RAL 93-084

The Solution of Augmented Systems

Iain S. Duff [1]

ABSTRACT

We examine the solution of sets of linear equations for which the coefficient matrix has the form

$$\begin{pmatrix} H & A \\ A^T & 0 \end{pmatrix}$$

where the matrix H is symmetric. We are interested in the case when the matrices H and A are sparse.

These augmented systems occur in many application areas, for example in the solution of linear programming problems, structural analysis, magnetostatics, differential algebraic systems, constrained optimization, electrical networks, and computational fluid dynamics. We discuss in some detail how they arise in the last three of these applications and consider particular characteristics and methods of solution.

We then concentrate on direct methods of solution. We examine issues related to conditioning and scaling, and discuss the design and performance of a code for solving these systems.

Keywords: augmented systems, constrained optimization, Stokes problem, indefinite sparse matrices, KKT systems, systems matrix, equilibrium problems, electrical networks, interior point methods, Gaussian elimination, multifrontal methods, direct methods.

AMS(MOS) subject classifications: 65F05, 65F15, 65F50, 65K05, 76D07, 90C30.

Computing Reviews classification: G.1.3

Computing Reviews General Terms: Algorithms

Text for invited lecture given at Dundee Biennial Conference from 29 June to 2 July, 1993. To appear in Proceedings edited by Griffiths and Watson.

[1] Current reports available by anonymous ftp from numerical.cc.rl.ac.uk (internet 130.246.8.23) in the directory "pub/reports".

Central Computing Department
Atlas Centre
Rutherford Appleton Laboratory
Oxon OX11 0QX

November 1, 1993.

Contents

1 Introduction

The augmented matrix

$$\begin{pmatrix} H & A \\ A^T & 0 \end{pmatrix} \tag{1.1}$$

is ubiquitous in the numerical solution of problems in applied mathematics (see, for example, [Strang, 1988]). Since we assume that H and A are sparse, then clearly the matrix (1.1) is even more so. We review a few simple properties of the matrix (1.1) in Section 2 and discuss, in Section 3, applications which give rise to the associated linear system

$$\begin{pmatrix} H & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}. \tag{1.2}$$

For historical or technical reasons, different approaches to the solution of system (1.2) are used in different application areas. We examine three major application areas and indicate system characteristics and usual solution techniques in each case. In many cases, H is positive definite and is often even diagonal. Also c is very often zero. This additional structure will also be exploited.

We introduce a direct method for the solution of system (1.2) in Section 4 and, in Section 5, consider issues related to the conditioning and scaling of the system. In Section 6, we describe briefly a new code, based on a multifrontal approach but respecting the structure of (1.1). We study the performance of this new code, MA47 from the Harwell Subroutine Library, in Section 7 and present some concluding remarks in Section 8.

2 Basic properties of augmented systems

There are some important facts that should be mentioned concerning the augmented system (1.2). We will assume throughout this paper that H is a square matrix of order m and that A has dimensions $m \times n$ with $m \geq n$. We will use the term *aspect ratio* to denote the ratio of m to n. We will see in Section 3 that the aspect ratio can have a strong influence on the choice of solution technique.

Clearly the matrix (1.1) is symmetric and indefinite. If A has full rank and H is nonsingular, then (1.1) is clearly nonsingular. Indeed, if H is positive definite, we see from the identity

$$\begin{pmatrix} H & A \\ A^T & 0 \end{pmatrix} = \begin{pmatrix} H & 0 \\ A^T & I \end{pmatrix} \begin{pmatrix} H^{-1} & 0 \\ 0 & -A^T H^{-1} A \end{pmatrix} \begin{pmatrix} H & A \\ 0 & I \end{pmatrix}$$

that, by Sylvester's Law of Inertia, we have m positive eigenvalues and n negative eigenvalues. Indeed, if μ_1 and μ_m are the largest and smallest eigenvalues of H and σ_1 and σ_n the largest and smallest singular values of A, then Rusten and Winther [Rusten and Winther, 1991] show that the spectrum of the matrix (1.1) is contained in the union of the intervals I^- and I^+, where

$$I^- = [\tfrac{1}{2}(\mu_m - \sqrt{\mu_m^2 + 4\sigma_1^2}), \tfrac{1}{2}(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_n^2})] \text{ and } I^+ = [\mu_m, \tfrac{1}{2}(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2})].$$

It is not necessary for H to be positive definite or even nonsingular for the augmented matrix to be nonsingular. A sufficient (and necessary) condition is that A has full column rank (rank(A)=n) and the columns of $\begin{pmatrix} H \\ A^T \end{pmatrix}$ are linearly independent. In this paper, we

will assume that the matrix (1.1) is nonsingular. A good reference for the basic properties of augmented systems, which includes the case where A is not full rank, is given by Wedin [Wedin, 1985].

Although we will never want to compute the inverse explicitly, it is useful to display it to help later in the understanding of ill conditioning. If H is nonsingular, the inverse of (1.1) is given by

$$\begin{pmatrix} S & B^T \\ B & -(A^T H^{-1} A)^{-1} \end{pmatrix} \tag{2.3}$$

where B is a left-generalized inverse of A given by $(A^T H^{-1} A)^{-1} A^T H^{-1}$ and S can be written as $H^{-1}(I - AB)$, where $(I - AB)$ is the orthogonal projector onto range$(A)^{\perp}$ (the null space of A^T).

