NASTRAN AS A RESOURCE IN CODE DEVELOPMENT*

E. L. Stanton and L. M. Crain
Prototype Development Associates, Inc.
Santa Ana, California

T. F. Neu
Naval Air Development Center
Warminster, Pennsylvania

SUMMARY

The heavy use of NASTRAN here and abroad has demonstrated its value as a resource for structural analysis. This paper presents another view of NASTRAN as a quite different resource. A case history is presented in which the NASTRAN system provided both guidelines and working software for use in the development of a major new discrete element program, PATCHES-III. At the 1973 Users' Colloquium, there were two papers presented on the addition of solid isoparametric elements to NASTRAN. At about the same time, development of PATCHES-III for general solids of composite material was begun. The grid point modeling system available in NASTRAN at that time was judged inadequate to efficiently support the modeling of general solids. However, there were many features of the program that would also be required in the new code. To avoid duplication and take advantage of the wide spread user familiarity with NASTRAN, the PATCHES-III system uses NASTRAN Bulk Data syntax, NASTRAN matrix utilities and the NASTRAN Linkage Editor. There were obvious problems that had to be overcome; GINO had to be transplanted to a new environment to mention only one. This paper reviews how these problems were solved and presents details on the architecture of the PATCHES-III parametric cubic modeling system. This system includes novel model construction procedures, new checkpoint/restart strategies and other features that may benefit future versions of NASTRAN.

*This development was sponsored by the Navy under contract number N62269-73-C-0736 and was performed at McDonnell Douglas Astronautics Company.

INTRODUCTION

At a recent NASTRAN Users' Colloquium, software eugenics was a topic of considerable interest, particularly as it related to the contiuned improvement of NASTRAN. Taking the symmetric view, NASTRAN can serve as a resource in the development of other finite element programs. A case history of one such development, PATCHES-III, is presented with emphasis on the mechanics of how this was accomplished. Among the reasons for taking this approach are the need for: (1) a standard finite element syntax familiar to all; (2) reliable, efficient, out-of-core matrix utilities; and (3) a standard file format for matrices. There are a great many finite element programs today (Ref. 1) and they seemingly all use a different syntax. Ideally the structural analyst would like to model his structure independently of the program selected for analysis just as one codes in FORTRAN independently (more or less) of the machine selected for processing. Although this goal remains unrealized, NASTRAN's Bulk Data syntax has proved durable in a dynamic growth period and is probably the most widely used data modeling system in structural mechanics today. Also the NASTRAN matrix utilities are evolving into a system that can serve as a standard both for operations and inter-program communication. For these reasons, as well as for economy, NASTRAN's Bulk Data syntax and matrix utilities were used in the development of PATCHES-III.

The first major integration step involved solving several interface problems between the Linkage Editor and the PATCHES-III program library. It was then necessary to create a NASTRAN environment in which the general input output system (GINO) could operate in support of the matrix utilities. This system functions in parallel with a random access utility system (RASTUS) in PATCHES-III. Installation of a NASTRAN module, such as XSCE1 or RBMG2, then consists primarily of matrix file management. Matrix files including USET , the basic partitioning vector in NASTRAN, are constructed by various routines in the host program, PATCHES-III. The detailed link structure of a modifield version of RBMG2 will be presented as an illustration of a major NASTRAN matrix module adapted for use in another finite element program.

There are important differences between PATCHES-III, a parametric cubic modeling system, and NASTRAN. The grid point modeling available in NASTRAN requires voluminous input for two-dimensional structures and truly staggering input for solid three-dimensional structures. To minimize this problem, PATCHES-III constructs finite line, surface and volume models using operations which can reference data to be created by other operations (construction-in-context); then a queing algorithm orders the operations for serial processing. In the case of one interlaminar stress analysis, input data requirements were reduced from 2750 NASTRAN Bulk Data cards (Ref. 2) to only 46 PATCHES-III Bulk Data cards (Ref. 3). There are also differences in the approach to spill during decomposition and the checkpoint/restart strategy that will be discussed. The final difference to receive attention is the way in which the two programs interface with post-processor programs. This issue is particularly important for graphic output and data editing.

NASTRAN USAGE IN CODE DEVELOPMENT

Early in the design phase of PATCHES-III it became obvious that we could not afford the luxury of reinventing the wheel. Numerous programs had solved in numerous ways many of the basic matrix processing functions that would be needed in the timely construction of this major new software system. However, the NASTRAN modules under consideration were shown to rely heavily upon the NASTRAN environment, in particular GINO and the open core concept as well as an intrinsic dependence upon the linkage editor. In addition, NASTRAN on the CDC system utilized a non-standard RUN compiler and timing considerations had already indicated the necessity for certain of the inner loop routines to be compiled with full optimization under the FTN compiler. Thus in order to use the NASTRAN utilities, it was necessary to simulate an environment within PATCHES-III in which an FTN NASTRAN system could be adapted.

The first major step in the incorporation of NASTRAN modules into PATCHES-III was made through the acquisition of an FTN NASTRAN Linkage Editor (Ref. 4) as well as an FTN version of NASTRAN level 15.1 from the Naval Ship Research and Development Center (NSRDC). Certain difficulties arose in the use of the linkage editor. In the first attempt, the LINKLIB file, from which externals are satisfied, contained relocatables from NSRDC as well as from our own site. Such a hybrid system must be avoided as system and site idiosyncracies can conflict. As long as the LINKLIB file is self-consistent, the resultant system will operate identically at all facilities with no changes required. A second problem with the use of the linkage editor arose from a small set of PATCHES-III routines written in CDC machine language, COMPASS. The loader places a word of traceback information in the word prior to the primary entry point of the routine. However, certain of the COMPASS routines in the PATCHES-III library had their primary entry point at other than the first word of the routine which would result in execution of the traceback word in some circumstances yielding obviously unpredictable results. When this and other operational problems were circumvented (Ref. 5), a fundamental drawback of the linkage editor became evident; it was designed for a large and relatively stable system - NASTRAN, and not for a program in its development cycle, subject to almost daily modifications. For example, if a referenced routine is not explicitly positioned, it will "float" to the highest segment in the link rather than being positioned in the segment from which it was referenced. The impact of this problem was diminished by breaking the system into a structure consisting of a resident link 0 and 18 subordinate parallel links, Figure 1, thereby also minimizing the restriction that a routine can exist at only one location within a link. Another difficulty is that any references to such unpositioned routines are satisfied only from the LINKLIB file therefore requiring concatenation of all program libraries onto LINKLIB. The linkage editor also contains little diagnostic capability.

A job procedure library, essentially a set of control card subroutines, was created to minimize the potential for error in the development cycle. One of these procedures, for example, effects a modification to the source code and to the program structure and also handles all file management associated with the creation of the new system. This procedure, diagramed in Figure 2, requires three input card files: the UPDATE modifications to the PATCHES-III source, the UPDATE modifications to the linkage editor directives and an input Case Control/Bulk Data deck to test the system. Typically, the linkage structure of the system remains unchanged and only the code in a few links is modified. For example, assume that subroutines referenced in links 1 and 17 are to be modified. Having placed the linkage editor directives for each link in a separate *DECK, it is possible in this instance to generate the input to the linkage editor with only one card:

*COMPILE LINKM, LINK1, LINK17, END

where deck LINKM contains thedirectives to modify an existing executable file and deck END contains the ENDLINKS directive. In our example then the following steps would be performed:

(1) UPDATE the PATCHES-III source and send the modified routines to the FTN compiler
(2) Augment/modify the previous version of the PATCHES-III relocatables per the updates
(3) Create the new LINKLIB file
(4) UPDATE the linkage directives
(5) Execute the linkage editor using directives created in step 4, and create the new executable file
(6) Execute the PATCHES-III system as created

Armed with a functioning linkage editor, we could then begin the process of adding selected portions of NASTRAN to the new system. The devices used to solve these problems may not always have been the most efficient or general, but they satisfied our needs given the time constraints. First it was necessary to add the basic NASTRAN utilities and resident common blocks to link 0 through the use of linkage editor directives. Figure 3, from Ref. 4, diagrams the current configuration of the merged PATCHES-III/NASTRAN link 0. Block data routines TIME and GINO66 should also be loaded.

To follow the flow of a typical execution will serve to illustrate the necessary steps in a combined PATCHES-III/NASTRAN run. Table 1 gives a brief description of each of the links diagramed in Figure 1. Figures 4, 5, and 6 describe the basic flow of a PATCHES-III execution. In links BEGIN and INITNS, the program initializes the NASTRAN subsystem. Toward this end, a call is made to BTSTRP which determines the machine type and sets machine dependent constants. Then the time-to-go clock is initialized via a CALL KLOCK (TBEGIN) where TBEGIN is located in common block/SYSTEM/. A call to GNFIAT is made to determine the number of logical files available and to place an entry for each in the file entry table XFIAT. A call to GNFIST then sets up proper linkage between data blocks and the files upon which they reside. This call to GNFIST should be followed by an OPEN and CLOSE operation on file NPTP setting a necessary first pass flag. Room must be allocated for the NASTRAN system working storage beyond the open core region and this is accomplished by decrementing word 76_8 of core by 4000_8 in a small COMPASS routine CORE76, called from the initialization link. It is this word which is queried on each call to CORESZ, the open core utility, and this action then prevents any utility from referencing the NASTRAN system region. The PATCHES-III executive modifies the field length of the program as necessary prior to the execution of each link. Therefore, the CORE76 routine must be utilized to modify word 76_8 on each change of field length. At the same time, the NASTRAN system region must be translated to its new location below open core. As NASTRAN modules utilizing the time-to-go feature are employed, a TIME Case Control card should be processed setting the first word of common block STIME to TBEGIN + 60 times the time estimate in minutes.

