On Equilibration and Sparse Factorization of Matrices Arising in Finite Element Solutions of Partial Differential Equations

Valmor F. de Almeida* and Jeffrey J. Derby
Department of Chemical Engineering and Materials Science,
Army High Performance Computing Research Center and
Minnesota Supercomputer Institute
University of Minnesota
Minneapolis, MN 55455-0132, U.S.A.

Andrew M. Chapman
NEC Systems Inc.,
4200 Research Forest #400
The Woodlands, TX 77381, U.S.A.

Received 2 October 1998; reviewed 3 February 1999; revised 5 August 1999; accepted 30 August 1999.

Investigations of scaling and equilibration of general matrices have been traditionally aimed at the effects on the stability and accuracy of LU factorizations—the so-called scaling problem. Notably, Skeel (1979) concludes that no systematic scaling procedure can be concocted for general matrices exempt from the danger of disastrous effects. Other researchers suggest that scaling procedures are not beneficial and should be abandoned altogether. Stability and accuracy issues notwithstanding, we show that this unglamorous technique has a profound impact on the sparsity of the resulting LU factors. In the modern era of fast computing, equilibration can play a key role in constructing incomplete sparse factorizations to *solve a problem unstably, but quickly and iteratively.* This article presents practical evidence, on the basis of sparsity, that scaling is an indispensable companion for sparse factorization algorithms when applied to realistic problems of industrial interest. In light of our findings, we conclude that equilibration with the ∞-norm is superior than equilibration with the 2-norm.

Keywords: Matrix equilibration, scaling, incomplete LU factorization, pivoting, preconditioning, sparse finite-element matrices, incompressible fluids, Navier-Stokes equations

* Corresponding author currently in the Chemical Technology Division at the Oak Ridge National Laboratory; dealmeidav@ornl.gov, http://www.msi.umn.edu/~dalmeida.

I. INTRODUCTION

The **LU** factorization of a large and very sparse $n \times n$ matrix **A**, where **L** is an unit lower triangular matrix and **U** is an upper triangular matrix, is one of the most widely used algorithms in scientific computing and computational engineering. It is the cornerstone of direct linear solvers, playing a central role in deriving robust preconditioners for iterative linear solvers (CHAPMAN *et al.*, 1996; HACKBUSH AND WITTUM, 1993). Consequently, there has been a vast amount of work on algorithms to perform such factorizations efficiently on digital computers. In this manuscript, we revisit some rather old ideas on matrix equilibration and present some convincing practical evidence that such ideas are extremely useful for solving modern problems arising from the modeling of materials processing systems.

Matrix equilibration comprises a number of techniques to scale matrix coefficients in some manner to improve the quality of a factorization. The main idea behind such scaling has typically been to "optimally" obtain one of the $(n!)^2$ possible non-unique LU factorizations associated with permutations of rows and columns for different choices of pivots. It has been long recognized (VAN DER SLUIS, 1970) that almost any pivotal sequence can be achieved by appropriately scaling the rows and the columns of a matrix. This fact spurred extensive theoretical investigation on how a matrix should be scaled to force a particular pivoting strategy to extract a stable and accurate factorization—the so-called scaling problem (BAUER, 1963; MCCARTHY AND STRANG, 1973; VAN DER SLUIS, 1969).

While much effort has been directed toward the scaling problem, there have been many conflicting results, and no general consensus has been obtained. Some of the early work focused on finding optimal scalings to minimize the classical condition number of the scaled matrix based on the argument that the quality of numerical computations generally improves if the condition number of the matrix is reduced. Results along this avenue conflict with a theorem of BAUER (1963), which states that in the absence of row or column interchanges, scaling with powers of the machine base does not lead to a better computed solution, but rather, to the same solution. In fact, contrived examples show that scaling may lead to less accurate solutions.

SKEEL (1979) investigated the effect of scaling a linear system of equations—rather than the matrix of coefficients—on the stability of Gauss elimination with partial pivoting. He concluded that the optimal way to scale for stability depended on the solution, and that the problem of scaling for accuracy had many solutions, one of which depended solely on the matrix of coefficients. He considered his findings unpractical regarding stability because he needed information on data that was unavailable. Finally he conjectured that there was no satisfactory algorithm for scaling a general matrix.

POOLE AND NEAL (1992) pointed out that reducing the condition number of a matrix via scaling is not relevant. Instead, they questioned how scaling improves the selection of pivots. They concluded, on the basis of a geometric analysis of the Gauss elimination method, that scaling or equilibration may be unnecessary in the presence of an adequate pivoting strategy. They extended their analysis and proposed the rook's pivoting strat-

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

egy as a superior alternative to the well-known partial pivoting and complete pivoting strategies.

Regrettably, it is not yet fully understood how scaling and equilibration really affect the choice of pivots for a general pivoting strategy. Typical practice is to either scale matrices on a problem-by-problem basis or to ignore scaling altogther. General scaling strategies are unreliable and not advisable due to potential disastrous effects on the accuracy and stability of factorizations (Dahlquist and Björck, 1974; Golub and Van Loan, 1989). Apparently, theoretical investigation on scaling has been abandoned. To the best of our knowledge, it has been considered an *ad hoc* technique seldom reported, but nevertheless, silently used. The question of whether or not to scale to improve stability and accuracy is a subject of dispute; however, we report here on a very practical benefit from matrix equilibration, namely the favorable effects of scaling on the resulting sparsity of the factorization.

Prohibitive memory requirements hamper the computation of accurate factorizations of large and sparse matrices because the factors are, in general, much denser than the original matrix. Therefore, a common practice is to compute a less accurate factorization with less fill-in and to resort to an iterative refinement technique to improve the accuracy of the factorization *a posteriori*. If the less-accurate factorization departs significantly from an accurate factorization, iterative refinement will certainly diverge. However, the factorization can still be useful as a preconditioner of an iterative linear solver, in which case the factorization is often referred to as incomplete.

The foregoing prompts the question whether scaling and equilibration have an impact on the sparsity of the factorization as well as on stability and accuracy. An inaccurate, but significantly sparser factorization is still computationally attractive because it is likely to be useful when embedded into other algorithms. For example, nonlinear solvers such as the modified Newton-like methods and Newton-Krylov methods are commonly used in the solution of nonlinear partial differential equations (Kelley, 1995).

We observe that in practice, equilibration has a profound effect on the sparsity of the **LU** factors of a class of matrices arising in important models of industrial applications. Equilibration is not only beneficial, but also indispensable for the analysis of these industrially relevant models of materials processing. Without some form of equilibration, excessive permutation of rows and columns occurs in the course of pivoting during the sparse factorization. Consequently, a large amount of fill-in quickly exhausts the available computational memory.

Section II. describes three-dimensional mathematical models of materials processing systems with representative computed solutions. The solution method is briefly outlined to clarify the origin and structure of the matrices obtained. In section III., a sparse LU factorization algorithm for finite-element matrices is presented. This algorithm computes stable and unstable factorizations employed respectively in a modified Newton-like method and in a Newton-Krylov method for solving parametrized nonlinear systems of equations. Two forms of equilibration are discussed in section IV., and the comparative results and conclusions are reported in sections V.–VI..

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

II. MODELS OF MATERIALS PROCESSING SYSTEMS

Three continuum models of important industrial materials processing systems are described next. The models contain a set of coupled multi-field partial differential equations in 3-D space representative of challenging *realistic* problems.

A. MCZ model

The magnetic Czochralski crystal growth method (fig. 1a) is widely used in the production of large diameter, single-crystal silicon wafers for very large and ultra large scale integrated (VLSI/ULSI) electronic devices (LIN AND BENSON, 1987).