In many cases, the second component in the right-hand side of system (1.2), c, is zero. Although this does not affect the augmented matrix or the properties that we have just described, it can have a significant effect on solution techniques for (1.2). A simple indication of this is that the (2,2) matrix of (2.1) will have no influence on the solution of the system. Some solution techniques fail to respect this and can suffer the consequences, particularly if the (2,2) block is ill conditioned. We return to this theme in Section 5.

3 Occurrence of augmented systems

Several authors (for example, [Fortin and Glowinski, 1983], [Strang, 1986], [Strang, 1988], and [Vavasis, 1993]) have emphasized the importance of the system (1.2) in applied mathematics, indeed Strang [Strang, 1986] calls this system the "fundamental system" and claims that it is indeed so. Principal areas in which such a system arises naturally are optimization, computational fluid dynamics, and electrical networks. We study these in more detail in Sections 3.1 to 3.3. Other important application areas include structural analysis, differential algebraic systems, heat equilibrium, and magnetostatics.

The language for systems of the form (1.2) is also highly dependent on the application area under consideration. In general, we have used the nomenclature from constrained optimization.

3.1 Optimization

Systems of the form (1.2) abound in applications in optimization. The most obvious example is in the minimization

$$\min(\tfrac{1}{2}(Hx, x) - (b, x)) \text{ subject to } A^T x = c$$

where $(\cdot\ ,\ \cdot)$ denotes the Euclidean inner product. The role of the variable y in (1.2) is more clearly seen if the above constrained minimization is expressed as finding the saddle point of the Lagrangian form

$$\tfrac{1}{2}(Hx, x) - (b, x) + (y, A^T x - c)$$

where the variables y are clearly the Lagrange multipliers.

A common name for system (1.2) in optimization is the KKT system, for three people involved in obtaining necessary optimality conditions for differentiable problems using this system, Karush, Kuhn, and Tucker [Kuhn and Tucker, 1951]. More generally, iterative methods for linearly and nonlinearly constrained minimization problems frequently solve a sequence of such quadratic minimization problems. Such techniques are normally known

as recursive (RQP) or successive (SQP) quadratic programming methods, see for example Gill, Murray, and Wright [Gill *et al.*, 1981]. In general, A is a matrix of constraint normals while H is a second derivative matrix or an approximation to it. In many cases H is diagonal, for example when solving linear programming problems by interior point methods [Karmarkar, 1984] or when solving weighted least-squares problems using the augmented system approach. Since we are normally solving a nonlinear system and the solution to the quadratic minimization gives the direction along which the current estimate is adjusted, we force this direction to be in the subspace orthogonal to the constraint normals so that c is zero. We will assume that this is so for the remainder of this subsection. Usually A has full column rank and H is symmetric semi-definite. The aspect ratio can occasionally be quite large but usually it is less than 4. Because optimizers like to use A as the Jacobian of the constraint matrix, they perversely use a notation for (1.1) with the transpose on the (1,2) block rather than on the (2,1) block.

There are two main methods for solving these systems in optimization, the range-space method and the null-space method. In range-space methods, variables corresponding to the (1,1) block are eliminated and the variables y are obtained from the equation

$$(A^T H^{-1} A)y = (A^T H^{-1})b \tag{3.4}$$

which, of course requires H to be nonsingular. The equations (3.1) are then solved by either an iterative method (for example, conjugate gradients) or a direct method (for example, sparse Cholesky). The main problems with this approach are that the system (3.1) can be much more ill conditioned than the original system and the coefficient matrix can be much denser than the augmented system.

In the null-space method use is made of the fact that, since $A^T x = 0$, x lies in the null space of A^T and so can be written as a linear combination of basis vectors of this null space. If these basis vectors are denoted by the matrix Z, then x can be written as Zs, for some vector s. If we then substitute in the equations in the first block of (1.2) we get the system

$$(Z^T H Z)s = Z^T b. \tag{3.5}$$

The coefficient matrix of (3.2) is called the reduced or projected Hessian. Potential problems with this approach are to obtain a good basis Z and the lack of sparsity in the reduced Hessian.

There are two cases where augmented systems arise in optimization that are so important that it is worth mentioning them explicitly. The first is in the subproblem at each stage of an interior point method for the solution of linear programming problems ([Karmarkar, 1984]). The matrix in this case has the form

$$\begin{pmatrix} D^{-2} & A \\ A^T & 0 \end{pmatrix}$$

where the diagonal matrix D has entries $\{x_1, x_2, \ldots, x_n\}$ which are all greater than zero. However, as we approach a solution, many of these components (which are the current estimate of the solution) will tend to zero so that the scaling of the D matrix can become very bad, although work of Stewart [Stewart, 1989] shows that the overall system is not so ill conditioned (see also discussion in [Vavasis, 1993]). Largely because of the economic importance of linear programming, this application has received great attention and many methods of solution have been proposed; including methods based on normal equations, QR, and conjugate gradients, as well as augmented systems. The use of the latter approach in this context is discussed by [Duff, 1990] and [Fourer and Mehrota, 1992] inter alios.