At this point, we are able to reference any of the NASTRAN utilities including GINO. The above steps have created entries in the FIAT and FIST tables such that references may be made to any GINO file numbered 201 through 216 and 301. References to other file numbers should not be made. For example, a reference to file number 301 + i results in an actual reference to file 200 + i. As this was an adequate number of GINO files for our application, we did not use pooling and unpooling of files. Whenever a file is no longer necessary, it is made available for use by a new module simply through a close with rewind request. Since the communication to most NASTRAN utilities references the GINO file number through either calling parameters or common blocks, in most cases only the driving routine needs consider the bookkeeping problem and the NASTRAN routines can usually remain unmodified.

Many of the NASTRAN modules require some additonal information in the form of the USET table and trailer. The major constituents of this file are mask words - one per degree of freedom of the problem indicating the types of constraints associated with each degree of freedom. The USET trailer contains in location three the number of degrees of freedom and in location five the logical OR of all the mask words in the table. The entire file can be constructed in approximately a dozen lines of FORTRAN, primarily through calls to GINO utilities.

The above operations have succeeded in creating an environment suitable for many NASTRAN utilities. All that remains is to establish the proper interface to drive them. A number of the modules in the PATCHES-III system such as the input processor, model generation and data verification links consist almost exclusively of PATCHES-III routines, Figure 7 (Ref. 6). Certain other links consist primarily of NASTRAN routines, for example XSCE1 the single point constraint eliminator, and SSG the static solution generator. The other links are a marriage of both systems and as an example, consider the matrix decomposition link, RBMG2D, shown schematically in Figure 8. The driver routine determines if a checkpoint or restart is requested and calls the PATCHES-III controller CKPTL or RESTL as appropriate. These routines then write or read the necessary files on the restart tape through the use of the NASTRAN utilities OUTPT1 or INPTT1. Matrix decomposition is performed, when required, under the supervision of RBMG2D through a call to the NASTRAN driver FACTOR which in turn calls RSPSPC, a slightly modified version of NASTRAN's RSPSDC. There are certain steps which, in general, must be performed to create a NASTRAN environment for a module such as RBMG2:

(1) The driver routine must allocate storage for the appropriate COMMON blocks used to communicate file numbers and header records within the module.

(2) The input, output, and scratch GINO file numbers should be placed in the appropriate common blocks or subroutine parameter lists consistent with their usage in other links.

(3) The linkage editor must be used to position any necessary open core arrays to a REGION following the segment from which it is referenced.

(4) Any files which are opened should be closed and rewound prior to exit from the link.

Experience has indicated that a typical major module such as the decomposition module requires approximately three weeks of iteration to install and debug. Adaptation of a smaller utility, such as INPTT1 and OUTPT1 requires substantially less time.

DIFFERENCES IN PHILOSOPHY

There are a number of significant differences in philosophy between PATCHES-III and NASTRAN that may be useful to consider for future versions of NASTRAN and other systems. Prior to illustrating these differences, an overview of the capabilities of PATCHES-III is appropriate. More detailed discussions can be found in References 3, 7, and 8. PATCHES-III determines the linear elastic response of a general heterogeneous anisotropic solid under thermal loads, mechanical loads and imposed displacements. The analysis model is based on a sixty-four point isoparametric solid discrete element with variable material properties. This element efficiently models the strain discontinuities at heterogeneous material boundaries as well as the continuous strains at homogeneous boundaries. The program constructs models for both geometry and physical data using parametric cubic interpolation over lines, surfaces and volumes. This system automates the construction of discrete element models and can reduce input data requirements by more than an order of magnitude. In this system, construction operations are available that reference data to be created by other operations (construction-in-context). Conceptually and in practice, this system is analogous to an interactive model generation system. Instead of sequentially processing requests from a terminal, all requests are made at once (batch mode via Bulk Data) and a queing algorithm orders the requests for serial processing. This approach is also used to create models for physical data and imposed displacement constraints. Thus the same interpolation functions and most of the construction tools used to model the geometry are also available to model any physical data input to PATCHES-III. Both zero and non-zero constraint options are available that constrain the entire face of an element with a single bulk data card using either the reference frame or the local surface frame.

Consider the construction of a hyperpatch for one segment of a thick-walled circular cylinder. Two quite different but nearly equal constructions illustrate the procedure for a simple shape. In the first, Figure 9, grid points 1 and 3 are input and a straight line connecting them is created with a LINEPC Bulk Data card. This line is rotated about the $\underset{\sim}{e}_3$ axis through 90° to form one quadrant of a cylindrical surface using a PATCHR card. In this process grid points 7 and 5 are automatically created. The surface 1-3-7-5 is expanded a unit amount in the direction of its normal to create a hyperpatch using the HPN card. The grid points 2, 4, 6 and 8 are automatically created in this process along with the parameters defining the element or hyperpatch. The final figure is a 90° segment of a circular cylinder of unit thickness. The construction of this 192 parameter hyperpatch required five (5) cards of very simple format. Now consider the same construction problem but this time input grid points 1, 2, 3 and 4, Figure 10. A quadrilateral surface is created with a PATCHQ card and this surface is rotated about the $\underset{\sim}{e}_3$ axis through 90° to form the hyperpatch using the HPR card. The construction this time required six (6) bulk data cards. These examples illustrate the method, not the complexity of the geometric shapes that can be created (c.f. Ref. 7, page 2-22). In fact the geometry for all of the general elements in NASTRAN, including isoparametric elements, can be constructed using the modeling system in PATCHES-III.

Consider next a PATCHES-III analysis of an interlaminar normal stress problem. One of the few three-dimensional composite laminate problems for which corroborative solutions exist is a four ply graphite-epoxy plate under uniaxial load. A finite difference solution of the elasticity equations (Ref. 9), a stress-function discrete element solution of the elasticity equations (Ref. 10), an analytic solution of certain higher order laminated plate theory equations (Ref. 11) and the PATCHES-III displacement-function discrete element solution of the elasticity equations all agree well for the interlaminar normal stress. Stresses are computed for the $0^{\circ}/90^{\circ}/90^{\circ}/0^{\circ}$ laminate shown in Figure 11 under a uniform imposed displacement in the axial direction. Taking full advantage of symmetry, a simple four element PATCHES-III model was created from the forty-six bulk data cards shown in Figure 12. This is a substantial reduction from the 2750 Bulk Data cards necessary for a comparable solution using NASTRAN (Ref. 2) and the savings in the user's model definition and debugging cycle is of comparable magnitude. Figure 13, from the PATCHES-III plotting post-processor, shows a six element model of the interlaminar stress problem; an additional pair of elements has been used in the vicinity of the free edge to improve transverse shear stress accuracy. Lines in addition to the element boundaries are also drawn to show surface effects. Figure 14 shows the deformed geometry view of the model with hidden surfaces removed. The magnitude of the displacements has been magnified substantially to make the deformations visible. Figure 15 is a data surface plot; the lower surface represents the undeformed geometry along the laminate mid-plane; the upper surface represents the normal strains such that each point in the upper surface lies in the normal direction to the undeformed surface and at a distance proportional to the magnitude of the normal strain at that point plus an initial offset.

Although similar in many ways, there are some basic philosophical differences in the operation of the programs. A fundamental policy in PATCHES-III which is sometimes absent from NASTRAN concerns the diagnostic package. In PATCHES-III, an attempt is made to detect every error in model or data generation including Case Control-Bulk Data cross references prior to execution of any of the time consuming modules. Any diagnostic messages are explicit and informative and reference the card which precipitated the error condition. In concert with this philosophy, a "DRY" Case Control directive exists which indicates that this is to be a dry run of the data for the purpose of diagnosing any input errors. Another difference is that the field length used by PATCHES-III varies automatically during a run. This can result in a significant reduction in computing charges for larger analyses in a multiprocessing environment.

Although conditions may change with the release of Level 16, the current release of NASTRAN consumes large amounts of I/O time for a matrix decomposition operation that "spills." In PATCHES-III when the spill condition is detected, the program switches to a combination decomposition-conjugate gradient solution scheme which allows the user to buy as much accuracy as he requires up to and including complete convergence at a substantially reduced cost.

Bulk Data processing and model generation consume relatively little time in PATCHES-III when compared to generation of the element stiffness matrices or decomposition. Accordingly, the checkpoint/restart strategies employed by PATCHES-III differ from those of NASTRAN in that a checkpoint dictionary is not required. Rather, the Bulk Data deck is resubmitted on a restart run and the necessary tables and files are recreated. For design applications, the ELEMENT checkpoint is of use when a few elements are being modified. The LOADS checkpoint is utilized in analysis applications in which only the loads are changed.

A special output called the User Information Messages file is printed subsequent to all output from PATCHES-III. This listing details CP times and field lengths of each of the major modules encountered during execution. It also summarizes additional information concerning the dimension of certain matrices, the usage of random access storage, and other salient features of the execution.

A major departure from NASTRAN involves PATCHES-III's generation of a post-processing data file, PPDATA. This file uses a straightforward format written with unformatted binary FORTRAN write statements, and serves as an easy-to-use data base for post-processor programs. By no means a replacement for the OUTPT2 capability in NASTRAN, the PPDATA file does offer significant advantages: (1) it makes possible a wide range of DMAP independent post-processors; (2) it is far easier for the average structural analyst to use and adopt to his own requirements; (3) use of the post-processing scheme permits support of the blossoming array of hardware devices, in particular plotters, to be offloaded from the primary system; and (4) the post-processing environment is a necessity for experimental or special case analyses such as failure models which would be inappropriate to include in NASTRAN. In PATCHES-III the PPDATA tape contains n+1 files: one file containing the invariant data such as geometry, and n subcase files containing the load conditions and the results of the analysis.

