NHG 1-259

ELEMENT-BY-ELEMENT APPROXIMATE FACTORIZATION ALGORITHMS FOR HEAT CONDUCTION(1)

Thomas J.R. Hughes(2)
Division of Applied Mechanics
Department of Mechanical Engineering
Durand Building
Stanford University
Stanford, California 94305

James M. Winget(3)
Division of Engineering and Applied Science
California Institute of Technology
Pasadena, California 91125

K. C. Park(4)
Applied Mechanics Laboratory
Lockheed Palo Alto Research Laboratories
3251 Hanover Street
Palo Alto, California 94304

---

(1) Supported by NASA Langley Research Grant No. 1-259

(2) Professor of Mechanical Engineering

(3) Graduate Research Assistant

(4) Staff Scientist

Abstract

Element-by-element solution strategies are developed for transient heat conduction problems. Results of numerical tests indicate the effectiveness of the procedures proposed. The small data base requirements and attractive architectural features of the algorithms suggest considerable potential for solving large scale problems.

Contents

1. Introduction

The first example of an element-by-element (EBE) algorithm for heat conduction was presented by Hughes, Levit and Winget [H4]. In that work the EBE concept was used to develop a non-iterative second-order time-accurate unconditionally stable transient algorithm for both linear and nonlinear problems. Element arrays could be processed individually with no need to construct a global coefficient matrix. Our initial numerical testing with this scheme proved satisfactory. However, later on we discovered that under certain circumstances the accuracy level attained by typical globally implicit methods, such as the Crank-Nicolson procedure, was not attained by the method of [H4]. The problem was traced to spatial truncation error terms such as those which afflict some classical split-operator finite difference methods such as the DuFort-Frankel method [A1]. To overcome these accuracy deficiencies we were led to reformulate the EBE procedure as an iterative linear equation solver so that standard time discretization techniques could be employed. In this way issues of stability and accuracy are obviated. The only question which remains is how fast does the iterative process converge? At the same time the small data base and attractive architectural features of the EBE process are retained. Another advantage which accrues is that coupled capacity matrices may be accomodated. This improves upon [H4] which was restricted to lumped capacity.

On the other hand, relegating the EBE concept to iterative linear equation solving does not seem to exploit its full potential. To the authors, this represents a conservative, interim strategy. In future research we hope to explore the use of EBE concepts throughout the entire problem solving spectrum.

There already seems clear paths for significant increases of efficiency in large, nonlinear problems by adopting this philosophy. It is interesting to note that the chronology of research developments in multigrid techniques followed along similar lines in that initial success was found in iterative linear equation solving, but subsequent improvement was obtained by procedures in which multigrid concepts permeated all aspects of the solution process (Brandt [B2]).

By restricting the use of the EBE concept to iterative equation solving the research-problem is rendered tractable in that other aspects of solution can be done by standard means. Despite this fact there still appears to be a great deal of variety to the types of EBE strategies which may be developed. Basically, three main ingredients are necessary for an EBE iterative linear-equation solver. They are an iterative driver strategy, an EBE approximate factorization scheme, and the definition of an array which approximates the global coefficient matrix and is amenable to EBE approximate factorization. These topics are explored in Sections 2 to 4, respectively. In Section 5, sample problems are presented.

For related developments in the area of structural analysis, the interested reader is urged to consult [H5, H10, H11, N1, O1]. A pilot study of a transonic flow, involving an unsymmetric coefficient matrix, is presented in [H11]. By virtue of the fact that the present thrust to research in EBE techniques has been concerned with aspects of linear equation solving, a certain synthesis of concepts has ensued. This, as remarked above, is believed to be a very temporary state of affairs. To further develop EBE algorithms which are truly effective for different problem classes, the physics and idiosyncracies of the individual classes will need to be accounted for in the structure of the

algorithms. For example, we may contrast the transient heat conduction and structural schemes presented in [H4] and [O1], respectively. The heat conduction scheme was formulated in terms of temperature degrees-of-freedom in a very natural way. No structural analog in terms of kinematical variables could be developed which attained unconditional stability. Rather, an entirely new global formulation had to be created with stresses and velocities as primary unknowns! Needless to say, the developmental implications of such schemes are significant. A unique procedure of this kind requires considerable research on all levels to be brought to fruition. We anticipate this being the case for the various problem classes to which the EBE concept will undoubtedly be applied.

2. Iterative Algorithms

Two candidate iterative algorithms which can be used in conjunction with approximately factorized arrays are described below.

2a. Parabolic Regularization

The parabolic regularization algorithm is derived in [H5]. Table 1 presents a flowchart of the procedure for symmetric positive-definite systems.

Table 1 Flowchart of the parabolic regularization (PR) algorithm with line search and BFGS updates

Step 1. Initialization:

$$m = 0 \quad , \quad \underset{\sim}{x}_0 = \underset{\sim}{0} \quad , \quad \underset{\sim}{r}_0 = \underset{\sim}{b}$$

$$\underset{\sim}{f}_k = \underset{\sim}{g}_k = \underset{\sim}{0} \qquad (\text{loop: } k = 1 , 2 , \ldots , n_{BFGS})$$

$$\Delta\underset{\sim}{x} = \underset{\sim}{B}^{-1} \underset{\sim}{r}_0$$

Step 2. Line search:

$$s = \Delta\underset{\sim}{x}^T \underset{\sim}{r}_m / \Delta\underset{\sim}{x}^T \underset{\sim}{A} \Delta\underset{\sim}{x}$$

$$\underset{\sim}{x}_{m+1} = \underset{\sim}{x}_m + s \, \Delta\underset{\sim}{x}$$

Step 3. Convergence check:

$$||\underset{\sim}{r}_{m+1}|| < \delta$$

Yes: Return

No : Continue

Step 4. Relabel old BFGS vectors:

$$\underset{\sim}{f}_{k-1} = \underset{\sim}{f}_k \quad , \quad \underset{\sim}{g}_{k-1} = \underset{\sim}{g}_k \quad , \quad (\text{loop: } k = 2 , 3 , \ldots , n_{BFGS})$$

Step 5. Calculate new BFGS vectors:

$$\underset{\sim}{f}_{n_{BFGS}} = (\Delta\underset{\sim}{x}^T \underset{\sim}{r}_m)^{-1} \Delta\underset{\sim}{x}$$

$$\underset{\sim}{g}_{n_{BFGS}} = \underset{\sim}{r}_{m+1} - (1 - s^{1/2})\underset{\sim}{r}_m$$

Step 6. New search direction:

$$\underset{\sim}{z} = \underset{\sim}{r}_{m+1}$$

$$\underset{\sim}{z} \leftarrow \underset{\sim}{z} + (\underset{\sim}{f}_k^T \underset{\sim}{z})\underset{\sim}{g}_k \quad \text{(loop: } k = n_{BFGS}, n_{BFGS} - 1, \ldots, 1)$$

$$\underset{\sim}{z} \leftarrow \underset{\sim}{B}^{-1} \underset{\sim}{z}$$

$$\underset{\sim}{z} \leftarrow \underset{\sim}{z} + (\underset{\sim}{g}_k^T \underset{\sim}{z})\underset{\sim}{f}_k \quad \text{(loop: } k = 1, \ldots, n_{BFGS})$$

$$\Delta\underset{\sim}{x} = \underset{\sim}{z}$$

Step 7. $m \leftarrow m + 1$, go to Step 2.

