrm{LDL^T}$ factors exist for any permutation P. Hence, P may be chosen for sparsity reasons alone, as for Cholesky on definite systems. We find that $K_{\delta\delta}$ is better conditioned than $A^TA + \delta^2 I$ and $AA^T + \delta^2 I$, and its Cholesky factors are potentially more sparse. Also, in finite precision the factors may be computed with "sufficient accuracy" if δ is not too small, and they may be used to implement iterative refinement for slightly different systems,

$$\begin{pmatrix} \delta_1 I & A \\ A^T & -\delta_2 I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix} \qquad \equiv \qquad K_{\delta_1\delta_2} z = d, \tag{2.5}$$

with $\delta_1,\ \delta_2 \ge 0$. We therefore propose the following method.

Algorithm RAS to solve $Ax = b$ and possibly $A^Ts = c$.

1. Choose $\delta > 0$ and compute the Cholesky-type factorization (2.4).
2. Solve $K_{\delta\delta}z = d$ (2.3) to obtain approximate solutions to $Ax = b$ and $A^Ts = c$.
3. Refine $K_{\delta\delta}z = d$ to solve the least-squares problems (2.2) more accurately.
4. Refine $K_{0\delta}z = d$ to solve $Ax = b$ more accurately.
5. Refine $K_{00}z = d$ to solve $Ax = b$ and $A^Ts = c$ more accurately.

If Step 4 is performed, one could set $c = 0$. In Step 4 or 5, one could use $0 < \delta_i \ll \delta$ if residual norms of order δ_i are considered small enough.

2.1. Previous work. Use of the system $K_{\delta\delta}z = d$ (2.3) was suggested in [Sau95, Sau96] for both square and rectangular systems. It was motivated by Vanderbei's indefinite $\mathrm{LDL^T}$ factorizations in LOQO [Van94, Van95] and by the stability analyses in [GV79, GSS96]. Further stability analysis is given by Mathias [Math92]. When applied to our matrix $K_{\delta\delta}$, the results of [Math92] and [GSS96] both indicate that $\mathrm{LDL^T}$ factorizations of $K_{\delta\delta}$ should be sufficiently stable as long as $\|A\|/\delta$ is not too large.

System (2.3) has been used to implement a primal-dual barrier method for regularized linear programs within the mathematical programming system OSL [Sau96, ST96, OSL]. In this context, "A" is a rectangular matrix of the form $\bar{D}\bar{A}^T$, where $\bar{D}$ is diagonal and $\bar{A}$ is the constraint matrix in the linear program. A sequence of problems is solved in which $\bar{D}$ and hence "A" become increasingly ill-conditioned. With $\delta \approx 10^{-4}\|A\|$, reliable performance has been obtained without the aid of iterative refinement.

Here we focus on systems where A is square, and we examine the convergence of iterative refinement when $\delta \approx 10^{-7}\|A\|$.

3. Condition of $K_{\delta\delta}$.

Let A be $n \times n$ with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$, so that $\|A\| = \sigma_1$ and $\text{cond}(A) = \sigma_1/\sigma_n$. Also, let

$$B_1 = \begin{pmatrix} A \\ \delta I \end{pmatrix}, \qquad B_2 = \begin{pmatrix} A^T \\ \delta I \end{pmatrix}, \qquad N = A^TA + \delta^2 I.$$

The eigenvalues of $K_{\delta\delta}$ in (2.3) are $\lambda_i(K_{\delta\delta}) = \pm\sqrt{\sigma_i^2 + \delta^2}$, so that

$$\begin{aligned} \text{cond}(K_{\delta\delta}) &= \sqrt{(\sigma_1^2 + \delta^2)/(\sigma_n^2 + \delta^2)} \\ &= \text{cond}(B_1) = \text{cond}(B_2) = \sqrt{\text{cond}(N)}. \end{aligned}$$

Thus when A is square, x and s in (2.2) are defined by least-squares problems with the same condition. Also, if A is square and nonsingular, x and s are well-defined for any b and c, even if $\delta \to 0$. (This is the principal difference from the rectangular or singular case.)