Another special case is (unweighted) linear least-squares where the (1,1) matrix is the identity. This case and that of the weighted least-squares just discussed in the interior point subproblem are extensively discussed in works by Björck (for example, [Björck, 1990], [Björck, 1991]). We will consider this special case further in our discussions in Sections 5–8.

Iterative methods are used widely in solving linear systems arising in optimization although they have not been used so often on system (1.1) because it is indefinite. Recent work, for example [Arioli *et al.*, 1993], has investigated extensions of conjugate gradients for this case.

3.2 Computational fluid dynamics

Although we concentrate in this section on applications from computational fluid dynamics, there are many other areas in differential equations giving rise to augmented systems. They occur in elasticity and heat equilibrium problems and indeed whenever second order elliptic equations are solved using a mixed formulation. There are several ways in which system (1.1) arises in computational fluid dynamics. The most obvious way is in the solution of the Stokes' equations for incompressible fluid flow

$$\begin{aligned} -\nabla^2 u + \nabla p &= f \\ \nabla u &= 0 \end{aligned}$$

with appropriate boundary conditions. The matrix H corresponds to the Laplacian and A comes from the discretization of the convective term that also appears in the continuity equations. The variables x are the velocity variables (u) and y are the pressure variables (p). Since in the finite element discretization the velocity function bases are typically quadratic over each element, while the pressure function bases are usually linear, the aspect ratio is sometimes as much as 20, particularly for three-dimensional problems (see, for example [Ng *et al.*, 1993]). Another characteristic of CFD equations is that the matrix H has very large order, often in the hundreds of thousands.

In these applications, in contrast to the general optimization case, the projected matrix $A^T H^{-1} A$ is not only well conditioned but has a condition number independent of the mesh size used in the discretization. The reduced system (3.1), used in pressure correction methods, is commonly solved using an iterative method like conjugate gradients. One problem with this approach is that it involves solving systems with H as coefficient matrix at every iteration of the conjugate gradient iteration. Since the system in H is often itself solved using an iterative method, we obtain an inner-outer iteration. If conjugate gradients is used for the outer iteration then an accurate answer is required from the inner iteration. However, recently Elman and Golub [Elman and Golub, 1993] view the so-called Uzawa method as a Richardson iteration on the reduced system and show that there is no need for an accurate solution of the inner iteration. Because the reduced system is not formed explicitly (it would involve too much work and would normally be dense), it is hard to obtain a good preconditioner. Elman and Golub use diagonal and tridiagonal approximations to the factors of the mass matrix associated with the pressures. Another possibility is to use the matrix $A^T A$ as a preconditioner. Since this matrix is symmetric but otherwise not easy to solve, another iteration is normally used to implement this preconditioner. Recently, Ng et al. [Ng *et al.*, 1993] have advocated using a direct method to solve the preconditioning equations since, unlike the reduced systems themselves, they can be very ill conditioned. A nice discussion of preconditioning the full augmented system is presented in [Rusten and Winther, 1991]. Ramage and Wathen [Ramage and Wathen,

1993] have experimented with solving the complete augmented system using the method of conjugate residuals with diagonal scaling.

3.3 Electrical networks

We comment on the application of electrical networks more briefly because they can produce augmented systems with quite different characteristics from those of the previous applications. We consider the case of purely resistive networks and show that augmented systems arise quite simply through an application of Ohm's Law and Kirchhoff's Law. The first Law relates resistances to current and voltages and the corresponding equations can be written

$$Dx - Au = b,$$

while the second Law, which ensures current conservation at the nodes, can be written

$$A^T x = 0,$$

where the variables x, u, and b are the currents in the edges, voltages at the nodes, and external voltages (batteries) respectively, D is a diagonal matrix of resistances, and A is the adjacency matrix for circuit connectivity ($a_{ij} = 1(-1)$ indicates that edge i originates (ends) at node j).

There are several differences between the augmented system consisting of the above equations and those we have considered earlier. The matrix A has only -1 or 1 as possible nonzero entries and its sparsity structure is determined by the physical connectivity of the network. Usually, the matrix D is very badly scaled (if an open circuit is included the corresponding value of D would be infinite while components corresponding to short circuits would be zero (or nearly so)).

Both Strang [Strang, 1988] and Vavasis [Vavasis, 1993] use electrical networks to introduce augmented systems and the latter develops a variant of his method particularly designed for the numerical character of A. Traditional solution methods are to use an LDL^T factorization with pivoting, but this can be very unstable if the scaling in D is bad enough.

4 Direct methods of solution

As we have seen earlier, particularly in Section 3.2, sometimes the systems can be so large that a direct solution technique by itself is impractical. However, there are many problems, even in computational fluid dynamics, where direct methods are competitive and additionally it is possible to combine such methods with iterative ones when problem sizes become huge. We restrict the rest of our discussion to direct methods of solution.

The most common approach is to form and solve the reduced equations (3.1), using a Cholesky or LDL^T factorization on H to generate the Schur complement matrix $-A^T H^{-1} A$. Cholesky or LDL^T factorization is then used to factorize the Schur complement matrix. This method is only applicable if H is nonsingular and even then can be unstable. Additionally the Schur complement is usually quite dense (and would be full if A had a full row), so that sparse techniques may be inefficient in the second factorization. It is, however, the basis for many of the methods discussed earlier and sometimes the dimensions of the Schur complement are small so the fill-in to the Schur complement is not a problem.