The notation in Table 1 is given as follows: m is the iteration counter; the $\underset{\sim}{f}_k$'s and $\underset{\sim}{g}_k$'s are the BFGS vectors; n_{BFGS} is the maximum number of BFGS vectors allowed; $\underset{\sim}{B}$ is a matrix which approximates $\underset{\sim}{A}$, but is more easily factorized; s is the search parameter; $\underset{\sim}{x}_m$ is the m^{th} approximation of $\underset{\sim}{x}$; $\underset{\sim}{r}_m = \underset{\sim}{b} - \underset{\sim}{A}\,\underset{\sim}{x}_m$ is the corresponding residual; $||\underset{\sim}{r}_m||$ is its Euclidean length; and δ is a preassigned error tolerance. The search parameter in step 2 is determined by minimizing the potential energy

$$P(s) = -(\underset{\sim}{x}_m + s\,\Delta\underset{\sim}{x})^T (\underset{\sim}{b} - \frac{1}{2}\underset{\sim}{A}(\underset{\sim}{x}_m + s\,\Delta\underset{\sim}{x})) \qquad (2.1).$$

2b. Preconditioned Conjugate Gradients

This algorithm is a generalization of the classical conjugate gradients method (see Hestenes-Stiefel [H1]) in which a "preconditioning" is performed

using $\underset{\sim}{B}$, the matrix approximating $\underset{\sim}{A}$. The algorithm is summarized in Table 2.

Table 2 Flowchart of preconditioned conjugate gradients (CG)

Step 1. Initialization:

$m = 0$, $\underset{\sim}{x}_0 = \underset{\sim}{0}$

$\underset{\sim}{r}_0 = \underset{\sim}{b}$

$\underset{\sim}{p}_0 = \underset{\sim}{z}_0 = \underset{\sim}{B}^{-1} \underset{\sim}{r}_0$

Step 2. $\alpha_m = \underset{\sim}{r}_m^T \underset{\sim}{z}_m / \underset{\sim}{p}_m^T \underset{\sim}{A} \underset{\sim}{p}_m$

Step 3. $\underset{\sim}{x}_{m+1} = \underset{\sim}{x}_m + \alpha_m \underset{\sim}{p}_m$

Step 4. $\underset{\sim}{r}_{m+1} = \underset{\sim}{r}_m - \alpha_m \underset{\sim}{A} \underset{\sim}{p}_m$

Step 5. Convergence check:

$||\underset{\sim}{r}_{m+1}|| < \delta$?

Yes: Return

No : Continue

Step 6. $\underset{\sim}{z}_{m+1} = \underset{\sim}{B}^{-1} \underset{\sim}{r}_{m+1}$

Step 7. $\beta_m = \underset{\sim}{r}_{m+1}^T \underset{\sim}{z}_{m+1} / \underset{\sim}{r}_m^T \underset{\sim}{z}_m$

Step 8. $\underset{\sim}{p}_{m+1} = \underset{\sim}{z}_{m+1} + \beta_m \underset{\sim}{p}_m$

Step 9. $m \leftarrow m + 1$, go to Step 2.

Remark 1. Glowinski et al. [G1, G2] (see also references therein) have successfully used the preconditioned conjugate gradients algorithm in their finite element work. The matrix which they employ as preconditioner is determined by way

of various "incomplete Cholesky factorizations" (see e.g. Thomasnet [T1] and references therein).

Remark 2. A fixed number of vectors is all that is needed in the CG method. This makes it computationally more attractive than the PR algorithm with BFGS updates, because a considerable number of BFGS vectors typically need to be stored.

3. Approximate Factorization

The convergence rate of the algorithms presented in the preceding section depend heavily upon the approximating matrix $\underset{\sim}{B}$. It may be noted that if $\underset{\sim}{B} = \underset{\sim}{A}$ then both algorithms immediately obtain the exact solution $\underset{\sim}{x}$. Numerous choices for $\underset{\sim}{B}$ are possible. To explore some of the possibilities we shall introduce the following notational scheme. Let

$$\underset{\sim}{A} = \underset{\sim}{L}_p(\underset{\sim}{A})\underset{\sim}{D}_p(\underset{\sim}{A})\underset{\sim}{U}_p(\underset{\sim}{A}) \qquad \text{(product decomposition)} \tag{3.1}$$

$$\underset{\sim}{A} = \underset{\sim}{L}_s(\underset{\sim}{A}) + \underset{\sim}{D}_s(\underset{\sim}{A}) + \underset{\sim}{U}_s(\underset{\sim}{A}) \quad \text{(sum decomposition)} \tag{3.2}$$

where the subscripts p and s indicate "product" and "sum", respectively.

Equation (3.1) represents the Crout factorization. Thus $\underset{\sim}{L}_p$ and $\underset{\sim}{U}_p$ are lower and upper triangular matrices, respectively, with diagonal entries equal to 1, and $\underset{\sim}{D}_p(\underset{\sim}{A})$ is a diagonal matrix. If $\underset{\sim}{A}$ is symmetric, then $\underset{\sim}{L}_p(\underset{\sim}{A}) = \underset{\sim}{U}_p^T(\underset{\sim}{A})$. If the entries of $\underset{\sim}{D}_p$ are nonnegative, then we can write

$$\underset{\sim}{A} = \tilde{\underset{\sim}{L}}_p(\underset{\sim}{A})\tilde{\underset{\sim}{U}}_p(\underset{\sim}{A}) \tag{3.3}$$

where

$$\tilde{\underset{\sim}{L}}_p = \underset{\sim}{L}_p\underset{\sim}{D}_p^{\frac{1}{2}} \tag{3.4}$$

$$\tilde{U}_p = D_p^{\frac{1}{2}} U_p \tag{3.5}$$

When $\underset{\sim}{A}$ is symmetric positive-definite, (3.3)-(3.5) defines the Cholesky, or square-root, factorization.

In equation (3.2), L_s and U_s are lower and upper triangular matrices with diagonal entries equal to 0, and D_s is diagonal. In analogy with the product decomposition, we may write

$$\underset{\sim}{A} = \tilde{L}_s(\underset{\sim}{A}) + \tilde{U}_s(\underset{\sim}{A}) \tag{3.6}$$

where

$$\tilde{L}_s = L_s + \frac{1}{2} D_s \tag{3.7}$$

$$\tilde{U}_s = U_s + \frac{1}{2} D_s \tag{3.8}$$

If $\underset{\sim}{A}$ is symmetric, then $L_s(\underset{\sim}{A}) = U_s(\underset{\sim}{A})^T$ and $\tilde{L}_s(\underset{\sim}{A}) = \tilde{U}_s(\underset{\sim}{A})^T$.

Remark 1. The decomposition (3.6)-(3.8) has figured in the transient analysis algorithms developed by Trujillo [T2, T3] and subsequently discussed by Park [P1].

Remark 2. Note that the net total storage required for the sum decomposition is exactly the same as for the original matrix. However, the product decomposition entails increased storage due to "fill-in" of zeros within the skyline. This is perhaps the major drawback of direct solution schemes such as Crout elimination.