In practice we expect to have $0 < \delta < \|A\|$. The condition of A is typically not known, but we see that it affects the condition of $K_{\delta\delta}$ as follows:

$$\begin{array}{ll} \text{If } A \text{ is well-conditioned } (\sigma_n > \delta), & \text{cond}(K_{\delta\delta}) \approx \text{cond}(A). \\ \text{If } A \text{ is ill-conditioned } (\sigma_n < \delta), & \text{cond}(K_{\delta\delta}) \approx \|A\|/\delta. \end{array} \tag{3.1}$$

3.1. Effective condition with indefinite factors.

As noted above, LDL$^\text{T}$ factors of $K_{\delta\delta}$ exist for any ordering P, the same as for definite systems. The factorizations may be somewhat unstable, but following [GV79, Math92, GSS96] we find that errors in computed solutions of systems $K_{\delta\delta}z = d$ are bounded by an "effective condition number", $\text{Econd}(K_{\delta\delta})$. This is proportional to the true condition number, magnified by a factor that happens to be $\|A\|/\delta$ [Sau95]. From (3.1), the following hold when A is square:

$$\begin{array}{ll} \text{If } A \text{ is well-conditioned } (\sigma_n > \delta), & \text{Econd}(K_{\delta\delta}) \approx (\|A\|/\delta)\,\text{cond}(A). \\ \text{If } A \text{ is ill-conditioned } (\sigma_n < \delta), & \text{Econd}(K_{\delta\delta}) \approx (\|A\|/\delta)^2. \end{array} \tag{3.2}$$

With machine precision ϵ, we can expect to solve $K_{\delta\delta}z = d$ accurately (perhaps with the aid of iterative refinement) as long as $\text{Econd}(K_{\delta\delta}) < 1/\epsilon$.

3.2. Well-conditioned A.

If A is reasonably well-conditioned, (3.2) implies that refinement will converge if $\delta > \|A\|\,\text{cond}(A)\epsilon$. On current machines with 64-bit arithmetic, this means we can achieve residuals as follows:

cond(A)	$\|b - Ax\|/\|b\|$
$\geq 10^8$	10^{-7}
10^6	10^{-9}
10^4	10^{-11}
10^2	10^{-13}
10	10^{-14}

If we really want to solve $Ax = b$, we can use LDL$^{\mathrm{T}}$ factors of $K_{\delta\delta}$ to refine the systems

$$\begin{pmatrix} & A \\ A^T & -\delta I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} & A \\ A^T & \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}.$$

In exact arithmetic, refinement converges with any $\delta > 0$, and the convergence is rapid if $\delta < 0.5\sigma_{\min}(A)$, say. In practice, we can achieve the following:

cond(A)	$\|b - Ax\|/\|b\|$
$\leq 10^7$	10^{-15}

3.3. Ill-conditioned A. When A is ill-conditioned (cond$(A) > 1/\sqrt{\epsilon}$), (3.2) implies that we must have $\delta > \|A\|\sqrt{\epsilon}$ for refinement of $K_{\delta\delta}z = d$ to converge. This means we can achieve $\|b - Ax\|/\|b\| \approx 10\sqrt{\epsilon} \approx 10^{-7}$ *regardless of the condition of* A. (If 128-bit precision becomes commonplace, the achievable residual will be about 10^{-15}.)

4. Preconditioning iterative methods. For the rare cases where iterative refinement fails, we consider using LDL$^{\mathrm{T}}$ factors of $K_{\delta\delta}$ to generate a preconditioner for certain iterative solvers.

4.1. SYMMLQ. We propose applying SYMMLQ [PS75] to the symmetric indefinite system

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

using the positive definite preconditioner $C = L|D|L^T$.

4.2. LSQR.

$$K_{\delta\delta} = K_{\delta_1\delta_2} = LDL^T$$

$$K_{\delta\delta}^{-1} = \begin{pmatrix} \delta(AA^T + \delta^2 I)^{-1} & (AA^T + \delta^2 I)^{-1}A \\ (A^TA + \delta^2 I)^{-1}A^T & -\delta(A^TA + \delta^2 I)^{-1} \end{pmatrix}.$$

Note that

$$\begin{pmatrix} 0 & I \end{pmatrix} K_{\delta\delta}^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} = (A^TA + \delta^2 I)^{-1}A^T \approx A^{-1}.$$

Hence define

$$C = \begin{pmatrix} 0 & I \end{pmatrix} (LDL^T)^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} \approx A^{-1}.$$

- Refine $K_{0\delta} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$ using $K_{\delta\delta} = LDL^T$ (as before).
- Apply LSQR to $(AC)\Delta y = b - Ax$.
- Set $\Delta x = C\Delta y$. Set $x = x + \Delta x$.