PATCHES-III itself assists in the creation of post-processor programs by supplying a library of utilities which the post-processor may reference within the framework of a procedure library designed to aid in the construction of such systems. The most obvious and highly used post-processing system for PATCHES-III is the plot system. This program accepts free form inputs similar to Case Control cards to direct the plotting of the deformed or undeformed three dimendional elements. Surfaces may be subdivided to any requested extent and labeled contours of data surfaces may be requested on the data or geometry surfaces. Many options similar to those in NASTRAN for view, perspective and hidden line control are available.

CONCLUDING REMARKS

NASTRAN has shown itself to be a valuable resource in code development, both as a reference and a source of software. Once the steps have been identified, it is not difficult to implement any of the NASTRAN utilities in an existing or developing code. Since a major expense in code development is usually associated with the debugging and optimization phases, it is a tremendous asset to have NASTRAN as a source of efficient and effectively error-free software.

The NASTRAN Case Control-Bulk Data syntax was used to make a new finite element program, PATCHES-III, easier to learn for many users. More needs to be done to make structural modeling less dependent on the syntax of individual applications programs.

A number of major differences in the operations of PATCHES-III and NASTRAN have been noted. In particular, the construction-in-context of geometry and data models and the post-processor data file should be given serious consideration for inclusion in future versions of NASTRAN.

REFERENCES

1. W. Pilkey, K. Saczalski, and H. Schaeffer, Structural Mechanics Computer Programs, University Press of Virginia (1974).

2. T. F. Neu, "Finite-Element Analysis of Edge Effects in Angle-Ply Composite Laminates," Naval Air Development Center Report No. NADC-74051-30, March 1974.

3. E. L. Stanton and L. M. Crain, "PATCHES-III User's Manual," McDonnell Douglas Astronautics Company, Report No. MDC G5538, November 1974.

4. Roger J. Martin, "A General Purpose Overlay Loader for CDC-6000 Series Computers; Modification of the NASTRAN Linkage Editor," Report 4062, Naval Ship Research and Development Center.

5. E. L. Stanton and L. M. Crain, "Three-Dimensional Parametric Discrete Element Program for the Analysis of Composite Structures," Progress Report Number 6, Contract Number N62269-73-C-0736, pp. 12-15, June 1974.

6. L. M. Crain, "PATCHES-III Program Structure and Subroutine Descriptions," McDonnell Douglas Automation Company, Report No. MDC G5795, April 1975.

7. E. L. Stanton, "A Three Dimensional Parametric Cubic Discrete Element Program for the Analysis of Composite Structures," McDonnell Douglas Astronautics Company, Report No. MDC G5716, January 1975.

8. E. L. Stanton, L. M. Crain, and T. F. Neu, "A Parametric Cubic Modeling System for General Solids of Composite Material." McDonnell Douglas Astronautics Company, Paper No. WD2606, July 1975.

9. R. Byron Pipes, "Solution of Certain Problems in the Theory of Elasticity for Laminated Anisotropic Systems," Ph. D. Dissertation, University of Texas (1972).

10. E. F. Rybicki, "Approximate Three-Dimensional Solutions for Symmetric Laminates Under Inplane Loading," Journal of Composite Materials, Vol. 5, 1971, pp. 354-360.

11. N. J. Pagano, "On the Calculation of Interlaminar Normal Stress in Composite Laminate," Journal of Composite Materials, Vol. 8, 1974, pp. 65-81.

Table 1

PATCHES-III LINK DESCRIPTIONS

LINK	NAME	FUNCTION
0	MAIN	Executive control and common storage
1	BEGIN	Initialize PATCHES-III system
2	INPCN	Input control. Construct geometry and data models
3	GET	Initialize integration tables
4	MATCN	Material properties module
18	LOADS	Generate element load vectors
6	INITNS	Initialize NASTRAN sub-system
7	BIGMSH	Generate element meshpoint connectivity
8	IMDISP	Imposed displacement module
9	MPC	Multipoint constraints (not active)
5	*EKIJ	Generate element stiffness matrices
10	TTEKTD	Transform element matrices to analysis coordinates
12	SMA	Structural matrix assembler
13	XSCE1	Single point constraint eliminator
14	MCE1	Multipoint constraint eliminator (not active)
15	*RBMG2D	Matrix decomposition
16	SSG	Static solution generator
17	SDR	Stress recovery
11	SUBCØM	Subcase combinations

*Links that use majority of the CP time.

