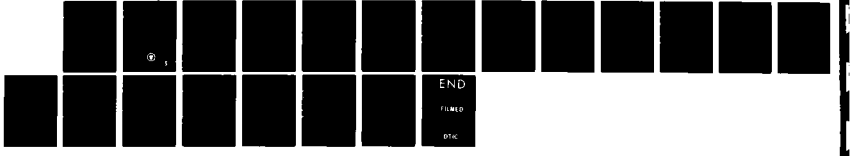
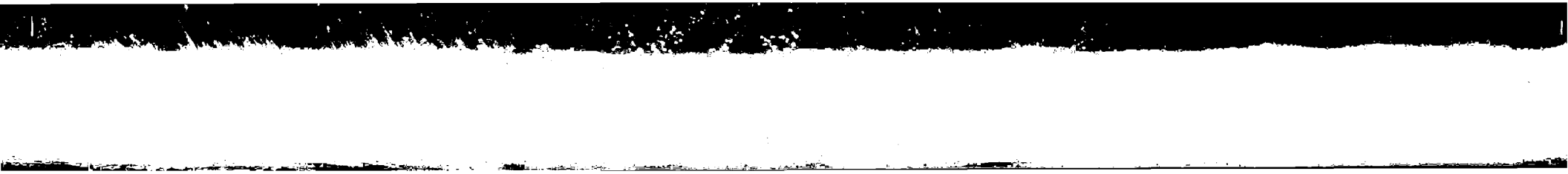
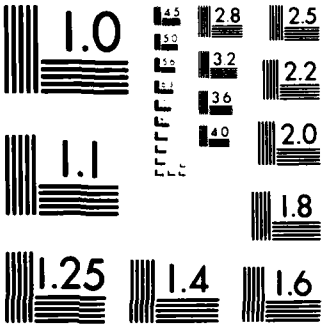
AD-A148 880 A PROGRAM DESIGN FOR AN ADAPTIVE NON-LINEAR FINITE ELEMENT SOLVER(U) PITTSBURGH UNIV PA INST FOR COMPUTATIONAL MATHEMATICS AND APP.. F R SLEDGE ET AL. 1/1

UNCLASSIFIED 1984 ICMA-84-68 N00014-80-C-0455 F/G 9/2 NL







(15)

INSTITUTE FOR COMPUTATIONAL MATHEMATICS AND APPLICATIONS

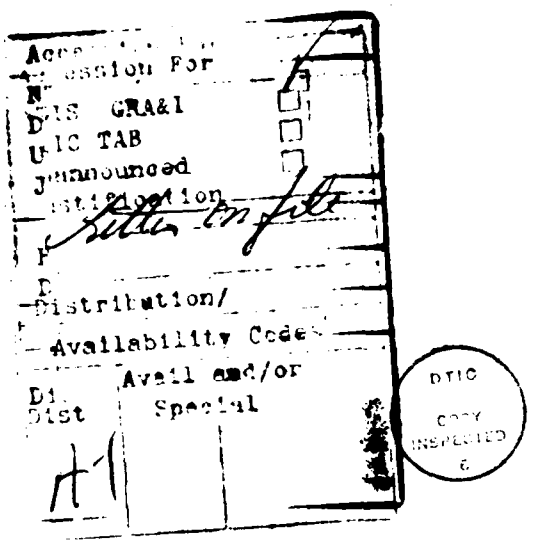
Technical Report ICMA-84-68

A PROGRAM DESIGN FOR AN ADAPTIVE, NON-LINEAR FINITE ELEMENT SOLVER[+)]

by

Frank R. Sledge and Werner C. Rheinboldt

Department of Mathematics and Statistics
University of Pittsburgh



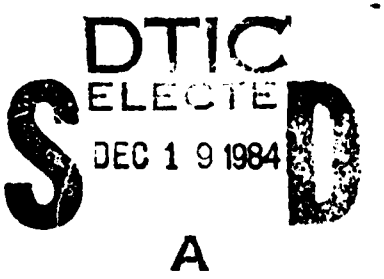
Technical Report ICMA-84-68

A PROGRAM DESIGN FOR AN ADAPTIVE, NON-LINEAR FINITE ELEMENT SOLVER[+)]

by

Frank R. Sledge and Werner C. Rheinboldt

Department of Mathematics and Statistics
University of Pittsburgh
Pittsburgh, PA 15261



+) This work was supported in part by the Office of Naval Research under Contract N00014-80-C-0455.