Our associated 3-D model for the melt flow in the Czochralski process (ROJO AND DERBY, 1999) comprizes the Navier-Stokes equations coupled with a reduced form of Maxwell's equations of electromagnetics and the energy balance equation (CHANDRASEKHAR, 1981). Denoting Ω the region of three-dimensional space occupied by the silicon melt, the governing equations of steady-state motion of a Newtonian, incompressible and electrically conductive fluid subjected to Fourier's law of heat conduction and Ohm's law of electric conduction are

$$\operatorname{div}_{\boldsymbol{x}}(\boldsymbol{v}\otimes\boldsymbol{v}) = \operatorname{div}_{\boldsymbol{x}}\boldsymbol{T} - Gr\,\theta\boldsymbol{g} + Ha^2\,\boldsymbol{j}(\boldsymbol{x})\times\boldsymbol{b} \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.1a}$$

$$\operatorname{div}_{\boldsymbol{x}}\boldsymbol{v} = 0 \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.1b}$$

$$Pr\,\operatorname{div}_{\boldsymbol{x}}(\theta\boldsymbol{v}) = \operatorname{div}_{\boldsymbol{x}}(\nabla\theta) \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.1c}$$

$$\operatorname{div}_{\boldsymbol{x}}\boldsymbol{j} = 0 \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.1d}$$

with constitutive equations of stress and electric current respectively

$$\boldsymbol{T}(\boldsymbol{x}) := -p(\boldsymbol{x})\boldsymbol{I} + \nabla_{\boldsymbol{x}}\boldsymbol{v} + \nabla_{\boldsymbol{x}}\boldsymbol{v}^{\mathsf{T}} \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.1e}$$

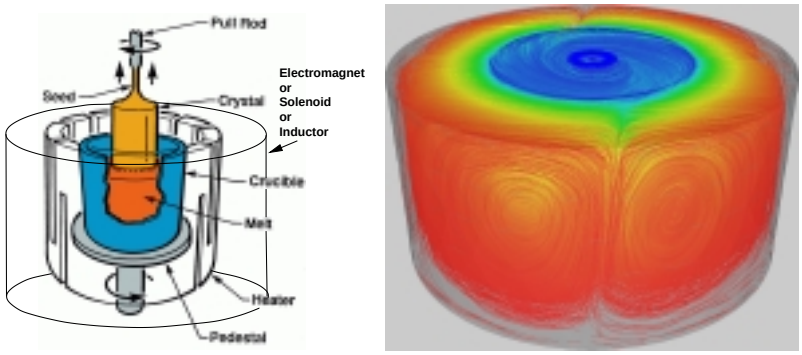
$$\boldsymbol{j}(\boldsymbol{x}) := -\nabla_{\boldsymbol{x}}\Phi + \boldsymbol{v}(\boldsymbol{x})\times\boldsymbol{b} \qquad \forall\ \boldsymbol{x}\in\Omega. \tag{2.1f}$$

Here, Gr is the Grashof number, ratio of buoyancy force to viscous force; Ha is the Hartmann number, square-root of the ratio of electric force to viscous force; and Pr is the Prandtl number, ratio of viscous diffusion to heat diffusion. The velocity, stress, temperature and electric current fields are denoted respectively $\boldsymbol{v}$, $\boldsymbol{T}$, θ and $\boldsymbol{j}$, while the gravitational acceleration vector is denoted $\boldsymbol{g}$, the transverse magnetic induction vector, $\boldsymbol{b}$; the pressure, p; and the electric potential, Φ.

Two extra physical parameters appear in the boundary conditions, namely, Re_x—the crystal rotational Reynolds number that represents the ratio of inertial to viscous force, and the radiation parameter $\mathscr{Q}$—the ratio of melt/ambient meniscus radiation to conduction. In figure 1b the ratio of crystal to crucible radius and the ratio of height to crucible diameter are equal to 0.5.

The system (2.1) with accompanying boundary conditions is nonlinear, coupled, constrained and parametrized by Gr, Ha, Pr, Re_x and $\mathscr{Q}$. Systematic study of this model by means of constructing a family of approximate solutions calls for state-of-the-art algorithms. Figure 1b shows a particular solution of this family.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.



(a) (b)

FIG. 1. MCZ simulated bulk flow for $Gr = 10^5$, $Ha = 45$, $Pr = 10^{-2}$, $Re_x = 150$ and $\mathscr{Q} = 2$: (a) Apparatus; (b) Computed fluid particle traces colored by temperature.

B. KTP model

Single crystals of nonlinear optical materials such as potassium titanyl phosphate are successfully grown by the solution crystal growth method. These crystals' properties are suitable for various electro-optics applications such as laser systems (BIERLEIN AND VANHERZEELE, 1989). There are outstanding continuum transport issues in the KTP system currently in study via the model illustrated in figure 2a (VARTAK *et al.*, 1998).

In the KTP model the motion of the solution can be studied independently from mass transfer by means of the Navier-Stokes equations for a Newtonian, incompressible mixture

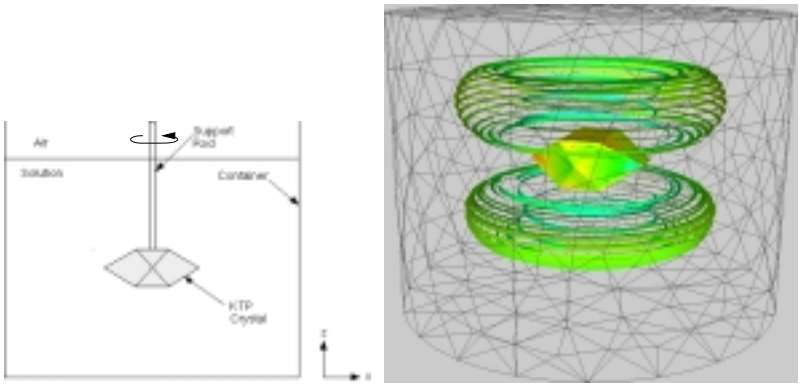
$$Re\left(\mathrm{div}_{\boldsymbol{x}}(\boldsymbol{v}\otimes\boldsymbol{v}) + 2\boldsymbol{\omega}\times\boldsymbol{v} + \boldsymbol{\omega}\times(\boldsymbol{\omega}\times\boldsymbol{x})\right) = \mathrm{div}_{\boldsymbol{x}}\,\boldsymbol{T} \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.2a}$$

$$\mathrm{div}_{\boldsymbol{x}}\,\boldsymbol{v} = 0 \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.2b}$$

where a rotating frame of reference is selected with angular velocity $\boldsymbol{\omega}$ equal to the angular velocity of the rotating crystal. The three-dimensional domain occupied by the mixture is denoted Ω and Re is the Reynolds number. Here $\boldsymbol{v}$ represents the mass average velocity field of the mixture and and $\boldsymbol{T}$, the associated stress tensor field.

By virtue of a rotating reference frame, (2.2) can be solved conveniently on a fixed domain (fig. 2b). Boundary conditions of no-slip are applied everywhere, except on the mixture/ambient interface, where the shear stress is assumed to be zero. For an observer on the rotating frame of reference, the mixture velocity on the crystal is zero, and the container's wall and bottom surfaces rotate with the same magnitude $\|\boldsymbol{\omega}\|$, but in the opposite direction of $\boldsymbol{\omega}$. Figure 2b illustrates a particular solution of (2.2) under industrially operating conditions.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.



(a) (b)

FIG. 2. KTP simulated bulk flow for $Re = 300$: (a) Bulk-flow model (VARTAK *et al.*, 1998); (b) Computed fluid particle traces colored by speed showing the secondary flow above and below the crystal.

C. MCVD model

Low-loss optical fibers are manufactured by a modified chemical deposition process illustrated in figure 3a. A mixture of gases flows into a rotating tube heated by a traversing torch. Because the mixture may be safely treated as an incompressible fluid, the governing equations for this model are similar to the MCZ model, except that they do not take the magnetic field into account. Thus,

$$Re\ \mathrm{div}_{\boldsymbol{x}}(\boldsymbol{v}\otimes\boldsymbol{v}) = \mathrm{div}_{\boldsymbol{x}}\,\boldsymbol{T} - \frac{Gr}{Re}\,\theta\boldsymbol{g} \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.3a}$$

$$\mathrm{div}_{\boldsymbol{x}}\,\boldsymbol{v} = 0 \qquad \forall\ \boldsymbol{x}\in\Omega, \tag{2.3b}$$

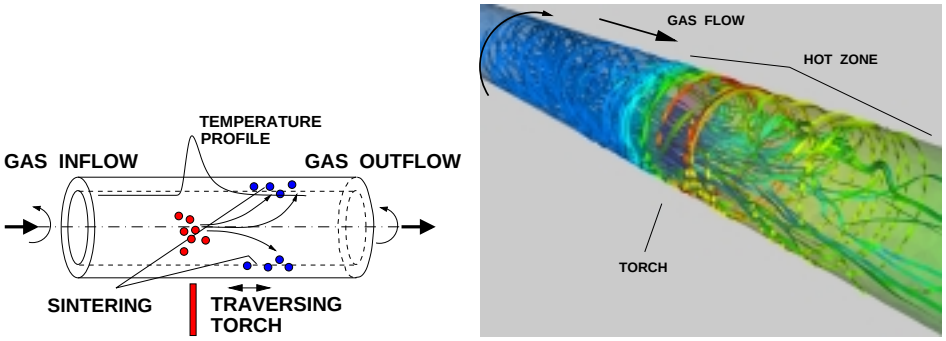
$$Pr\,\mathrm{div}_{\boldsymbol{x}}(\theta\boldsymbol{v}) = \mathrm{div}_{\boldsymbol{x}}(\nabla\,\theta) \qquad \forall\ \boldsymbol{x}\in\Omega. \tag{2.3c}$$

The physical parameters and the fields have the same meaning as in the MCZ governing equations (2.1). The Reynolds number Re is the ratio of inertia of the incoming gas flow to viscous force. Evidently the essential difference in the model, when compared to the MCZ model, is not only the value of the parameters, but also the boundary conditions, notably inflow and outflow. When the no-slip boundary condition is applied to the wall of the rotating tube, a rotational Reynolds number, Re_r, emerges. Figure 3b presents evidence of significant buoyancy disturbance in the gas flow at the hot zone for industrially operating values of physical parameters.