5. Numerical results.

Here we give results for solving square systems $Ax = b$ by three different methods: LU, AS and RAS.

As test data we used the optimal basis matrices from some of the Netlib linear programming (LP) problems [Gay85], and most of the unsymmetric examples in the Harwell-Boeing collection [DGL89, DGL92], now conveniently available from the Matrix Market [MM96]. Genuine values were available for the nonzeros in A. We applied geometric-mean scaling (several passes through the columns and rows of A) to ensure that $\|A\| \approx 1$. We then defined $b = Ae$, where e is a vector of 1s, and we set $c = 0$.

Double precision arithmetic was used ($\epsilon \approx 10^{-16}$), and cpu times are for a DEC Alpha 3000/400 workstation. The regularization used for $K_{\delta\delta}$ in (2.3) was $\delta = 10^{-6}$.

5.1. AS versus RAS.

To compare the AS and RAS methods we used the MA47 package of Duff and Reid [DR95, DR96b] to factorize $K_0 = LBL^T$ and $K_{\delta\delta} = LDL^T$. The threshold pivoting parameter was set to *pivtol* = 0.01 and 0.0 respectively.

Storage-wise, Table 5.1 shows that the $\mathrm{LBL^T}$ factorizations of K_0 perform well for most of the LP bases, especially for ones that are somewhat triangular. (The shell basis is exactly triangular.) Indeed, the number of nonzeros is comparable to the LU factors of A (Table 5.3). However, pilots shows considerable fill-in during the Factor phase, and Table 5.2 shows that for both Analyze and Factor, MA47's $\mathrm{LDL^T}$ factorizations of $K_{\delta\delta}$ were always more efficient than $\mathrm{LBL^T}$.

Similar experience with MA47 is reported by Duff [Duf94] on augmented systems constructed from rectangular LP matrices.

The same trend is shown on the Harwell-Boeing matrices, many of which have a sparse banded structure common to finite-difference systems arising from PDEs. After sherman5 strained the workstation limits, we didn't run $\mathrm{LBL^T}$ on larger examples. (We tried *pivtol* $= 10^{-6}$ on sherman5 and found that the $\mathrm{LBL^T}$ Factor nonzeros decreased from 942000 to 588000, and the Factor time decreased from 806 seconds to 180. However, the computed x had no digits of accuracy.)

These results show the RAS method to be quite promising.

5.2. LU versus RAS.

Two efficient packages for solving unsymmetric systems $Ax = b$ are SuperLU (Demmel *et al.* [DGL99, DGLL]) and MA48 (Duff and Reid [DR93, DR96a]).

Here we used MA48's Analyze and First-factorize with *pivtol* = 0.01 to obtain $A = LU$, and compared with the RAS method of (2.3): MA47 factorizing $K_{\delta\delta} = LDL^T$. This is a fair comparison for any given matrix A.

(If there are further matrices with exactly the same sparsity pattern, both approaches can economize by omitting their Analyze phase. MA48's First-factorize makes use of the row and column orderings from Analyze, but may alter the row permutation to retain stability (perhaps losing some sparsity in the factors). MA48's Fast-factorize keeps the orderings from Analyze (perhaps sacrificing stability). In contrast, the RAS method retains the same sparsity and stability for all matrices of the same sparsity pattern, assuming they are well scaled.)

Returning to single systems, Tables 5.3 and 5.4 show that $A = LU$ is significantly more efficient than $K_{\delta\delta} = LDL^T$ on the LP examples and for some of the Harwell-Boeing matrices (especially orani678). However, on the sparse banded systems (sherman5 and later), the trend is reversed. The number of LU nonzeros is rather similar to the number of $\mathrm{LDL^T}$ nonzeros. For the last six examples, the RAS method is significantly more efficient.

5.3. Residuals and errors. Table 5.5 lists residuals and errors for the LU and RAS methods. From the LU results we can estimate that the largest condition numbers were $\mathrm{cond}(A) \approx 10^5$ for greenbea, fs_760_3 and lnsp3937, and $\mathrm{cond}(A) \approx 10^7$ for gre_1107. In all cases, the LU and RAS methods achieved $\|b - Ax\|/\|b\| \leq 10^{-12}$ and 10^{-9} respectively, which seems adequate in practice. The errors in x are certainly larger for the RAS method (10^{-2} for the last two cases), but with these moderate condition numbers, iterative refinement would have reduced the largest residuals and errors significantly.