A PROGRAM DESIGN FOR AN ADAPTIVE, NON-LINEAR FINITE ELEMENT SOLVER[+)]

by

Frank R. Sledge and Werner C. Rheinboldt

Department of Mathematics and Statistics
University of Pittsburgh
Pittsburgh, PA 15261

1. INTRODUCTION

An experimental software system for the solution of a class of non-linear, stationary boundary-value problems is currently under development at the University of Pittsburgh. The program NFEARS (Non-linear Finite-Element Adaptive Research Solver) is a further development of the program FEARS [1-3], which utilized bilinear elements to solve linear elliptic problems. NFEARS retains the functionality of the earlier program, but incorporates a continuation procedure to solve non-linear problems, using biquadratic Hermitian elements. The NFEARS design properties include the following:

(1) The system constitutes an applications-independent finite-element solver for a certain class of two-dimensional, non-linear, stationary, boundary-value problems defined by a weak mathematical formulation.

(2) Adaptive approaches are employed extensively. The a posteriori error indicators developed in [2] are used to control the adaptive processes and to provide a solution with near optimal error within a prescribed cost range.

[+)] This work was supported in part by the Office of Naval Research under Contract N-00014-80-C-0455.

(3) In the system design, advantage was taken of the inherent parallelism and modularity of the finite element method. In particular, a two-level data structure has been employed to take maximum advantage of the parallelism in the continuation process.

(4) The system is highly modular in structure, reflecting not only the natural separation by distinct function, but also the isolation of those processes, particularly error analysis, which are anticipated to be of the greatest experimental interest. In this way, the migration of NFEARS from convenient research vehicle to efficient production tool will be gradual and controlled. Extensive provisions for evaluating the performance are incorporated.

A principle feature of (2) is an adaptive algorithm for mesh refinement and/or derefinement (or simply (de)refinement). Briefly, after a solution has been obtained on some mesh, error indicators are computed on each individual element. These indicators are used, both individually and in patterns, to compose an estimate of the error in an appropriate norm, and to specify what, if any, changes are to be made to the mesh. The (de)refinement algorithm in essence divides elements and/or consolidates others so as to achieve a more equal distribution of error indicators.

This paper presents the overall structure of NFEARS. The weak formulation for the admissible class of problems is given in Section 2. Sections 3 and 4 discuss the design criteria of, respectively, the control and data structures of the program.

2. MATHEMATICAL BASIS OF THE DESIGN

By necessity our attention had to be restricted to a specific class of problems that is sufficiently broad to be of interest both for research and practical appli-

cations, yet narrow enough to allow for easy implementation as an experimental vehicle. Our choice was a fairly general class of non-linear, stationary boundary-value problems on certain two-dimensional domains which admits also many linear elliptic problems as special cases. Throughout the design, we have refrained from using approaches which would limit extension to more general problems.

The permissible domains are of the same type as in FEARS [1-3]. The domain Ω in $\mathbb{R}^2$ is taken to be an open, connected, and simply connected set with boundary $\partial\Omega$. We denote points of $\bar{\Omega}$ by $\bar{x} := (x_1, x_2)^T$. The system will have two modules, namely, for the solution of problems with one and two unknowns, respectively. In the case of one unknown function, we seek a function $u(\bar{x})$ defined on $\bar{\Omega}$ such that

(i) u is a stationary point of the functional

$$\int_\Omega \Phi(I^{[1]}, I^{[2]}, \lambda_1, \lambda_2) d\bar{x} = \tag{2.1}$$

$$\lambda_1 \int_\Omega u\, f_1(\bar{x}) d\bar{x} + \lambda_2 \int_\Omega u\, f_2(\bar{x}) d\bar{x}$$

$$+ \lambda_1 \int_{\partial\Omega} u\, g_1(s) ds + \lambda_2 \int_{\partial\Omega} u\, g_2(s) ds$$

where $d\bar{x} := dx_1 dx_2$ and

$$I^{[1]} = I^{[1]}\left(\frac{\partial u}{\partial x_1} + \frac{\partial u}{\partial x_2}\right) \tag{2.2}$$

and

$$I^{[2]} = I^{[2]}(u) \tag{2.3}$$

are invariants with respect to the rotation of coordinate axes.