D. Origin and structure of matrices

The methods used for constructing parametrized solution curves for the aforementined models are:

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.



(a) (b)

FIG. 3. MCVD simulated flow $Gr = 10^6$, $Re = 1\,400$, $Re_r = 50$, Pr $= 0.72$: (a) Fixed torch model; (b) Fluid particle traces colored by temperature showing the gas flow structure at the hot zone.

1. Weak reformulation of the mathematical problem and linearization (BREZZI AND FORTIN, 1991; GUNZBURGER, 1989)
2. Fixed-point iteration with high convergence rate—typically a Newton-like method (KELLEY, 1995)
3. Finite element approximation in stable functional spaces—second-order accurate methods were preferred (DE ALMEIDA AND DERBY, 1999b; GIRAULT AND RAVIART, 1986)
4. Solution path following methods—a zeroth-order method was employed, but first-order methods are recommended (ALLGOWER AND GEORG, 1990; DE ALMEIDA AND DERBY, 1999a; KELLER, 1992)
5. Large and sparse linear system solvers—iterative methods with factorization-based preconditioning (SAAD, 1996; SMITH *et al.*, 1996) or sparse direct linear solvers (sec. III.)

A sequence of large sparse linear system of equations results from the above. The corresponding matrices of coefficients possess no particular *nice* properties. They are largely nonsymmetric, indefinite and nondiagonal dominant because the discretized operators are nonselfadjoint, the fields are coupled strongly through large values of physical parameters, and physical constraints are enforced on selected fields.

Preconditioned iterative techniques play an important role in solving large and sparse systems of linear equations. Robustness depends critically on the preconditioner employed. In fact, the choice of the preconditioner is far more important than the choice of the accelerator. For realistic problems, factorization-based preconditioners, when judiciously applied, are often the most reliable preconditioners; evidence of the robustness of this approach is presented in the ensuing sections.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

III. SPARSE (INCOMPLETE) FACTORIZATION OF MATRICES

The unique factorization $\mathbf{P}\,\mathbf{A}\,\mathbf{Q} = \mathbf{L}\,\mathbf{U}$ of a large and sparse $n \times n$ matrix $\mathbf{A}$ into its $\mathbf{L}$ and $\mathbf{U}$ factors—where $\mathbf{L}$ is unit lower triangular; $\mathbf{U}$ is upper triangular; and $\mathbf{P}$ and $\mathbf{Q}$ are rows and columns permutation matrices—is the cornerstone of direct methods for solving systems of linear equations, and it plays a key role in deriving preconditioning for iterative methods. We follow a systematic approach to construct *robust* preconditioners by modifying a sparse factorization algorithm that *ignores* judiciously chosen *small* numbers. Although there exists a number of known drawbacks in this approach when applied to *general* matrices, it can be efficiently applied to matrices embedded in other iterative algorithms (DE ALMEIDA AND DERBY, 1999a).

Factorization of large and sparse matrices is typically obtained by compressing its rows and columns into a dense matrix $\mathbf{F}$ and, simultaneously, computing the corresponding rows and columns of the $\mathbf{L}$ and $\mathbf{U}$ factors by elimination in $\mathbf{F}$ (DUFF *et al.*, 1989). Eliminated rows and columns are deleted from $\mathbf{F}$ in order to keep the matrix dense and small. It is known that high computational throughput is obtained on a wide range of computer architectures by applying standard level 2 and/or 3 BLAS (Basic Linear Algebra Subroutines) operations on $\mathbf{F}$ (DONGARRA *et al.*, 1991). Details of the factorization method follow next.

A. Finite element matrices

Large sparse finite element matrices $\mathbf{A}$ naturally arise as an *assembling sum* of N dense element $m \times m$ matrices $\mathbf{A}_e$ (not necessarily all the same size)

$$\mathbf{A} \leftarrow \sum_{e}^{N} \mathbf{A}_e. \tag{3.1}$$

Typically, N is much smaller than the number of rows and columns n of $\mathbf{A}$ when high-order finite elements are used. Table I shows the values of N, n, and m for the system of equations in the models described in section II..

TABLE I. Number of element matrices and dimensions of matrices for the model problems. Not all element matrices have the same size for a given model; m is the maximum value

Model	N	n	m
MCZ	2 564	74 531	79
KTP	4 332	77 119	49
MCVD	9 921	242 152	64

When $\mathbf{A}$ is expressed as in (3.1), row and column compression is obtained by mapping or assembling each of the element matrices $\mathbf{A}_e$ into $\mathbf{F}$ in a similar manner. The assembling of $\mathbf{A}_e$ and the factorization of $\mathbf{A}$ are done simultaneously as explained below. In the finite-element literature, $\mathbf{F}$ is known as the frontal matrix, a designation we adopt acquiescently.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

Although multiple frontal matrices can be used simultaneously, this work is concerned with the use of one single frontal matrix.

Let $\mathbf{F}_e$ be the frontal matrix obtained by assembling e element matrices $\mathbf{F}_e \leftarrow \sum_1^e \mathbf{A}_e$. A suitable permutation of the frontal matrix results in the following block structure

$$\begin{pmatrix} \mathbf{F}_{1,1} & \mathbf{F}_{1,2} \\ \mathbf{F}_{2,1} & \mathbf{F}_{2,2} \end{pmatrix}. \tag{3.2}$$

Rows in $\mathbf{F}_{1,1}$ and $\mathbf{F}_{1,2}$ represent fully summed compressed rows of the original matrix $\mathbf{A}$, and columns in $\mathbf{F}_{1,1}$ and $\mathbf{F}_{2,1}$ represent fully summed compressed columns. These rows and columns are fully summed in the sense that no further assemblies of new element matrices will contribute to existing entries in the fully summed rows and columns. As new element matrices are assembled, the blocks increase in size, but the value of the existing fully summed entries remains unchanged.

The factorization of $\mathbf{A}$ can start with the selection of pivots within $\mathbf{F}_{1,1}$ as soon as this block has nonzero size. For every selected pivot, one row of the $\mathbf{U}$ factor and one column of the $\mathbf{L}$ factor are obtained by eliminating each corresponding pivotal row and column. That is, after the selection of the k-th pivot $\alpha^{(k)}$ and the appropriate permutations of the pivotal row and pivotal column, the frontal matrix $\mathbf{F}_e^{(k)}$ can be organized in the following form

$$\begin{pmatrix} \alpha^{(k)} & \mathbf{r}^{(k)\mathsf{T}} \\ \mathbf{c}^{(k)} & \mathbf{D}^{(k)} \end{pmatrix}. \tag{3.3}$$

Elimination is performed by replacing the frontal matrix $\mathbf{F}_e^{(k)}$ with the Schur complement

$$\mathbf{F}_e^{(k+1)} \leftarrow \mathbf{D}^{(k)} - \frac{\mathbf{c}^{(k)}\mathbf{r}^{(k)\mathsf{T}}}{\alpha^{(k)}}. \tag{3.4}$$

Consequently, the dimension of $\mathbf{F}_e^{(k)}$ is reduced by one. In practice, the assembling and elimination operations take place in the array space created to store the frontal matrix.

A permutation of the row $(\alpha^{(k)}, \mathbf{r}^{(k)\mathsf{T}})$ must be stored in the $\mathbf{U}$ factor before the pivotal row is deleted. Similarly, the vector of multipliers must be stored in the $\mathbf{L}$ factor by permuting $(1, \mathbf{c}^{(k)}/\alpha^{(k)})^{\mathsf{T}}$ accordingly. These permutations are best carried out *in place* at the end of the factorization when the permutation matrices $\mathbf{P}$ and $\mathbf{Q}$ are available. Therefore we store, temporarily, the pivotal row and column to the corresponding factors without permuting them. Since the pivotal row and column are compressed vectors, $\mathbf{L}$ and $\mathbf{U}$ can be efficiently stored with the compressed storage column format (CSC) and the compressed storage row format (CSR), respectively (SAAD, 1996). A variety of matrix operations can be readily performed on sparse matrices stored in standard formats. Notably, the factors can finally be permuted by applying the row and column permutation matrices, and standard sparse triangular substitution can be employed to solve or precondition a system of equations.