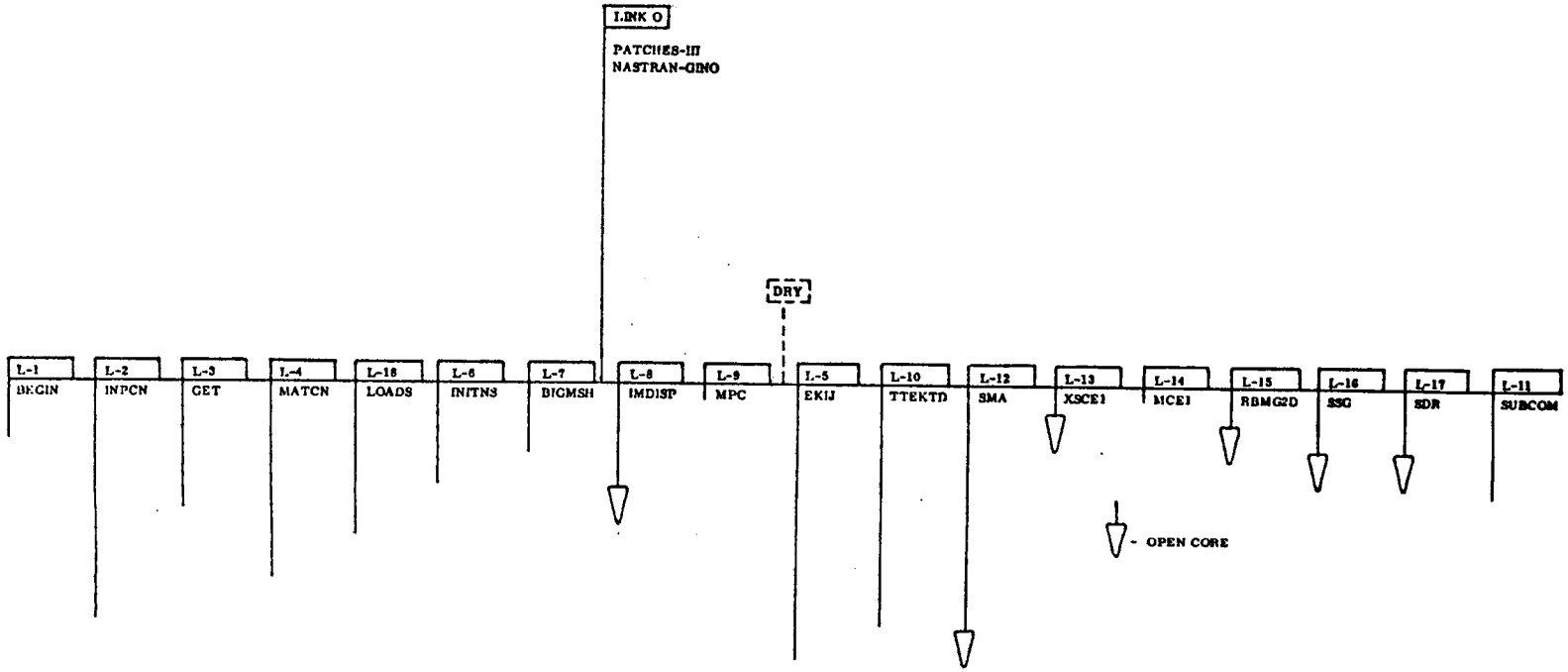


Figure 1: PATCHES-III/NASTRAN Schematic Load Map

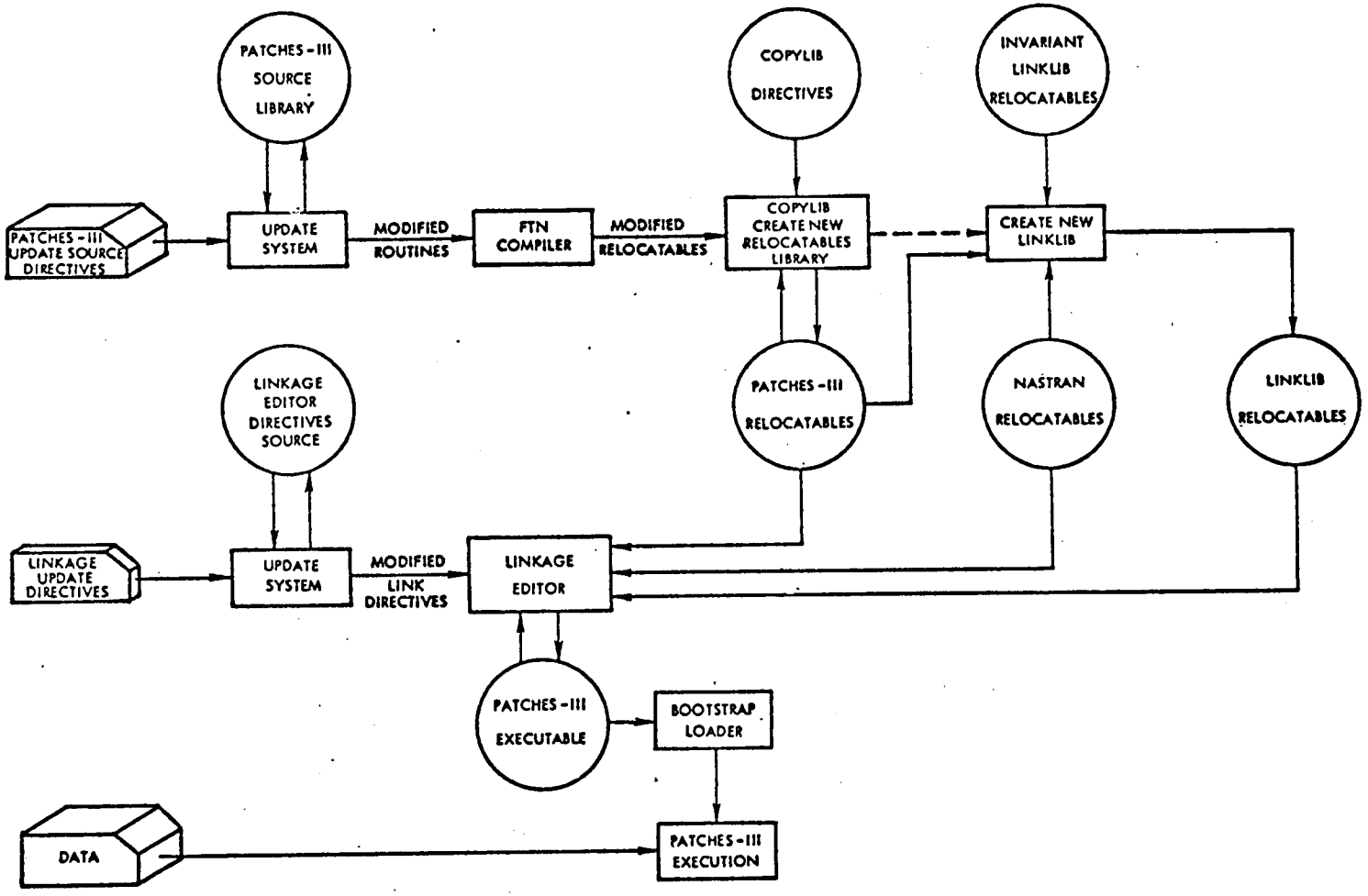


FIGURE 2. PROCEDURE TO MODIFY AND EXECUTE PATCHES - III

LINK 0 Structure

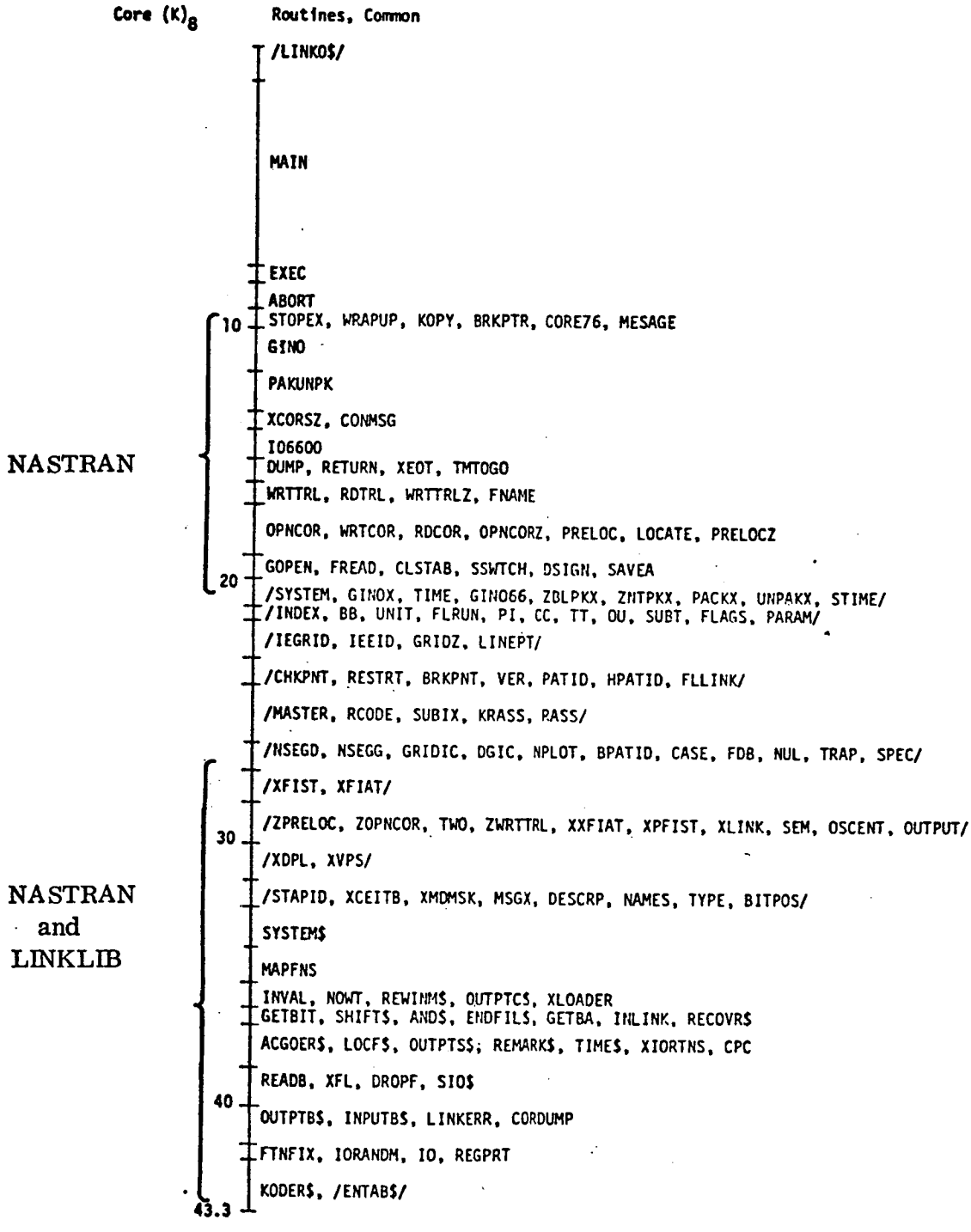


Figure 3: PATCHES-III Link 0 Structure