(ii) If Γ_0, Γ_1, and Γ_2 are appropriately chosen subsets of $\partial\Omega$, then u satisfies boundary conditions of the form

$$u = \lambda_1 h_1(\bar{x}) + \lambda_2 h_2(\bar{x}) \quad \text{on} \quad \Gamma_o \tag{2.4}$$

or

$$\frac{\partial u}{\partial n} = \lambda_1 k_1(\bar{x}) + \lambda_2 k_2(\bar{x}) \quad \text{on} \quad \Gamma_1 \tag{2.5}$$

or

$$u + \frac{\partial u}{\partial n} = \lambda_1 (h_1 + k_1) + \lambda_2 (h_2 + k_2) \quad \text{on} \quad \Gamma_2 \tag{2.6}$$

where n is the outward pointing unit normal vector on $\partial\Omega$.

Similarly, in the case of two unknown functions, we seek a vector function $\bar{u} = (u_1, u_2)^T$ defined on $\bar{\Omega}$, such that again conditions of the form (i) and (ii) hold. In (i) the functional (2.1) is replaced by

$$\int_\Omega \Phi(I^{[1]}, I^{[2]}, I^{[3]}, \lambda_1, \lambda_2) d\bar{x} = \tag{2.7}$$

$$\lambda_1 \int_\Omega \bar{u} \cdot \bar{f}_1(\bar{x}) d\bar{x} + \lambda_2 \int_\Omega \bar{u} \cdot \bar{f}_2(\bar{x}) d\bar{x}$$

$$+ \lambda_1 \int_{\partial\Omega} \bar{u} \cdot \bar{g}_1(s) ds + \lambda_2 \int_{\partial\Omega} \bar{u} \cdot \bar{g}_2(s) ds$$

and the invariants become

$$\begin{aligned} I^{[1]} &= I^{[1]} \left(\frac{\partial u_1}{\partial x_1}, \frac{\partial u_1}{\partial x_2}, \frac{\partial u_2}{\partial x_1}, \frac{\partial u_2}{\partial x_2}\right) \\ I^{[2]} &= I^{[2]} \left(\frac{\partial u_1}{\partial x_1}, \frac{\partial u_1}{\partial x_2}, \frac{\partial u_2}{\partial x_1}, \frac{\partial u_2}{\partial x_2}\right) \\ I^{[3]} &= I^{[3]} (\bar{u}). \end{aligned} \tag{2.8}$$

In (ii) the scalar functions h_1, h_2, k_1, and k_2 are likewise replaced by vector functions.

This class of problems is fairly general and includes, in particular, most of the basic problems in elasticity theory. Two parameters, λ_1 and λ_2, are incorporated into the formulation, hence the equilibrium surface of the problem under study is two-dimensional.

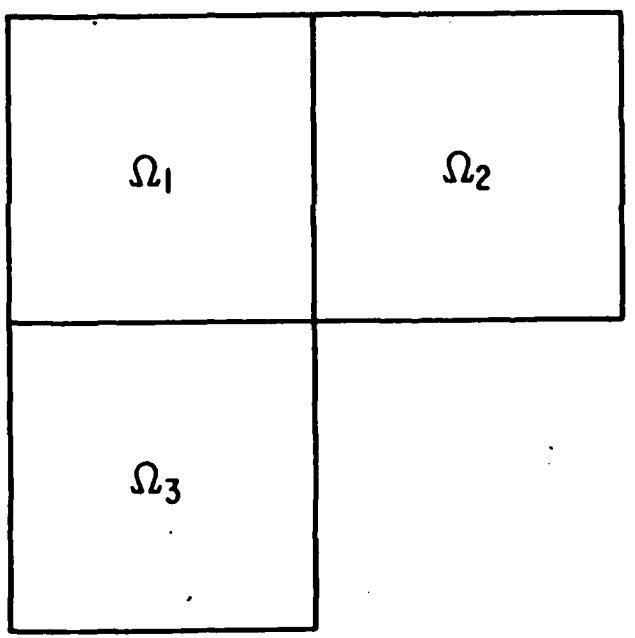
As in FEARS, the domain $\bar{\Omega}$ is defined as a union of subdomains Ω_i, upon each of which is introduced a finite-element mesh. In order to minimize the amount of redundant data, we work, as far as possible, with non-intersecting open subdomains. The set of discrete points necessary to close the union of these open subdomains is treated as a separate case. Figure 1 illustrates how this process works for a simple (straightlined) domain. Here the domain $\bar{\Omega}$ is decomposed into a union of