Another approach is to ignore symmetry and use an unsymmetric sparse solver. At first glance this may appear daft but it looks more attractive if the structure of the overall

system is considered. In particular, if we are solving a (weighted) least-squares problem, c is zero and we normally wish to solve (1.2) only for the variables y. If the residual, corresponding to variables x, is required we would normally obtain it from the direct computation $b - Ay$. Note that if we choose early pivots in Gaussian elimination from rows corresponding to zero right-hand side components and later pivots from columns corresponding to the variables y then much of the factorization need not be used to obtain the solution. Unfortunately, taking advantage of this in our pivoting algorithm may conflict with preserving sparsity so a compromise must be achieved. In some early experiments [Duff and Reid, 1976] this method did show promise but was generally inferior to symmetric factorization of augmented systems or normal equations.

QR based methods will normally suffer from too much fill-in but they are of interest for cases with particular structure in A, for example if A is block diagonal. This class has been examined recently by [Gulliksson and Wedin, 1992] who have developed a modified form of QR factorization although they have not been concerned with sparsity.

The system (1.2) is indefinite so it would be possible to use a modified Cholesky scheme similar to those proposed for such systems in the optimization context (for example, [Eskow and Schnabel, 1992]). However, these methods do not produce an accurate solution of (1.2) and do not respect the structure.

We conclude this section by introducing the approach that we will consider in the remainder of this paper. That is to use an LDL^T factorization of (1.1) where the matrix D is block diagonal (blocks of order 1 and 2) and L is block lower triangular. Our method will inherit much of the stability of the scheme for full systems proposed by Bunch [Bunch, 1971], [Bunch and Parlett, 1971]. Clearly the use of 2×2 pivots is necessary, as can be seen from the matrix

$$\begin{pmatrix} 0 & \times \\ \times & 0 \end{pmatrix}. \tag{4.6}$$

The above papers show also that 2×2 pivots are sufficient to ensure stability assuming numerical pivoting is used. As usual with sparse factorizations, we relax the numerical pivoting controls in order to preserve sparsity better. If the augmented matrix (1.1) is denoted by the matrix M with entries m_{ij}, we choose a 1×1 pivot if

$$|m_{kk}| \geq u . \max_{j \neq k} |m_{kj}|, \tag{4.7}$$

and a 2×2 pivot

$$\left| \begin{pmatrix} m_{kk} & m_{k\,k+1} \\ m_{k+1\,k} & m_{k+1\,k+1} \end{pmatrix}^{-1} \right| \begin{pmatrix} \max_{j \neq k,k+1} |m_{kj}| \\ \max_{j \neq k,k+1} |m_{k+1j}| \end{pmatrix} \leq \begin{pmatrix} u^{-1} \\ u^{-1} \end{pmatrix}, \tag{4.8}$$

where u is a threshold parameter ($0 \leq u \leq 0.5$), subject to the entry being suitable on sparsity grounds. We will discuss some aspects of sparsity pivoting later in Section 7.

5 Conditioning and Scaling

Clearly the use of the augmented system should be better for stability than the normal equations. We can see this trivially by observing that the pivot choice of selecting pivots down the diagonal of H first gives the normal equations; so normal equations are a particular case of using the augmented systems. On another level, we note that, although the normal equations appear in the inverse of the augmented system (the (2, 2) block in (2.1)), it was already observed that this block is not involved in the solution if c is zero. We now study the conditioning and error estimation for systems (1.1) where we assume H

is diagonal and c is zero. We would like any condition number or error estimate to reflect the structure, and we find that the work in [Arioli *et al.*, 1989] is ideally suited since most of the second block of equations will lie in their category 2 because often the residual is close to zero and we have assumed that c is. If we then define two scaled residuals ω_1 and ω_2, where ω_1 corresponds to the category 1 equations and ω_2 to the others, with corresponding condition numbers K_{ω_1} and K_{ω_2} then the estimate of the error in the solution is given by $\omega_1 K_{\omega_1} + \omega_2 K_{\omega_2}$.

As an example, we compare this estimate on a sequence of three problems where H is γI, $\gamma = 1, 10^3, 10^5$ and A is the matrix WELL1850 from the Harwell-Boeing test set [Duff *et al.*, 1992]. We have the results shown in Table 5.1, where K_∞ is the classical ∞-norm condition number.

Table 5.1: Comparison of error estimation

K_∞	$\omega_1 K_{\omega_1} + \omega_2 K_{\omega_2}$	actual error
3.10^{18}	9.10^{-1}	1.10^{-1}
1.10^{16}	2.10^{-10}	8.10^{-12}
2.10^{10}	1.10^{-10}	7.10^{-12}

The scaling of such systems that arise in the solution of sparse least-squares problems was addressed at length by Björck at the last Dundee meeting [Björck, 1992]. He uses an α scaling of the form

$$\begin{pmatrix} I & 0 \\ 0 & \alpha^{-1}I \end{pmatrix} \begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} \alpha I & 0 \\ 0 & I \end{pmatrix}$$

to give the scaled system

$$\begin{pmatrix} \alpha I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} r/\alpha \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}.$$

The main unresolved issue is how to choose *a priori* a suitable value for α. We are currently experimenting with some ideas for this and presently favour a scheme that uses a crude method to choose an initial α which is then refined if necessary using information from the first attempted solution. We will report on this separately [Duff *et al.*, 1993] but should mention here that we have found it very beneficial to first scale the matrix (1.1) using the Harwell Subroutine Library scaling routine MC30 (a symmetric variant of the algorithm described in [Curtis and Reid, 1972]) before applying the α scaling.