TABLE 5.1
Nonzeros for $K_0 = LBL^T$ and $K_{\delta\delta} = LDL^T$ (MA47, pivtol = 0.01 and 0.0 respectively).

	Analyze		Factor	
	$\mathrm{LBL^T}$	$\mathrm{LDL^T}$	$\mathrm{LBL^T}$	$\mathrm{LDL^T}$
shell	2577		2577	7762
25fv47	6354	Same	16018	20568
stair	7632	as	8371	34598
greenbea	9584	Factor	17215	52248
pilotja	19714		20386	101154
degen3	27261		35455	163514
pilots	43090		126121	198698
west0655	6167		10117	33596
west2021	13805		17033	44863
nnc1374	30292		62229	73393
gemat12	60677		82094	142228
steam2	48592		89188	91528
fs_760_3	26848		139666	79976
pores_2	39707		216628	101426
sherman5	138061		941853	305715

TABLE 5.2
Times for $K_0 = LBL^T$ and $K_{\delta\delta} = LDL^T$.

	Analyze		Factor	
	$\mathrm{LBL^T}$	$\mathrm{LDL^T}$	$\mathrm{LBL^T}$	$\mathrm{LDL^T}$
shell	.06	.10	.02	.02
25fv47	.36	.12	.27	.09
stair	.22	.19	.43	.14
greenbea	1.29	.19	.74	.30
pilotja	.58	.54	1.47	.46
degen3	2.65	1.86	3.74	1.56
pilots	5.44	1.31	20.20	2.60
west0655	.28	.11	.33	.13
west2021	1.96	.28	3.48	.14
nnc1374	1.22	.31	2.75	.28
gemat12	1.09	.65	1.93	.46
steam2	.13	.28	2.32	.68
fs_760_3	1.54	.27	31.35	.49
pores_2	2.47	.27	144.60	.62
sherman5	9.15	1.20	806.47	3.04

TABLE 5.3
Nonzeros for $A = LU$ (MA48, pivtol = 0.01) and $K_{\delta\delta} = LDL^T$ (MA47, pivtol = 0.0).

		Factor	
	n	LU	LDL$^{\mathrm{T}}$
shell	537	3006	7762
25fv47	357	13194	20568
stair	822	9386	34598
greenbea	2393	16870	52248
pilotja	941	27354	101154
degen3	1504	37850	163514
pilots	1442	88977	198698
west0655	655	8603	33596
west2021	2021	17998	44863
nnc1374	1374	47297	73393
gemat12	4929	89541	142228
steam2	600	75786	91528
fs_760_3	760	50639	79976
pores_2	1224	62078	101426
gre_1107	1107	66145	170913
jpwh_991	991	69726	156322
mcfe	765	116273	117728
orani678	2529	190211	464924
sherman5	3312	282956	305715
orsreg_1	2205	294085	412200
watt__2	1856	346499	295907
sherman3	5005	490741	531832
lns_3937	3937	721472	569300
lnsp3937	3937	595816	568353

TABLE 5.4
Times for $A = LU$ (MA48, pivtol = 0.01) and $K_{\delta\delta} = LDL^T$ (MA47, pivtol = 0.0).

	Analyze		Factor	
	LU	$\mathrm{LDL^T}$	LU	$\mathrm{LDL^T}$
shell	.00	.10	.01	.02
25fv47	.04	.12	.02	.09
stair	.06	.19	.03	.14
greenbea	.08	.19	.02	.30
pilotja	.11	.54	.03	.46
degen3	.22	1.86	.07	1.56
pilots	1.37	1.31	.35	2.60
west0655	.04	.11	.02	.13
west2021	.08	.28	.03	.14
nnc1374	.28	.31	.15	.28
gemat12	.47	.65	.17	.46
steam2	.43	.28	.33	.68
fs_760_3	.34	.27	.21	.49
pores_2	.43	.27	.23	.62
gre_1107	.56	.44	.36	1.74
jpwh_991	.30	.42	.38	1.87
mcfe	.83	1.16	.69	1.34
orani678	1.86	86.16	.66	10.56
sherman5	4.11	1.20	2.98	3.04
orsreg_1	3.75	.86	2.76	4.94
watt__2	2.57	.74	5.40	3.00
sherman3	9.95	1.25	8.71	6.84
lns_3937	13.50	1.37	12.73	5.09
lnsp3937	22.76	1.31	10.99	5.12

Table 5.5

Residuals $\|b - Ax\|/\|b\|$ and errors $\|x - e\|$ for computed x via $A = LU$ (MA48, pivtol = 0.01) and $K_{\delta\delta} = LDL^T$ (MA47, pivtol = 0.0).