(1) Three 2-dimensional open subdomains (or 2-subdomains), here indexed as 2-1 through 2-3,

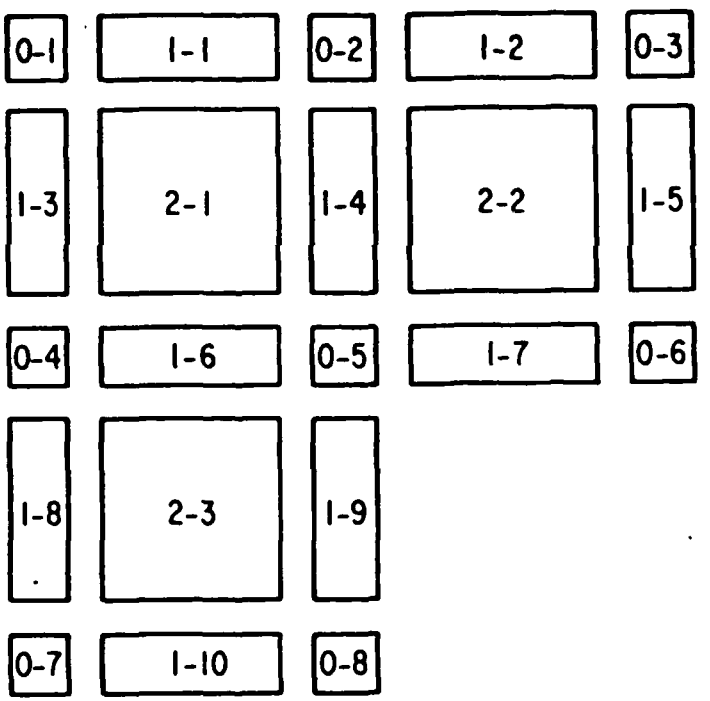
(2) Ten 1-dimensional open line segments (or 1-subdomains) on the boundaries of the 2-subdomains, here indexed as 1-1 through 1-10, so that each 1-subdomain is on the boundary of at most 2 2-subdomains, and,

(3) A set of eight discrete points, or 0-subdomains, here indexed as 0-1 through 0-8, which close the decomposition. Each 0-subdomain is thus on the boundary if at most 4 2-subdomains and at most 4 1-subdomains.

For the design of a reasonably efficient mesh refinement algorith, further restrictions on the choice of the subdomains are desirable. We assume that the closure $\overline{\Omega_i}$ of 2-subdomain Ω_i is a diffeomorphic image of the closed unit square, $\overline{Q_0}$, on which a simple hierarchy of subdivisions can be defined. Moreover, the closure of each 1-subdomain is assumed to be a diffeomorphic image of the unit interval $[0,1]$.



domain



subdomain processes

FIGURE 1

The mesh on each $\overline{\Omega_i}$ consists of curvilinear elements which are first defined on Q_0 and then mapped onto Ω_i. Then an admissible mesh on $\overline{Q_0}$ is defined as a collection M of closed squares $\bar{Q}$ in $\overline{Q_0}$ which are generated by recursive application of the two rules:

(2.9)

(1) The mesh $M = M_0$ consisting only of $\overline{Q_0}$ itself is admissible.

(2) If M is an admissible mesh on $\overline{Q_0}$, then the mesh M' that is obtained from M by subdividing any one closed square $\bar{Q}$ of M into four congruent squares of half the side length of $\bar{Q}$ is admissible.

A typical mesh on $\overline{Q_0}$ generated in this way is shown in Figure 2. (For clarity, some of the mid-side and center nodes of the biquadratic Hermitian element have been omitted.) The refinement introduces two types of intersection point. We call a point *regular* if it is a corner of all undivided squares incident with it; all others are termed *irregular*. All points on the boundary of $\overline{Q_0}$ are always regular. In Figure 2, the irregular points are marked by small circles. In order to obtain continuity of the solution at irregular points, we constrain the solution to take on values interpolated from nearby regular points. Thus the irregular point no longer carry independent information of their own, and must be treated separately. In Figure 2, the elemental stiffness matrix K if the hatched element depends on the 5 numbered regular points. If K_0 represents the usual stiffness matrix of that element, then $K := L^T K_0 L$, where L is an interpolation matrix whose entries depend only on the mesh M on $\overline{Q_0}$.