Remark 3. If we ignore the line search and quasi-Newton update ingredients of the PR algorithm, then classical iterative algorithms are obtained by choosing

$\underset{\sim}{B}$ as follows:

$$\underset{\sim}{B} = \underset{\sim}{D}_{\underset{\sim}{A}}(\underset{\sim}{A}) \qquad \text{(Jacobi method)} \tag{3.9}$$

$$\underset{\sim}{B} = \underset{\sim}{L}_{\underset{\sim}{A}}(\underset{\sim}{A}) + \underset{\sim}{D}_{\underset{\sim}{A}}(\underset{\sim}{A}) \qquad \text{(Gauss-Seidel method)} \tag{3.10}$$

To describe the procedures that are emphasized herein, we first consider matrices, $\tilde{\underset{\sim}{A}}$, written in the following form:

$$\tilde{\underset{\sim}{A}} = \underset{\sim}{W}^{\frac{1}{2}}(\underset{\sim}{I} + \varepsilon\, \bar{\underset{\sim}{A}})\underset{\sim}{W}^{\frac{1}{2}} \tag{3.11}$$

where $\underset{\sim}{I}$ is the identity matrix, $\underset{\sim}{W}$ is a positive-definite diagonal matrix, ε is a scalar, and $\bar{\underset{\sim}{A}}$ is a matrix which has the same sparsity pattern as $\underset{\sim}{A}$. $\tilde{\underset{\sim}{A}}$ is to be thought of as an approximation of $\underset{\sim}{A}$. Specific choices of $\underset{\sim}{W}$, ε and $\bar{\underset{\sim}{A}}$ are considered later in section 4. The second and final stage of the approximation is to define

$$\underset{\sim}{B} = \underset{\sim}{W}^{\frac{1}{2}}\, \underset{\sim}{C}\, \underset{\sim}{W}^{\frac{1}{2}} \tag{3.12}$$

where $\underset{\sim}{C}$ is an approximation of $\underset{\sim}{I} + \varepsilon\, \bar{\underset{\sim}{A}}$. Various choices are considered below:

3a. <u>Two-component Splitting</u>

Let $\bar{\underset{\sim}{A}}$ be decomposed as follows:

$$\bar{\underset{\sim}{A}} = \bar{\underset{\sim}{A}}_1 + \bar{\underset{\sim}{A}}_2 \tag{3.13}$$

Then a possible definition of $\underset{\sim}{C}$ is

$$\underset{\sim}{C} = (\underset{\sim}{I} + \varepsilon\, \bar{\underset{\sim}{A}}_1)(\underset{\sim}{I} + \varepsilon\, \bar{\underset{\sim}{A}}_2) = \underset{\sim}{I} + \varepsilon\, \bar{\underset{\sim}{A}} + \varepsilon^2\, \bar{\underset{\sim}{A}}_1\bar{\underset{\sim}{A}}_2 = \underset{\sim}{I} + \varepsilon\, \bar{\underset{\sim}{A}} + O(\varepsilon^2) \tag{3.14}$$

The last part suggests the nature of the approximation. Computational

simplicity is gained if $\bar{A}_1$ and $\bar{A}_2$ are very sparse and more easily factorized than $\bar{A}$.

For example, let

$$\bar{A}_1 = L_s(A) \tag{3.15}$$

$$\bar{A}_2 = U_s(\bar{A}) \tag{3.16}$$

Thus B has the following simple form

$$B = W^{\frac{1}{2}}(I + c\, L_s(\bar{A}))\,(I + \varepsilon\, U_s(\bar{A}))W^{\frac{1}{2}} \tag{3.17}$$

As may be seen, B is already factored and the factors require no more storage than that for A. Only diagonal scaling, and forward reductions and back substitutions with sparse triangular arrays are needed to solve equations with B as coefficient matrix. This eliminates the cost of factorization and obviates the storage penalties due to "fill-in". Equation (3.17) represents a symmetrized Gauss-Seidel type approximate factorization.

3b. One-pass Multi-component Splitting

Consider a multi-component sum decomposition of $\bar{A}$:

$$\bar{A} = \sum_{i=1}^{n} \bar{A}_i \tag{3.18}$$

Let

$$\begin{aligned} C &= \prod_{i=1}^{n} (I + \varepsilon \bar{A}_i) \\ &= (I + \varepsilon \bar{A}_1)(I + \varepsilon \bar{A}_2) \ldots (I + \varepsilon \bar{A}_n) \\ &= I + \varepsilon \bar{A} + O(\varepsilon^2) \end{aligned} \tag{3.19}$$

Clearly, this is just a straightforward generalization of the two-component splitting.

3c. Two-pass Multi-component Splitting

This generalization of the preceding case has qualitative advantages under certain circumstances (Marchuk [M1]). Let

$$\begin{aligned} \underset{\sim}{C} &= \prod_{i=1}^{n} (\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_i) \prod_{i=n}^{1} (\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_i) \\ &= (\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_1)(\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_2) \cdots (\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_n) \times \\ &\quad \times (\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_n)(\underset{\sim}{I} + \frac{1}{2} \overline{\underset{\sim}{A}}_{n-1}) \cdots (\underset{\sim}{I} + \frac{\epsilon}{2} \overline{\underset{\sim}{A}}_1) \\ &= \underset{\sim}{I} + \epsilon \overline{\underset{\sim}{A}} + O(\epsilon^2) \end{aligned} \tag{3.20}$$

If each $\overline{\underset{\sim}{A}}_i$ is symmetric and positive semi-definite, then $\underset{\sim}{C}$ is symmetric and positive-definite.

3d. Element-by-element (EBE) Approximate Factorizations

The EBE approximate factorization is simply a multi-component splitting in which the components are the finite element arrays themselves. That is we assume

$$\overline{\underset{\sim}{A}} = \sum_{e=1}^{n_{el}} \overline{\underset{\sim}{A}}^e \tag{3.21}$$

where $\overline{\underset{\sim}{A}}^e$ is the e^{th} element contribution to $\overline{\underset{\sim}{A}}$. Then $\underset{\sim}{C}$ may be defined by either the one-pass or two-pass formulae, viz.

$$\underset{\sim}{C} = \prod_{e=1}^{n_{el}} (\underset{\sim}{I} + \epsilon \overline{\underset{\sim}{A}}^e) \tag{3.22}$$

$$\underset{\sim}{C} = \prod_{e=1}^{n_{el}} (\underset{\sim}{I} + \frac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e) \quad \prod_{e=n_{el}}^{1} (\underset{\sim}{I} + \frac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e) \qquad \text{("Marchuk EBE")} \qquad (3.23)$$

Remark 1. We wish to use the term element in the generic sense of a "subdomain model", where an element could be an individual finite element or a subassembly of elements. Thus we allow limited assembly. Various equivalent terminologies have been used to define this concept, such as "substructures" and "superelements". Subdomain finite element models inherit the symmetry and definiteness properties of the global array. Consequently, the remark made after (3.20) applies.