Operation (3.4) requires a single rank-one update thus, a level 2 BLAS can be used. The updating process (3.4) is recursively applied to the frontal matrix (3.2) by selecting

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

pivots from the leading block until no pivots can be found (selection of pivots is explained below). At this point, the assembling of the next element matrix is resumed and the updating process (3.4) is restarted. The process will naturally stop when $e = N$ and $k = n$, that is, when all element matrices are assembled and n pivots are found.

For large systems of equations, the performance of the overall factorization is dominated by a level 2 BLAS operation. In vector machines with shared memory, it is possible to reach 89% of the theoretical performance (DE ALMEIDA AND DERBY, 1999a). In cache-based computer architectures, the method performs poorly because the predominant expense is the data movement of $O(\bar{m}_{\mathbf{F}}^2)$ numbers from the computer's main memory to cache and vice-versa. Note that (3.4) operates, in average, on a $\bar{m}_{\mathbf{F}} \times \bar{m}_{\mathbf{F}}$ frontal matrix (see table II for values of $\bar{m}_{\mathbf{F}}$ obtained from the models). Therefore, $O(2\,\bar{m}_{\mathbf{F}}^2)$ operations are performed on the data stored in cache per elimination.

TABLE II. Typical dimensions of frontal matrices ($m_{\mathbf{F}} \times m_{\mathbf{F}}$) $\mathbf{F}$ for the model problems. Average size $\bar{m}_{\mathbf{F}}$ and maximum size.

Model	$\bar{m}_{\mathbf{F}}$	max.
MCZ	2 056	2 898
KTP	1 898	2 656
MCVD	366	971

DUFF AND SCOTT (1993) suggest an alternative implementation of sparse factorization rich in level 3 BLAS operations. The cost of moving the data into cache is at least the same $O(\bar{m}_{\mathbf{F}}^2)$, but the number of operations performed on the data in cash increases to $O(\bar{m}_{\mathbf{F}}^3)$. This seems to be the preferred implementation in cash-based computers.

It is typical for element matrices $\mathbf{A}_e$ derived from higher-order finite element methods, to possess fully summed rows and columns. These rows and columns are associated to interpolating polynomials, whose support are contained entirely either in one or a few finite elements. Therefore, it is advantageous to eliminate these equations from the element matrix before assembling them into the frontal matrix. This elimination operation is referred to as static condensation and is carried out on a condensation matrix analogous to the frontal matrix. Details of the static condensation employed in this work are reported in DE ALMEIDA *et al.* (1998).

B. Fill-in and drop strategy

Drop strategies are often used in conjunction with a factorization to reduce storage requirements. Although $\mathbf{A}$ is sparse, the factors $\mathbf{LU}$ are not. The nonzero entries in $\mathbf{LU}$ that correspond to zero entries in $\mathbf{A}$ are called fill-in. They are created during the update (3.4). A limited amount of fill-in may be discarded with little or no impact on the stability of the factorization to obtain factors that are as sparse as possible. When the factorization is sought to be used as a preconditioner, large amounts of judiciously chosen fill-in may be discarded before it becomes excessively unstable. Controlling fill-in

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

is necessary for two reasons: first, computer memory is usually unavailable for storing all the fill-in (often the bottleneck); second, operation count obviously grows with the size of the stored data.

We follow a dual threshold drop strategy (SAAD, 1996). Entries in the factors are dropped first, accordingly to their relative magnitudes, and secondly, based on a maximum number of retainable entries. Two specified parameters, namely, *tol* and *lfil* control the amount of discarded entries as follows. Values in the pivotal row $\mathbf{r}$ and column $\mathbf{c}$

$$r_i < \frac{tol}{\sqrt{l}} \left\|\mathbf{r}\right\|_2 \quad \text{and} \quad c_i < \frac{tol}{\sqrt{l}} \left\|\mathbf{c}\right\|_2, \tag{3.5}$$

respectively, are dropped relatively to the root-mean-square value where l is the dimension of the vectors. After numerical dropping is enforced, if the number of surviving entries is larger than *lfil*, the largest *lfil* entries, in absolute value, are retained; the remaining entries are simply discarded. The second dropping strategy must be used with caution. Because it neglects numerical values arbitrarily, a disastrous effect on the stability of the factorization may result. Our suggestion is to avoid using the *lfil* parameter whenever possible. Practical values of *tol* and *lfil* are presented in the results section.

C. Pivoting: stability versus fill-in

Pivoting is mandatory to obtain a backward stable factorization of matrices in realistic applications. Because the factorization and the assembling of element matrices into the frontal matrix occur simultaneously, neither partial pivoting nor complete pivoting are feasibly implemented without increasing the frontal matrix to a prohibitive size. Since the number of entries per column and rows, respectively, of the $\mathbf{L}$ and $\mathbf{U}$ factors depend on the size of the frontal matrix, it is crucial that $\mathbf{F}$ be as small as possible. The threshold pivoting method of DUFF *et al.* (1989) is often employed as an alternative to partial and complete pivoting methods. In the threshold pivoting method, the fully summed columns $\mathbf{F}_{1,1}$ and $\mathbf{F}_{2,1}$ are searched in the increasing order of their global indices in $\mathbf{A}$ for a pivot. This search involves sorting the fully summed columns *and* fully summed rows in $\mathbf{F}_{1,1}$ and $\mathbf{F}_{2,1}$. The resulting pivoting strategy will be biased towards the initial ordering of rows and columns of the original matrix presumed reordered *a priori* for small fill-in. For a given fully summed column c, the element $a_{r,c}$ of $\mathbf{F}_{1,1}$ with smallest index r is selected as a pivot if it satisfies the criterion

$$\left|a_{r,c}\right| \geq u \max \left|a_{i,c}\right| \quad \forall\, a_{i,c} \in \mathbf{F}_{1,1} \text{ and } \mathbf{F}_{2,1}, \tag{3.6}$$

where $0 < u \leq 1$ is a threshold (stability) parameter. Note that the search for the maximum value extends to the block $\mathbf{F}_{2,1}$ since the entries in this block are fully summed. However, these entries are not pivot candidates because the corresponding rows in $\mathbf{F}_{2,2}$ are not fully summed. When $u < 1$ and (3.6) is satisfied, a pivot with smaller magnitude than the pivot obtained by partial pivoting will be selected and the resulting vector of multipliers will store values with magnitude at most $1/u$ in the $\mathbf{L}$ factor. Therefore, u has a direct impact on the magnitude of the entries in $\mathbf{U}$. If u is chosen too small, unacceptable growth can occur and the resulting factorization can be too unstable. If

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

$u = 1$, then threshold pivoting will be identical to partial pivoting modulo columns permutations.

The criterion (3.6) attempts to create more pivotal options to preserve the sparsity of the resulting $\mathbf{L}$ and $\mathbf{U}$ factors by allowing a controlled growth of entries in $\mathbf{U}$. Hopefully the small entries in $\mathbf{U}$ will grow without affecting the stability of the resulting factorization adversely.

Column permutations are often used in combination with $u < 1$. Once the threshold criterion fails for a particular column, a second column is tested and so forth. If an acceptable pivot cannot be found within all column candidates, an extra element matrix is assembled into the frontal matrix so that additional pivot candidates become available. Here we chose to search as many fully summed columns as possible before assembling an extra element matrix. This strategy should only be used when valid pivots can be found by searching a small number of columns, typically 10–20 when n is large. Otherwise, a limited number of column searches must be imposed so searching for a pivot does not become extremely time consuming. Limiting the number of column searches is a trade-off between computer memory allocated for the frontal matrix and time spent in pivoting. Algorithm 3.1 summarizes the factorization method described so far (see DE ALMEIDA *et al.*, 1998, for a modified version including static condensation).