The decomposition of Ω makes evident a natural parallelism in the incremental construction and solution of the macro-stiffness system. The overall macro-stiffness matrix has the form

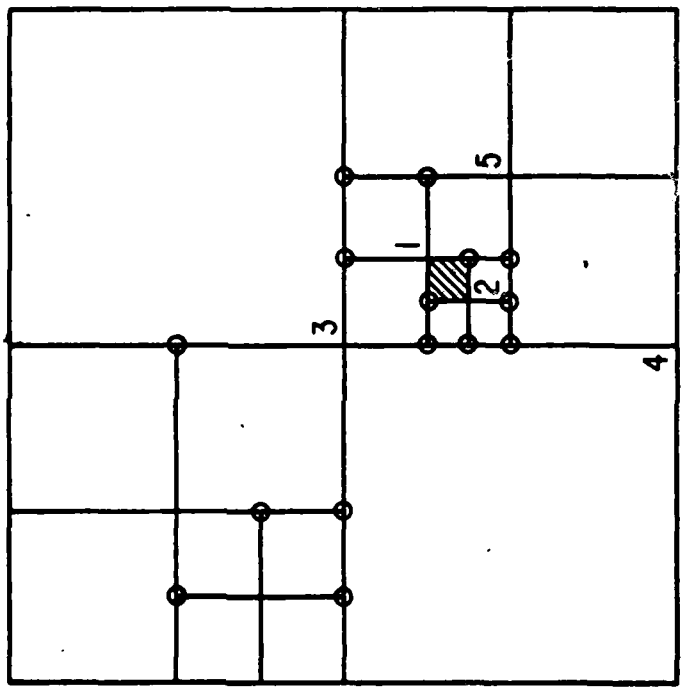


FIGURE 2

$$\begin{bmatrix} A_1 & & & 0 & C_1 \\ & \ddots & & & \vdots \\ 0 & & \ddots & & \vdots \\ & & & A_N & C_N \\ C_1^T & \cdots & \cdots & C_N^T & B \end{bmatrix} \tag{2.10}$$

where A_i corresponds to the interior of the 2-subdomain Ω_i, C_i corresponds to the 0- and 1-subdomains on the boundary of Ω_i, and B contains information on all, but only, the 0- and 1-subdomains. Using a symmetric-decomposition solution method, we may perform the following steps for each Ω_i in parallel

(2.11)

(i) Generate the matrices A_i and C_i.

(ii) Generate the contributions to the matrix B and send them to a designated linear solver-process.

(iii) Compute the decomposition of A_i and the corresponding modification of C_i.

(iv) Compute the resulting modification of B and send it to the solver-process.

A similar procedure applies to the processing of the right-hand side. (Less optimistically, the Ω_i may be processed serially, but even here, the inherent parallelism implies that the order of processing is irrelevant.) Upon completion of the parallel construction/decomposition, the matrix B is decomposed and the corresponding portion of the solution obtained. The remainder of the solution, in the interiors of the Ω_i, is obtained by back-substitution for each Ω_i, and is once again a parallel process.

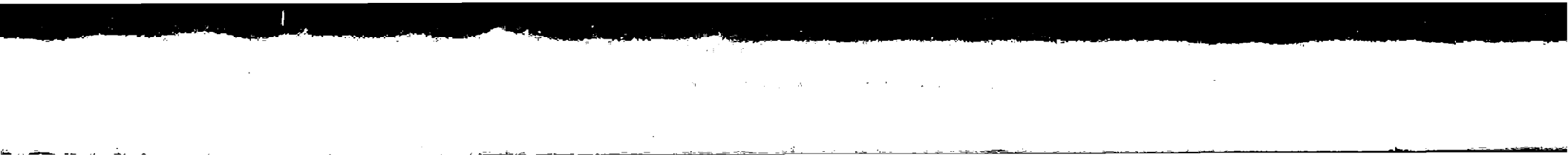
The ability of this scheme to deal with multiple right-hand sides without altering the matrix decomposition is of particular importance in the continuation

process, since it allows us to use modified rather than full Newton refinement. The macro-stiffness matrix is calculated only once in each continuation step, at the starting solution point for the step. The refinement of the predicted solution then uses this same matrix, resulting in a marked improvement in efficiency. Full Newton refinement remains possible, of course, if it is needed.