Remark 2. The element arrays in (3.22) and (3.23) need to be factorized into triangular form. This can be done exactly using product decompositions or approximately using sum decompositions as in section 3a, equations (3.15)-(3.17):

one-pass

Corresponding to (3.22) we have

$$\underset{\sim}{C} = \prod_{e=1}^{n_{el}} \underset{\sim}{L}_p(\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}}^e)\underset{\sim}{D}_p(\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}}^e)\underset{\sim}{U}_p(\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}}^e) \qquad \text{(product)} \qquad (3.24)$$

or

$$\underset{\sim}{C} = \prod_{e=1}^{n_{el}} (\underset{\sim}{I} + \varepsilon\,\tilde{\underset{\sim}{L}}_s(\overline{\underset{\sim}{A}}^e))\,(\underset{\sim}{I} + \varepsilon\,\tilde{\underset{\sim}{U}}_s(\overline{\underset{\sim}{A}}^e)) \qquad \text{(sum)} \qquad (3.25)$$

Note (3.24) is identical to (3.22) whereas (3.25) is an approximation of (3.22).

two-pass

Corresponding to (3.23) we have

$$\underset{\sim}{C} = \prod_{e=1}^{n_{el}} \underset{\sim}{L}_p(\underset{\sim}{I} + \tfrac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e)\underset{\sim}{D}_p(\underset{\sim}{I} + \tfrac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e)\underset{\sim}{U}_p(\underset{\sim}{I} + \tfrac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e) \times$$

$$\times \prod_{e=n_{el}}^{1} \underset{\sim}{L}_p(\underset{\sim}{I} + \tfrac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e)\underset{\sim}{D}_p(\underset{\sim}{I} + \tfrac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e)\underset{\sim}{U}_p(\underset{\sim}{I} + \tfrac{\varepsilon}{2}\overline{\underset{\sim}{A}}^e) \qquad \text{(product)} \qquad (3.26)$$

or

$$\underset{\sim}{C} = \prod_{e=1}^{n_{el}} (\underset{\sim}{I} + \tfrac{\varepsilon}{2}\tilde{\underset{\sim}{L}}_s(\overline{\underset{\sim}{A}}^e))\ (\underset{\sim}{I} + \tfrac{\varepsilon}{2}\tilde{\underset{\sim}{U}}_s(\overline{\underset{\sim}{A}}^e)) \times$$

$$\times \prod_{e=n_{el}}^{1} (\underset{\sim}{I} + \tfrac{\varepsilon}{2}\tilde{\underset{\sim}{L}}_s(\overline{\underset{\sim}{A}}^e))\ (\underset{\sim}{I} + \tfrac{\varepsilon}{2}\tilde{\underset{\sim}{U}}_s(\overline{\underset{\sim}{A}}^e)) \qquad \text{(sum)} \qquad (3.27)$$

Note (3.26) is identical to (3.23) whereas (3.27) is an approximation of (3.23).

Whether to use product or sum factorizations of the element arrays is a question of efficiency. Belytschko and Liu [B1] have proposed a fast exact inversion procedure for 4-node heat conduction elements. For subassemblies, the approximate sum factorizations may have advantages.

Remark 3. Note that storage demands are vastly less in the EBE case. Only one element at a time need be stored and processed. Whether or not it is desirable to save factorized element arrays depends upon the availability of high speed RAM, and the trade-off between CPU and disk I/O costs.

Remark 4. The ordering of the factors influences how well $\underset{\sim}{C}$ approximates $\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}}$. The global product decomposition,

$$\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}} = \underset{\sim}{L}_p(\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}})\underset{\sim}{D}_p(\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}})\underset{\sim}{U}_p(\underset{\sim}{I} + \varepsilon\,\overline{\underset{\sim}{A}}) \quad , \qquad (3.28)$$

suggests that it might be worthwhile to reorder the factors in (3.24)-(3.27) such that all lower triangular factors precede diagonals which in turn precede upper triangular factors. This results in the following "reordered" schemes:

$$\underset{\sim}{C} = \left[\prod_{e=1}^{n_{el}} \underset{\sim}{L}_p(\underset{\sim}{I} + \varepsilon\, \overline{\underset{\sim}{A}}^e)\right] \left[\prod_{e=1}^{n_{el}} \underset{\sim}{D}_p(\underset{\sim}{I} + \varepsilon\, \overline{\underset{\sim}{A}}^e)\right] \times$$

$$\times \left[\prod_{e=n_{el}}^{1} \underset{\sim}{U}_p(\underset{\sim}{I} + \varepsilon\, \overline{\underset{\sim}{A}}^e)\right] \qquad \text{("Crout EBE")} \qquad (3.29)$$

$$\underset{\sim}{C} = \left[\prod_{e=1}^{n_{el}} (\underset{\sim}{I} + \varepsilon\, \tilde{\underset{\sim}{L}}_s(\overline{\underset{\sim}{A}}^e))\right] \left[\prod_{e=n_{el}}^{1} (\underset{\sim}{I} + \varepsilon\, \tilde{\underset{\sim}{U}}_s(\overline{\underset{\sim}{A}}^e))\right]$$

("symm. Gauss-Seidel EBE") (3.30)

Note that in the case of symmetric $\overline{\underset{\sim}{A}}$, symmetry is preserved by (3.29) and (3.30). Thus there seems little motivation for similarly reordering the two-pass versions.

In the case of positive $\underset{\sim}{D}_p(\underset{\sim}{I} + \varepsilon\, \overline{\underset{\sim}{A}}^e)$'s, the Crout factorizations can be reordered in terms of Cholesky factors. For example, a variant of (3.29) is

$$\underset{\sim}{C} = \left[\prod_{e=1}^{n_{el}} \tilde{\underset{\sim}{L}}_p(\underset{\sim}{I} + \varepsilon\, \overline{\underset{\sim}{A}}^e)\right] \left[\prod_{e=n_{el}}^{1} \tilde{\underset{\sim}{U}}_p(\underset{\sim}{I} + \varepsilon\, \overline{\underset{\sim}{A}}^e)\right]$$

("Cholesky EBE") (3.31)

Note (3.31) and (3.29) are not generally identical.

Remark 5. If elements are segregated into non-contiguous subgroups then calculations are parallelizable. For example, brick-like domains can be decomposed into eight non-contiguous element groups (see Figure 1). Because the elements in each subgroup have no common degrees-of-freedom, they can be processed in parallel. The eight groups, however, need to be processed sequentially. For analogous two-dimensional domains, four element groups need to be employed.

Remark 6. It has been our computational experience that if $\underset{\sim}{A}$ is symmetric and positive-definite, then qualitatively faithful approximate factorizations, which preserve these properties, perform much better than those that do not.

4. Selection of $\underset{\sim}{W}$, ε and $\overline{\underset{\sim}{A}}$.

The following two definitions of $\underset{\sim}{W}$, ε and $\overline{\underset{\sim}{A}}$ have been employed:

a.) This choice is motivated by the derivation of the PR algorithm (see [H13])

$$\underset{\sim}{W} = \underset{\sim}{D}_S(\underset{\sim}{A}) \tag{4.1}$$

$$\overline{\underset{\sim}{A}} = \frac{1}{\varepsilon} \underset{\sim}{W}^{-\frac{1}{2}} \underset{\sim}{A} \underset{\sim}{W}^{-\frac{1}{2}} \tag{4.2}$$

Thus

$$\tilde{\underset{\sim}{A}} = \underset{\sim}{D}_S(\underset{\sim}{A}) + \underset{\sim}{A} \tag{4.3}$$

b.) In this case

$$\underset{\sim}{W} = \underset{\sim}{D}_S(\underset{\sim}{A}) \tag{4.4}$$

$$\overline{\underset{\sim}{A}} = \frac{1}{\varepsilon} \underset{\sim}{W}^{-\frac{1}{2}}(\underset{\sim}{A} - \underset{\sim}{D}_S(\underset{\sim}{A}))\underset{\sim}{W}^{-\frac{1}{2}} \tag{4.5}$$

which leads to

$$\underset{\sim}{A} = \underset{\sim}{A} \tag{4.6}$$

This procedure was proposed in Winget [W1].

Remark 1.