Several complementary strategies for controlling fill-in can be added to threshold pivoting. DUFF *et al.* (1989) suggest, on a practical basis, the Markowitz algorithm. We find that in most practical cases of the uni-frontal factorization scheme, sparsity is best served by appropriate matrix reordering and equilibration (sec IV.). We employ the wavefront reduction method described in (SLOAN, 1989) to reorder the graph associated with the finite element mesh. The parameters w_1 and w_2, present in the function that assigns labelling priorities to the vertices of the graph, were varied systematically until the frontwidth of the reordered graph was the smallest found. Typically we found $w_1 = 1$ to 2 and $w_1 = 7$ to 11.

Algorithm 3.1 Level 2 BLAS sparse LU factorization.

Select u, tol and lfil
Set $k := 0$
For $e = 1, \ldots, N$ *Do:*
  Set $flag := true$
  Assemble $\mathbf{F}_e^{(k)} \leftarrow \mathbf{A}_e$
  While $flag = true$ *Do:*
    Search for pivot $\alpha^{(k+1)}$ *in* $\mathbf{F}_{1,1}$ *satisfying* (3.6)
    If pivot search fails then
      Set $flag := false$
    Else
      Compute $\mathbf{F}_e^{(k+1)}$ *with* (3.4)
      Apply dual threshold dropping (3.5)
      Store $\mathbf{r}$ *in* $\mathbf{U}$ *(CSR) and* $\mathbf{c}/\alpha^{(k+1)}$ *in* $\mathbf{L}$ *(CSC)*
      Set $k := k + 1$

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

EndIf
EndWhileDo
EndDo
Permute $\mathbf{L}$ *and* $\mathbf{U}$ *in place.*

D. Accuracy and stability issues

Triangular substitution is the operation that leads to the solution of a system of linear equations whose matrix of coefficient is triangular. It is a fundamental backward stable operation. In other words, the computed solution is the exact solution of a nearby system—a standard result from backward error analysis (GOLUB AND VAN LOAN, 1989). Although stability does not imply accuracy, results of forward error analysis indicate that in practice, even when triangular factors are ill-conditioned, computed solutions are far more accurate than standard error bounds predict (HIGHAM, 1989).

When constructing incomplete factors, it is instructive to monitor indicators of the factorization's stability and of the computed solution's accuracy as well. A stable factorization with excessive dropping may result in ill-conditioned factors which will impact accuracy adversely. Usually only an indication of whether accuracy and/or stability have been seriously compromised by dropping and/or pivoting strategies is needed. Stability and accuracy can be estimated inexpensively by the following indicators

1. $\left\|\mathbf{L}\right\|, \left\|\mathbf{U}\right\|,$
2. $\left\|\mathbf{L}^{-1}\right\|, \left\|\mathbf{U}^{-1}\right\|,$
3. $1/\min_k \left|\alpha^{(k)}\right|,$ and
4. $\left\|(\mathbf{LU})^{-1}\mathbf{A} - \mathbf{I}\right\|,$

where only a bound for the matrix norms may be computed. In the results presented in section V., we used the infinity norm. A lower bound for the infinity norm of a general matrix $\mathbf{M}$ is simply $\left\|\mathbf{Me}\right\|_\infty$, where $\mathbf{e}$ is the vector with elements $e_i = 1 \; \forall \; i$. This matrix norm estimate is easily evaluated when computing the norms in the indicators (2) and (4) because the inverse of the matrices is never required explicitly.

Indicators (1) appear in standard formulae of backward error analysis of LU factorizations (GOLUB AND VAN LOAN, 1989) where $\left\|\mathbf{L}\right\| \left\|\mathbf{U}\right\| = O(\left\|\mathbf{A}\right\|)$ implies a backward stable factorization. In practice, we look for unacceptable growth of the entries in $\mathbf{U}$, observing the magnitude of the largest entry. As we mentioned earlier in section C., the value of the largest entry in $\mathbf{L}$ is exactly $1/u$. Indicators (2), in conjunction with indicators (1), measure the forward error (accuracy) of the triangular solutions. Recall that the condition number of the $\mathbf{LU}$ factors is the product of the corresponding norms in (2) and (1); therefore, an estimate of how ill-conditioned the computed factors are is readily available.

In view of (3.4) the value of indicator (3), that is, the reciprocal of the smallest pivot, indicates how large the entries in $\mathbf{U}$ become during the factorization process. Hence, small pivots will eventually cause large $\left\|\mathbf{U}\right\|$, which brings forth instability.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

The indicator (4), estimated as $\left\| \left((\mathbf{LU})^{-1} \mathbf{A} - \mathbf{I} \right) \mathbf{e} \right\|_\infty$, is a measure of the forward error of the computed solution of the system $\mathbf{LU}\,\mathbf{x} = \mathbf{A}\,\mathbf{e}$. Thus, an accurate factorization yields $\mathbf{x}$ close to $\mathbf{e}$ and a small value of indicator (4).

Values of indicators (1)–(4) are shown in our experiments in section V. as complementary data. The accuracy and/or stability of a factorization is ultimately assessed by the overall performance of the particular algorithm in which the factorization is embedded. For example, ILU preconditioned iterative linear solvers or Newton-like nonlinear solvers. The indicators (1)–(4) are not sufficient to predict the performance of a factorization, but they are useful in identifying the causes of the factorization's failure.

E. Incomplete factorization for preconditioners

Incomplete factorizations can be derived from the algorithm 3.1 by simply enforcing aggressive dropping and pivoting when choosing the parameters u, *tol* and *lfil*, that is, small values of u and *lfil*, and large values of *tol*. Since this simple derivation would still require $O(n\,\bar{m}_{\mathbf{F}}^2)$ operations, the savings in operation count will be slight when compared with strict choices for the same parameters, that is, large values of u and *lfil*, and small values of *tol*.

As a stand-alone algorithm for deriving preconditioners and factorizations for general-purpose linear solvers, algorithm 3.1 may perform poorly due to a large floating-point operation count. Nevertheless for applications that can amortize the cost of a factorization, the algorithm is valuable because of its intrinsic robustness. For instance, the cost of factorizations embedded into algorithms designed for solving parametrized nonlinear systems of equations can be dramatically reduced in practice by recycling an existing factorization (DE ALMEIDA AND DERBY, 1999a).

In the results section, we expose practical values for the parameters u, *tol* and *lfil* that generate incomplete LU robust preconditioners with outstanding performance when applied to the class of problems described in section II..

IV. EQUILIBRATION OF FINITE ELEMENT MATRICES

Equilibration is a particular form of scaling, where the rows and columns of a matrix are modified to possess approximately the same norm. We demonstrate that equilibrated sparse matrices, when operated by the LU factorization algorithm of section III., give rise to sparser factors than their unequilibrated counterparts. Therefore equilibration appears to be an important artifice to control fill-in. In the remaining sections, we explore this serendipitous property. Analysis of the effects of equilibration on stability and accuracy of LU factorizations is controversial and beyond the scope of this article. For details, we recommend the eloquent exposition of POOLE AND NEAL (1992).

The matrix $\tilde{\mathbf{A}} := \mathbf{R}\,\mathbf{A}\,\mathbf{C}$, where $\mathbf{R}$ and $\mathbf{C}$ are diagonal matrices, is an equilibration of $\mathbf{A}$ if the norm of its columns and rows possesses approximately the same magnitude.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

Therefore, one possibility for defining the diagonal matrices is

$$\boxed{R_{i,i} := \left\|A_{i,\bullet}\right\|^{-1}} \quad \text{and} \quad C_{i,i} := \left\|A_{\bullet,i}\right\|^{-1} \quad \text{where} \quad R_{i,j} = C_{i,j} = 0 \text{ if } i \neq j, \qquad (4.1)$$

$A_{i,\bullet}$ is the ith row vector of $\mathbf{A}$, and $A_{\bullet,i}$, the ith column vector. Thus, if $\mathbf{A}$ is the matrix of coefficients of a linear system of equations, $\mathbf{R}$ scales the equations and $\mathbf{C}$ scales the unknowns. In our experiments, we consider the 2-norm and ∞-norm in (4.1).

A simple modification of the algorithm 3.1 allows the computation of $\mathbf{R}$ and $\mathbf{C}$. The basic structure needed is the compression of rows and columns into the frontal matrix. We describe the construction of equilibration matrices with 2-norm of rows and columns. The case for the ∞-norm is analogous.