3. NFEARS CONTROL STRUCTURE

To discuss the control and data structures of the program NFEARS, it is first necessary to discuss the philosophy of the error analysis in the continuation process. The mathematical basis of the error analysis is discussed in [4], and the various aspects of its implementation are, at root, the raison d'etre for NFEARS as a research tool in the first place. For this reason, the error analysis portions must be implemented so as to provide maximum flexibility and eash of experimentation. However, error analysis in NFEARS does not occupy a readily isolated module, but rather, various aspects of it occur throughout the code. Those aspects of NFEARS which are not under such intense research scrutiny, such as continuation per se, Newton refinement and linear solution, are still of sufficient complexity and cost that efficient coding is essential. Thus, the exigencies of error analysis determine not only that portion of the code, but dictate design criteria for all of NFEARS. To the extent that speed and ease of alteration of code are conflicting design goals, the primary design criterion for any portion of code will largely rest on whether that code is, or is not, part of the error analysis.

The global structure of the NFEARS design is such that, once some particular mesh has been specified and a solution point on that mesh calculated, several continuation steps may occur before the error analysis deems mesh alterations necessary. Such changes are dictated by the error indicators calculated for each element at each step, and will involve either refinement of the mesh if the errors



subordinate to the executive.

The mesh management module is responsible for the creation and maintenance of a disk data structure reflecting the current mesh and current solution. As noted, changes to the mesh are anticipated to be relatively rare in relation to continuation steps on a given mesh. However, when mesh changes do occur, their effect on the total NFEARS data structure will be extensive. The data structure used to store mesh information is a straightforward extension of the tree structure used in FEARS, which is described in [3]). Additionally, whenever the mesh is changed, a Transient Data Set (see below) is created afresh, containing mesh-dependent data in a form most efficiently available to the continuation module.

Within each continuation step, error indicators are calculated on each element, as are summary error indicators on each 2-subdomain and for the region as a whole. These indicators depend on the new solution point, and are calculated as soon as the new solution is available. Their purpose is two-fold, and it is important to delineate the two functions. On the one hand, it must be ascertained whether the current mesh has become untenable, and must be (de)refined. This is, in essence, a go-no-go decision to be made after each continuation step. It depends not simply on individual error indicators but on patterns of errors within each 2-subdomain. This error pattern analysis is thus a submodule of the FMC process.

On the other hand, once the current mesh has been deemed unacceptable, it still remains to specify (and effect) precisely what changes must be made to the mesh. This logically separate step occurs in the (de)refinement specification module. This module is yielded control only after an FMC episode has been terminated by directive from the error pattern analysis. In turn, the mesh management module will then effect the specified changes, and the executive will calculate a new starting solution and begin another FMC episode.

Apart from the error pattern analysis submodule, the FMC module poses somewhat the converse design criteria than the error analysis. Continuation on a fixed mesh

has already been researched (see PITCON [5,6]) and, subject to minor and anticipated variations in technique, is not considered to be of major research interest in NFEARS. This module, then, as well as the linear solver and user-function evaluation submodules subordinate to it, are optimized for speed, even if at some expense in size, clarity and flexibility. Of particular importance is that mesh-dependent data, unchanging during several continuation steps, is calculated only once and is available in a sequential, unstructured stream. This data, the Transient Data Set, is described in the Section 4 below.

Repeated continuation steps within an FMC episode will occur until one of the following sequential conditions arises:

(1) The target values of the parameters λ_1 and λ_2 are achieved, with no contro-indications in terms of error or cost control measures,

(2) Any of the cost-control parameters are violated, such as the maximum number of continuation steps or the minimum acceptable step size,

(3) The error pattern analysis indicates that the current mesh has become untenable, requiring refinement in some portions of the region and/or derefinement in others.

(4) Unanticipated serious errors, such as the singularity of the Jacobian/continuation matrix.

At the conclusion of an FMC episode, control is yielded either to the (de)refinement specification and mesh refinement modules for mesh (de)refinement, or the the executive module to terminate the run.

As in PITCON, a single continuation step consists of a controlled sequence of six sub-modules, namely

(1) The calculation of a tangent vector at the current solution point,

(2) The selection of the new continuation variable,

(3) The selection of the step size,

(4) The construction of the predicted solution and the evaluation of the functional (2.1) at this point via user-supplied functions,

(5) Modified (optionally full) Newton refinement of the predicted solution, including the calculation of the elemental and summary error indicators, and

(6) The error pattern analysis on the error indicators to make the go-no-go decision on mesh (de)refinement.