Matrices of the form $\tilde{\underset{\sim}{A}} = \underset{\sim}{D}_s(A) + \epsilon \underset{\sim}{A}$ were introduced in [H5]. Nour-Omid and Parlett [N1] analytically investigated the effectiveness of matrices of this type on a model problem and concluded that the optimal valve of ϵ was ∞. This limit is achieved by the definitions (4.4) and (4.5).

Remark 2.

The implicit-explicit finite element concept [H2, H3, H6-H9] has a very simple and clean implementation within EBE approximate factorizations. Recall that an explicit element contributes only its diagonal mass matrix to the coefficient matrix $\underset{\sim}{A}$. Thus W, according to any one of the preceding definitions, totally accounts for the explicit element contributions and the corresponding $\overline{\underset{\sim}{A}}^e$'s are identically zero. What this means is that explicit elements may be simply omitted from the formula for $\underset{\sim}{C}$. In nonlinear problems this opens the way to time-adaptive implicit-explicit element partitions. In calculating the element contributions to the residual (i.e. "$\underset{\sim}{b}$"), a check can be made whether or not the critical time step is exceeded for the element. If it is not exceeded, a flag is set to indicate that element contributions to $\underset{\sim}{C}$ may be simply ignored. The potential savings in nonlinear transient analysis procedures incorporating these ideas if clearly considerable.

5. Sample Problems

The computed results were obtained on a VAX computer using single precision (32 bits per floating point word). Critical time steps were computed from $\Delta t_{crit} = 2/\lambda_{max}$ where λ_{max} is the maximum element eigenvalue. Unless otherwise noted, bilinear quadrilaterals were employed with 2 × 2 Gauss integration.

NASA Insulated Structure Test Problem

The problem description is illustrated in Figure 2. A number of comparisons of the various techniques proposed were made for this problem. In Table 1 the selection of $\tilde{\underset{\sim}{A}} = \underset{\sim}{A}$ [W1] is seen to converge faster than $\tilde{\underset{\sim}{A}} = \underset{\sim}{D}_s(\underset{\sim}{A}) + \underset{\sim}{A}$. In addition, the steady-state residual potential energy, measured by $\log_{10}(-P_f)$, attains a smaller value when $\tilde{\underset{\sim}{A}} = \underset{\sim}{A}$. This and other calculations have indicated that $\tilde{\underset{\sim}{A}} = \underset{\sim}{A}$ is the superior choice. It is used in the comparisons shown in Table 2. The first observation which may be made here is that the CC algorithm is more effective than PR with line searches. The use of BFGS updates would doubtlessly improve upon the performances of PR, however, the increased data pool required to store the BFGS vectors is a significant disadvantage. Thus our current preference in symmetric positive-definite cases is the CG method. The EBE factorizations, ranked from best to worst, are: Crout, Cholesky, Marchuk, and symmetrized Gauss-Seidel. Nevertheless, it must be kept in mind that overall computational efficiency may alter this ordering. For example, although symmetrized Gauss-Seidel was the slowest to converge, it does not require element factorization, an advantage. A final point to observe is that convergence is typically slower during the larger time step sequences (i.e. steps 21-50) than the smaller step sequences (i.e. steps 1-20). There appear to be two reasons for this. Firstly, for the larger steps the solution closely

approximates the steady state. Thus the initial residual is fairly small, resulting in a more stringent convergence criterion. (A more reasonable convergence criterion would no doubt result in faster termination for the larger steps than for the smaller. In fact, even the "non-converged" solutions possessed adequate accuracy from a practical standpoint.) Secondly, the conditioning of element factors deteriorates for larger steps in that the element arrays become nearly singular.

Table 1. Comparison of $\tilde{A} = D_s(A) + A$ with $\tilde{A} = A$.

Algorithm: PR + LS (no BFGS)

Approximate factorization: Marchuk EBE

$\tilde{A}$	$\log_{10}(-P_f)^{(\dagger)}$	ave. it's. per step	
		steps 1-20	steps 21-50(‡)
$D_s(A) + A$	- 13.0	6.0	10
A	- 15.4	5.03	10

Table 2. Comparison of PR and CG algorithms and various EBE approximate factorizations. In each case $\tilde{A} = A$.

Algorithm: PR + LS (no BFGS)

EBE approx. fact.	$\log_{10}(-P_f)^{(\dagger)}$	ave. it's. per step	
		steps 1-20	steps 21-50(‡)
symm. Gauss-Seidel	- 13.4	5.41	10
Marchuk	- 15.4	5.03	10
Cholesky	- 15.8	3.95	10
Crout	- 15.1	3.95	10

Algorithm: CG

EBE approx. fact.	$\log_{10}(-P_f)^{(\dagger)}$	ave. it's. per step	
		steps 1-20	steps 21-50(§)
symm. Gauss-Seidel	- 25.3	3.95	9.0
Marchuk	- 25.3	3.50	8.3
Cholesky	- 25.3	3.45	7.9
Crout	- 25.3	2.95	8.0

Notes:

(†) P_f, the final value of potential energy, is minimized by the exact steady-state solution. Consequently, the more negative $\log_{10}(-P_f)$, the better the approximation of the steady-state.

(‡) The maximum number of iterations was 10. The PR algorithm failed to converge for steps 21-50.

(§) The CG algorithm converged in less than or equal to 10 iterations in all cases.

Parallel/Sequential Test Problem

The problem description is given in Figure 3. The purpose of this problem is to compare convergence characteristics for "natural" element orderings, which necessitate sequential processing, with orderings that lend themselves to parallel computations. The comparisons were all performed with the CG algorithm, $\tilde{\underline{A}} = \underline{A}$ and the Cholesky EBE approximate factorization.

Over the thirty time steps the sequential ordering averaged 2.53 iterations per step to attain convergence, whereas the parallel ordering averaged 3.47 iterations. Despite the fact that the parallel ordering is slower, which might be anticipated, the fact that it is reasonably fast is extremely encouraging. For the 256 element mesh shown a 64-processor computer could attain speeds 64 times faster than a single processor. This more than compensates for the somewhat slower convergence of the parallel ordering. The gains in larger problems are potentially even more spectacular.

6. Conclusions

The numerical comparisons between the PR and CG algorithms indicate that the CG algorithm is superior to PR. It is likely that BFGS updates would have improved the performance of PR, however, the need to store the BFGS vectors is considered a serious drawback when compared with the small, fixed-storage requirements of the CG algorithm.

Among the EBE approximate factorizations, the Crout variant seemed best, however, the Cholesky, Marchuk, and symmetrized Gauss-Seidel versions were also effective and thus a preference for one over another may need to be based on other computational considerations.

The calculations comparing parallel and sequential orderings are very exciting. The preliminary indications are that parallel processing with EBE factorizations may be a very efficient computational strategy.

Although a number of possible improvements may still be envisioned, we believe that the methodology developed is at a stage where it may be incorporated as an option in production heat conduction codes. Future hardware developments, such as parallel multi-processor computers, promise to further enhance the performance of EBE techniques on large problems.

References

A1. W. F. Ames, Numerical Methods for Partial Differential Equations, second edition, Academic Press, New York, 1977.

B1. T. Belytschko and W. K. Liu, "On Reduced Matrix Inversion for Operator Splitting Methods," preprint.

B2. A. Brandt, "Guide to Multigrid Development," in Multigrid Methods (W. Hackbusch and U. Trottenberg, eds.) Springer-Verlag, Berlin-Heidelberg, New York, 1982.