Consider first the construction of $\mathbf{R}$. With the available representation (3.2) of the frontal matrix

$$\begin{pmatrix} \mathbf{F}_{1,1} & \mathbf{F}_{1,2} \\ \mathbf{F}_{2,1} & \mathbf{F}_{2,2} \end{pmatrix},$$

the entries in the fully summed rows of the block $\mathbf{F}_{2,1}$ can be squared, summed and temporarily stored in the corresponding global locations in $\mathbf{R}$. Likewise, the rows spanning the blocks $\mathbf{F}_{1,1}$ and $\mathbf{F}_{1,2}$ can be squared, summed and *added* to existing values in the corresponding global location in $\mathbf{R}$. The existing values in $\mathbf{R}$ arise because of the temporary storage of the blocks $\mathbf{F}_{1,2}$ and $\mathbf{F}_{2,1}$ from previous states of the frontal matrix. Next the fully summed blocks can be removed from the frontal matrix and a new element matrix can be assembled. The process is repeated until all element matrices are assembled and all rows processed. At the end of the process, the diagonal entry $R_{i,i}$ becomes the sum of squares of the row entries $A_{i,\bullet}$. Therefore, computing the correct $R_{i,i}$ is a simple matter of taking the reciprocal of the square-root of each entry in $\mathbf{R}$. The algorithm below summarizes the procedure.

Algorithm 4.1 Construction of row equilibration matrix with vector 2-norm.

Set $\mathbf{R} := 0$
For $e = 1, \ldots, N$ *Do:*
  Assemble $\mathbf{F}_e \leftarrow \mathbf{A}_e$
  For $I = 1, \ldots, s$ *Do: (*$s \times s$ *is the dimension of* $\mathbf{F}_{1,1}$*)*
    $i \leftarrow I$ *(global mapping)*
    $R_{i,i} = \sum_{J=1}^{t} (\mathbf{F}_{2,1})_{I,J}^2$ *(*$t \times t$ *is the dimension of* $\mathbf{F}_{2,2}$ *in* $\mathbf{F}_e$*)*
    $R_{i,i} = R_{i,i} + \sum_{J=1}^{t} (\mathbf{F}_{1,1})_{I,J}^2$
    $R_{i,i} = R_{i,i} + \sum_{J=1}^{s} (\mathbf{F}_{1,2})_{I,J}^2$
  EndDo
  Delete $\mathbf{F}_{1,1}$, $\mathbf{F}_{1,2}$ *and* $\mathbf{F}_{2,1}$
  Replace $\mathbf{F}_e$ *with* $\mathbf{F}_{2,2}$
EndDo
Set $R_{i,i} := 1/\sqrt{R_{i,i}}$

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

The construction of $\mathbf{C}$ is similar to the construction of $\mathbf{R}$. The modification of the algorithm for computing equilibration matrices based on the ∞-norm is straightforward. In the numerical experiments of the next section, the rows of $\mathbf{A}$ are equilibrated first and the columns of the resulting row-equilibrated matrix second. Thus, the modified definition of $\mathbf{C}$ is

$$\boxed{C_{i,i} := \left\| (\mathbf{RA})_{\bullet,i} \right\|^{-1}} \quad \text{where} \quad C_{i,j} = 0 \text{ if } i \neq j.$$

V. RESULTS AND DISCUSSION

Table III presents results for two cases of incomplete factorization for each model problem. The factorizations in Case 1 are used as robust preconditioners in a Newton-GMRES(100) nonlinear solver described by BROWN AND SAAD (1990) and are constructed with relaxed values of the pivoting (stability) threshold parameter u (sec. C.), the fill-in numerical dropping parameter *tol*, and the fill-in cut-off parameter *lfil* (sec. B.).

Factorizations in Case 2 are used in conjunction with SHAMANSKII'S acceleration scheme for Newton's nonlinear solver (DE ALMEIDA AND DERBY, 1999a). Here an existing LU factorization is used to accelerate the convergence of the nonlinear solver by solving many linear systems with updated right sides at a given iteration of Newton's method. The cost of the acceleration is only that of two triangular substitutions and the resulting convergence is substantially improved. The values for the parameters u, *tol* and *lfil* must be more strict (sec. E.) than those in Case 1 otherwise the acceleration may not converge.

In both cases, a suitable set of values for u, *tol* and *lfil* is found so that a family of finite element approximate solutions is satisfactorily computed for each model problem—second-order accurate mixed tetrahedral finite elements are employed and the convergence criteria for the nonlinear solvers depend on the particular approximating functional spaces (DE ALMEIDA AND DERBY, 1999b). The matrices are constructed by linearizing the governing system of partial differential equations *at* a representative fully converged solution, and subsequently discretizing the resulting set of linear equations. The solutions selected for each model problem (sec. II.) correspond to the dimensionless numbers reported in figures 1, 2 and 3. The naturally appearing matrices are called Jacobian matrices. We extracted the Jacobian matrices from the first nonlinear iteration and factored them with two different degrees of accuracy according to the cases 1 and 2 mentioned above.

In the third to fifth columns of the tables, results for factorizations of matrices with no equilibration, ∞-norm equilibration and 2-norm equilibration, respectively are exhibited. The matrices are equilibrated according to the 2-norm and ∞-norm procedure (sec. IV.). Indicators of the difficulties encountered when factoring the equilibrated matrices are compared with similar results obtained in the absence of equilibration. We are particularly interested in the first three indicators of the tables, namely, the number of nonzero entries in the factors $nnz_{\mathbf{LU}}$, the average size of the frontal matrix $\bar{m}_{\mathbf{F}}$, and the

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

TABLE III. Indicators for incomplete factorization of $\tilde{\mathbf{A}}$ with different equilibration procedures. Gflops is the total number of billions of floating-point operations in the LU factorization of $\tilde{\mathbf{A}}$.

MCZ model. Solution at $Gr = 10^5$, $Ha = 45$, $Pr = 10^{-2}$, $Re_x = 150$ and $\mathscr{Q} = 2$.
$n_{\mathbf{A}} = 74\,531$, $nnz_{\mathbf{A}} = 62\,609\,619$. In Case 1, $u = 0.1$, $tol = 10^{-4}$ and $lfil = 2\,500$.
In Case 2, $u = 0.1$, $tol = 10^{-7}$ and $lfil = \infty$.

Case	Indicator	No Equil.*	∞-norm Equil.	2-norm Equil.
1	$nnz_{\mathbf{LU}}$	297 024	162 277 622	164 858 500
	$\bar{m}_{\mathbf{F}}$	12 080†	2 056	2 011
	Gflops	0.027	400	389
	$1/\min_k \lvert\alpha^{(k)}\rvert$	$2.97\,10^5$	$2.34\,10^2$	$7.07\,10^2$
	$\max \lvert U_{i,j}\rvert$	$1.71\,10^1$	$7.95\,10^3$	$2.58\,10^3$
	$\lVert(\mathbf{LU})^{-1}\tilde{\mathbf{A}} - \mathbf{I}\rVert$	—‡	$5.29\,10^4$	$2.08\,10^2$
2	$nnz_{\mathbf{LU}}$	316 432	177 701 597	179 943 964
	$\bar{m}_{\mathbf{F}}$	12 080†	2056	2011
	Gflops	0.027	400	389
	$1/\min_k \lvert\alpha^{(k)}\rvert$	$2.97\,10^5$	$2.34\,10^2$	$7.07\,10^2$
	$\max \lvert U_{i,j}\rvert$	$1.71\,10^1$	$7.95\,10^3$	$2.58\,10^3$
	$\lVert(\mathbf{LU})^{-1}\tilde{\mathbf{A}} - \mathbf{I}\rVert$	—‡	$2.40\,10^1$	$4.00\,10^1$

KTP model. Solution at $Re = 300$. $n_{\mathbf{A}} = 77\,119$, $nnz_{\mathbf{A}} = 77\,346\,872$.
In Case 1, $u = 0.1$, $tol = 10^{-3}$ and $lfil = 2\,000$. In Case 2, $u = 0.1$, $tol = 10^{-7}$ and $lfil = \infty$.