For example, we consider the problem where H is the identity matrix and A is the matrix FFFFF800 from the test set of LP problems distributed by David Gay [Gay, 1985]. FFFFF800 is of dimensions 1028×524 and has 6401 nonzero entries. The resulting augmented matrix is scaled to create 5 matrices which are increasingly ill conditioned, with classical condition numbers shown in Table 5.2. We then ran a range of scaling options on these matrices. We denote these five strategies by:

1. Just α scaling.
2. Just scaling by MC30.
3. α scaling followed by MC30.
4. Row scaling of A followed by α scaling.

5. Scaling by MC30 and then α scaling.

Note that when we row scale A in strategy 4, the augmented matrix has the form

$$\begin{pmatrix} D^{-2} & C \\ C^T & 0 \end{pmatrix}$$

where C is the row-equilibrated A ($A = DC$, D diagonal). This is the strategy recommended by Björck [Björck, 1992] when A is badly scaled. We obtain the results in Table 5.2 which clearly indicate that it is vital to scale A prior to using the α scaling and illustrates the advantage of strategy 5. We study various scaling strategies and their effect on the solution of augmented systems in [Duff *et al.*, 1993].

Table 5.2: Error in solution for various scalings

Matrix	Condition number	Scaling					
	K_∞	None	1	2	3	4	5
FFFFF_02	1.10^{09}	4.10^{-13}	3.10^{-13}	5.10^{-13}	4.10^{-13}	1.10^{-12}	5.10^{-13}
FFFFF_04	1.10^{13}	1.10^{-08}	6.10^{-13}	4.10^{-12}	1.10^{-12}	5.10^{-13}	5.10^{-13}
FFFFF_06	1.10^{16}	4.10^{-05}	5.10^{-13}	2.10^{-06}	2.10^{-05}	3.10^{-13}	1.10^{-13}
FFFFF_08	1.10^{20}	6.10^{-02}	2.10^{-11}	5.10^{-03}	3.10^{-03}	6.10^{-13}	2.10^{-13}
FFFFF_10	1.10^{23}	5.10^{+00}	9.10^{-05}	4.10^{-03}	9.10^{-02}	2.10^{-11}	2.10^{-13}

6 MA47, a sparse structured symmetric indefinite solver

We will use a multifrontal approach in designing our algorithm for the LDL^T factorization of sparse symmetric indefinite matrices. The details of such an approach are not necessary to understand this paper but further background can be obtained from the original papers by Duff and Reid [Duff and Reid, 1983], [Duff and Reid, 1984]. We now describe the features of multifrontal methods that are needed to follow the subsequent discussion. As is common in sparse elimination, the factorization is split into a symbolic phase, which performs an analysis using only the sparsity pattern of the matrix, and a numerical factorization phase.

In a multifrontal method, the sparse factorization proceeds by a sequence of factorizations on small dense matrices, called frontal matrices. The frontal matrices and a partial ordering for the sequence are determined by a computational tree where each node represents a full matrix factorization and each edge the transfer of data from child to parent node. This tree, which can be obtained from the elimination tree ([Duff, 1981], [Liu, 1990]), is determined solely by the sparsity pattern of the matrix and an initial pivot ordering that can be obtained by standard sparsity preserving orderings such as minimum degree. When using the tree to drive the numerical factorization, eliminations at any node can proceed as soon as those at the child nodes have completed, giving added flexibility for issues such as exploitation of parallelism. Normally, the complete frontal matrix cannot be factorized but only a few steps of Gaussian elimination are possible, after which the remaining reduced matrix (the Schur complement) needs to be summed (assembled) with other data at the parent node before further factorizations can take place. Thus the frontal matrices can be written

$$\begin{pmatrix} F_{11} & F_{12} \\ F_{12}^T & F_{22} \end{pmatrix}$$

where the matrix F_{11} is factorized and the Schur complement $F_{22} - F_{12}^T F_{11}^{-1} F_{12}$ is assembled with other Schur complements and original data at the parent node. Note that the elimination is done using full linear algebra and direct addressing while all the indirect addressing is confined to the assembly.

Since the symbolic phase chooses pivots only on the basis of the structure of the system, it is quite possible that they cannot be used during the numerical factorization because of the failure of the pivots to satisfy the criterion (4.2) or (4.3). If there are pivots rejected in this way during the numerical factorization, all that happens is that the corresponding block F_{11} is smaller than forecast by the analysis and the Schur complement correspondingly larger. Thus the amount of work and storage for the factorization can rise although the factorization can always be performed since, at the root node, any entry in the frontal matrix can be chosen as pivot and all the matrix can be factorized respecting inequalities (4.2) or (4.3).