G1. R. Glowinski, J. Periaux and Q. V. Dinh, "Domain Decomposition Methods for Nonlinear Problems in Fluid Dynamics," INRIA Report No. 147, July 1982.

G2. R. Glowinski, Numerical Methods for Nonlinear Variational Problems, second edition, Springer Series in Computational Physics, Springer-Verlag, New York-Heidelberg-Berlin, to appear.

H1. M. R. Hestenes and E. Steifel, "Methods of Conjugate Gradients for Solving Linear Systems," Journal of Research of National Bureau of Standards, Vol. 49, No. 6, 409-436, December 1952.

H2. T.J.R. Hughes, "Implicit-explicit Finite Element Techniques for Symmetric and Non-symmetric Systems," pp. 255-267 in Recent Advances in Non-linear Computational Mechanics, (ed. E. Hinton, D.R.J. Owen and C. Taylor) Pineridge Press, Swansea, U.K., 1982.

H3. T.J.R. Hughes, "Analysis of Transient Algorithms with Particular Reference to Stability Behavior," Computational Methods in Transient Analysis, Eds. T. Belytschko and T.J.R. Hughes, North-Holland Publishing Co., Amsterdam, 1983.

H4. T.J.R. Hughes, I. Levit and J. Winget, "Implicit, Unconditionally Stable, Element-by-element Algorithms for Heat Conduction Analysis," Journal of the Engineering Mechanics Division, ASCE, Vol. 109, No. 2, 576-585, April 1983.

H5. T.J.R. Hughes, I. Levit and J. Winget, "An Element-by-element Solution Algorithm for Problems of Structural and Solid Mechanics," Computer Methods in Applied Mechanics and Engineering, Vol. 36, No. 2, 241-254 (1983).

H6. T.J.R. Hughes and W. K. Liu, "Implicit-Explicit Finite Elements in Transient Analysis: Stability Theory," Journal of Applied Mechanics, Vol. 45, 371-374 (1978).

H7. T.J.R. Hughes and W. K. Liu, "Implicit-Explicit Finite Elements in Transient Analysis: Implementation and Numerical Examples," Journal of Applied Mechanics, Vol. 45, 375-378 (1978).

H8. T.J.R. Hughes, K. S. Pister, and R. L. Taylor, "Implicit-Explicit Finite Elements and Nonlinear Transient Analysis," Computer Methods in Applied Mechanics and Engineering, Vols. 17/18, 159-182 (1979).

H9. T.J.R. Hughes and R. A. Stephenson, "Convergence of Implicit-Explicit Algorithms in Nonlinear Transient Analysis," International Journal of Engineering Science, Vol. 19, No. 2, 295-302 (1981).

H10. T.J.R. Hughes, J. Winget and I. Levit, "Element-by-element Solution Procedures for Nonlinear Structural Analysis," Proceedings of the NASA Lewis Workshop, April 19-20, 1982.

H11. T.J.R. Hughes, J. Winget, I. Levit and T. E. Tezduyar, "New Alternating Direction Procedures in Finite Element Analysis based upon EBE Approximate Factorizations," Symposium on Recent Developments in Computer Methods for Nonlinear Solid and Structural Mechanics, ASME Meeting, University of Houston, Houston, Texas, June 20-22, 1983.

M1. G. I. Marchuk, Methods of Numerical Mathematics, Springer-Verlag, New York-Heidelberg-Berlin, 1975.

N1. B. Nour-Omid and B. N. Parlett, "Element Preconditioning," Report PAM-103, Center for Pure and Applied Mathematics, University of California, Berkeley, October 1982.

O1. M. Ortiz, P. M. Pinsky and R. L. Taylor, "Unconditionally Stable Element-by-element Algorithms for Dynamic Problems", Computer Methods in Applied Mechanics and Engineering, Vol. 36, No. 2, 223-239 (1983).

P1. K. C. Park, "Semi-Implicit Transient Analysis Procedure for Structural Dynamics Analysis," Int. J. for Num. Meth. in Engng., 18 604-622 (1982).

T1. F. Thomasset, Implementation of Finite Element Methods for Navier-Stokes Equations, Springer-Verlag, New York-Heidelberg-Berlin, 1981.

T2. D. M. Trujillo, "An Unconditionally Stable, Explicit Algorithm for Finite-Element Heat Conduction Analysis", Journal of Nuclear Engineering and Design, 41, 175-180, 1977.

T3. D. M. Trujillo, "An Unconditionally Stable, Explicit Algorithm for Structural Dynamics," International Journal for Numerical Methods in Engineering, Vol. 11, 1579-1592 (1977).

W1. J. Winget, Ph.D. Thesis, California Institute of Technology, Pasadena, forthcoming.

Appendix I - Derivation of Linear Algebraic Systems in the Finite Element Analysis of Heat Conduction Problems

Consider the following semi-discrete system:

$$\underset{\sim}{M}\,\underset{\sim}{a} + \underset{\sim}{C}\,\underset{\sim}{v} = \underset{\sim}{F} \tag{I.1}$$

where $\underset{\sim}{M}$ is the capacity matrix, $\underset{\sim}{C}$ is the conductivity matrix, $\underset{\sim}{F}$ is the heat-supply vector, $\underset{\sim}{v}$ is the temperature vector and $\underset{\sim}{a} = \dot{\underset{\sim}{v}}$ is the time-rate-of-temperature vector. We assume that $\underset{\sim}{M}$ and $\underset{\sim}{C}$ are symmetric and positive definite, and they may depend upon $\underset{\sim}{v}$ and t (time). We employ a predictor-multicorrector method to time-discretize the system. To this end we define the temperature "predictor" by

$$\tilde{\underset{\sim}{v}}_{n+1} = \underset{\sim}{v}_n + \Delta t(1 - \gamma)\underset{\sim}{a}_n \tag{I.2}$$

where subscripts refer to the step number; Δt is the time step; $\underset{\sim}{v}_n$ and $\underset{\sim}{a}_n$ are the approximations to $\underset{\sim}{v}(t_n)$ and $\underset{\sim}{a}(t_n)$, respectively; and γ is a parameter governing stability and accuracy characteristics of the algorithm [H3]. Calculations begin with the initial data $\underset{\sim}{v}_0$ and $\underset{\sim}{a}_0$; $\underset{\sim}{a}_0$ may be calculated from

$$\underset{\sim}{M}_0\,\underset{\sim}{a}_0 = \underset{\sim}{F}_0 - \underset{\sim}{K}_0\,\underset{\sim}{d}_0 \tag{I.3}$$

In each time step a nonlinear algebraic problem arises which may be solved by Newton-type iterative procedures:

$$i = 0 \qquad \text{(i is the iteration counter)} \tag{I.4}$$

$$\left.\begin{aligned} \underset{\sim}{v}_{n+1}^{(i)} &= \tilde{\underset{\sim}{v}}_{n+1} \\ \underset{\sim}{a}_{n+1}^{(i)} &= \underset{\sim}{0} \end{aligned}\right\} \text{(predictor phase)} \qquad \text{(I.5), (I.6)}$$

$$\underset{\sim}{R} = \underset{\sim}{F}_{n+1} - \underset{\sim}{C}_{n+1}^{(i)} \underset{\sim}{v}_{n+1}^{(i)} - \underset{\sim}{M}_{n+1}^{(i)} \underset{\sim}{a}_{n+1}^{(i)} \qquad \text{(residual)} \qquad \text{(I.7)}$$

$$\underset{\sim}{M}^{*} = \underset{\sim}{M}_{n+1}^{(i)} + \gamma\Delta t \, \underset{\sim}{C}_{n+1}^{(i)} \qquad \text{(effective capacity)} \qquad \text{(I.8)}$$

$$\boxed{\underset{\sim}{M}^{*} \, \Delta\underset{\sim}{a} = \underset{\sim}{R}} \qquad \text{(I.9)}$$

$$\left.\begin{aligned} \underset{\sim}{a}_{n+1}^{(i+1)} &= \underset{\sim}{a}_{n+1}^{(i)} + \Delta\underset{\sim}{a} \\ \underset{\sim}{v}_{n+1}^{(i+1)} &= \underset{\sim}{v}_{n+1}^{(i)} + \gamma\Delta t \, \Delta\underset{\sim}{a} \end{aligned}\right\} \text{(corrector phase)} \qquad \text{(I.10), (I.11)}$$