Note that the failure of the Newton refinement to converge will not, by itself, terminate on FMC episode; rather, it would simply indicate the need for a smaller step. Cost control restrictions would then terminate this process if needed.

The linear solver and user-supplied function evaluation modules present no unusual design criteria other than as noted above.

4. NFEARS DATA STRUCTURE

The data structure for NFEARS is composed, is broad terms, of two distinct structures, the permanent data set (PDS) and the transient data set (TDS). This division directly reflects the corresponding division of an NFEARS execution run into FMC episodes punctuated by mesh re-specification and (de)refinement. The distinction between permanent and transient data is precisely the distinction of data that transcends any particular mesh versus data that is dependent on a given mesh. The importance of the distinction lies in the fact that the PDS is the primary structure for the mesh specification and management modules (which are o .imized

for speed).

The PDS contains data that is intended to survive not only a given mesh selection, but even the termination of the NFEARS execution. It is created immediately upon successfully editing user input, and contains all data regarding the geometry of the region, as well as user-supplied run-control and cost-control parameters. In the course of the run, it is augmented with the current mesh, solution point, and error indicators. These are stored in a flexible tree structure (similar to that in [3]), enabling mesh (de)refinement to be made at a reasonable cost. Also stored in the PDS are such historical data as may be needed for summary reports to trace the events of an NFEARS run.

Once the PDS has been created from user input, and some mesh specified with a corresponding solution point, the execution of NFEARS may be interrupted without loss of data, and resumed "in place" at some later date. This "dump/restart" capability may be used for interim analysis of results, post-processing, or simply to alleviate machine-time restrictions. It also provides the ability for the user to follow different curves on the solution surface, starting from a common point.

The TDS, conversely, does not survive any instance of mesh (de)refinement. It is created in its entirety by the mesh management module whenever the mesh is changed, and contains all data from the PDS necessary to do continuation on the current mesh. It is used by the FMC process, mostly on a read-only basis. It consists of some core-resident portions relating to the 0- and 1-subdomains, with the remainder placed in blocks of a sequential disk file. Each block corresponds to one 2-subdomain, and the blocks are processed either in parallel, or serially, independent of order. Input buffering overlap on the TDS is tuned to keep disk overhead to a minimum.

The contents of the TDS for a given mesh includes the following types of mesh-dependent data:

(1) Integration-related data for each element and boundary line segment, such as quadrature coordinates and the determinant of the Jacobian if the isoparametric domain transformation,

(2) Index information needed for the construction of the linear system, especially in the assembly of the Jacobian matrix of the function (2.1),

(3) The interpolation matrix L by which the solution values at irregular nodes are expressed in terms of regular nodes, and

(4) Pivoting strategies for the incremental decomposition and solution, by 2-subdomain, of the linear system.

Some data in the TDS is organized by element, some by node; minor repetition of data occurs in a few cases. In addition to the TDS, tabulations of shape function evaluations and derivatives for the biquadratic Hermitian element are stored in the PDS, again to obviate repetitive calculation within the FMC process.

REFERENCES

[1] P. Zave and W. Rheinboldt, Design of an Adaptive Parallel Finite Element System, ACM Trans. on Math. Softw. 5, 1979, pp. 1-17.

[2] C. Mesztenyi, A. Miller and W. Szymczak, FEARS, Details of Mathematical Formulation, Univ. of Maryland, Inst. f. Phys. Science and Technology, Tech. Note BN-994, December 1982.

[3] W. Rheinboldt and C. Meszentyi, On a Data Structure for Adaptive Finite Element Mesh Refinement, ACM Trans. on Math. Softw. 6, 1980, pp. 166-187.

[4] W. Rheinboldt, Error Estimates for Non-Linear Finite Element Computations, Univ. of Pittsburgh, Inst. for Comp. Math. and Appl., Tech. Report 84-69, 1984.

[5] W. Rheinboldt and J. V. Burkardt, A Locally Parameterized Continuation Process, ACM Trans. on Math. Softw. 9, 1983, pp. 215-235.

[6] W. Rheinboldt and J. V. Burkardt, Algorithm 596: A Program for a Locally Parameterized Continuation Process, ACM Trans. on Math. Softw. 9, 1983, pp. 236-241.

END

FILMED

2-85

DTIC