In our original code [Duff and Reid, 1983], the symbolic phase used a minimum degree ordering that assumed any diagonal entry could be chosen as pivot. This did not cause many problems with the numerical factorization in our early experiments with indefinite systems, the overheads because of delayed pivots being seldom more than 10%. However, the situation with augmented systems is quite different since the symbolic phase might choose a sequence of pivots from the (2,2) block early in the factorization. If there has not been fill-in to these positions, then they will be zero during numerical factorization and the effect on subsequent frontal matrix sizes can be very significant. We were alerted to this problem by Gill, Murray, and Saunders (personal communication, 1989) and designed an analysis and factorization scheme that took account explicitly of the zero block [Duff *et al.*, 1991]. In this work, we were concerned not only to avoid choosing zero diagonal entries as pivots during the analysis but also to take into account the structure of the matrix during subsequent factorization. Based on the work in [Duff *et al.*, 1991], we have recently developed a new code which has been released in the Harwell Subroutine Library [Anon, 1993] under the name MA47.

The earlier code, MA27 [Duff and Reid, 1982] in the Harwell Subroutine Library, was run on the matrix

$$\begin{pmatrix} I_{m-n} & & A_1 \\ & 0_n & A_2 \\ A_1^T & A_2^T & 0_n \end{pmatrix} \tag{6.9}$$

where the matrix A_2 is a nonsingular $n \times n$ submatrix of the matrix FFFFF800 that was used in the experiments in Section 5. The symbolic phase of MA27 forecasts that 1.5 million floating-point operations are needed for the numerical factorization. However, the subsequent numerical factorization of MA27 requires 16.5 million floating-point operations when the threshold parameter u of (4.2) and (4.3) has the value 0.1.

In MA47, not only do we avoid pivoting on zeros in the symbolic phase but we also take account of the block structure. For example, in the matrix

$$\begin{pmatrix} \times & \times & \times & \times & \times & \times & \times \\ \times & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & 0 & & & & & \\ \times & 0 & & & & & \\ \times & 0 & & & & & \\ \times & 0 & & & & & \\ \times & 0 & & & & & \end{pmatrix},$$

if the (1,1) entry is chosen as pivot, the whole matrix is thereafter full. However, if the top right block of order two is chosen first (mathematically and structurally equivalent to

pivoting on the (1,2) entry followed by the (2,1) entry), the remaining Schur complement is completely zero. Notice that this pivot choice and that of (1,1) followed by (2,2) are mathematically equivalent although the sparse code will not recognize the cancellation caused by the (2,2) pivot. In our new code, we want to choose 2×2 pivots of this form, called *tile* pivots and pivots of the form (4.1), called *oxo* pivots.

In general, the Schur complement from these structured systems will not be either zero or full but they can all be represented by the block form

$$\begin{pmatrix} 0 & X & X \\ X & X & X \\ X & X & 0 \end{pmatrix}. \tag{6.10}$$

Our new code thus allows for non-full frontal matrices of this type and chooses pivots in symmetric pairs based on an unsymmetric Markowitz count.

One consequence of our strategy of allowing non-full Schur complements is that a front matrix is not always absorbed by the parent node and can propagate up the tree. For example, if an *oxo* pivot is chosen at the parent node, where the (1,1) entry is not in the Schur complement but the (2,2) entry is in the (1,1) block, then the resulting Schur complement will comprise two overlapping matrices of the form (6.2). Our alternatives are to store the Schur complement as a general sparse matrix or to exploit structure but then have multiple fronts. We choose the second alternative but notice that either causes our new code to be much more complicated than the old. A second issue, the effect of which we will observe in the next section, is that our sparsity criterion will seldom choose structured pivots when H is diagonal and the aspect ratio is high because normally the Markowitz cost for an off-diagonal will be significantly higher than for the entry on the diagonal. Thus we would not expect many *tiles* to be chosen in such cases. Since, in such examples, MA27 is likely for similar arguments to pivot first in the (1,1) block of (1.1), we would expect little improvement for the new code and in fact possibly worse performance because of the much more complicated structure of the MA47 code.

We now examine this performance briefly in the following section.

7 Preliminary experiments with MA47

There are some good reasons for favouring the selection of *tiles* or *oxo* pivots because the structure of the augmented system is unchanged.

We adapted the symbolic phase of MA47 to force the selection of more *tiles* and show the performance on the problem

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix}, \tag{7.11}$$

where the matrix A is the aforementioned FFFFF800 matrix.

We can see from the results in Table 7.1 that the performance is very flat over a wide range of forcing *tiles*. Also, although not noted in the table, the numerical accuracy was essentially unchanged. From this run and several others, we conclude that there is little point in modifying MA47 but that approaches that favour selecting *tiles* (for example, [Gould, 1993]) should not be too disadvantaged from a point of view of sparsity unless they severely overdo the amount of forcing.

Finally, when we ran MA47 on the example (6.1) of Section 6, only 7,954 floating-point operations were forecast by the symbolic phase, nearly a two-hundred fold improvement. The subsequent numerical factorization of MA47 also required only 7,954 floating-point

Table 7.3: Forcing of *tiles*

number *tiles*		Ops	Storage	Time
Forecast	Actual	(×1000)	(Kwords)	(SPARC-1 secs)
89	77	6.8	69.3	10.1
150	137	8.7	78.6	12.3
230	217	6.7	66.2	10.0
249	234	7.0	69.2	11.0
371	302	12.1	85.7	21.9

operations, now an improvement of over two thousand fold! Additionally, the accuracy was slightly better with MA47 although, because of the more complicated code, the factorization times for MA27 and MA47 on a SPARC-10 were 3.15 and 0.07 seconds respectively, for a still creditable ratio of 45.