If additional iterations are to be performed, i is replaced by $i+1$, and calculations resume with (I.7). Either a fixed number of iterations may be performed, or iterating may be terminated when $\Delta\underset{\sim}{a}$ and/or $\underset{\sim}{R}$ satisfy preassigned convergence conditions. When the iterative phase is completed, the solution at step $n+1$ is defined by the last iterates (viz. $\underset{\sim}{v}_{n+1} = \underset{\sim}{v}_{n+1}^{(i+1)}$ and $\underset{\sim}{a}_{n+1} = \underset{\sim}{a}_{n+1}^{(i+1)}$). At this point, n is replaced by $n+1$, and calculations for the next time step may begin.

So-called implicit-explicit element partitions [H2, H3, H6-H9] may be encompassed by the above formulation simply be excluding explicit element contributions from $\underset{\sim}{C}$. A totally explicit formulation is attained by ignoring $\underset{\sim}{C}$. In these cases it is necessary to employ a diagonal capacity matrix in explicit regions to attain full computational efficiency.

To simplify the writing in the body of this paper we adopt the following

notations in place of (I.9):

$$\underset{\sim}{A}\,\underset{\sim}{x} = \underset{\sim}{b} \tag{I.12}$$

Thus during each step, at each iteration, we wish to solve (I.12) in which $\underset{\sim}{A}$ is assembled from element arrays, that is

$$\underset{\sim}{A} = \sum_{e=1}^{n_{el}} \underset{\sim}{A}^{e} \tag{I.13}$$

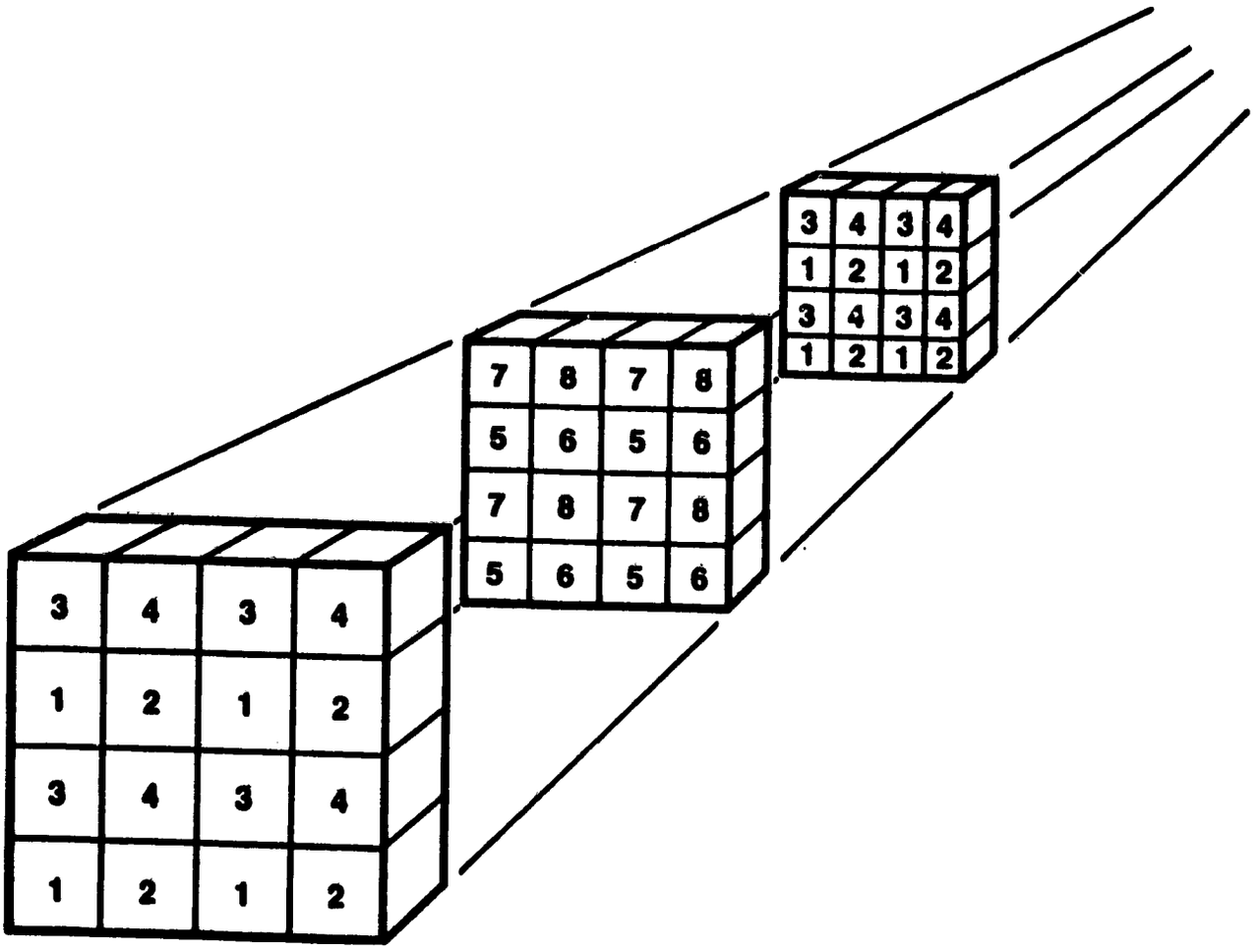
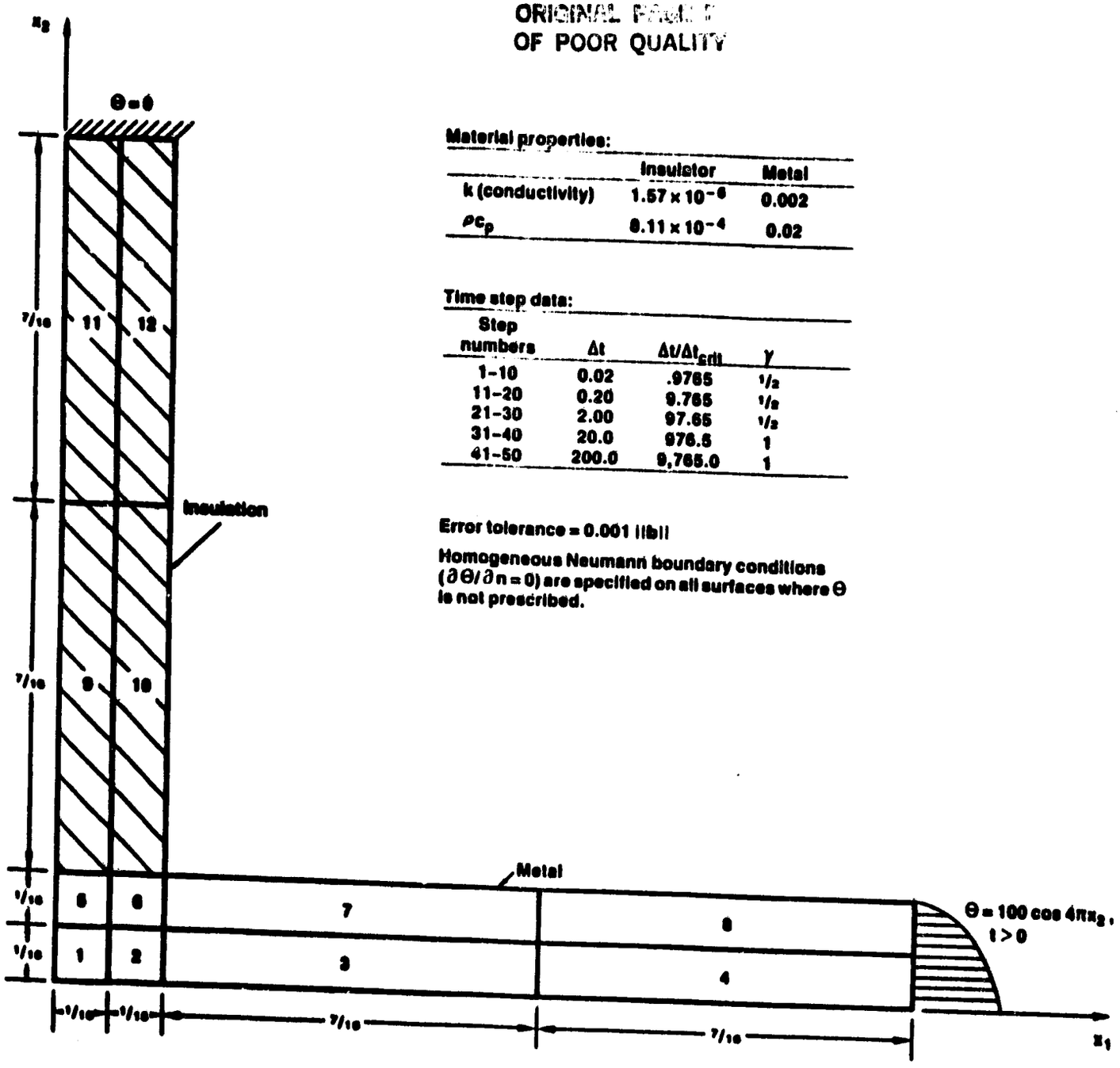