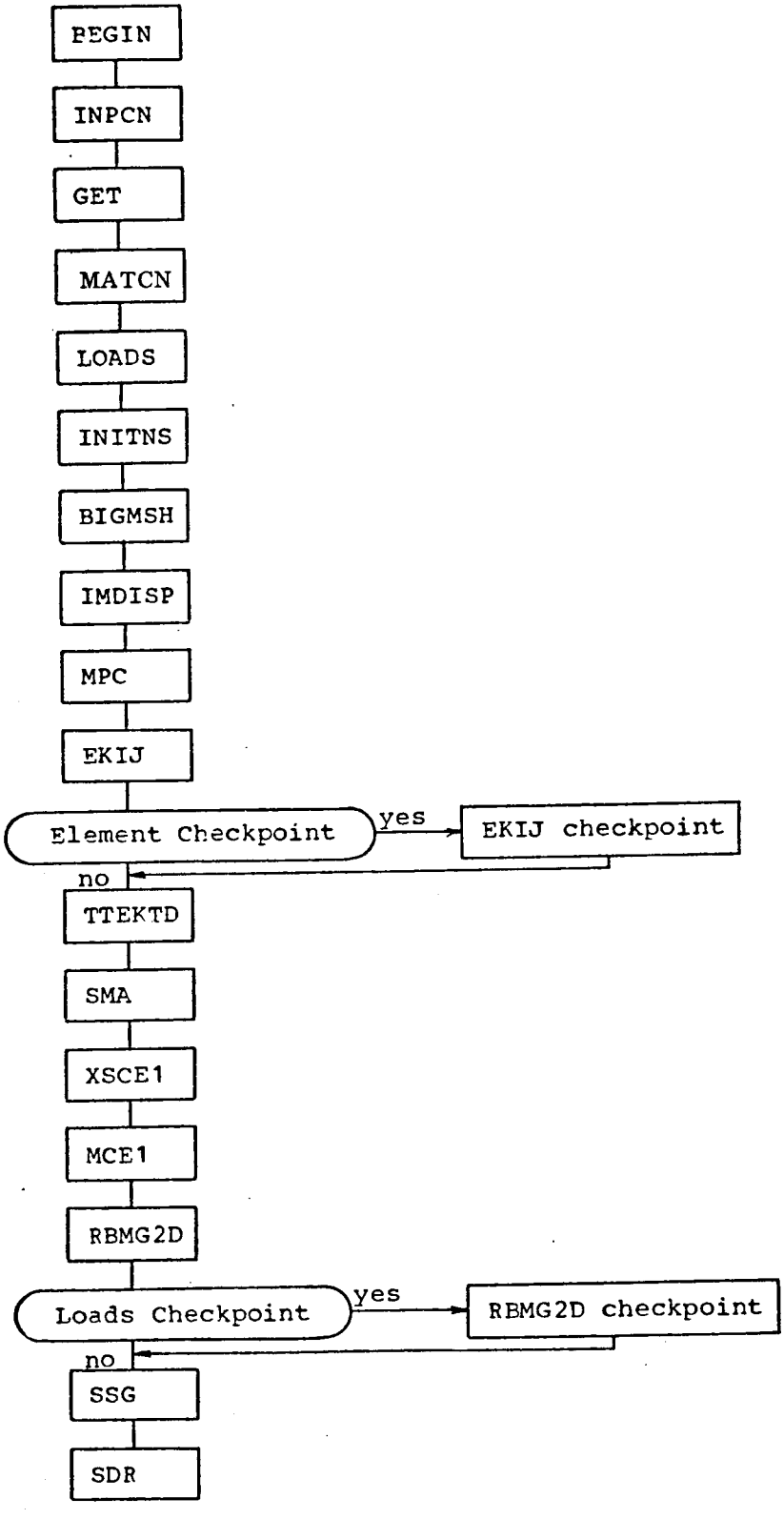


Figure 4: Basic PATCHES-III Flow

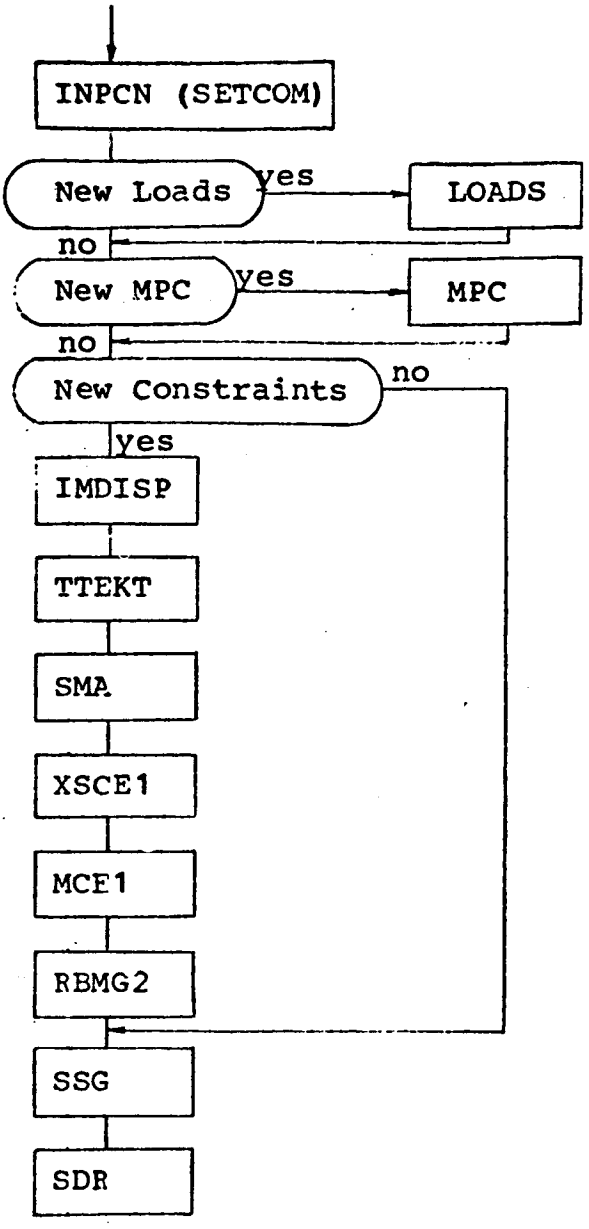


Figure 5: Additional Subcase Flow

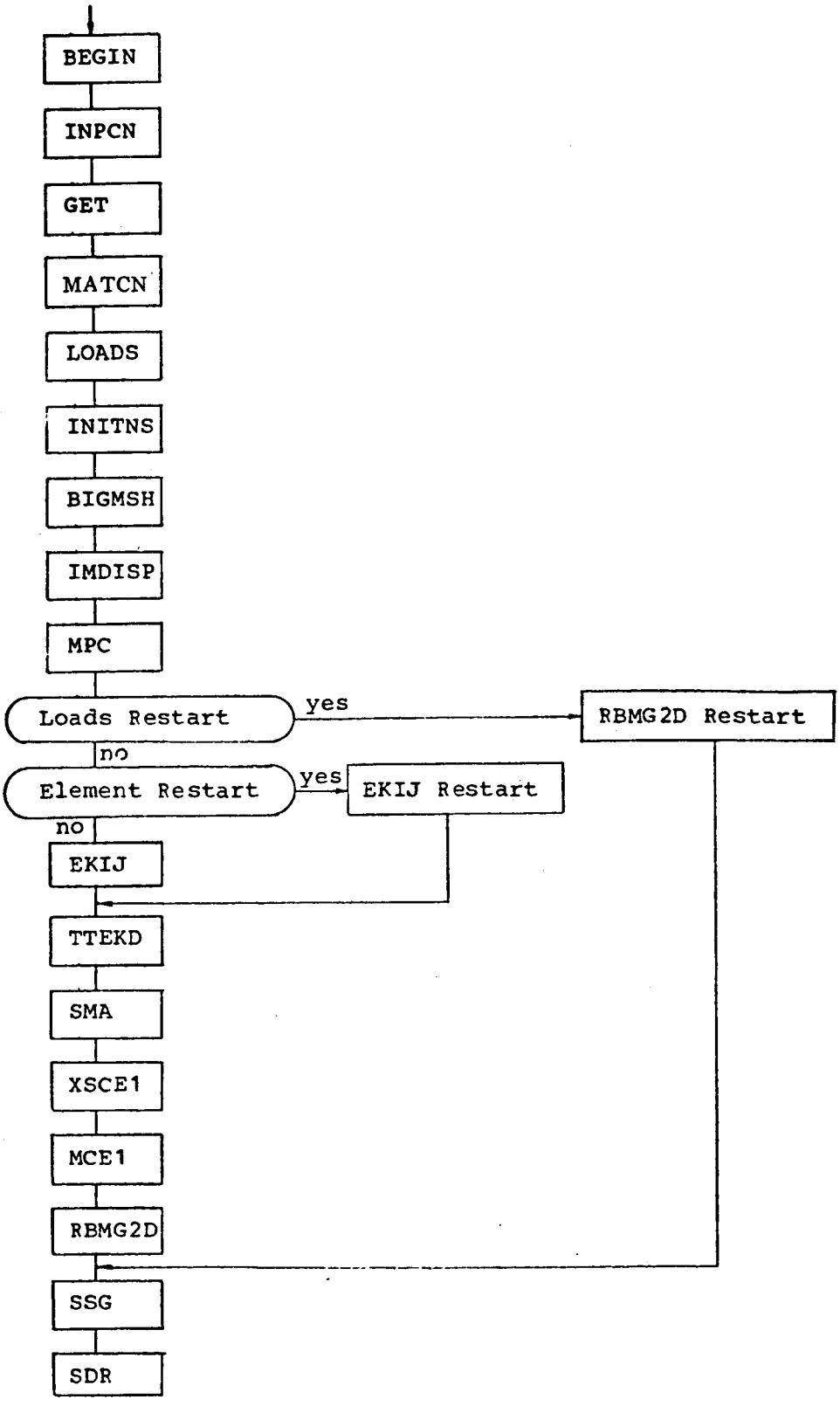


Figure 6: Restart Flow

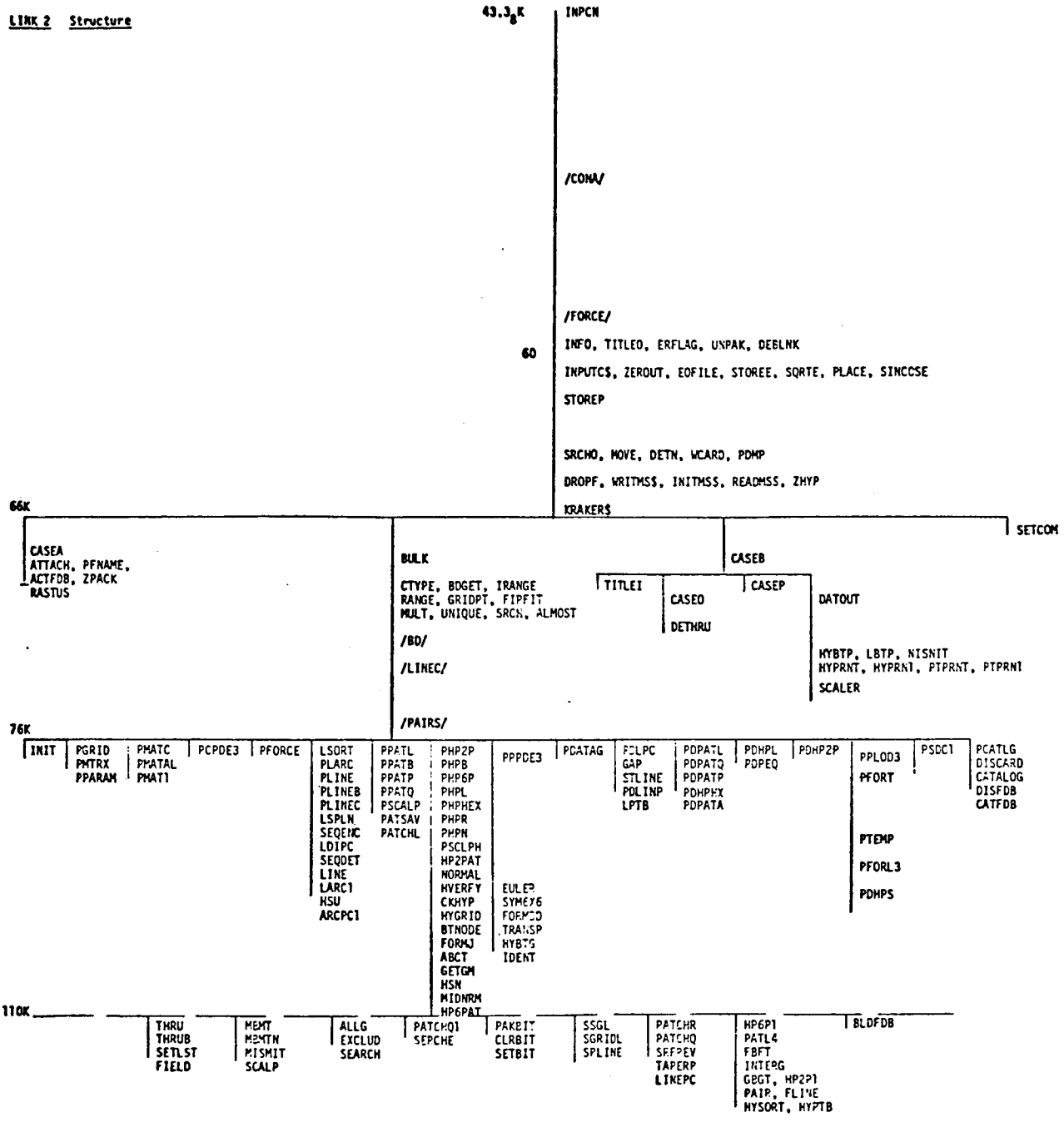