Case	Indicator	No Equil.*	∞-norm Equil.	2-norm Equil.
1	$nnz_{\mathbf{LU}}$	220 806	148 713 671	148 150 737
	$\bar{m}_{\mathbf{F}}$	12 050†	1 898	2 151
	Gflops	0.017	357	376
	$1/\min_k \lvert\alpha^{(k)}\rvert$	$8.30\,10^1$	$6.87\,10^1$	$1.20\,10^2$
	$\max \lvert U_{i,j}\rvert$	$2.15\,10^2$	$7.75\,10^4$	$2.15\,10^3$
	$\lVert(\mathbf{LU})^{-1}\tilde{\mathbf{A}} - \mathbf{I}\rVert$	—‡	$2.46\,10^4$	$5.93\,10^2$
2	$nnz_{\mathbf{LU}}$	223 094	174 484 100	183 242 153
	$\bar{m}_{\mathbf{F}}$	12 050†	1 898	1903
	Gflops	0.017	357	376
	$1/\min_k \lvert\alpha^{(k)}\rvert$	$8.30\,10^1$	$6.87\,10^1$	$1.20\,10^2$
	$\max \lvert U_{i,j}\rvert$	$2.15\,10^2$	$7.75\,10^4$	$2.15\,10^3$
	$\lVert(\mathbf{LU})^{-1}\tilde{\mathbf{A}} - \mathbf{I}\rVert$	—‡	$5.98\,10^1$	$7.73\,10^1$

MCVD model. Solution at $Gr = 10^6$, $Re = 1\,400$, $Re_r = 50$, Pr = 0.72.
$n_{\mathbf{A}} = 242\,152$, $nnz_{\mathbf{A}} = 42\,622\,355$. In Case 1, $u = 0.1$, $tol = 10^{-3}$ and $lfil = \infty$.
In Case 2, $u = 0.1$, $tol = 10^{-7}$ and $lfil = \infty$.

Case	Indicator	No Equil.*	∞-norm Equil.	2-norm Equil.
1	$nnz_{\mathbf{LU}}$	341 923	74 203 733	77 419 110
	$\bar{m}_{\mathbf{F}}$	10 026†	366	371
	Gflops	0.454	41	43
	$1/\min_k \lvert\alpha^{(k)}\rvert$	$4.97\,10^5$	$8.59\,10^1$	$1.31\,10^2$
	$\max \lvert U_{i,j}\rvert$	$2.10\,10^1$	$2.02\,10^3$	$1.35\,10^3$
	$\lVert(\mathbf{LU})^{-1}\tilde{\mathbf{A}} - \mathbf{I}\rVert$	—‡	$1.42\,10^2$	$2.58\,10^2$
2	$nnz_{\mathbf{LU}}$	327 968	88 527 545	91 038 795
	$\bar{m}_{\mathbf{F}}$	10 026†	366	371
	Gflops	0.020	41	43
	$1/\min_k \lvert\alpha^{(k)}\rvert$	$6.84\,10^4$	$8.59\,10^1$	$1.31\,10^2$
	$\max \lvert U_{i,j}\rvert$	9	$2.02\,10^3$	$1.35\,10^3$
	$\lVert(\mathbf{LU})^{-1}\tilde{\mathbf{A}} - \mathbf{I}\rVert$	—‡	$1.44\,10^2$	$2.48\,10^2$

* Factorization was terminated before completion.
† Size of the frontal matrix at the point where factorization was terminated.
‡ Quantity not available at the point where factorization was terminated.

total number of billions of floating-point operations Gflops. These indicators help us to evaluate the feasibility of a factorization. The remaining set of indicators, $1/\min_k \left|\alpha^{(k)}\right|$, $\max \left|U_{i,j}\right|$, and $\left\|(\mathbf{LU})^{-1}\,\tilde{\mathbf{A}} - \mathbf{I}\right\|$ described in section D., is related to the accuracy and stability of the factorization and is considered supplementary data (values of $\left\|\mathbf{L}^{-1}\right\|$ and $\left\|\mathbf{U}^{-1}\right\|$ are reported in DE ALMEIDA *et al.*, 1998). In practice, the quality of the factorization is assessed on the value of these indicators and the actual performance of the factorization with the algorithm in which it is embedded. Typically when indicators display relatively large values, the quality of the factorization is poor. However, relatively small values are not a firm ground to assure the factorization's needed quality.

In all cases for all models, factorization of unequilibrated Jacobian matrices was not possible with the available computational resources whereas factorization of equilibrated versions of the same matrices was successful.

In all cases without equilibration, a prohibitive size of the frontal matrix consumed the available computational memory space at the initial stages of the factorization—the relatively small number of floating-point operations justify this conclusion. The uncontrolled growth in size of the frontal matrix $\mathbf{F}$ was caused by excessive pivotal failure. As line 8 in algorithm 3.1 indicates, for each failure, a new element matrix is added to the frontal matrix. If the rate of incoming element matrices is substantially greater than the rate at which pivotal rows and columns are eliminated, $\mathbf{F}$ grows rapidly. The third column in the tables reports the available data for cases without equilibration at the point where the factorization was terminated. In the absence of equilibration, in order to force a factorization to proceed without increasing the size of the frontal matrix, the threshold (stability) parameter u had to be relaxed to very small values, resulting in LU factors too unstable to be useful either as a preconditioner or as an accelerator.

As for sparsity, results obtained with ∞-norm equilibration were consistently superior than the ones obtained with 2-norm equilibration. In general, the number of floating-point operations and the number of nonzero entries in the factors of ∞-norm equilibrated matrices are smaller than the factors of 2-norm equilibrated matrices. Surprisingly, the storage requirement of one factor of the LU fatorization of the matrices is approximately the same as that needed to store the original matrix. This can be verified from the data in the tables by comparing the number of nonzero entries of the factors and the number of nonzero entries $nnz_{\mathbf{A}}$ in the matrix. The relative small values of the indicators of stability and accuracy in all cases are consistent with the satisfactory performance of the factorizations in cases 1 and 2 within their respective nonlinear solvers. However, the factorizations in Case 1, aimed at preconditioning the Newton-GMRES nonlinear solver, failed when used with the SHAMANSKII'S acceleration scheme for Newton's nonlinear solver. This illustrates our assertion that relative small values of the stability and accuracy indicators are not a firm basis for concluding that a factorization will perform adequately. Ultimately, the accuracy and stability of a factorization, combined with subtleties of the embedding algorithm, will determine its performance.

Other experiments with equilibrated matrices were conducted with relaxed choices for the u, *tol*, and *lfil* parameters. Equilibration with ∞-norm led to the smallest number of

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

floating-point operations in most cases. The sparsity of the factorization monotonically increased when *lfil* decreased and *tol* increased. The behavior of u was deceptive. In the range $0.01 < u < 1$, sparsity monotonically increased when u decreased; otherwise, in other instances, it decreased when u decreased.

Stability and accuracy of factorizations deteriorated significantly when *lfil* was smaller than the average size of the frontal matrix $\bar{m}_{\mathbf{F}}$. Best results were obtained when *lfil* was greater than or equal to $\bar{m}_{\mathbf{F}}$. We observed that equilibration helped to preserve the original ordering of the matrix by reducing the amount of pivoting required to carry out the factorization. In the presence of equilibration, fully summed rows and columns present in the frontal matrix often passed the threshold criterion (3.6) without requiring row and column interchanges. Consequently *a priori* estimate of $\bar{m}_{\mathbf{F}}$ computed with a sparse symbolic factorization of $\mathbf{A}$ was often satisfactory. Values of *tol* did not affect the quality of the factorization so dramatically as *lfil* values did, so we recommend that at least $tol < 0.01$. We also suggest that the value for u should be on the order of 0.1 based on extensive testing conducted by DUFF *et al.* (1989) and the results presented here.

In view of the values of the accuracy indicator $\left\| (\mathbf{LU})^{-1} \tilde{\mathbf{A}} - \mathbf{I} \right\|$ for Cases 1 and 2 of the models, sparser factors are less accurate than less sparse factors. Comparing the results obtained for ∞-norm equilibrated matrices with 2-norm equilibrated matrices for each case of the models, it is observed that LU factorizations of ∞-norm equilibrated matrices are sparser and cheaper to compute but they are also less accurate and less stable.

VI. CONCLUSIONS AND FINAL REMARKS

We have examined the effects of equilibrating large and sparse matrices that arise from the discretization and linearization of a class of coupled nonlinear system of elliptic partial differential equations. Many mathematical models of continuum transport phenomena in engineering systems fall into this class, notably, the incompressible Navier-Stokes equations, the magnetohydrodynamics equations, and the convection-diffusion equation. Specifically, we have tested equilibration on matrices arising in the finite element solution of three models of materials processing systems. These systems are appropriate for the testing of iterative solution techniques which rely on the robustness provided by LU factorizations such as incomplete LU preconditioning—simple diagonal preconditioning is often insufficent for the solution of these problems (YECKEL AND DERBY, 1997)—and accelerated Newton-like methods that explore triangular solves with an existing factorization (DE ALMEIDA AND DERBY, 1999a).