Figure 1. Decompostion of three-dimensional domain into eight groups of brick elements for parallel processing.



Material properties:

	Insulator	Metal
k (conductivity)	1.57×10^{-6}	0.002
ρc_p	8.11×10^{-4}	0.02

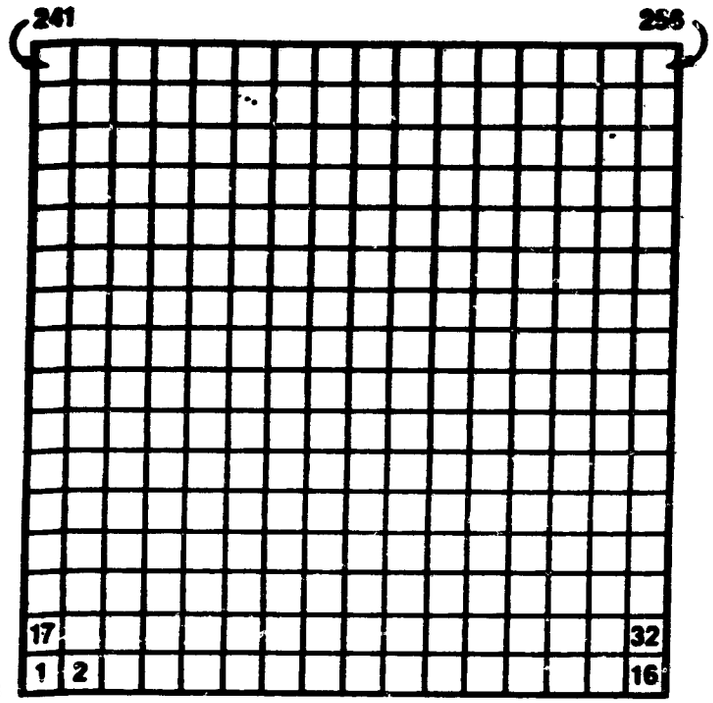
Time step data:

Step numbers	Δt	$\Delta t/\Delta t_{crit}$	γ
1-10	0.02	.9765	1/2
11-20	0.20	9.765	1/2
21-30	2.00	97.65	1/2
31-40	20.0	976.5	1
41-50	200.0	9,765.0	1

Error tolerance = 0.001 ||b||

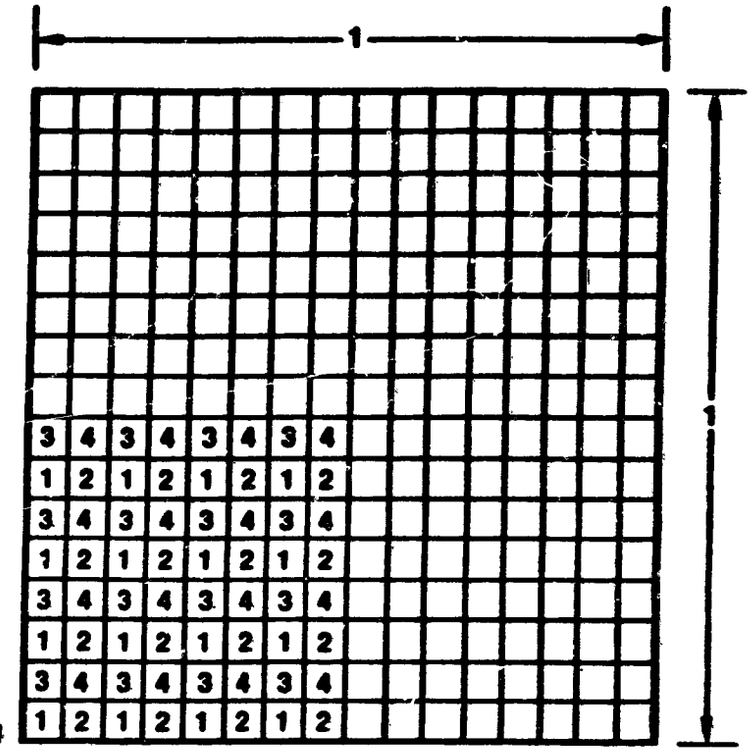
Homogeneous Neumann boundary conditions ($\partial\Theta/\partial n = 0$) are specified on all surfaces where Θ is not prescribed.

Figure 2. Problem description for NASA insulated structure test problem.



Element numbers

Natural element ordering
for sequential computation



Element group numbers, typical

Decomposition into four element
groups for parallel computation

Initial condition: $\Theta = 0$
Boundary condition: $\Theta = 1, t > 0$
Material properties: $k = 1, \rho c_p = 1$

Time step data:

Step numbers	$\Delta t/\Delta t_{crit}$
1–10	1
11–20	10
21–30	100

Figure 3. Problem description for parallel/sequential comparison