We show in Table 7.2, results comparing MA47 with MA27 on a range of matrices of this form, where the threshold parameter for both codes has been set to 0.001. These results emphasize that significant gains can be obtained with the new code. However, unfortunately sometimes a penalty is paid for the extra complications in the new code and, on unconstrained problems of the form of (7.1) with the same LP matrices as in Table 7.2, MA27 usually outperforms MA47. For example, on problem (7.1) where A is equal to the FFFFF800 problem, the numerical factorization of MA47 requires .95 seconds on a SPARC-10 while MA27 needs only .57 seconds. It was results of this kind that prevented us making MA27 an obsolescent routine in the Harwell Subroutine Library. I should stress that, although not recorded in the table, the solution was obtained with a similar satisfactory accuracy for all the runs of both MA27 and MA47.

Table 7.4: Comparison between MA27 and MA47 on constrained least-squares problems. Time in secs on Sun SPARC-10.

		CPU time			No. Operations		Nonzeros
matrix A	Method	Symb	Fact	Total	Forecast	Actual	in factors
ADLITTLE	MA47	0.020	0.010	0.030	771	771	619
	MA27	0.010	0.020	0.030	4356	12533	1303
AFIRO	MA47	0.010	0.000	0.010	180	180	180
	MA27	0.000	0.000	0.000	518	1297	283
ISRAEL	MA47	0.140	0.020	0.160	3129	3129	2961
	MA27	0.070	0.120	0.190	103528	333566	8247
CAPRI	MA47	0.080	0.020	0.100	2612	2612	2604
	MA27	0.050	0.140	0.190	134972	415996	11188
SHARE1B	MA47	0.070	0.030	0.100	4487	4482	1743
	MA27	0.020	0.040	0.060	32741	58287	3834
VTPBASE	MA47	0.040	0.020	0.060	1471	1471	1471
	MA27	0.030	0.060	0.090	16566	60992	3742
E226	MA47	0.110	0.030	0.140	3463	3463	3463
	MA27	0.080	0.210	0.290	189126	583512	12913
BEACONFD	MA47	0.150	0.030	0.180	3876	3876	3876
	MA27	0.130	0.160	0.290	152572	438060	10941
FFFFF800	MA47	0.290	0.060	0.350	7954	7954	7954
	MA27	0.380	1.030	1.410	1512220	4336104	44377

8 Conclusions

We have reviewed the use of augmented systems illustrating application areas in which they appear and commenting on a range of solution techniques used. We have described a general approach for their solution based on a direct method using 2×2 pivoting and have illustrated that it could prove useful in certain cases. Sometimes our new code yields a substantial improvement over an earlier code that did not respect the structure.

Acknowledgement
I would like to thank my colleagues at CERFACS and RAL for their assistance. Ali Bouaricha and João Patricio for helping obtain some of the numerical results and Nick Gould, John Reid, and Jennifer Scott for commenting on the manuscript. I also thank Åke Björck for discussions on Section 5.

References

[Anon, 1993] Anon. *Harwell Subroutine Library A Catalogue of Subroutines (Release 11).* Theoretical Studies Department, AEA Industrial Technology, 1993.

[Arioli *et al.*, 1989] M. Arioli, J. Demmel, and I. S. Duff. Solving sparse linear systems with sparse backward error. *SIAM J. Matrix Anal. Appl.*, 10:165–190, 1989.

[Arioli *et al.*, 1993] M. Arioli, T. F. Chan, I. S. Duff, N. I. M. Gould, and J. K. Reid. Computing a search direction for large-scale linearly-constrained nonlinear optimization calculations. Technical Report RAL 93-066, Central Computing Department, Rutherford Appleton Laboratory, 1993.

[Björck, 1990] Å. Björck. Least squares methods. In P. G. Ciarlet and J. L. Lions, editors, *Handbook of Numerical Analysis*, volume I: Finite Difference Methods-Solution of Equations in R^n. Elsevier/North Holland, Amsterdam, 1990.

[Björck, 1991] Å. Björck. Error analysis of least squares algorithms. In G. H. Golub and P. Van Dooren, editors, *Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms*, NATO ASI Series, pages 41–73, Berlin, 1991. Springer.

[Björck, 1992] Å. Björck. Pivoting and stability in the augmented system method. In Griffiths and Watson, editors, *Numerical Analysis Proceedings, Dundee 91*, pages 1–16, London, 1992. Longman.

[Bunch and Parlett, 1971] J. R. Bunch and B. N. Parlett. Direct methods for solving symmetric indefinite systems of linear systems. *SIAM J. Numer. Anal.*, 8:639–655, 1971.

[Bunch, 1971] J. R. Bunch. Analysis of the diagonal pivoting method. *SIAM J. Numer. Anal.*, 8:656–680, 1971.

[Curtis and Reid, 1972] A. R. Curtis and J. K. Reid. On the automatic scaling of matrices for Gaussian elimination. *J. Inst. Maths. Applics.*, 10:118–124, 1972.

[Duff and Reid, 1976] I. S. Duff and J. K. Reid. A comparison of some methods for the solution of sparse overdetermined systems of linear equations. *J. Inst. Maths. Applics.*, 17:267–280, 1976.