While there is little formal theory to substantiate the benefits of matrix equilibration, our computational test cases point to a dramatic practical benefit. We find that LU factors of equilibrated matrices are substantially sparser than their unequilibrated counterparts; therefore, they are not as expensive to compute and store. This issue is of enormous relevance for solution of the systems arising from realistic problems, such as those resulting from three-dimensional, finite element models. In fact, all attempts to factor unequilibrated matrices failed for the cases considered here due to prohibitive piv-

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

oting costs and untolerably large amounts of fill-in which exhausted the computational storage resources in the very early stages of the factorization process. Factorizations with relaxed pivoting (small threshold parameter) were also attempted so that pivoting and fill-in were tolerable. However the resulting factors were too unstable to serve either as a preconditioner or as an accelerator. In contrast, LU factors from equilibrated matrices were sucessfully employed to solve all of the test cases.

LU factors of matrices equilibrated with the ∞-norm of rows and columns were sparser than those obtained by equilibrating the matrices with the 2-norm. Surprisingly, the storage requirement for each factor was nearly the same as for the original matrix. That is, relatively small storage was required to store the factors. LU factorizations of ∞-norm equilibrated matrices were less backward stable and less accurate than those computed with the 2-norm, yet this was not an important issue for the overall problem solution. By solving the test models to the desired degree of accuracy, we show that judiciously computed incomplete LU factorizations are invaluable for the analysis of an important class of problems in engineering and applied sciences.

Finally, understanding how exactly matrix equilibration affects a particular pivoting strategy is still lacking. Our experiments show that matrix equilibration alters a given pivoting sequence so that the sparsity of the LU factors is preserved. In practice, this desirable property is as essential as the accuracy and stability of the factorization. For the future, it would be interesting to investigate alternative pivoting strategies that would outperform the threshold pivoting method combined with equilibration.

This work was supported by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement DAAH04-95-2-0003/contract DAAH04-95-C-0008, the content of which does not necessarily reflect the position or policy of the government, and no official endorsement should be inferred. Additional computational resources were provided by the University of Minnesota Supercomputer Institute. VFdA is grateful to Dr. J. C. Rojo and Mr. B. Vartak for discussions on crystal growth issues, and to Drs. T. Salamon and A. Bose for discussions on optical fiber processing.

References

ALLGOWER, E. L. AND GEORG, K. **1990** *Numerical Continuation Methods*, volume 15 of *Springer Series in Computational Mathematics*. Springer-Verlag, New York, New York.

DE ALMEIDA, V. F. AND DERBY, J. J. **1999a** Construction of solution curves for large 2-D problems of steady-state flows of incompressible fluids. *SIAM J. Sci. Comput.* **0** 0–0. In press.

DE ALMEIDA, V. F. AND DERBY, J. J. **1999b** Practical mixed finite elements for 3-D flows of incompressible fluids. Minnesota Supercomputer Institute Research Report 99-

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

, University of Minnesota, Department of Chemical Engineering and Materials Science, Minneapolis, MN 55455-0132, U.S.A. In preparation.

de Almeida, V. F., Chapman, A. M., and Derby, J. J. **1998** On equilibration and sparse factorization of matrices arising in finite element solutions of partial differential equations. Minnesota Supercomputer Institute Research Report 98/165, University of Minnesota, Department of Chemical Engineering and Materials Science, Minneapolis, MN 55455-0132, U.S.A. Also available under request to `de_almeida@na-net.ornl.gov` or at URL: `http://www.msi.umn.edu/~dalmeida`.

Bauer, F. L. **1963** Optimally scaled matrices. *Numer. Math.* **5** 73–87.

Bierlein, J. D. and Vanherzeele, H. **1989** Potassium titanyl phosphate: properties and new applications. *J. Opt. Soc. Am. B* **6** 622–33.

Brezzi, F. and Fortin, M. **1991** *Mixed and Hybrid Finite Element Methods*, volume 15 of *Computational Mathematics*. Springer-Verlag, New York, New York.

Brown, P. N. and Saad, Y. **1990** Hybrid Krylov methods for nonlinear systems of equations. *SIAM J. Sci. Stat. Comput.* **3** 450–81.

Chandrasekhar, S. **1981** *Hydrodynamic and Hydromagnetic Stability*. Dover Publications, Inc., New York, New York. Originally published in 1961 by Clarendon Press.

Chapman, A. M., Saad, Y., and Wigton, L. **1996** High order ILU preconditioners for CFD problems. Minnesota Supercomputer Institute Research Report 96/14, University of Minnesota, Department of Computer Science and Engineering, Minneapolis, MN 55455-0132, U.S.A.

Dahlquist, G. and Björck, Å. **1974** *Numerical Methods*. Series in Automatic Computation. Prentice-Hall, Englewood Cliffs, New Jersey.

Dongarra, J. J., Duff, I. S., Sorensen, D. C., and van der Vorst, H. A. **1991** *Solving Linear Systems on Vector and Shared Memory Computers*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

Duff, I. S. and Scott, J. A. **1993** MA42—a new frontal code for solving sparse unsymmetric systems. Research Report RAL 93-064, Central Computing Department, Atlas Centre, Rutherford Appleton Laboratory, Oxon OX11 0QX, U.K.

Duff, I. S., Erisman, A. M., and Reid, J. K. **1989** *Direct Methods for Sparse Matrices*. Monographs on Numerical Analysis. Clarendon Press, Oxford, Great Britain.

Girault, V. and Raviart, P.-A. **1986** *Finite Element Methods for Navier-Stokes Equations*. Springer Series in Computational Mathematics. Springer-Verlag, New York, extended version of 1979 edition.

Golub, G. H. and Van Loan, C. F. **1989** *Matrix Computations*. Johns Hopkins Series in the Mathematical Sciences. The Jonhs Hopkins University Press, Baltimore, second edition.

Gunzburger, M. D. **1989** *Finite Element Methods for Viscous Incompressible Flows*. Computer Science and Scientific Computing. Academic Press, Inc, Boston, Massachusets.

Hackbush, W. and Wittum, G., editors **1993** *Incomplete Decomposition (ILU)—Algorithms, Theory, and Applications*. Proceedings of the Eigth GAMM-Seminar, January 24–26, 1992. Vieweg, Germany.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.

HIGHAM, N. J. **1989** The accuracy of solutions to triangular systems. *SIAM J. Numer. Anal.* **26** 1252–65.

KELLER, H. B. **1992** *Numerical Methods for Two-Point Boundary-Value Problems.* Dover Books in Advanced Mathematics. Dover Publications, Inc., New York, New York. Corrected and expanded edition of the work first published by Blaisdell Publishing Company in 1964.

KELLEY, C. T. **1995** *Iterative Methods for Linear and Nonlinear Equations.* Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

LIN, W. AND BENSON, K. E. **1987** The science and engineering of large-diameter Czochralski silicon crystal growth. *Ann. Rev. Mater. Sci.* **17** 273–98.

MCCARTHY, C. AND STRANG, G. **1973** Optimal conditioning of matrices. *SIAM J. Numer. Anal.* **10** 370–88.

POOLE, G. AND NEAL, L. **1992** Gaussian elimination: when is scaling beneficial. *Linear Algebra Appl.* **162** 309–24.

ROJO, J. C. AND DERBY, J. J. **1999** On the formation of rotational spoke patterns on the melt surface during czochralski growth of molten bismuth silicon oxide. *J. Crystal Growth* **198** 154–60.

SAAD, Y. **1996** *Iterative Methods for Sparse Linear Systems.* The PWS Series in Computer Science. PWS Publishing, New York, New York.

SKEEL, R. D. **1979** Scaling for numerical stability in Gaussian elimination. *J. ACM* **26** 494–526.

SLOAN, S. W. **1989** A FORTRAN program for profile and wavefront reduction. *Int. J. Num. Meth. Eng.* **28** 2651–79.

SMITH, B. F., BJØRSTAD, P. E., AND GROPP, W. D. **1996** *Domain Decomposition Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, New York.

VAN DER SLUIS, A. **1969** Condition numbers and equilibration of matrices. *Numer. Math.* **14** 14–23.

VAN DER SLUIS, A. **1970** Condition, equilibration and pivoting in linear algebraic systems. *Numer. Math.* **15** 74–86.

VARTAK, B., KWON, Y.-I., YECKEL, A., AND DERBY, J. J. **1998** Fluid flow and mass transfer during the solution growth of potassium titanyl phosphate. *J. Crystal Growth.* Submitted.

YECKEL, A. J. AND DERBY, J. J. **1997** Parallel computation of incompressible flows in materials processing: Numerical experiments in diagonal preconditioning. *Parallel Computing* **23** 1379–400.

Submitted to *Numerical Methods for Partial Differential Equations*, An International Journal, pp. 1–22 on 23Sep98. Reviewed on 3Feb99. Revised on 5Aug99. Accepted on 30Aug99.