Figure 7: PATCHES-III Link 2 Structure

LINK 15, STRUCTURE

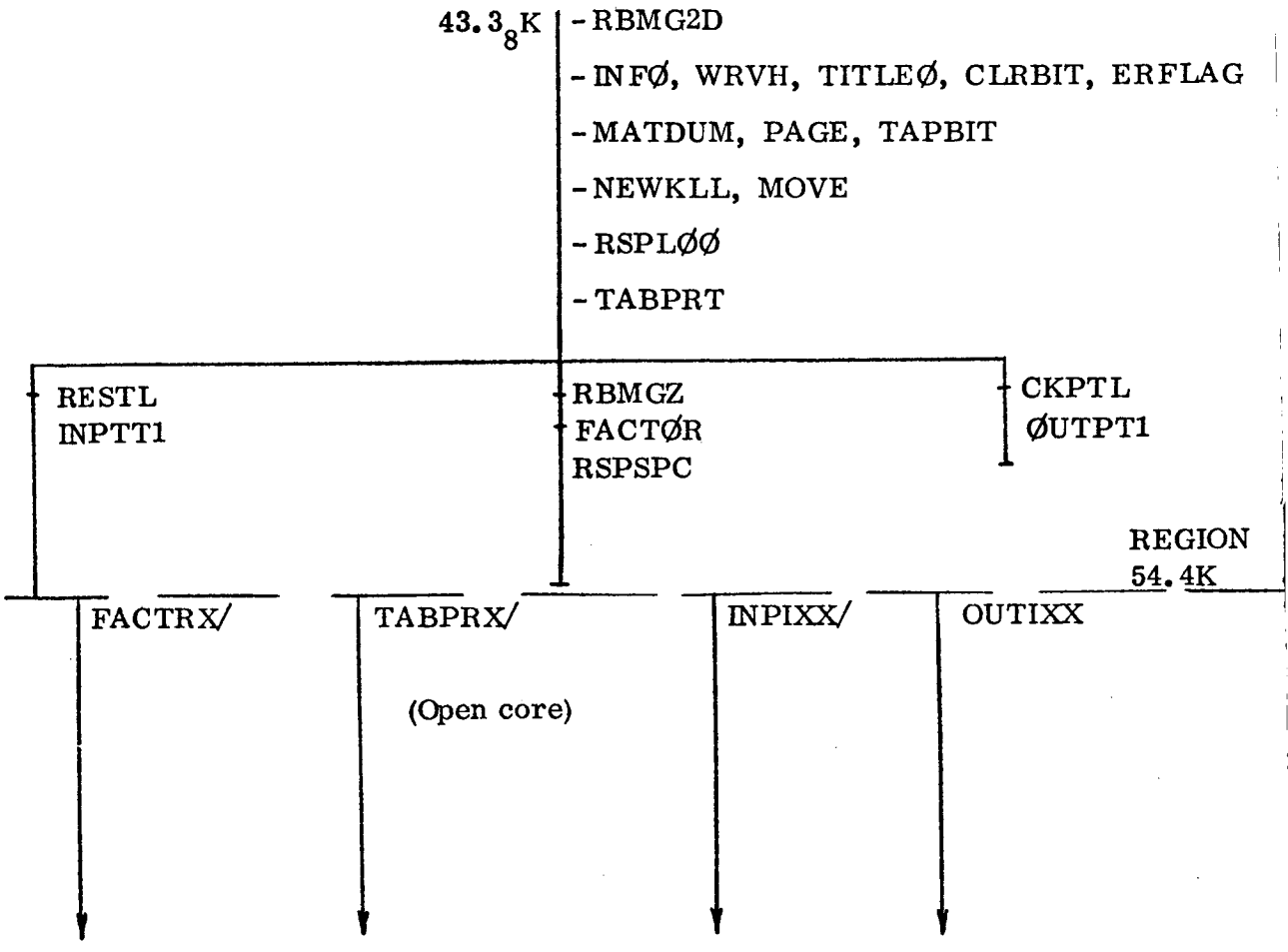
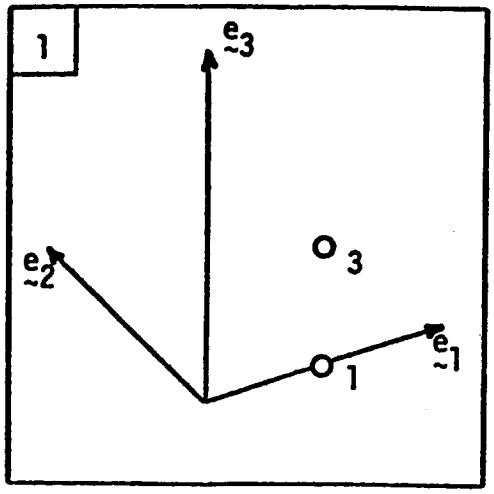
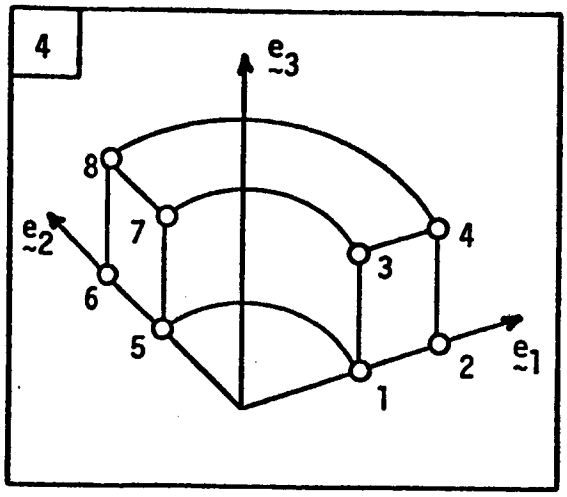
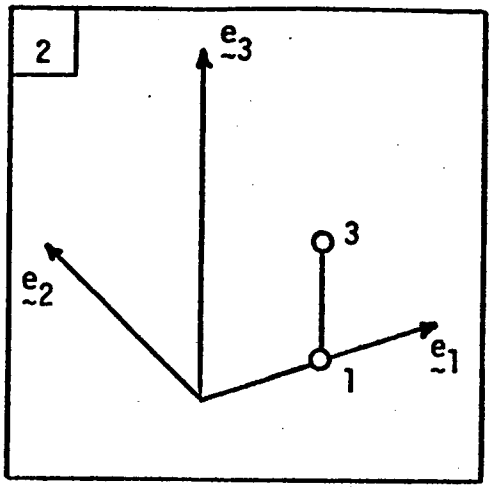
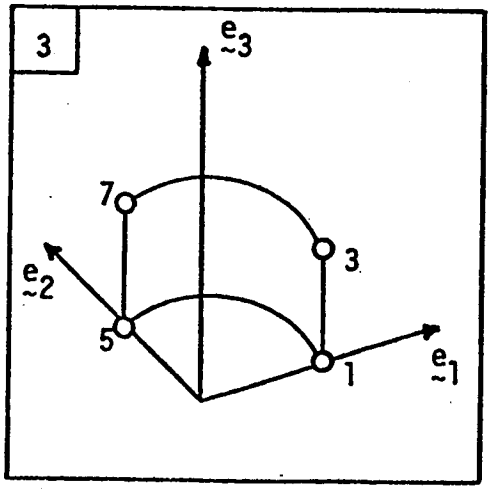