[Duff and Reid, 1982] I. S. Duff and J. K. Reid. MA27—a set of Fortran subroutines for solving sparse symmetric sets of linear equations. Technical Report R.10533, AERE, Harwell, England, 1982.

[Duff and Reid, 1983] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear systems. *ACM Trans. Math. Software*, 9:302–325, 1983.

[Duff and Reid, 1984] I. S. Duff and J. K. Reid. The multifrontal solution of unsymmetric sets of linear systems. *SIAM J. Scient. Statist. Comput.*, 5:663–641, 1984.

[Duff *et al.*, 1991] I. S. Duff, N. I. M. Gould, J. K. Reid, J. A. Scott, and K. Turner. Factorization of sparse symmetric indefinite matrices. *IMA Journal of Numerical Analysis*, 11:181–204, 1991.

[Duff *et al.*, 1992] I. S. Duff, R. G. Grimes, and J. G. Lewis. Users' guide for the Harwell-Boeing sparse matrix collection (Release I). Technical Report RAL 92-086, Rutherford Appleton Laboratory, 1992.

[Duff *et al.*, 1993] I. S. Duff, N. I. M. Gould, and J. Patricio. Experiments in scaling augmented systems. Technical Report (to appear), Rutherford Appleton Laboratory, 1993.

[Duff, 1981] I. S. Duff. Full matrix techniques in sparse Gaussian elimination. In G.A. Watson, editor, *Numerical Analysis Proceedings, Dundee1981*, Lecture Notes in Mathematics 912, pages 71–84, Berlin, 1981. Springer-Verlag.

[Duff, 1990] I. S. Duff. The solution of large-scale least-squares problems on supercomputers. *Annals of Operations Research*, 22:241–252, 1990.

[Elman and Golub, 1993] H.C. Elman and G.H. Golub. Inexact and preconditioned uzawa algorithms for saddle point problems. Technical Report CS-TR-3075, Compuetr Science Department, University of Maryland, 1993.

[Eskow and Schnabel, 1992] E. Eskow and R. B. Schnabel. A preconditioned iterative method for saddlepoint problems. *SIAM Journal on Matrix Analysis and Applications*, 13:887–904, 1992.

[Forsgren and Murray, 1993] A. Forsgren and W. Murray. Newton methods for large-scale linear equality-constrained minimization. *SIAM Journal on Matrix Analysis and Applications*, 14:560–587, 1993.

[Fortin and Glowinski, 1983] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems. Volume 15 of Studies in Mathematics and its Applications.* North-Holland, 1983.

[Fourer and Mehrota, 1992] R. Fourer and S. Mehrota. Solving symmetric indefinite systems in an interior-point method for linear programming. Technical Report Technical Report 92-01, Department of Industrial Engineering and Management Sciences, Northwestern University, 1992.

[Gay, 1985] D. M. Gay. Electronic mail distribution of linear programming test problems. Mathematical Programming Society. COAL Newsletter, 1985.

[Gill *et al.*, 1981] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, 1981.

[Gould, 1993] N. I. M. Gould. KKTSOL/MA49: A set of Fortran subroutines for solving sparse symmetric sets of linear equations which arise in constrained optimization calculations. Technical Report (to appear), CERFACS, Toulouse, France, 1993.

[Gulliksson and Wedin, 1992] M. Gulliksson and P.-Å. Wedin. Modifying the QR-decomposition to constrained and weighted linear least squares. *SIAM Journal on Matrix Analysis and Applications*, 13:1298–1313, 1992.

[Karmarkar, 1984] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[Kuhn and Tucker, 1951] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In J. Neyman, editor, *Proceedings of Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, California, 1951. University of California, Press.

[Liu, 1990] J. H. W. Liu. The role of elimination trees in sparse factorization. *SIAM J. Matrix Anal. Appl.*, 11:134–172, 1990.

[Ng *et al.*, 1993] E. Ng, B. Nitrosso, and B. Peyton. On the solution of Stokes' pressure system within N3S using supernodal Cholesky factorization. Technical Report (to appear), Electricité de France, 1993.

[Ramage and Wathen, 1993] A. Ramage and A.J Wathen. Iterative solution techniques for the Navier-Stokes equations. Technical Report AM-93-01, School of Mathematics, University of Bristol, 1993.

[Rusten and Winther, 1991] T. Rusten and R. Winther. A preconditioned iterative method for saddlepoint problems. *ACM Transactions on Mathematical Software*, 17:306–312, 1991.

[Stewart, 1989] G. W. Stewart. On scaled projections and pseudoinverses. *Linear Algebra and its Applications*, 112:189–193, 1989.

[Strang, 1986] G. Strang. *Introduction to Applied Mathematics.* Wellesley-Cambridge Press, 1986.

[Strang, 1988] G. Strang. A framework for equilibrium equations. *SIAM Review*, 30:283–297, 1988.

[Vavasis, 1993] S. A. Vavasis. Stable numerical algorithms for equilibrium systems. *SIAM Journal on Matrix Analysis and Applications*, 12:000–000, 1993.

[Wedin, 1985] P-Å. Wedin. Perturbation theory and condition numbers for generalized and constrained linear least squares problems. Technical Report UMINF 125.85, Inst Inf Processing, University of Umeå, 1985.