	$\|b - Ax\|/\|b\|$		$\|x - e\|$	
	LU	$\mathrm{LDL^T}$	LU	$\mathrm{LDL^T}$
shell	1e-15	4e-13	3e-14	1e-07
25fv47	7e-16	1e-12	2e-14	3e-09
stair	1e-16	1e-12	3e-13	2e-07
greenbea	1e-16	2e-10	1e-11	7e-04
pilotja	6e-16	1e-11	7e-14	8e-06
degen3	1e-15	3e-14	8e-15	5e-10
pilots	2e-15	1e-11	2e-12	7e-08
west0655	4e-14	6e-11	1e-12	1e-06
west2021	5e-14	3e-11	5e-14	7e-07
nnc1374	4e-14	2e-09	7e-11	8e-03
gemat12	1e-13	1e-09	8e-12	8e-04
steam2	5e-17	8e-16	2e-15	5e-14
fs_760_3	4e-13	1e-09	3e-08	5e-02
pores_2	1e-13	1e-09	4e-12	1e-07
gre_1107	3e-13	7e-12	1e-06	9e-03
jpwh_991	2e-16	2e-12	8e-17	1e-11
mcfe	5e-14	3e-15	2e-14	1e-14
orani678	3e-15	2e-11	3e-11	2e-06
sherman5	9e-13	1e-10	1e-11	2e-08
orsreg_1	7e-14	3e-12	2e-13	8e-11
watt__2	1e-12	2e-11	3e-11	1e-07
sherman3	3e-12	1e-11	3e-11	8e-07
lns_3937	3e-13	2e-10	7e-09	4e-02
lnsp3937	5e-13	2e-10	3e-08	4e-02

6. Conclusions.

6. Conclusions. Developers of sparse QR packages normally have rectangular systems in mind [GN84, Mats92, Mats94, Sun96]. In general they would be surprised to find that they could compete with sparse LU packages on square systems $Ax = b$.

Similarly, the augmented-system approach is normally suited to rectangular systems, but in this paper we restricted ourself to square systems in order to compare with sparse LU. On the Harwell-Boeing test set, a surprise did eventuate—the RAS approach of (2.3) proved to be more efficient than MA48's First-factorize $A = LU$ on sparse banded systems, such as those arising from discretized Navier-Stokes equations. Iterative methods such as GMRES [SS86] are normally applied to such problems (since they are often far larger than the examples used here). Perhaps the RAS approach will be a useful direct method for solving smallish examples without concerns about stagnation, and for obtaining preconditioners for larger cases.

The primary benefit of the RAS approach to unsymmetric $Ax = b$ is that it can utilize any new sparse $\mathrm{LDL^T}$ package (as long as D is allowed to have negative elements!). Storage and work are determined by the Analyze phase, and small residuals are attainable *regardless of the condition of A*.

Acknowledgements. We thank Iain Duff and John Reid for providing MA47 and MA48 for the experiments described. Comments from Iain Duff on the 1996 draft are also gratefully acknowledged.

REFERENCES

[Adl99] M. Adlers, *Computing sparse orthogonal factors in* Matlab, Report LiTH-MAT-R-1998-19, Department of Mathematics, Linköping University, Linköping, Sweden (1999).

[Bjo67] Å. Björck, *Iterative refinement of linear least squares solutions I*, BIT, 7 (1967), 257–278.

[Bjo96] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[DEG99] J. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li and J. W. H. Liu, SIAM J. Matrix Anal. Appl., 20(3) (1999) 720–755.

[DGL99] J. Demmel, J. R. Gilbert and X. S. Li, SIAM J. Matrix Anal. Appl., 20(4) (1999) 915–952.

[DGLL] ———, The SuperLU library, http://www.nersc.gov/~xiaoye/SuperLU/, 1999.

[Duf94] I. S. Duff, *The solution of augmented systems*, in D. F. Griffiths and G. A. Watson (eds.), *Numerical Analysis 1993*, Longman, UK, 1994, 40–55.