Figure 8: PATCHES-III Link 15 Structure



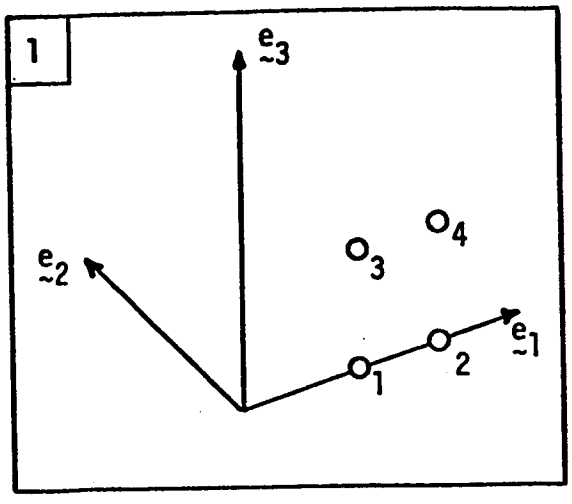


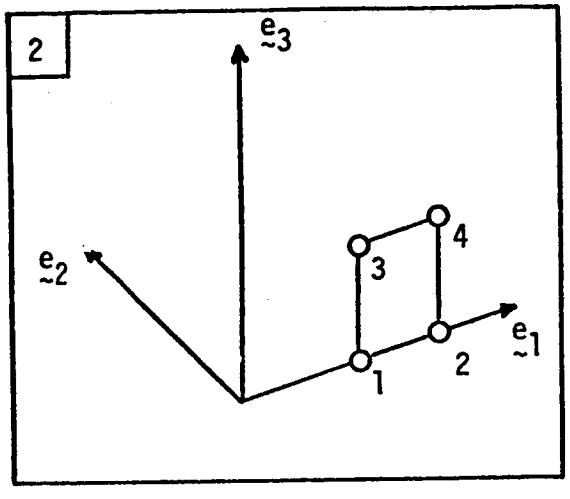


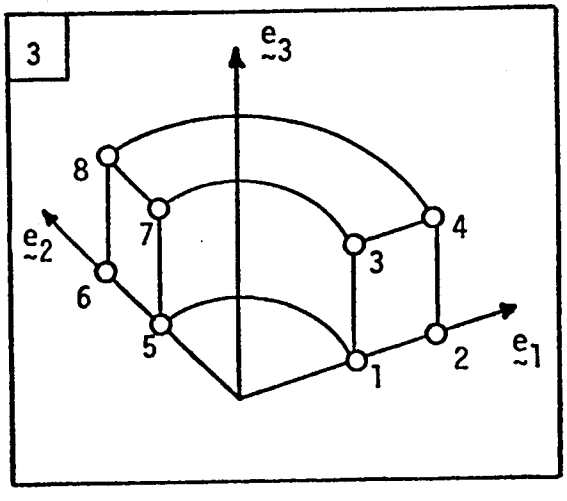


Step	Bulk Data Option	Cards
1	GRID	2
2	LINEPC	1
3	PATCHR	1
4	HPN	1

Figure 9: Hyperpatch Construction Example - Method One







Step	Bulk Data Option	Cards
1	GRID	4
2	PATCHQ	1
3	HPR	1

Figure 10: Hyperpatch Construction Example - Method Two

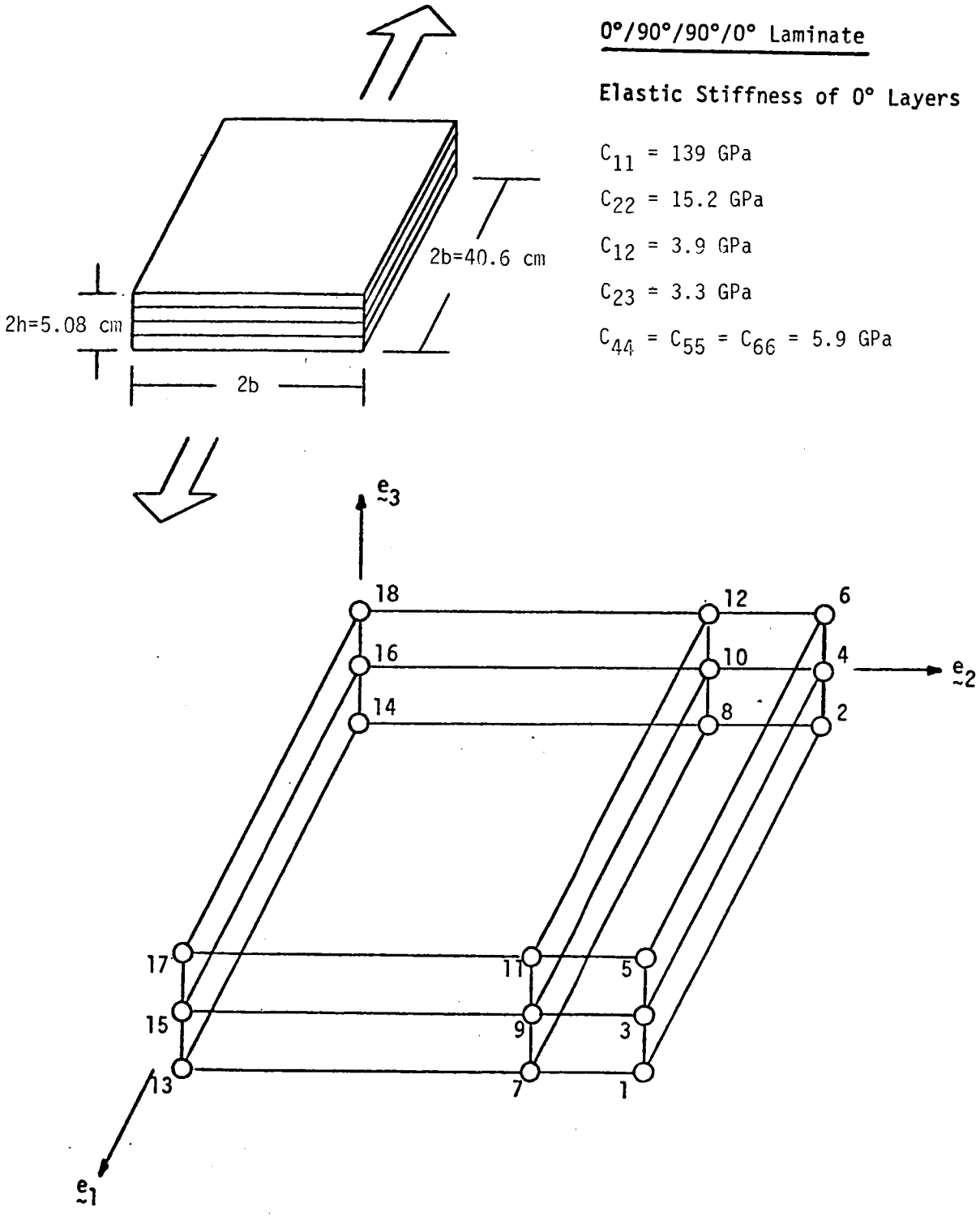


Figure 11: Four Element Model of a Graphite Epoxy Laminate

B U L K   D A T A   C A R D S   PAGE 1

CARD	FIELD-1	FIELD-2	FIELD-3	FIELD-4	FIELD-5	FIELD-6	FIELD-7	FIELD-8	FIELD-9	FIELD-10
1-	CPDE3	10	14	8	7	13				+CP1
2-	+CP1		16	10	9	15				
3-	CPDE3	20	8	2	1	7				+CP2
4-	+CP2		10	4	3	9				
5-	CPDE3	30	16	10	9	15				+CP3
6-	+CP3		18	12	11	17				
7-	CPDE3	40	10	4	3	9				+CP4
8-	+CP4		12	6	5	11				
9-	DATAG	1		1	0,01	3	0,01	5	0,01	+D1
10-	+D1	7	0,01	9	0,01	11	0,01	13	0,01	+D2
11-	+D2	15	0,01	17	0,01					
12-	DPATQ	10	1	13	15	9	7			
13-	DPATQ	20	1	7	9	3	1			
14-	DPATQ	30	1	15	17	11	9			
15-	DPATQ	40	1	9	11	5	3			
16-	GRID	15		8,	0,	0,				
17-	GRID	9		8,	6,	0,				
18-	GRID	3		8,	8,	0,				
19-	GRID	16		0,	0,	0,				
20-	GRID	10		0,	6,	0,				
21-	GRID	4		0,	8,	0,				
22-	HPN	10	1	r,5						
23-	HPN	20	2	r,5						
24-	HPN	30	1	,5						
25-	HPN	40	2	,5						
26-	PPDE3	10	1		90,					
27-	PPDE3	20	1		90,					
28-	PPDE3	30	1							
29-	PPDE3	40	1							
30-	MATC	1	1	1	1					
31-	MTRX-1	20.2+6	0,56+6	0,56+6	2,21+6	0,48+6	2,21+6	0,85+6	0,0	+M1
32-	+M1	0,0	0,85+6	0,0	0,85+6					
33-	PATCHQ	1	16	10	9	15				
34-	PATCHQ	2	10	4	3	9				
35-	SDC10	10	10	1	14	16	10	8		
36-	SDC10	10	10	2	14	13	15	16		
37-	SDC10	10	10	3	14	8	7	13		
38-	SDC10	10	20	1	8	10	4	2		
39-	SDC10	10	20	3	8	2	1	7		
40-	SDC10	10	30	1	16	18	12	10		
41-	SDC10	10	30	2	16	15	17	18		
42-	SDC10	10	40	1	10	12	6	4		
43-	SDC1	10	10	10						
44-	SDC1	10	20	20						
45-	SDC1	10	30	30						
46-	SDC1	10	40	40						