[DGL89] I. S. Duff, R. G. Grimes and J. G. Lewis, *Sparse matrix test problems*, ACM Trans. Math. Softw., 15(1) (1989), 1–14.

[DGL92] ———, *User's guide for the Harwell-Boeing sparse matrix collection*, Technical Report TR/PA/92/86, ftp://ftp.cerfacs.fr/pub/harwell_boeing/userguide.ps.Z, 1992.

[DR93] I. S. Duff and J. K. Reid, *MA48, a Fortran code for direct solution of sparse unsymmetric linear systems of equations*, Report RAL 93-072, Rutherford Appleton Laboratory, Oxfordshire, England, 1993.

[DR95] ———, *MA47, a Fortran code for direct solution of indefinite symmetric systems of linear equations*, Report RAL 95-001, Rutherford Appleton Laboratory, Oxfordshire, England, 1995.

[DR96a] ———, *The design of MA48, a code for the direct solution of sparse unsymmetric linear systems of equations*, ACM Trans. Math. Softw., 22 (1996), 187–226.

[DR96b] ———, *Exploiting zeros on the diagonal in the direct solution of indefinite sparse symmetric linear systems*, ACM Trans. Math. Softw., 22 (1996), 227–257.

[Gay85] D. M. Gay, *Electronic mail distribution of linear programming test problems*, Mathematical Programming Society COAL Newsletter, 13 (1985), 10–12.

[GN84] A. George and E. Ng, *SPARSPAK: Waterloo Sparse Matrix Package; User's guide for SPARSPAK-B*, Report CS-84-37, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 1984.

[GSS96] 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), 35–46.

[GV79] G. H. Golub and C. F. Van Loan, *Unsymmetric positive definite linear systems*, Linear Algebra Appl., 28 (1979), 85–98.

[Math92] R. Mathias, *Matrices with positive definite Hermitian part: inequalities and linear systems*, SIAM J. Matrix Anal. Appl., 13(2) (1992), 640–654.

[Mats92] P. Matstoms, *QR27: Specification Sheet*, Department of Mathematics, University of Linköping, Linköping, Sweden, 1992.

[Mats94] ———, *Sparse QR factorization in MATLAB*, ACM Trans. on Math. Softw., 20(1) (1994), 136–159.

[MM96] The Matrix Market, Web pages and tools, R. Boisvert, R. Pozo, K. Remington, R. Barrett and J. Dongarra, http://math.nist.gov/MatrixMarket/, 1996.

[OSL] OSL, Optimization Subroutine Library, IBM Watson Research Center, Yorktown Heights, NY.

[PS75] C. C. Paige and M. A. Saunders, Solution of sparse indefinite systems of linear equations, SIAM J. Numerical Analysis, 12 (1975), 617–629.

[Rot96] E. Rothberg, *Performance of panel and block approaches to sparse Cholesky factorization on the iPSC/860 and Paragon multicomputers*, SIAM J. Sci. Comput., 17(3) (1996), 699–713.

[SS86] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving unsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), 856–869.

[Sau95] M. A. Saunders, *Solution of sparse rectangular systems using LSQR and CRAIG*, BIT, 35 (1995), 588–604.

[Sau96] ———, *Cholesky-based methods for sparse least squares: The benefits of regularization*, in L. Adams and J. L. Nazareth (eds.), *Linear and Nonlinear Conjugate Gradient-Related Methods*, SIAM, Philadelphia, 92–100, 1996.

[Sau99] ———, Solution of sparse linear equations using Cholesky factors of augmented systems, Report SOL 99-1, Dept of EESOR, Stanford University, 9 pages.

[ST96] M. A. Saunders and J. A. Tomlin, *Solving regularized linear programs using barrier methods and KKT systems*, Report SOL 96-4, Department of EESOR, Stanford University, Stanford, CA, 1996.

[Sun96] C. Sun, *Parallel sparse orthogonal factorization on distributed-memory multiprocessors*, SIAM J. Sci. Comput., 17(3) (1996), 666–685.

[Van94] R. J. Vanderbei, *LOQO: An interior-point code for quadratic programming*, Report SOR 94-15, Department of Statistics and Operations Research, Princeton University, Princeton, NJ, 1994.

[Van95] ———, *Symmetric quasi-definite matrices*, SIAM J. Optim., 5(1) (1995), 100–113.