Figure 12: Bulk Data for Laminate

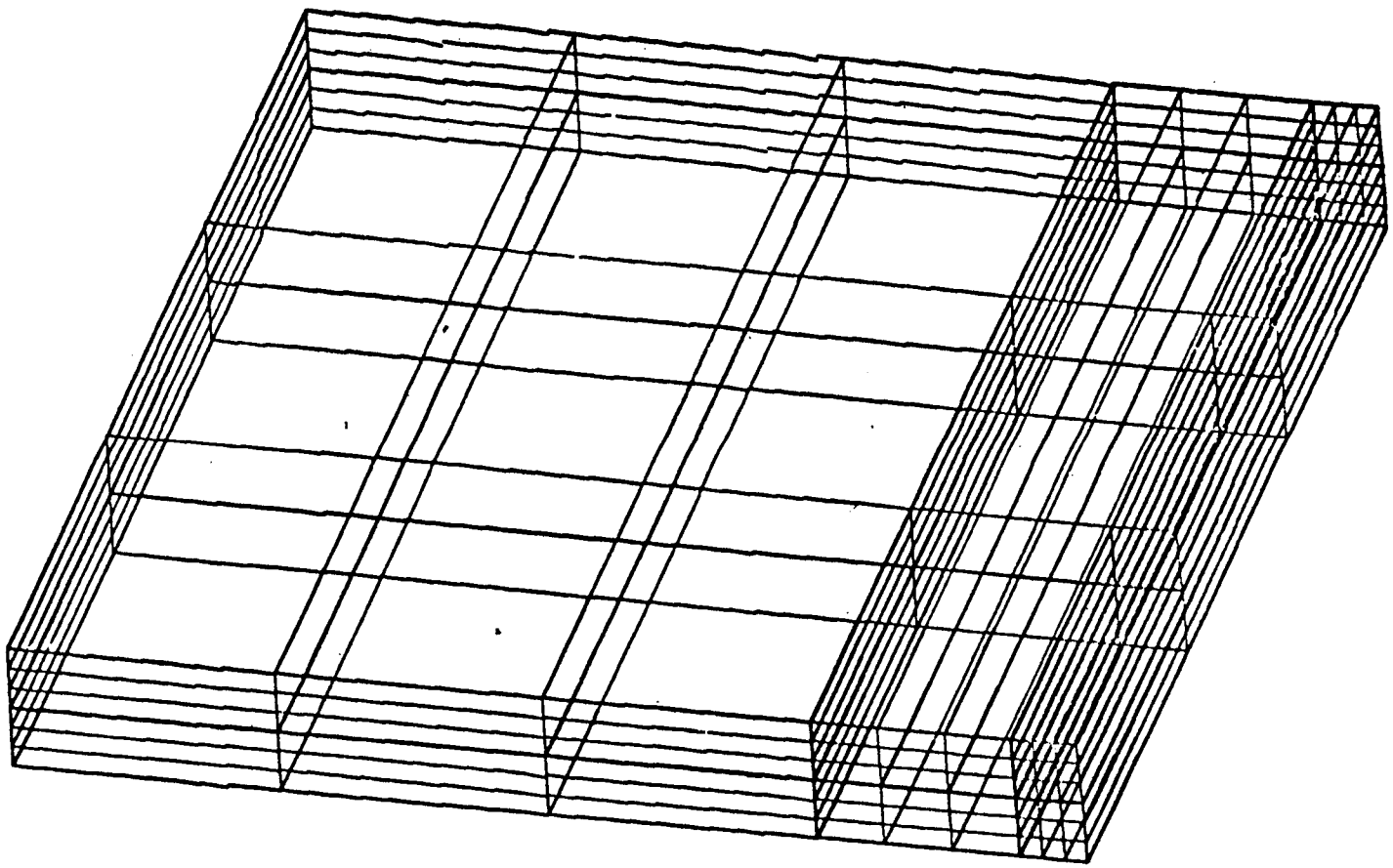


Figure 13: Six Element Model, Undeformed

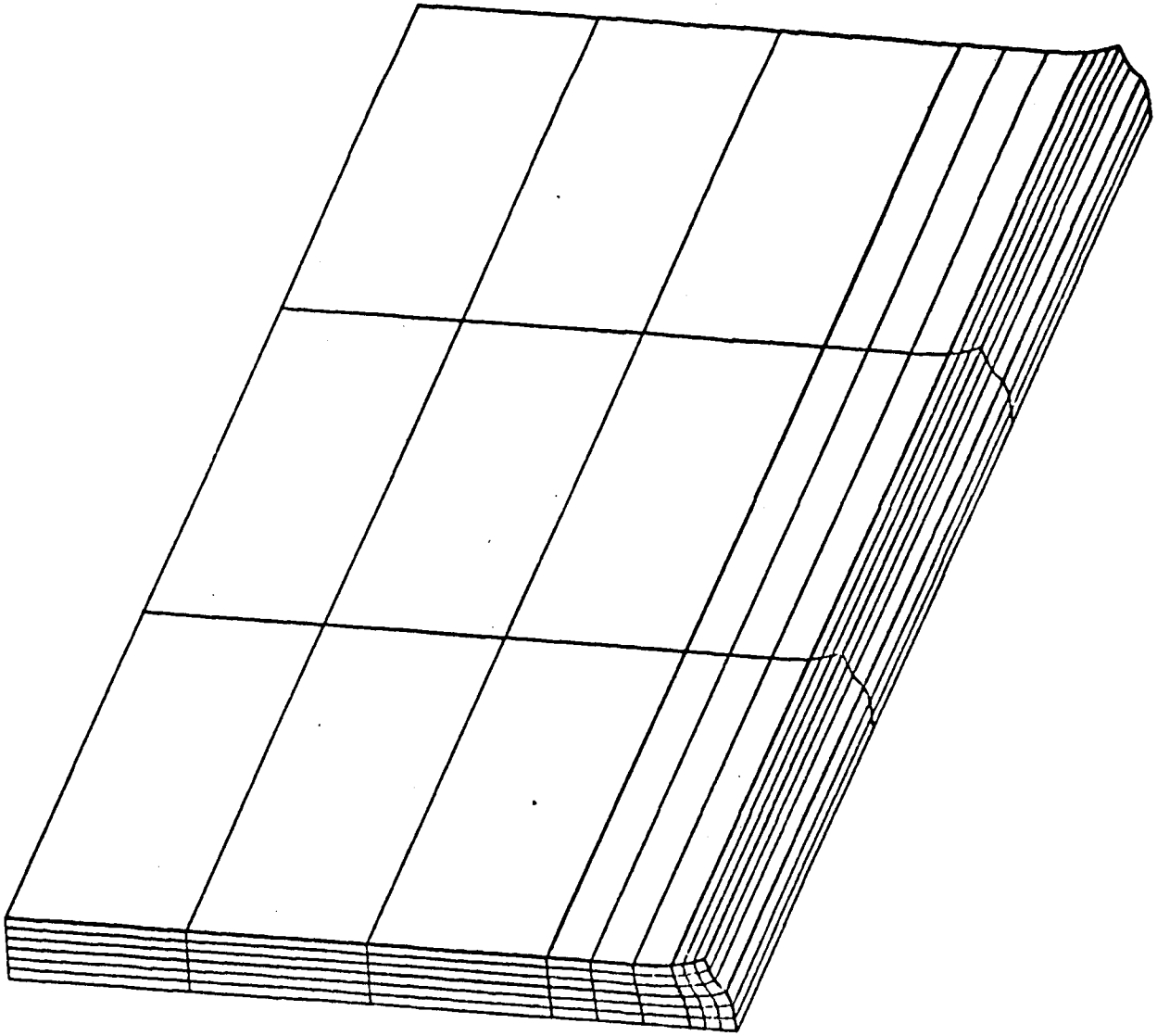


Figure 14: Six Element Model, Deformed

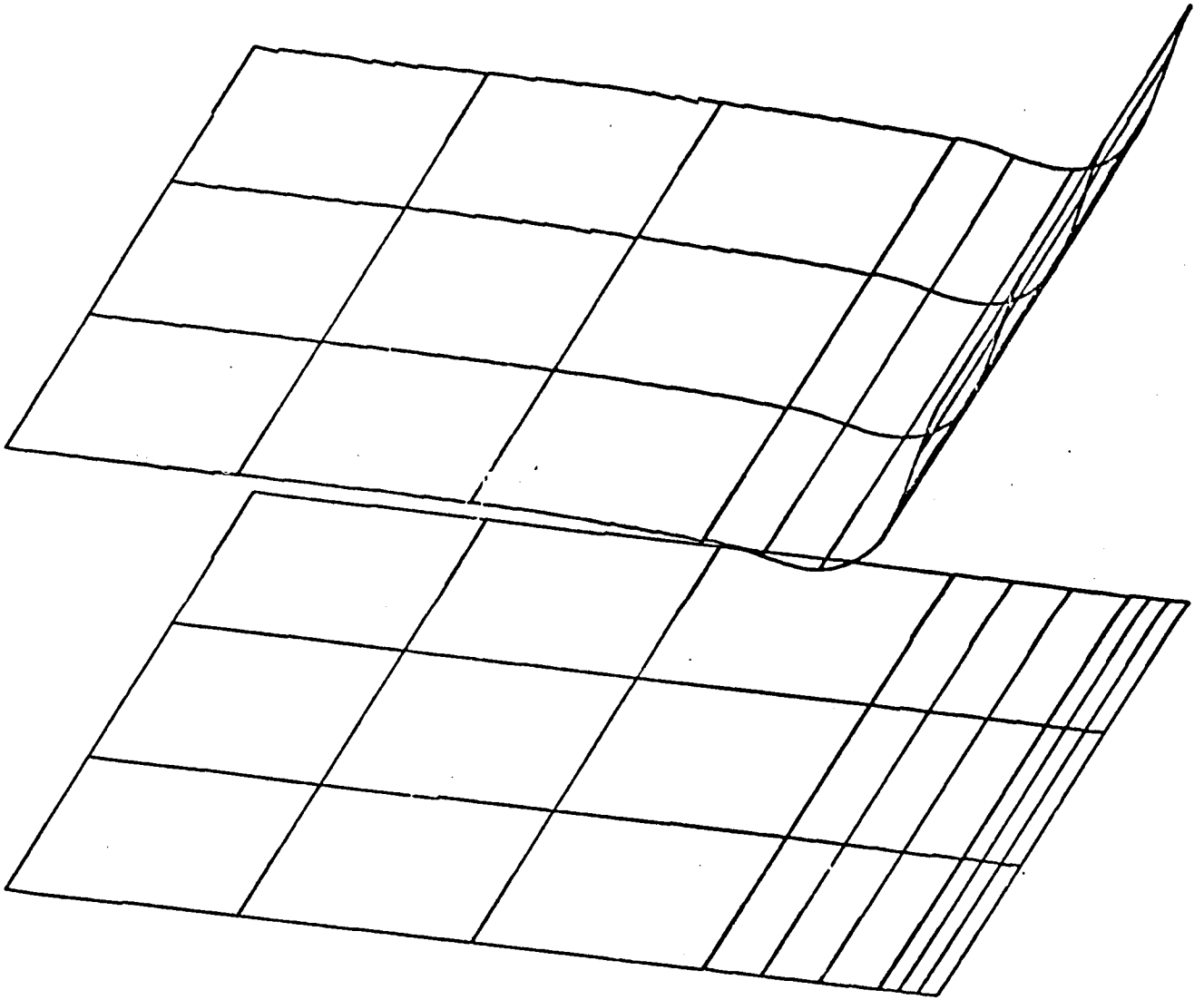


Figure 15: Normal Stress Data Surface